유니뷰 자바 디버깅 시스템의 설계 및 구현

옥 재 호* · 정 연 정** · 이 공 선** · 윤 기 송**

Design and Implementation of Uniview Java Debugging System

Jae-Ho Ock* · Yeon-Jeong Jeong** · Kong-Seon Lee** · Ki-Song Yoon**

Abstract

Uniview system is a client-server system that runs on heterogeneous distributed systems and supports the debugging of various kinds of programs. Its client system provides a unified debugging concept and interface on various debuggers of heterogeneous hosts. Its server system provides debugging services and features openness and scalability by interoperating with one or more debugger processes existing on the same host. Based on FSF(Free Software Foundation)'s gdb and Sun Microsystems's dbx, Uniview system supports C/C++ programming language in various UNIX environments as well as Windows environments. The proposed system was designed and implemented to support the JAVA language, which is prevalently used in recent heterogeneous distributed systems and was partly extended to make a clear analysis of JAVA class file structure. Sun Microsystems's jdb supplied as a JAVA debugger has very limited functions compared to other programming language debugger. In this paper, Uniview as a JAVA debugging system was implemented to provide debugging technologies which are necessary to debug Java applications but missing in current JAVA language as well as to provide its users with various information.

* (주)에듀뱅크
** 한국전자통신연구원

1. 서 론

최근 중소형급 컴퓨터 시스템이 다양해지면서 클라이언트 서버 시스템과 같은 이기종 컴퓨터에서 동작하는 분산 시스템의 중요도가 높아지고 있다. 그러나 다양한 방법론과 개발환경이 제시 되고 있음에도 불구하고 분산 디버깅의 어려움으로 인하여 분산 시스템 개발에 여전히 많은 비용이 소요되며, 순차적 프로그램의 디버깅보다 훨씬 복잡하다.

이러한 배경에서 개발된 유니뷰 시스템은 클라이언트 서버 모델의 분산처리 디버거로써 유니뷰 시스템 클라이언트는 이기종 호스트의 각 디버거들에 대한 단일화된 디버깅 개념과 인터페이스를 제공하며, 유니뷰 디버깅 서버는 동일 호스트에 존재하는 한 개 이상의 디버거 프로세스와 연동함으로써 디버깅 서비스를 제공하며 개방성과 확장성을 가지므로 분산 시스템을 효과적으로 디버깅 할 수 있는 도구를 제공한다. 또한 유니뷰 시스템은 네트워크로 연결된 다양한 시스템의 프로그램을 디버깅하기 위해 다양한 플랫폼과 프로그래밍 언어를 지원한다. 최근 자바는 플랫폼 독립적인 특성으로 인하여 많은 이기종간의 분산 네트워크 환경에서 프로그래밍 언어로 활용되고 있지만 현재 제공되는 자바 언어용 디버거는 다른 언어용 디버거에 비해 기능이 떨어지거나 제한되어 있다.

따라서 본 논문에서는 자바 언어에 대한 부족한 디버깅 기술을 해결하고 사용자에게 다양한 디버깅 정보를 제공하기 위하여 유니뷰 자바 디버깅 시스템을 개발하였다. 유니뷰 자바 디버깅 시스템은 자바 디버거로 알려진 SUN사의 jdb를 바탕으로 디버깅 서버를 구현하였으며 jdb가 제공하지 않는 디버깅 정보를 위해 자바 가상기계의 명세를 바탕으로 자바 클래스 파일을 분석하고 자바 디버깅 정보를 추출하였으며. 이들 정보를 이용하여 디버깅 환경을 구축하고 통합된 환경에서 상호 작용하여 디버깅할 수 있도록 설계 및 구현하였다.

본 논문의 구성은 다음과 같다. 2절에서는 유니뷰 시스템에 관하여 간략히 소개하며, 3절에서는 유니뷰 자바 디버깅 시스템의 설계에 관해 설명한다. 4절에서는 구현된 유니뷰 자바 디버깅 시스템에 관해 설명하며, 5절에서는 타 시스템과의 비교/평가에 관해 설명한다. 끝으로 6절에서는 연구의 결과를 요약하고 이후 연구 방향을 살펴봄으로써 결론을 맺는다.

2. 유니뷰 시스템

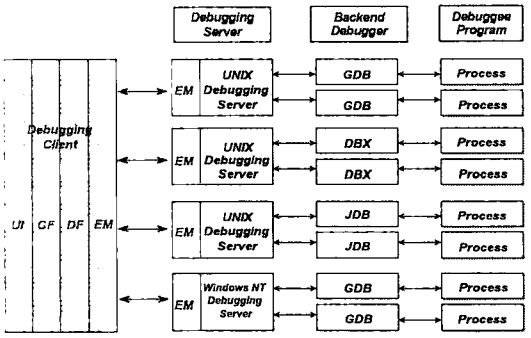
유니뷰 시스템은 디버깅 클라이언트, 디버깅 서버, 이벤트 매니저, 모니터의 4부분으로 구성되어 있다. 디버깅 서버는 후단부 디버거와의 인터페이스를 통해 프로그램의 디버깅을 제어한다. 디버깅 서버는 디버깅 클라이언트로부터 전달되는 사용자의 디버깅 요청을 전달 받아 후단부 디버거를 이용하여 디버깅을 수행하며, 그 결과를 다시 디버깅 클라이언트에게 전해주는 역할을 한다. 디버깅 클라이언트는 사용자 인터페이스를 제공하여 여러 개의 프로그램을 디버깅할 수 있게 해 주며 서버들과의 통신을 통해 사용자의 디버깅 요구를 전달한다. 이벤트 매니저는 디버깅 클라이언트와 디버깅 서버의 통신을 담당하는 부분 시스템으로 디버깅 클라이언트와 디버깅 서버에 의해 공유한다. 모니터는 디버깅 클라이언트와 디버깅 서버를 연결시키는 일종의 브로커이다[2, 3, 7].

2.1 디버깅 서버 (Debugging Server)

디버깅 서버는 각 호스트 상에서 동작하는 프로그램을 제어하고 디버깅 정보를 유지 관리하

는 역할을 한다. (그림 1)에서 보는 바와 같이 여러 종류의 유닉스 환경과 마이크로소프트 윈도우 NT 시스템을 각 플랫폼의 기존 네이티브 디버거를 후단부로 하여 서버를 구현하였다[11].

디버깅 서버는 디버깅 클라이언트로부터 생성 요청이 있을 때마다 생성되며, 각 디버깅 서버는 자신에게 포함된 프로세스들을 디버깅하고 있는 여러 후단부 디버거들을 관리할 수 있는 구조로 되어있다. 유닉스 시스템과 마이크로소프트 윈도우 NT 시스템에서는 gdb를 후단부 디버거로 사용하였다. 선사의 솔라리스 네이티브 컴파일용 서버는 dbx를 후단부 디버거로 사용하였으며, 자바 언어용 디버깅 서버는 jdb를 후단부 디버거로 사용하였다.



(그림 1) 유니뷰 시스템의 클라이언트/서버 구조

이러한 구조는 각 플랫폼마다 벤더들에 의해 제공되는 디버거를 이용하여 시스템을 재구성하는데 용이하며 서버의 이식성을 증대 시킬 수 있는 이점을 가지게 된다. FSF(Free Software Foundation)의 gdb의 장점은 gdb의 소스 코드를 자유롭게 수정 가능하다는 점과 gdb 자체가 높은 이식성을 가지고 있다는 점이다. 선사의 dbx는 상업용 디버거이므로 선사의 지원을 받을 수 있으며 솔라리스 플랫폼에서 개발시 많이 사용되고 있다. jdb는 자바 언어용 개발 도구의 표준으로, 자유롭게 사용할 수 있다. 결과적으로 이 모델은 별도의 디버거가 존재하는 다양한 플랫폼용 서버의 개발이 가능하다는 강점을 가지게 된다. gdb 기반 서버는 다음 일련의 과정을 통하여 클라이언트의 요청을 서비스하게 된다.

- 클라이언트의 디버깅 요청을 일련의 디버거 명령어로 번역한다.
- 해당 프로세스를 디버깅하고 있는 디버거에게 명령어를 전달한다.
- 디버거의 처리 결과가 도착하면 이를 분석하여 클라이언트에게 응답할 수 있는 형태로 가공한다.
- 가공한 결과를 서비스를 요청한 클라이언트로 이벤트 매니저를 통해 전달한다.

위의 마지막 단계에서 gdb로부터 오는 응답은 비동기적으로 발생하게 되므로 클라이언트로 전달되는 결과는 콜백(callback) 메커니즘을 사용한다. 예를 들어, 클라이언트로부터 디버깅 요청을 받게 되면 서버는 gdb의 명령어를 이용하여 gdb에게 디버깅을 요청한다. gdb에 의해 디버깅 실행 결과가 서버로 반환되면 콜백 함수를 통해 클라이언트로 그 결과가 전달된다.

2.2 디버깅 클라이언트(Debugging Client)

클라이언트는 사용자 인터페이스를 제공하고 디버깅 서버들과 통신하면서 디버깅 정보를 총괄 처리하는 역할을 한다. (그림 1)과 같이 4개의 모듈로 구성되어있다[1].

- UI(User Interface) : 사용자 인터페이스만을 담당하는 부분이다
- GF(GUI Framework) : 그래픽 사용자 인터페이스를 위한 정보 및 논리를 가지고 있으며 DF와 UI사이의 연결하는 부분으로 인터

페이스의 단순화를 위한 계층이다. UI에게 윈도우 메시지를 통해 시그널을 보내는 부분이기도 하다.

- DF(Debugging Framework) : 디버깅 정보 및 논리를 실제로 관리하는 클라이언트의 핵심 부분으로 UI에서 요구하는 디버깅 정보를 정의하고 관리하며, 자주 사용되는 정보를 캐싱하는 기능과 서버로부터 받은 정보의 변화를 UI에게 알리는 역할을 담당하는 부분이다.
- EM(Event Manager) : 서버와 클라이언트와의 통신을 담당하는 부분이다.

2.3 모니터(Monitor)

디버깅 서버가 실행되는 시스템에는 디버깅 클라이언트와 디버깅 서버의 연결시켜주는 모니터가 실행되고 있으며, 이 프로세스는 일종의 데몬 프로세스로서 하나의 시스템에 항상 하나만 존재한다. 모니터는 클라이언트의 서버 요청에 의해 서버를 생성시키고, 생성된 서버에게 클라이언트의 포트를 알려준다.

2.4 이벤트 매니저(Event Manager)

서버와 클라이언트 그리고 모니터 상호간의 통신 메커니즘을 제공하는 라이브러리로써 산업계의 표준으로 가장 널리 사용하고있는 ONC/RPC를 기반으로 구현되었으며[12], 다양한 플랫폼에 동작할 수 있도록 이식성과 전체 시스템의 모니터에 의한 서버와 클라이언트의 연결 구성에 따라 유연하게 대처할 수 있는 확장성 그리고 다국어 지원이 가능하도록 멀티 바이트 문자의 지원 등을 고려하여 설계 구현되었다[4]. 현재 구현된 라이브러리는 마이크로소프트 윈도우 NT, SunOS, 솔라리스 2.x, 리눅스, HP-UX 10.20 등에 이식되어 사용되고 있다.

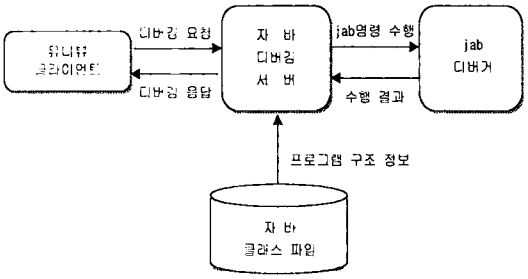
3. 유니뷰 자바 디버깅 시스템의 설계

3.1 유니뷰 자바 디버깅 시스템의 개요

유니뷰 자바 디버깅 시스템은 기존의 gdb를 이용한 유니뷰 시스템에 jdb를 이용한 디버깅 서버를 추가 함으로써 C/C++뿐만 아니라 자바 언어에 대한 디버깅이 가능하도록 한 시스템이다. gdb는 강력한 디버깅 기능을 제공함에도 불구하고 소스가 공개되어 있는 디버거이다[10]. 유니뷰 시스템에서는 디버깅할 프로그램의 프로그램 구조 분석을 위해 gdb의 일부 코드를 수정하였다[6]. gdb는 프로그램 구조 분석을 위해 다양한 정보를 제공하지만 정보의 양이 방대하고 유니뷰 시스템에 불필요한 정보를 포함하고 있기 때문에 gdb에 일부 명령을 추가 함으로써 유니뷰 시스템에서 사용될 수 있는 정보만을 포함할 수 있도록 하였다. 추가한 명령은 프로그램을 구성하는 링키지 유닛의 리스트를 출력하는 명령어, 해당 링키지 유닛의 구성 모듈들의 리스트를 출력하는 명령어, 모듈의 구성 함수들의 리스트를 출력하는 명령어 그리고 해당 함수를 구성하는 스테이트먼트 테이블을 구성하는 명령이 있다. 그러나 유니뷰 자바 디버깅 시스템에서 사용되는 jdb는 프로그램 구조를 이해할 수 있는 명령어를 제공하지 않고, gdb처럼 소스가 공개되어 있지 않아 jdb만으로는 디버깅 서버의 역할을 할 수 없다.

하지만 자바 프로그램은 컴파일시 생성되는 중간코드인 클래스파일내에 프로그램 구조에 대한 정보를 포함하고 있어, 디버깅하는 클래스파일에 대한 분석을 통해 jdb가 제공하지 못하는 프로그램 구조에 대한 정보를 얻을 수 있다. (그림 2)는 클라이언트, 서버 및 jdb, 자바 클래스파일 사이의 연관성을 나타내는 그림이다. 그림에서 유니뷰 자바 디버깅 서버는 후단부 jdb

와 수도 터미널(psedo terminal)을 통해 디버깅 실행 명령어와 실행 결과를 주고 받으며, 자바 클래스 파일을 통해 프로그램 구조에 대한 정보를 분석하여 클라이언트의 요구에 맞게 디버깅 정보를 보내준다[5].

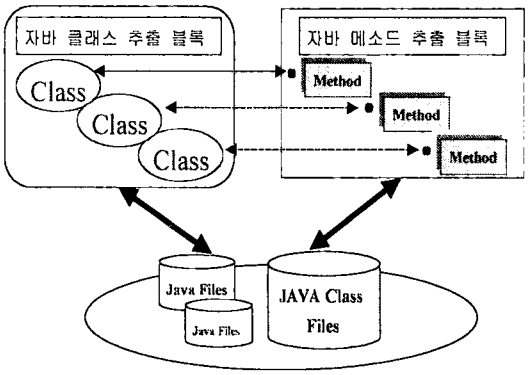


(그림 2) 클라이언트, 서버 및 jdb사이의 연관도

유니뷰 클라이언트의 요청에 따라 디버깅 정보를 생성하여 응답하는 유니뷰 자바 디버깅 서버는 클래스파일을 분석하는 자바 클래스파일 분석부, 메소드 사이의 관계를 규명하는 자바 메소드 호출 분석부 그리고 자바 클래스파일 분석부의 분석결과와 jdb의 실행 결과를 바탕으로 디버깅 정보를 생성하는 자바 디버깅 정보 추출부로 나눌수 있다. 다음 절에서는 자바 디버깅 서버 각각에 대한 구조와 기능에 관해서 설명한다.

3.2 자바 클래스파일 분석부

자바 클래스파일 분석부는 자바 클래스파일로부터 클래스의 속성(Attributes), 상수집합(ConstantPool), 필드(Fields), 메소드(Methods) 등을 분석하여 클래스 리스트와 메소드 리스트를 구성한다. 자바 클래스 파일은 일정한 형식의 자료 구조를 가지고 있는데, 각각의 필드 별로 구분된 자료 구조로 저장되며 이 자료구조는 자바 가상 머신 스팩[8, 9, 13]에 제시된 형태를 따른다. (그림 3)은 자바 클래스파일 분석부의 구조를 보여준다.

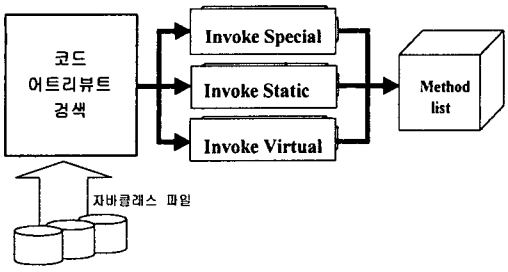


(그림 3) 자바 클래스파일 분석부의 구조

자바 디버깅 서버에서 기본이 되는 자바 클래스파일 분석부는 자바 클래스 추출 블록과 메소드 추출 블록으로 나눌 수 있다. 자바 클래스 추출 블록은 메인 메소드가 들어 있는 클래스부터 메소드 영역의 바이트 코드를 분석하여 실행시에 나타날 수 있는 모든 클래스를 추적하고 그 내용을 추출하는 기능을 제공한다. 한 클래스 파일의 피호출부에서 호출부의 클래스를 읽어 들이고 이러한 정보는 일련의 클래스 계층 구조도와 비슷한 구성을 가진다. 자바 클래스 메소드 추출 블록은 자바 클래스에 포함되는 생성자 및 일반 사용자 정의 메소드를 추출하는 기능을 제공한다. 이것은 자바 클래스 추출 블록의 클래스 리스트를 모두 검색하여 각각의 메소드 리스트를 추출하게 된다.

3.3 자바 메소드 호출 분석부

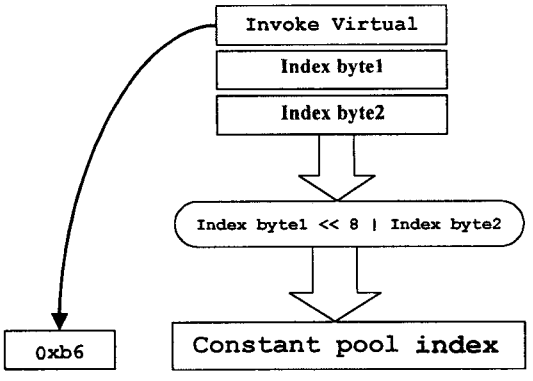
자바 메소드 호출 분석부는 3.2에서 기술한 자바 클래스 파일 분석부의 결과를 입력으로 하여 메소드 사이의 호출 관계를 규명하고 디버거의 사용자가 프로그램의 구조를 쉽게 파악할 수 있도록 하며 실제 사용자가 작성한 메소드 사이의 호출 관계를 보여준다. (그림 4)는 자바 메소드 호출 분석부의 구성을 보여준다.



(그림 4) 자바 메소드 호출 분석부의 구성

자바 바이트 코드에서 메소드의 호출에 사용되는 명령어는 모두 세 가지이다. 첫째, 일반적인 메소드의 호출에 사용되는 "invoke virtual", 둘째, 클래스의 생성자와 같은 특수한 메소드의 호출에 사용되는 "invoke special", 마지막으로 정적 메소드 호출에 사용되는 "invoke static"으로 구성된다. 각각의 메소드 호출 명령어들은 일정한 형태를 지니고 있는데, 명령어 다음에 2개의 연속된 1 바이트 크기의 인덱스를 가진다. 이 인덱스는 (그림 5)의 연산과정을 통하여 클래스 파일 객체내의 상수집합을 가리키게 된다.

상수 집합 인덱스는 호출하려는 메소드의 이름을 가리키는 인덱스이며, (그림 5)에서 인덱스를 생성하는 연산 과정을 보여 주고 있다. 상수집합 정보를 통해 호출할 메소드의 클래스와 modifier 등도 알 수 있으며, 검색된 결과는 리스트에 저장된다.

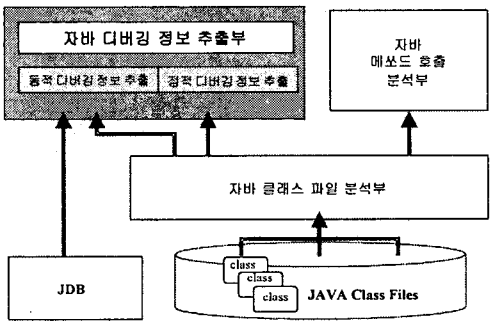


(그림 5) 인덱스 연산 과정

3.4 자바 디버깅 정보 추출부

자바 디버깅 정보 추출부는 자바 클래스 파일 분석부의 내용을 기반으로 하여 디버깅 작업시 필요로 하는 정보를 구성하고 사용자의 요구에 의해 필요로 하는 디버깅 정보를 클라이언트에 전달한다. 클라이언트는 서버에서 제공하는 정보를 바탕으로 디버깅 정보 및 결과를 사용자에게 보여준다.

(그림 6)은 자바 디버깅 정보 추출부의 구성을 보여준다.



(그림 6) 자바 디버깅 정보 추출부의 구성

자바 디버깅 정보 추출부는 정적 디버깅 정보 추출 블록과 동적 디버깅 정보 추출 블록으로 나뉜다. 정적 디버깅 정보 추출 블록은 자바 클래스파일에서 얻어지는 객체들로서 프로그램 정보, 스테이트먼트 정보, 소스 코드 정보, 속성 정보로 구성된다. 동적 디버깅 정보 추출 블록은 프로그램이 정지할 때마다 값이 수정되는 객체들로서 스택 정보, 브레이크 포인트 정보, 파라미터 변수 정보, 지역 변수 정보로 구성된다.

각각을 좀더 세분화 하여 살펴 보면 정적 디버깅 정보 추출 블록의 프로그램 정보 기능은 디버깅되고 있는 프로그램의 현재 스택 정보, 브레이크포인트 관리자, 링키지 유닛 관리자, 모듈 관리자 등의 각종 관리자들의 포인터, 정

적 정보의 시작점이 되는 링키지 유닛의 포인터, 레지스터 정보, 프로세스 ID, 메인 스코프 등의 다양한 프로그램의 현재 상태를 나타내는 정보를 가진다. 스테이트먼트 정보 기능은 자바 클래스 파일로부터 그 클래스의 메소드가 갖고 있는 실행가능한 스테이트먼트에 대한 정보를 추출한다. 소스 코드 정보 기능은 디버깅시 사용자가 디버깅하고 있는 메소드의 소스를 추출하여 보여주는 기능이며, 속성 정보 추출 기능은 자바 클래스와 관련된 일반적인 정보, 필드 정보, 메소드 정보를 자바 클래스 파일로 부터 추출하는 기능이다. 일반적인 정보로는 소스 파일 이름, 검색시간(Access Time), 수정시간(Modify Time), 클래스 파일 이름 등이고, 필드 정보는 자바 클래스가 가지는 모든 필드에 대한 이름, 데이터 타입이 있다. 메소드 정보는 자바 클래스가 가지는 메소드의 리턴 타입, 메소드 이름, 파라미터 이름, 파라미터 데이터 타입이 있다.

동적 디버깅 정보 추출 블록의 스택 정보 기능은 jdb의 "info stack" 명령을 이용하여 디버깅하고 있는 프로그램의 변화를 분석한다. 프로그램의 변화를 알기위해 사용되는 정보로는 소스파일이름, 라인정보, 코드어드레스, 스코프 이름 등이 있다. 브레이크 포인트 정보 기능은 jdb의 "stop" 명령의 결과값을 이용하여 소스파일이름, 라인정보, 코드어드레스, 스코프 이름, 스테이트먼트값 오프셋값 등의 정보를 연결리스트에 저장한다. 예를 들어 라인으로 브레이크 포인트를 지정한 경우 라인 정보를 제외한 나머지 정보는 코드록을 분석하여 원하는 값을 채워 넣는다. 파라미터 변수 정보 기능은 디버깅 되고 있는 클래스의 메소드가 가지는 파라미터 변수에 대한 정보를 추출한다. 파라미터 변수에 대한 정보는 이름, 데이터 타입, 배열의 크기이다. 또한 변수가 클래스인 경우 클래스가 가지는 클래스변수에 대한 이름, 데이터 타입 정보를 추출한다. 지역 변수 정보 기능은 디버깅 되고 있는 클래스의 메소드가 가지는 지역 변수에 대한 정보를 추출한다. 지역 변수에 대한 정보는 이름, 데이터 타입, 배열의 크기이다. 파라미터 변수와 같이 변수가 클래스인 경우 클래스가 가지는 클래스변수에 대한 이름, 데이터 타입 정보를 추출한다.

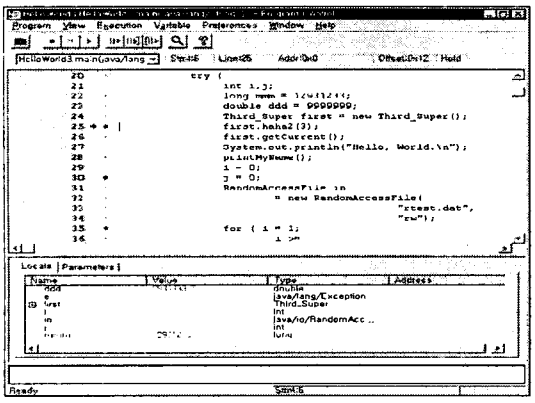
4. 유니뷰 자바 디버깅 시스템의 구현

유니뷰 자바 디버깅 서버는 후단부 jdb를 기반으로 하는 서버 시스템이다. 기존의 gdb를 이용한 디버깅 서버와의 호환성을 그대로 유지하기 위해 기본적인 아키텍처를 그대로 유지하면서 jdb 명령을 기반으로 작성되었다. 그리고 jdb가 제공하지 못하는 기능을 위해서 자바 가상머신에서 동작하는 자바 클래스 파일을 이용하였다.

초기 개발 시스템으로는 솔라리스 2.5.1기반하에 JDK(Java Developt Kit)1.2.1에서 제공하는 jdb를 이용하였으며, 여러 시스템에서의 분산 디버깅을 지원하기 위해 Linux, HP 10.20, Windows NT, Windows 98시스템에 이식하였다.

(그림 7)은 유니뷰 클라이언트 시스템의 프로그램 제어 윈도우에서 "resume"명령을 수행한 화면이다. "resume"은 프로그램 수행 제어 명령 중의 하나로서 현 정지 상태에서 프로그램의 수행을 재개하여 다음 정지할 원인이 발생할 때까지 수행을 계속하는 명령어로서 jdb명령어의 "cont"에 상응한다. 프로그램 제어 윈도우는 분산 시스템을 구성하는 프로그램들에 대해 각각 그 프로그램을 디버깅하는 주요 윈도우로서 프로그램의 수행 제어 명령을 위한 메뉴 및 툴바와 프로그램의 정지 위치를 보여주는 창, 소스 부분, 그리고 변수 부분으로 이루어져 있다. 소

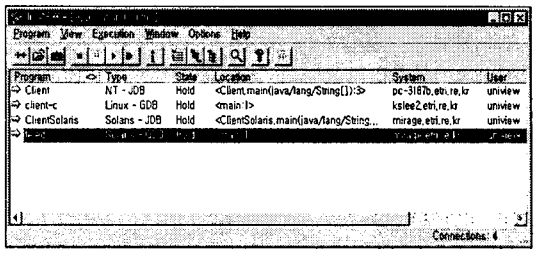
스 부분은 소스 내용과 현재 위치, 그리고 브레이크포인트 상태를 보여주며 변수창에서는 프로그램이 정지 상태에 있을 때 현재 위치에서 지역변수 및 매개변수의 값을 보여준다. 그림에서 현재 25라인, 30라인, 35라인에 브레이크포인트가 설정되어있으며, resume명령 실행후 25라인에서 프로그램이 정지상태임을 나타낸다. 그림 하단에는 현 정지상태에서 로컬 변수값의 변화를 빨간 색으로 나타내고 있다.



(그림 7) 프로그램 제어 윈도우에서 resume명령을 수행한 화면

(그림 8)은 유니뷰 클라이언트의 어플리케이션 윈도우이다. 어플리케이션 윈도우는 분산 시스템의 모든 프로그램들을 한눈에 파악하고 제어하는 중앙 제어 윈도우로서 프로그램의 상태를 보여주는 아이콘과 프로그램의 이름, 수행서버의 플랫폼 종류, 현재 프로그램의 위치, 호스트 주소, 사용자 등의 정보를 보여주다. 그림에서 자바 디버거 서버를 포함한 4개의 디버거 서버가 윈도우NT,Linux,Solaris 플랫폼에서 동작하는 모습을 나타내고 있다. 오픈된 프로그램들은 수행가능한 첫번째 문장에서 정지하여 수행제어 명령을 기다리고 있으며 상태는 HOLD임을 알 수 있다.

유니뷰 시스템은 하나 이상의 언어로 구축된 분산 시스템을 디버깅하는 기능을 지원한다. 이



(그림 8) 유니뷰 클라이언트의 어플리케이션 윈도우 화면

것은 유니뷰 시스템이 클라이언트 서버 아키텍처를 기반으로 하고 있으며 서버 부분의 후단부 디버거 및 디버거 인터페이스 부분만 수정함으로써 다양한 언어지원이 가능하기 때문이다. 유니뷰 자바 디버깅 시스템은 기본 디버깅 기능을 타 언어와 동일하게 지원하는 것외에 자바 언어 고유의 특성을 지원하기 위하여 클래스 파일의 구조를 분석하여 보여주는 클래스 브라우저 윈도우와 클래스 간의 계층구조를 보여주는 클래스 계층구조 윈도우를 추가하였으며 이를 위해 기존의 클라이언트 시스템 및 서버 시스템을 새로운 기능을 위해 확장하였다. 각각에 대해 설명하면 다음과 같다.

클래스 브라우저 윈도우는 지정된 하나의 클래스에 대해 관련된 정보를 보여주는 윈도우로서 클래스 전체에 관한 요약정보, 메소드와 필드의 리스트, 그리고 각 메소드 및 필드에 대하여 상세 정보 등을 보여준다. 두개의 패인윈도우로 이루어져 있으며 왼쪽은 트리구조를 가지고 오른쪽은 리스트 구조를 가진다. 왼쪽 트리구조는 클래스의 이름을 나타내는 루트 노드에 인터페이스, 메소드, 필드에 해당하는 1차 자식 노드 들이 있고 메소드와 필드 노드는 각각 이 클래스의 메소드 리스트 및 필드 리스트를 표시하게 확장(expand)될 수 있다. 오른쪽 리스트 구조는 왼쪽의 트리구조에서 선택된 노드에 따라 아래와 같은 내용의 상세정보를 표시한다. 그 내용은 다음과 같다

- 클래스 이름이 선택된 경우
  - Magic Number(매직 넘버)
  - Major Version(메이저 버전)
  - Minor Version(마이너 버전)
  - Access Flags(엑세스 플래그)
  - This Class(현 클래스 이름)
  - Super Class(슈퍼 클래스 이름)
  - Constant Pool Length(콘스탄트 풀 길이)
  - Number of Interfaces(인터페이스의 수)
  - Number of Fields(필드 수)
  - Number of Methods(메소드 수)
  - Number of Attributes(어트리뷰트 수)

- 인터페이스 노드가 선택된 경우
  이 클래스의 인터페이스의 이름을 나타낸다. 인터페이스의 개수는 제한이 없으므로 리스트에서 하나의 인터페이스에 한 줄씩을 표시하게 된다.

- 하나의 메소드 이름 노드가 선택된 경우
  - Name(메소드 이름)
  - Descriptor(메소드에 대한 디스크립터)
  - Access Flags(엑세스 플래그)
  - Number of Attributes(어트리뷰트 수)

- 하나의 필드 이름 노드가 선택된 경우
  - Name(필드 이름)
  - Descriptor(필드에 대한 디스크립터)
    Access Flags(엑세스 플래그)
    Number of Attributes(어트리뷰트 수)

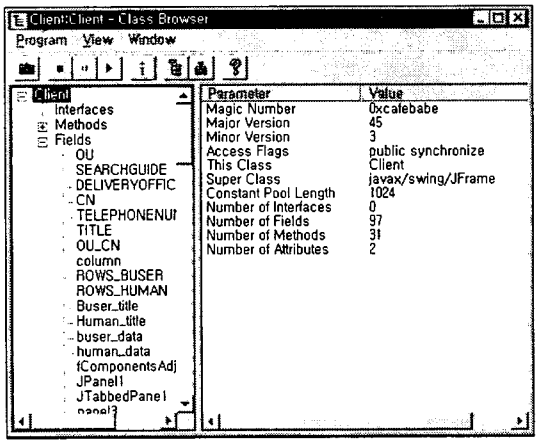
유니뷰 시스템에서는 확장된 자바 기능을 지원하기 위해 유니뷰 클라이언트에 대한 확장이 필요하다. 유니뷰 클라이언트는 사용자 인터페이스를 담당하는 UI 계층과 서버로부터 받아온 정보를 직접 관리하는 DF 계층 그리고 중간에 두 계층을 연결해주는 GF 계층을 가지는데 각 계층에서 클래스 브라우저 윈도우를 위하여 다음과 같은 일을 수행한다.

UI 계층은 사용자 요청을 받아 해당 클래스 정보를 이용하여 GF를 통하여 서버에 정보를 요청한다. 또한 사용자의 노드 선택이나 여러 액션을 받아 필요한 처리를 수행한다. GF 계층에서는 클래스 브라우저 윈도우를 감시 지원하는 클래스(GFClassBrowser)가 하나 있어 클래스 브라우저가 하나 생성될 때마다 지원하는 객체가 하나씩 생성된다. 또한 사용자의 클래스에 대해 대응하는 하나의 객체가 생성되는 클래스(GFObjectModule)가 있어서 클래스 브라우저가 생성되는 클래스에 대해 하나씩 존재한다. 이 두개의 클래스에 의해 사용자가 클래스 브라우저 윈도우가 이미 열려있는 클래스에 대해 다시 브라우저 윈도우를 요청했을 때 대응하는 윈도우를 찾아 활성화(activate)시킬 수 있다. 또한 GFClassBrowser 클래스에서는 아래 계층인 DF 계층에 서버로부터의 정보를 요청할 수있으며 가져온 정보를 UI 계층에서 사용자에게 보여주기 쉬운 형태로 변환하는 일을 담당한다. DF 계층에서는 해당 사용자 클래스에 대해 서버에 클래스 정보를 요청한다. 이 때 해당 요청에 대해 하나의 타스크를 발생시켜 서버로부터 결과가 왔을 때 그 타스크를 활성화하여 GF 클래스에 알려준다(notify). 또한 서버로부터 받은 정보를 DFModule 클래스에 저장한다.

(그림 9)는 클래스 브라우저의 구현된 화면이다. 왼쪽 패인에서 클래스 이름이 선택된 경우 선택된 클래스에 대한 클래스 상세 정보를 오른쪽 패인윈도우에 리스트 형식으로 출력한다.

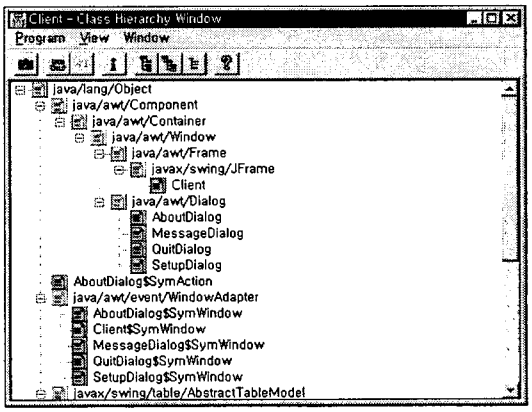
클래스 계층구조 윈도우는 클래스 브라우저 윈도우와 비슷한 기능으로서 지정된 클래스에 대해 계층구조 정보를 보여주는 윈도우이다. 클래스의 계층구조는 프로그램의 정적인 분석을 통해서 알수 있다. 일반적으로 각 클래스는 자신의 상위 클래스에 대한 정보를 가지고, 프로

그램 전체를 분석하면 하위 클래스에 대한 정보도 얻을 수 있다. 본 시스템에서는 먼저 상위 클래스에 대한 정보를 최초의 베이스 클래스까지 표현하고 하위 클래스는 사용자의 요청이 있을 때 표시한다.



(그림 9) 클래스 브라우저 윈도우

클래스 계층구조 윈도우는 클래스 브라우저 윈도우와 달리 프로그램 하나에 대해 하나의 윈도우만 존재한다. 사용자는 윈도우의 메뉴를 이용하여 지정된 클래스에 대한 트리의 확장(expand)를 요청할 수 있다. 이 때 트리는 해당 클래스까지의 최소한의 경로만 확장한다. 또한 요청된 클래스가 이미 확장된 노드인 경우 해당 노드를 하이라이트한다. 그리고 사용자는 어느 특정 노드에 대해 자식 노드들을 모두 확장하도록 요청할 수 있다. 이상의 두가지 사용자 액션을 지원하기 위하여 클라이언트 시스템은 확장된 사용자 클래스에 대해 GF에 GFClassNode라는 타입의 객체를 하나씩 가진다. 이 노드는 해당 클래스의 노드가 트리에 삽입되었는가, 그리고 현재 확장되었는가 여부를 나타낸다. (그림 10)은 "Client"클래스를 기준으로 각 계층관계를 표시한 클래스 계층 구조 윈도우의 화면이다. 툴바의 아이콘을 이용하여 선택된 노드에 대한 확장과 전체 노드에 대한 확장등의 기능을 가진다.



(그림 10) 클래스 계층 구조 윈도우

5. 비교/평가

유니뷰 자바 디버깅 시스템은 유니뷰 시스템에서 자바언어를 지원하기 위한 서버 시스템이다. 본 논문에서는 자바 디버깅 시스템을 포함하는 유니뷰 시스템과 타 시스템을 기술 분야로 나누어 비교/분석하였다.

<표 1>은 유니뷰 시스템과 메시지 기반의 분산 어플리케이션을 디버깅하는 분산된 디버거의 개념을 도입한 타 상용 시스템과 기술분야로 나누어 비교 분석한 표이다. 그리고 표에서 비교하고 있는 토탈뷰 시스템(Total View)[14]은 돌피닉스 사에서 개발하여 상품화한 분산 이기종 다중 프로세스 시스템의 디버거로서 사용하기 쉬운 사용자 인터페이스, 높은 수준의 순차 디버깅 기능, 멀티프로세싱과 병렬 및 분산 프로그래밍의 지원기능, 여러 호스트에 이식 기능 등을 지원한다. 토탈뷰는 분산 병렬 처리 라이브러리인 PVM 및 MPI등의 미들웨어에 기반한 멀티프로세싱 병렬 시스템의 디버깅 인터페이스를 지원하는 것이 주요한 특징이다. 그러나 이기종 플랫폼의 호스트가 혼재하는 경우는 지

〈표 1〉 타 시스템과의 기술 비교

기 술 분 야	본 시스템	기 술 비 교
분산 시스템 관리 기능	• 확장성이 높은 구조로 설계 • 백여개까지 한꺼번에 관리, 제어 가능 • 지리적으로 원격지에서 분산 시스템을 제어 수행가능함	• 타 상용 시스템은 지원하지 않음 • 토탈뷰와 비교하여 분산 시스템의 제어 기능 중 배치 오픈, 동시 제어 등의 기능은 본 제품만 제공됨
분산 시스템 분석 기술	분산된 각 프로그램의 구조를 정적, 동적으로 분석해주는 윈도우 제공	• 타 상용 시스템은 하나의 프로그램에 대해서만 지원 • 정적 분석만 제공됨.(클래스 계층구조는 온라인 도움말 형태로 제공됨)
사용자 인터페이스	• 상용화된 도구의 사용자 인터페이스 수준으로 지원 : 소스/변수, 프로그램 제어, 브레이크포인트 윈도우 등 • 모든 기종과 언어에 대해 동일한 인터페이스 제공	• 상용 시스템과 동일한 수준. • 토탈뷰에서는 유닉스 X 윈도우 환경의 클라이언트만 지원됨.
분산 시스템 관리 기능	• 확장성이 높은 구조로 설계 • 백여개까지 한꺼번에 관리, 제어 가능 • 지리적으로 원격지에서 분산 시스템을 제어 수행가능함	• 타 상용 시스템은 지원하지 않음 • 토탈뷰와 비교하여 분산 시스템의 제어 기능 중 배치 오픈, 동시 제어 등의 기능은 본 제품만 제공됨.
분산 시스템 분석 기술	분산된 각 프로그램의 구조를 정적, 동적으로 분석해주는 윈도우 제공	• 타 상용 시스템은 하나의 프로그램에 대해서만 지원. • 정적 분석만 제공됨.(클래스 계층구조는 온라인 도움말 형태로 제공됨)
사용자 인터페이스	• 상용화된 도구의 사용자 인터페이스 수준으로 지원 : 소스/변수, 프로그램 제어, 브레이크포인트 윈도우 등 • 모든 기종과 언어에 대해 동일한 인터페이스 제공	• 상용 시스템과 동일한 수준. • 토탈뷰에서는 유닉스 X 윈도우 환경의 클라이언트만 지원됨.

원하지 못한다는 점에서 본 시스템과는 차이가 있다.

<표 2>는 현재 어플리케이션의 디버거로써 많이 사용되는 시스템과 본 시스템을 분산 시스

〈표 2〉 타 시스템과의 기능 비교

분산시스템 디버깅기능 구분	GDB / DBX	JDB	비쥬얼 스튜디오	시맨텍 비쥬얼 카페	토탈뷰	유니뷰
이기종 지원 (지원 플랫폼)	유닉스 계열만 지원	JDK 지원 기종	윈도우 NT, 윈도우 98	윈도우 NT, 윈도우 98	유닉스 계열만 지원	유닉스 계열, 윈도우 NT, 윈도우 98
다중언어 지원 (지원 언어)	C, C++	자바	C, C++, Basic, 자바	자바	C, C++, 포트란	C, C++, 자바
통신 이벤트 추적 기술	×	×	×	×	○	○
분산 시스템 관리 기능	×	×	×	○ (원격 디버깅 방식)	○	○
분산 시스템 분석 기술	×	×	△ (정적 분석만 제공)	△ (정적 분석만 제공)	×	○
사용자 인터페이스	명령어 기반 인터페이스	명령어 기반 인터페이스	그래피컬 인터페이스	그래피컬 인터페이스	× 윈도우 기반 인터페이스	MFC 기반 인터페이스

템 디버깅 기능 중심으로 비교한 표이다.

6. 결 론

유니뷰 시스템은 이기종 분산환경에서 동작하는 프로그램을 디버깅하는 시스템으로써, 단일 사용자 인터페이스를 통해서 각 프로그램의 디버깅을 수행한다. 디버깅 가능한 언어로써는 C/C++ 뿐만 아니라 이기종간의 분산 네트워크 환경에서 프로그래밍 언어로 현재 많이 활용되고 있는 자바 언어를 지원한다. 본 논문에서는 자바 언어를 지원하기 위해 후단부 jdb를 이용한 유니뷰 자바 디버깅 시스템을 개발하였으며, 자바 가상머신에서 동작하는 클래스 파일의 분석을 통해 jdb가 제공하지 못하는 기능을 해결하였다. 또한 자바 언어에 대한 더욱더 정확한 디버깅 정보를 제공하기 위해 클래스 브라우저와 클래스 계층 윈도우를 추가하였다.

향후 연구 계획으로는 자바 디버거 서버 개발 중에 발견된 jdb의 버그 및 기능들에 대한 좀더 세밀한 연구 과정을 통해 자바 언어를 지원하기 위한 강력한 디버거를 개발 해야 하겠다. 또한 유니뷰 시스템이 이기종 분산환경에서 동작하는 다양한 프로그램을 디버깅하는 도구이긴 하지만, 제한된 운영환경 및 기능만을 갖고 있다. 따라서 분산환경 시스템 개발시 필요한 새로운 디버깅 기능들이 요구되며, 보다 다양한 운영체제를 위한 디버깅 서버와 디버깅 클라이언트의 개발 및 마이크로 소프트 비쥬얼 C++등과 같이 여러 종류의 컴파일러와 다양한 프로그램 개발 언어에 대한 지원이 필요하다.

참 고 문 헌

[1] 서영애 외 2인, "RPC에 기반한 분산 시스템의 디버경을 위한 이벤트 추적 기능의 실계 및 구현", 정보과학회논문지(C), 제5권 제3호, 1999, pp.313-325.

[2] 성명제 외 4인, "분산처리 진단/교정 시스템의 개발에 관한 연구", 한국정보과학회 97년 춘계학술발표논문집, 제24권 제1호(A), 한림대학교, 1997, pp.37-40.

[3] 성명제 외 3인, "분산 디버거 유니뷰 시스템의 개발", 한국정보과학회, Vol.4, 1998, pp.527-541.

[4] 이은정 외 4인, "분산 처리 진단/교정 시스템의 이벤트 추적 기능의 설계 및 구현", 정보과학회 97년 추계학술발표 논문집, 제24권 2호(IV), 이화여자대학교, 1997, pp. 135-138.

[5] 옥재호 외 2인, "유니뷰 jdb서버의 수행 제어 명령의 설계 및 구현", 한국정보처리학회 99년 추계학술발표논문집, 제6권 제2호, 광운대학교, 1999, pp.131-133.

[6] 조영욱 외 4인, "분산처리 디버거 유니뷰 서버의 구조", 정보과학회 97년 추계학술발표논문집, 제24권 제2호(I), 이화여자대학교, 1997, pp.327-330.

[7] 정보통신부, "분산처리 진단/교정 소프트웨어 개발에 관한 연구", 1차년도 연구개발 결과 보고서, 1997.

[8] A. Taivalsaari, "Implementation a Java Virtual Machine in the java programming Language," SUN Lab, 1997.

[9] B. Venners, "Inside Java Virtual Machine," McGraw-Hill, 1997.

[10] Cygnus support, *Working with gdb*. 매뉴얼.

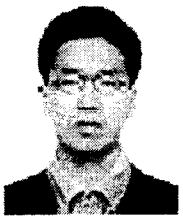
[11] E. Lee, et. al., "UniVIEW : A distributed debugger for client/server systems," SERI journal, 1998, 1.

[12] J. Bloomer, *Power programming with RPC*, O'Reilly & Associates Inc.

[13] T. Lindholm and F. Yellin, The JavaTM Virtual Machine Specification ADDISON-WESLEY, 1997.

[14] Total View Home Page, http://www.dolphincis.com/tw/tvfandb.htm, 1997.

■ 저자소개



옥 재 호

경남대학교 전자계산학과를 졸업하고, 경남대학교 컴퓨터공학과에서 공학 석사 학위를 취득하였으며, 한국전자통신연구원 네트워크 소프트웨어팀에서 하이브리드 메시징 시스템을 개발하고 있다. 주요 관심분야는 분산네트워킹, 전자상거래, 자바 가상머신 등이다.

정 연 정

부산대학교 전자계산학과를 졸업하고, 부산대학교 전자계산학과에서 이학석사 학위를 취득하였으며, 한국전자통신연구원 네트워크 소프트웨어팀에서 하이브리드 메시징 시스템을 개발하고 있다. 주요 관심분야는 정보보호, 분산처리, 실시간 처리등이다.

이 공 선

연세대학교에서 전자계산학 석사, 학위를 취득하였으며 한국전자통신연구원에서 소프트웨어공학관련 업무를 담당하고 있으며, 주요관심분야는 소프트웨어공학분야이다

윤 기 송

부산대학교 조선공학과를 졸업하고, City University of New York에서 전산학석사 학위를 취득하였으며, 한국전자통신연구원 책임연구원으로 네트워크 소프트웨어팀에서 하이브리드 메시징 시스템을 개발하고 있다. 주요 관심분야는 정보보호, 메시지, 분산처리 등이다.