

ERLANG으로 구현한 웹서버 YAWS의 기능개선

장재원^o 양하영 유정목 이만호
 충남대학교 컴퓨터학과
 {jaeback^o, hyyang, jmyoo, mhlee}@cs.cnu.ac.kr

Improvement of YAWS implemented by ERLANG

Jae-Won Jang, Ha-Young Yang, Jeong-Mok Yoo, Mann-Ho Lee
 Dept. of Computer Science, Chungnam National University

요 약

함수형 언어를 사용하여 작성한 응용 어플리케이션은 성능이 떨어진다는 생각이 지배적이다. 함수형 언어 ERLANG은 성능이 우수한 통신응용프로그램을 작성하기 위하여 개발되었다. ERLANG으로 작성한 웹서버인 YAWS는 범용 웹서버와 성능면에서 뒤지지 않지만 모든 웹프로그래밍 언어를 지원하지 않는다. 본 연구는 Jsp/Servlets을 서비스 할 수 있도록 YAWS의 기능을 개선하였다.

1. 서론

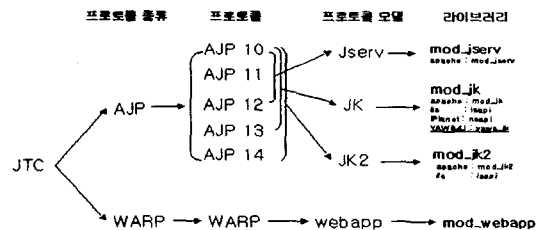
스웨덴의 통신회사 Ericsson에서 1998년에 개발한 ERLANG은 함수형 언어의 일반적인 특징 중에서 currying, high-order functions, lazy evaluation, ZF comprehension등을 제거한 작고 가벼우면서도 신뢰할 만한 대용량의 산업 어플리케이션을 프로그래밍할 수 있는 함수형 언어다. ERLANG은 동시적(concurrent), 분산된 환경에서의 장애 허용(distributed fault-tolerant), 실시간시스템을 프로그래밍하기 적합하여 통신, 네트워크 분야에서 활발한 연구가 진행중이다. [1][2] 본 논문은 ERLANG으로 구현한 웹서버 YAWS에서 Jsp/Servlets를 서비스 할 수 있도록 YAWS4J(YAWS for Jsp)을 설계하고 구현하였다. 이를 통해 ERLANG을 이해하고 프로그래밍 언어로서 발전가능성을 알아보고자 한다. 구성은 다음과 같다. 2장에서는 관련 연구로서 Jsp/Servlets을 서비스하기위해 필요한 사항들을 살펴본다. 3장에서는 Erlang의 특징을, 4장에서는 YAWS의 특징을 살펴보고, 5장에서 YAWS4J의 전체적인 구조와 구현내용을 알아본다. 그리고 마지막으로 결론을 맺는다.

2. 관련 연구

2.1 Jsp/Servlets 서비스

웹이 일반화되면서 정적 콘텐츠의 한계와 콘텐츠의 효율적인 개발을 위해서 새로운 스크립트언어(PHP, ASP, JSP 등)들이 개발되었다. 웹서버들은 다양한 스크립트

언어들을 보다 빠르고 효율적으로 서비스할 수 있도록 기능 개선이 이루어지고 있다. 다양한 스크립트언어 중 에서 Jsp/Servlets을 서비스할 수 있는 Tomcat은 범용 웹서버(Apache, IIS 등)들과 비교하였을 때 효율성과 안정성면에서 성능이 떨어진다. 이러한 단점을 해결하기 위하여 콘텐츠 제공자들은 백엔드에서는 Tomcat을 통해 Jsp/Servlets, 프론트엔드에서는 범용 웹서버를 통해 Jsp/Servlets 이외의 문서요청을 서비스하는 구조를 사용한다. 이러한 구조를 위해 Jakarta 프로젝트[3]는 웹서버와 Tomcat간의 통신을 담당할 수 있는 JTC(Jakarta Tomcat Connector)를 개발하였다. JTC는 AJP(Apache Jserv Protocol)와 WARP(Web Application Remote access/control Protocol)로 나누어진다.



[그림 1] JTC 분류 및 구현 모델들

AJP13은 JTC중에서 안정적이고 보편적인 프로토콜이며 [그림 1]처럼 mod_jk라고 하는 동적 라이브러리를 제공하는데, 이 동적 라이브러리를 Apache에서는 mod_jk, IIS에서는 isapi라고 부른다.

3. ERLANG

ERLANG은 함수형 언어의 특징요소를 제거함으로써 통신, 네트워크 어플리케이션 분야에 적합한 언어가 되었다. ERLANG의 특징은 다음과 같다.

Declarative syntax. Erlang은 선언적 구문구조를 가지고 있으며 side-effects가 없다.

Concurrent, lightweight 프로세스와 비동기식 메시지 전달구조를 가진다.

Real-time. Erlang은 milliseconds 수준의 soft real-time 시스템에 적합하다.

Continuous operation. 실행중인 시스템을 멈추지 않고 코드대체와 시스템 업그레이드가 가능하다.

Robust. 시스템에 에러가 발생했을 때 자동적으로 중지하고 재시작할 수 있는 라이브러리를 제공함으로써 안정적이다.

Memory management. Erlang은 자체적으로 메모리 관리를 수행함으로써 메모리 관련 에러가 없다.

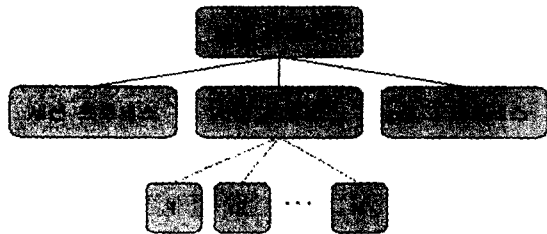
Distribution. 공유메모리를 지원하지 않고 비동기적 메시지 교환을 통하여 프로세스간에 상호작용을 한다

Integration. 다른 언어로 작성한 어플리케이션과 통신할 수 있는 다양한 인터페이스를 제공한다.

ERLANG은 이러한 특징적인 요소들 때문에 통신, 네트워크 어플리케이션을 작성하는데 적절한 언어로서 자리매김하고있다. 현재 ERLANG을 사용하여 개발한 대표적인 산업 어플리케이션으로 ATM Switch AXD301, 블루투스 위치정보시스템 BLISPlatform등이 있다.

4. YAWS

YAWS[8]는 ERLANG lightweight 프로세스를 사용하여 대용량의 동시적 커넥션을 제어할 수 있도록 구현된 웹서버이다. YAWS는 정적 콘텐츠인 HTML 문서에 대한 처리뿐만 아니라 ERLANG 스크립트로 작성된 동적 콘텐츠인 YAWS 문서를 처리하는 기능을 갖는다. YAWS 어플리케이션은 ERLANG/OTP의 디자인패턴에 따라 설계되었으며, YAWS의 전체적 구조는 [그림 3]과 같다.



[그림 2] YAWS 프로세스 구조

YAWS는 크게 시스템을 제어, 세션, 서버, 그리고 로그 프로세스들로 구성된다. 세션 프로세스는 특정 사용자와 세션유지를 위해 쿠키 값을 생성, 저장하는 역

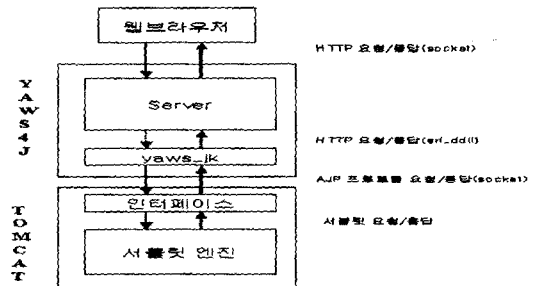
할을 한다. 서버 프로세스는 클라이언트의 요청 처리와 관련된 작업을 한다. 서버 프로세스는 클라이언트의 요청을 받을 때마다 요청 처리를 위한 자식 프로세스를 동적으로 생성한다. 동적으로 생성된 프로세스들은 서비스를 처리 한 후 소멸된다. 로그 프로세스는 클라이언트의 요청 및 응답결과에 대한 정보와 서버 프로세스 실행 시 발생하는 에러 사항들을 작성하고 저장한다. 마지막으로 제어 프로세스는 각각의 프로세스들을 시작, 중단, 재시작하고 상태를 모니터링(Monitoring) 한다.

5. YAWS4J

Apache는 mod_jk를 이용하여 Tomcat과 연동함으로써 Jsp/Servlets를 서비스하고, YAWS4J는 Apache와 마찬가지로 yaws_jk를 이용하여 Tomcat과 연동함으로써 Jsp/Servlets를 서비스한다.

5.1 YAWS4J 구조

[그림 3]은 YAWS4J에서 Jsp/Servlets를 서비스하는 과정을 나타낸다.



[그림 3] YAWS4J 구조 및 Jsp/Servlets 처리과정

YAWS4J는 웹브라우저로부터 Jsp/Servlets 요청을 받으면 yaws_jk를 통하여 Tomcat의 인터페이스로 서비스를 요청한다. Tomcat은 서블릿 엔진을 통하여 얻은 결과를 HTML형태로 만들어서 yaws_jk로 전달한다. YAWS4J는 이 결과를 웹브라우저로 전송한다. 이러한 구조는 범용 웹서버 Apache, IIS와 같은 방식이다.

5.2 YAWS4J

[그림3]에서 본 바와 같이 yaws_jk는 서버와 Tomcat 사이에서 연동할 수 있도록 하는 YAWS4J의 한 요소이다. yaws_jk를 개발하는데 참고한 mod_jk는 웹서버 인터페이스와 AJP 커넥터 라이브러리로 이루어진다. 마찬가지로 yaws_jk는 YAWS용 인터페이스와 AJP 커넥터 라이브러리로 이루어졌고 C로 구현되었다.

YAWS4J는 ERLANG으로 구현된 YAWS와 C로 구현된 yaws_jk사이의 통신을 위한 방법이 필요하다.

[표 1]은 ERLANG에서 지원하는 다른 언어와의 인터페이스의 종류와 모델들을 나타낸다. YAWS4J는 여러

모델 중에서 erl_ddll을 사용하였다. erl_ddll은 포트 인터페이스를 통하여 바이너리형태로 ERLANG 어플리케이션과 C 어플리케이션 사이의 통신을 담당한다.

[표 1] ERLANG과 다른 언어간의 통신모델

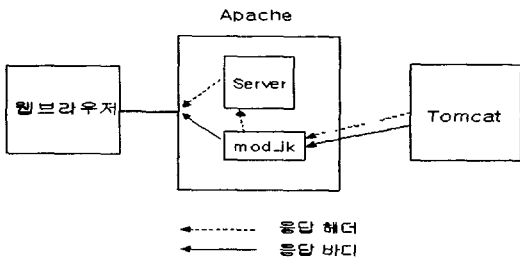
모델	방법	내용
erl_ddll	ERLANG->C	포트, 바이너리
erl_interface	C->ERLANG	포트,인코드/디코드
Jinterface	JAVA->ERLANG	포트,인코드/디코드
IC	ERLANG<->OTHERS	stub, IDL, CORBA
socket	ERLANG<->OTHERS	소켓, HTTP, IIOP

본 연구에서 개발한 yaws_jk는 [그림 3]과 같이 JTC에서 제공하는 AJP 커넥터 라이브러리를 이용하여 Tomcat과 연동하고 YAWS용 인터페이스를 이용하여 서버와 연동한다.

[그림 4]와 [그림 5]는 Apache와 YAWS4J에서 Jsp/Servlets을 처리하는 과정을 나타낸다. 진한 실선은 웹브라우저와 서버간의 연결을 나타내고 점선은 응답헤더를, 가는 실선은 응답바디의 전달과정을 나타낸다.

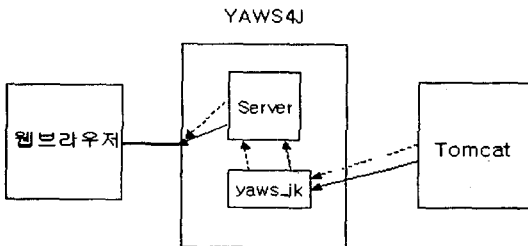
웹브라우저와 서버 사이에는 HTTP 프로토콜, 커넥터와 Tomcat사이에는 AJP 프로토콜에 따라서 통신한다.

하지만 [그림 4]와 [그림 5]는 많은 부분이 유사하지만 Tomcat으로부터 전달 받은 결과를 웹브라우저로 전송하는 과정에서 차이점이 있다.



[그림 4] Apache에서 Jsp/Servlets 서비스과정

[그림 4]에서 mod_jk는 Tomcat으로부터 응답헤더와 응답바디를 각각 전달 받는다. 응답헤더는 서버로 전달되어 Apache 웹서버정보와 추가되어 웹브라우저로 전송되고 응답바디는 바로 웹브라우저로 전송된다.



[그림 5] YAWS4J에서 Jsp/Servlets 서비스과정

[그림 5]에서 yaws_jk는 Tomcat으로부터 전달 받은 응답헤더와 응답바디를 서버로 전달한다. 서버는 부가정보와 추가된 응답헤더와 응답바디를 각각 웹브라우저로 전달한다. 이러한 차이점은 mod_jk와 달리 yaws_jk는 웹브라우저와 YAWS4J사이의 소켓 디스크립터를 사용할 수 없기 때문이다.

6. 결론

ERLANG은 함수형 언어의 복잡성과 특정기능을 제거함으로써 네트워크 분야에 적절한 높은 생산성과 효율성을 제공할 수 있다. 본 논문에서는 ERLANG으로 구현된 웹서버 YAWS의 기능을 개선하였다. YAWS는 lightweight 프로세스를 사용하여 대용량의 동시적 커넥션을 처리할 수 있고 동적 콘텐츠를 서비스하기 위하여 여러 웹스크립트언어를 지원한다. 그러나 Jsp/Servlets, ASP는 지원하지 않는다. 본 논문에서는 YAWS와 Tomcat의 연동을 통하여 Jsp/Servlets을 서비스할 수 있도록 YAWS4J를 설계하고 구현하였다.

참고문헌

[1] Joe Armstrong, " Concurrent Programming in ERLANG" published Prentice Hall
 [2] J.H. Nyström " Evaluating Distributed Functional Languages for Telecommunications Software" Heriot-Watt University, ACM SIGPLAN Erlang Workshop ' 03
 [3] <http://jakarta.apache.org>
 [4] Vivek Chopra와 1인, " Professional Apache Tomcat Chapter 13" Wrox Press 2003
 [5] Ulf Ekstrom, " Design Patterns for Simulations in Erlang/OTP" Uppsala University
 [6] http://Erlang.org/doc/r9c/doc/system_principles/part_frame.html
 [7] http://Erlang.org/doc/r9c/doc/design_principles/part_frame.html
 [8] <http://www.yaws.hyber.org>