제33회 한국정보처리학회 춘계학술발표대회 논문집 제17권 제1호 (2010. 4)

DBRS:B2J를 이용한 BPEL 엔진 기반 로봇 시스템 설계 및 구현

이종화*, 곽동규*, 최재영*
*숭실대학교 컴퓨터학과
{hayashi18, coolman}@ss.ssu.ac.kr, choi@ssu.ac.kr

DBRS:Design and Implementation of BPEL Engine based Robot System using B2J

Jonghwa Lee*, Donggyu Kwak*, Jaeyoung Choi*
Dept. Computer, Soong-Sil University*

요 약

URC 로봇 클라이언트는 원가 절감 및 하드웨어 경량화를 위해 최소한의 요소로 구성되며, 필요한 기능을 외부 디바이스와의 연동 또는 서버를 통해 전송받아 작업을 수행하게 된다. 기존의 URC 로봇 시스템은 이기종간의 제약 없는 서비스를 제공하기 위해서 웹 서비스를 사용하고, 워크플로우 표준 언어인 BPEL 을 기반으로 한 워크플로우 엔진을 사용하고 있다. 로봇 클라이언트의 제한 된 자원으로 인해 대부분의 URC 로봇 시스템은 서버에서 BPEL 엔진을 동작시키고, 네트워크를 통해서 클라이언트의 서비스를 호출하는 구조를 가지게 된다. 이에따라 기존 로봇 시스템은 클라이언트 서비스 호출로 인한 네트워크 부하가 발생하는 문제가 있다. 본 논문에서는 경량화 된 B2J 시스템을 이용하여 로봇 클라이언트에서 BPEL의 실행을 수행하는 DBRS를 제안한다. 기존의 B2J는 BPEL 문서를 Java 코드로 변환, 컴파일, 실행하는 과정을 같은 환경에서 수행했으나 DBRS에서 사용되는 경량화 된 B2J 는 변환자와 실행자를 분리함으로써 요구사항을 낮추고 로봇의 제한된 자원을 절약한다. DBRS 는 다른 BPEL 엔진을 사용하는 로봇 시스템에 비해 클라이언트의 제한된 자원을 절약하고, 로봇 서비스 호출에 의해 발생하는 네트워크 부하가 감소한다.

1. 서론

URC (Ubiquitous Robotic Companion) 은 지능형 서비스 로봇으로, “언제 어디서나, 나와 함께 하며, 나에게 필요한 서비스를 제공하는 로봇”을 의미하며, 네트워크 환경을 통해 로봇기술을 활용하는 것을 목표로 한다 [1]. 정해진 일을 반복 수행하는 기본의 산업용 로봇과 달리 지능형 서비스 로봇은 유비쿼터스 환경에서 인간과 상호 작용하고 주어진 상황에 맞게 적절한 서비스를 수행하는 로봇이다. URC 로봇은 한정된 자원으로 구성되며, 로봇의 경량화를 통해서 일반 가정에서도 로봇을 사용할 수 있도록 한다. 이를 위해서 네트워크를 통해 로봇에 적재되지 않은 기능이나 서비스를 제공할 수 있어야 한다. 즉, 부족한 기능이나 서비스는 외부 디바이스와 연동 및 외부 서버와의 협력을 통해 제공해야 한다.

이 기종간의 상호 연동의 비용과 복잡성을 줄이기 위해 나온 기술 중에 웹 서비스가 있다[2]. 외부와의 상호 연동을 위해서 웹 서비스는 여러 연구에서 적용 되고 있다. BPEL4WS(Business Process Execution Language for Web Services) 은 웹 서비스를 사용해서 웹 서비스 환경에서 비즈니스 프로세스를 정의하고 실행하는 워크플로우 표준 언어이다 [3]. 현재 BPEL을 실행할 수 있는 다양한 BPEL 엔진이 개발 연구되었다 [4][5]. BPEL은 정의된 서비스 전이 조건에 따라 워크플로우를 실행한다. 이는 표준이 가지는 장점을 수용하고 XML과 웹 서비스를 통해 기종에 독립적인 장점을 그대로 살릴 수 있다.

워크플로우 엔진은 비즈니스 프로세스 분야에서 프로세스의 자동화 수행을 통해 업무 처리 효율성을 극대화 시켜주는 응용 시스템으로 비즈니스 서비스의 실질적인 수행을 관리해주는 역할을 한다. 기존의 BPEL 엔진은 로봇 클라이언트에서 동작하기에 요구사항이 높고 모든 작업을 로봇 클라이언트가 수

행해야 하는 단점이 있다 [7]. 이러한 문제 때문에 기존 BPEL 엔진은 URC 로봇 클라이언트에 적용하기에 적합하지 않다.

본 논문에서는 변환자와 실행자를 분리한 B2J (BPEL to Java) [6]를 응용한 URC 시스템 DBRS (Distributed B2J based Robot System) 를 제안한다. BPEL 엔진 중에서 BPEL을 자바소스로 변환하여 실행하는 B2J가 있다. B2J는 BPEL을 입력받아 분산 처리 환경의 원격지 머신에서 이를 분석하고 파싱하여 자바소스 파일로 변환한다. 변환된 자바코드를 컴파일하여 자바 클래스로 변환시킨 후 실행한다. 이러한 특징은 외부와의 협력을 통해 서비스를 제공하는 URC 로봇 클라이언트에 유용하게 사용할 수 있다.

하지만 B2J는 소스코드 생성, 컴파일과 자바 실행파일로 변환을 실행환경에서 함께 수행하는 구조를 가지고 있기 때문에 B2J 의 변환환경과 실행환경을 분리해서 로봇 클라이언트에 적합하게 수정 할 필요성이 있다. 작성된 자바 실행파일은 로봇 클라이언트에서 실행되며, 자바 프로그램이 로봇 클라이언트의 로컬 서비스를 호출하기 위해서 서비스들은 웹서비스의 인터페이스를 사용하여 작성되어야 한다.

DBRS 는 B2J를 통해서 외부와 협력하여 외부 자원을 활용할 수 있다. DBRS 는 다른 BPEL 엔진을 사용한 로봇 시스템에 비해 로봇 클라이언트의 제한된 자원을 절약할 수 있으며, 로봇 서비스 호출에 의해 발생하는 네트워크 부하를 감소한다.

2. 관련 연구

현재 BPEL을 사용한 워크플로우 기반의 로봇 시스템이 연구개발 되고 있다. 그 예로 BPEL을 통한 모바일 로봇제어 시스템과 경량화 된 BPEL 엔진을 사용한 관련연구가 있다.

2.1 BPEL을 통한 모바일 로봇 제어

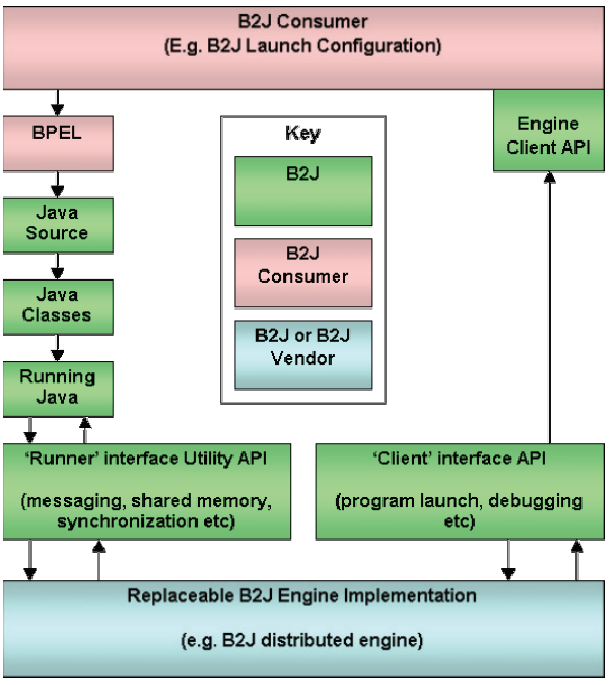
Tavetanov 연구자는 BPEL을 사용하여 로봇의 서비스 프로세스를 제어하는 방법을 연구하였다 [8]. Tavetanov 가 제안하는 시스템은 사용자가 GUI 환경에서 자신이 요구하는 로봇 응용에 따라 BPEL 편집기를 이용하여 BPEL 문서를 작성하고, 작성한 문서를 통해서 모바일 로봇을 제어할 수 있다.

워싱턴 대학에서는 모바일 디바이스를 위한 BPEL 워크플로우 프로세스 실행 엔진 Silver를 연구하였다 [7]. 기존의 BPEL 엔진은 대규모 서버에서 동작하는 아파치 톰캣에서 동작하게끔 만들어져 있다. 대표적인 오픈소스 BPEL 엔진인 ActiveBPEL은 JRE 1.4.2 환경과 Apache Tomcat 서버, 92MB 이상의 디스크공간과 22MB 이상의 메모리 공간을 요구한다 [4]. 유비쿼터스 환경의 로봇 클라이언트에서 해당 BPEL 엔진은 적합하지 않다.

그에 비해 워싱턴 대학에서 개발한 BPEL 엔진 Silver는 디스크 사용공간이 112KB 밖에 되지 않는다. Silver는 요구사항의 최소화를 통해서 유비쿼터스 환경에서의 로봇시스템을 지원할 수 있다. 하지만 Serializable Scopes 와 Event Handlers를 비롯한 BPEL 의 몇몇 기능을 지원하지 않고 BPEL 엔진에서 수행하는 모든 작업을 클라이언트에서 실행하므로 서버와의 분산처리를 통해 경량화를 할 여지가 있다.

2.2 B2J

B2J 는 IBM에서 개발한 STP(SOA Tools Platform) 프로젝트에서 BPEL을 실행하기 위한 서브 프로젝트로 개발 되었다.



<그림 1> B2J 시스템 구조

그림 1은 B2J 의 시스템 구조를 나타내고 있다. B2J 는 크게 두 개의 파트로 나뉜다. 첫 번째 파트는 B2J 엔진 API를 통해서 BPEL 문서에서 자바 소스코드를 생성한다. 생성되는 자바 소스 파일은 러

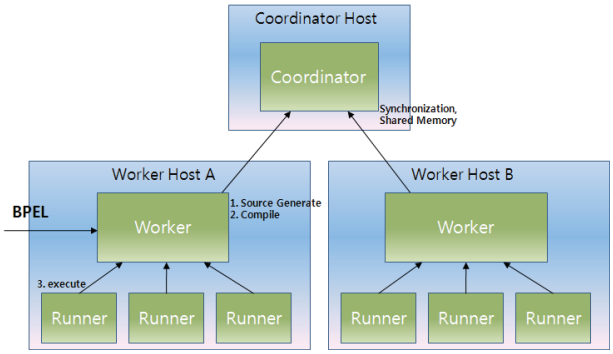
너 인터페이스를 상속한다. B2J는 러너 인터페이스를 통해서 메모리 공유와 동기화를 제공한다. 그리고 생성된 자바 소스코드를 자바 클래스 파일로 컴파일 한다. 컴파일은 사용자가 지정한 컴파일러를 통해 수행된다. 두 번째 파트는 컴파일 된 자바 클래스 파일을 러너를 통해서 실행시킨다. B2J는 자바 바 프로그램들의 메모리 공유와 동기화를 제공한다.

3. 설계 및 구현

기존의 B2J 는 BPEL에서 자바 코드로의 변환/컴파일 하는 작업과 실행하는 작업을 같은 환경에서 수행한다. 이 장에서는 기존의 B2J의 변환자와 실행자를 분리하여 경량화 한다. 그리고 경량화 된 B2J를 적용한 DBRS 의 시스템 구조에 대해서 설명한다. 그리고 실행자가 로컬 로봇 서비스를 사용하기 위해서 사용하는 로봇 서비스 인터페이스를 보인다.

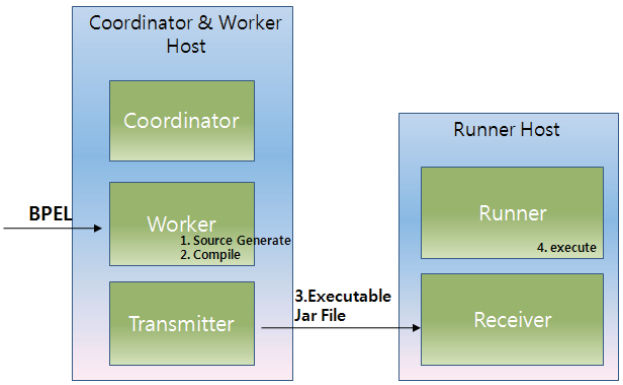
3.1 B2J 엔진의 수정

그림 2 에서는 기존 B2J 엔진의 분산처리 구조를 나타내고 있다. B2J 는 코디네이터, 워커 그리고 러너로 이루어져있다. 코디네이터는 B2J 시스템에서 스타 토폴리지의 형태를 가지며 분산처리 트렌젝션을 위한 동기화를 관리하고 모든 워커 호스트에 속한 러너들의 공유 메모리와 동기화를 관리한다. 워커는 BPEL 문서를 러너 인터페이스를 상속받은 자바 소스코드로 변환하고 컴파일한다. 러너는 워커와 물리적으로 같은 기계에서 동작하며 워커에서 제공하는 실행 가능 한 자바 클래스 파일을 실행한다.



<그림 2> 기존 B2J 엔진의 분산처리

하지만 기존의 B2J 엔진을 로봇 클라이언트에 적용시키면 자바코드 생성과 컴파일 작업으로 인해 발생하는 부하를 모두 로봇 클라이언트가 부담한다.

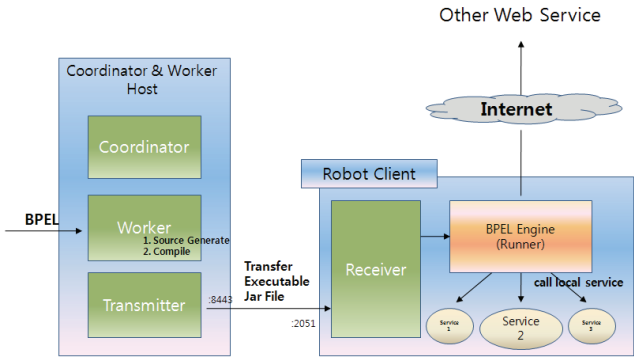


<그림 3> 수정한 B2J 엔진의 분산처리

그림 3은 이 문제를 해결하기 위해서 수정한 B2J 분산처리 과정을 나타낸다. 변경된 B2J 는 워커와 러너의 실행 머신을 분리함으로써 로봇 클라이언트의 자원 소모 없이 BPEL 파일을 Java 파일로 변경/컴파일 할 수 있다. 워커는 작성 된 클래스 파일을 실행 환경으로 전송하기 위해서 Transmitter에게 클래스 파일을 넘겨준다. Transmitter는 완성 된 자바 클래스 파일을 자바 실행파일로 묶어서 러너 호스트에게 전송한다. 러너 호스트는 Receiver를 통해서 워커 호스트가 전송하는 자바 실행파일을 전송 받아 실행한다.

3.2 DBRS 시스템 설계 및 구현

DBRS 시스템은 물리적으로 묶여있던 워커와 러너를 분리한 B2J를 적용한 구조를 가진다. 기존의 B2J 엔진은 워커와 러너가 물리적으로 같은 기계에서 동작하게끔 설계 되 있었다. DBRS 시스템은 B2J를 로봇 클라이언트에 적합한 경량화 된 구조로 만들기 위해서 워커와 러너를 분리하였다(3.1 절 B2J 엔진의 수정).



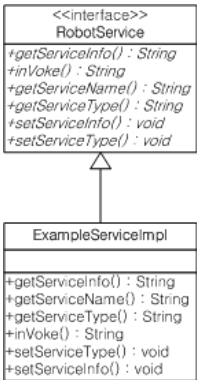
<그림 4> DBRS 시스템 구조

그림 4는 경량화 된 B2J를 적용한 DBRS 시스템의

구조를 나타내고 있다. 코디네이터와 워커는 러너 인터페이스를 상속한 자바 소스코드로 변환하여 컴파일 한다. Transmitter는 컴파일 된 자바 소스를 자바 실행파일 상태로 전송 한다. 로봇 클라이언트의 Receiver는 코디네이터/워커 호스트의 Transmitter를 통해 자바 실행파일을 다운로드 받는다. 다운로드 된 자바 실행파일은 러너에 의해서 실행된다. 자바 실행파일에는 BPEL 수행을 위한 프로그램과 수행에 필요한 의존적 라이브러리가 포함된다.

3.3 로봇 서비스의 인터페이스

BPEL 엔진이 로봇의 로컬 서비스를 웹 서비스처럼 사용하기 위해서, 로봇 클라이언트에서 지원하는 서비스의 형태를 웹 서비스와 유사한 형태로 작성했다.



<그림 5> 로컬 서비스를 위한 인터페이스

표 1 은 DBRS의 로봇 클라이언트에서 제공하는 서비스에서 사용되는 인터페이스 이다. 로봇 클라이언트에서 제공되는 로봇 서비스들은 이 인터페이스를 구현하는 형태로 작성된다.

4. 결론

기존의 URC 로봇 시스템은 로봇 서비스의 흐름을 제어하기 위해서 BPEL을 사용했다[7][8]. 그러나 URC의 서버-클라이언트 특성상 다수의 서비스를 호출하기 위한 네트워크 부하가 발생한다. 본 논문에서는 로봇의 워크플로우 프로세스의 경량화를 위한 B2J 기반의 DBRS 를 제안하였다. DBRS 에서는 B2J 엔진을 변경/적용함으로써, 사용자가 직접 로봇 클라이언트에게 명령을 내리는 구조를 통해서 로봇 서비스 호출 시 발생하는 네트워크 부하를 최소화 했다. B2J는 BPEL 문서를 자바소스 로 변환하여 실행한다. B2J 는 코디네이터와 워커, 러너의 구조로 이루어져 있고 코디네이터와 워커는 각각 물리적으로 다른 기계에서 동작한다.

B2J 엔진은 하나의 코디네이터 호스트로 통해 다수의 워커 호스트를 제어하고 워커 호스트에서 동작하는 러너들의 동기화와 메모리 공유를 지원한다. 하지만 워커의 기능을 클라이언트에서 수행할 경우 클라이언트의 제한된 자원을 소모한다. 본 논문에서는 B2J 엔진을 로봇 클라이언트에 적합한 구조로 만들기 위해서 워커와 러너를 물리적으로 다른 머신에서 동작 할 수 있게 수정했다. 자바 소스 변환과 컴파일에 의한 부하를 외부 워커 호스트로 분산시켰고, 로봇 클라이언트의 러너가 컴파일 된 클래스를 실행함으로써 로봇의 서비스 호출에 따른 네트워크 부하 없이 로컬에서 서비스를 직접 호출 할 수 있다. 그러기 위해서 로봇 서비스들은 웹 서비스에 인터페이스에 맞춰서 작성 되야 한다.

DBRS 시스템은 URC 환경에서 서버가 클라이언트의 서비스를 호출하는데 발생하는 네트워크 부하를 최소화하고, 기존 B2J 엔진에서 BPEL을 실행가능한 자바 클래스 파일로 변환하는 과정을 분리함으로써 B2J 의 경량화를 꾀했다. 향후 다른 웹 서비스들과의 상호 연동을 통해서 유비쿼터스 환경을 제공하기 위한 시스템으로 유용하게 이용할 수 있을 것으로 예상 한다.

Acknowledgment.
본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음" (NIPA-2009-(C1090-0902-0007))

참고문헌

[1] Young-Guk Ha, "Towards a Ubiquitous Robotic Companion: Design and Implementation of Ubiquitous Robotic Service Framework.", *ETRI Journal Volume 27 Number 6*, pp 666 - 676, December 2005.
[2] Web service, http://www.w3.org/TR/ws-arch/ .
[3] BPEL, http://www.eclipse.org/bpel/ .
[4] ActiveBPEL, http://java-source.net/open-source/workflow-engines/activebpel .
[5] YAWL, http://java-source.net/open-source/workflow-engines/yawl .
[6] B2J(BPEL to Java),http://www.eclipse.org/stp/b2j/ .
[7] Gregory Hackmann, Sliver: "A BPEL Workflow Process Execution Engine for Mobile Devices", 2006.
[8] Simeon Tsvetanov, "Using Some Motion Devices for Easily Workflows Illustration.", *International Scientific Conference Computer Science'2008*, pp. 681 - 684.