

OSGi 아키텍처에서 이기종간의 분산 컴퓨팅 지원을 위한 확장된 서비스 레지스트리

김동욱⁰ 최성술, 지충원, 김정선
한양대학교

{dukim⁰, sschoi, cwji, jskim}@cse.hanyang.ac.kr

Extended Service Registry for Heterogeneous Distributed Computing Support in OSGi Architecture

Donguk Kim⁰, Sungsul Choi, Chungwon Ji, Jungsun Kim
Hanyang Univ.

요약

홈 네트워크 관련 기술들 중 여러 종류의 홈 네트워크 미들웨어간 상호연동의 지원 및 서비스 배포 등의 문제를 해결하기 위한 환경을 제공하는 OSGi에 대한 관심이 높아지고 있다. 그러나, 현 OSGi 시스템은 이웃 OSGi 시스템간, 그리고 이기종 시스템간에 분산 컴퓨팅을 지원하지 않아 그 활용 범위가 제약된다. 따라서 본 논문에서는 외부 시스템이 가지는 서비스에 대한 투명한 접근이 가능하고, 로컬 시스템에 존재하는 다른 번들에 대한 접근에 동일한 접근 방법을 가지도록 하기 위하여, 웹 서비스를 이용하는 확장 레지스트리를 제안한다. 이를 통해 이웃 OSGi 시스템들 뿐만 아니라 이기종 시스템과도 연동을 할 수 있으며, 현재 OSGi 시스템이 가지는 기능성에 분산 컴퓨팅능력의 확장을 통해 OSGi 시스템의 활용 범위를 크게 늘릴 수 있다.

1. 서론

최근 유비쿼터스 환경에서 홈네트워킹에 관련한 연구가 활발해지고 있다. 사용자의 장비는 점점 작아지고 있으며, 주위의 자원들과의 통신도 필요하게 되었다. 이러한 장비들은 동적인 컴포넌트의 이용과 함께 seamless operation을 항상 제공할 필요가 있다. 따라서, Component-based Development (CBD) 접근과 잘 정의된 메커니즘이 필요하게 되었고, 좋은 예들 중 하나가 OSGi(Open Service Gateway Initiative)이다.

OSGi는 컴포넌트 모델과 서비스 지향 아키텍처(SOA)를 제공하며, 원거리 네트워크를 근거리 네트워크나 홈 네트워크와 연결해 준다. 또한 집이나 자동차 그리고 휴대 장치와 같은 OSGi 임베디드 장치에서 원격으로 관리되는 서비스 어플리케이션과 상호연동되는 서비스 어플리케이션을 위한 컴포넌트 실행 환경을 제공한다[1][2].

OSGi에 의한 서비스 지향 수행은 중앙 집중식 구조를 가지며, 로컬 머신에 등록된 서비스들만이 각각의 서비스들에게 공유하기 때문에 다른 컴퓨터에 존재하는 자원을 사용할 수 없다. 따라서 현재의 OSGi 아키텍처는 이웃한 시스템의 계산 능력을 동적으로 이용할 수 없다.

그러므로, 본 논문에서는 유비쿼터스 자원의 공유를 통해 분산 컴퓨팅을 지원할 수 있는 확장된 OSGi 서비스 레지스트리를 제안한다. 새로운 레지스트리는 홈 서

버나 자동차 서버가 서버 시스템과 상호작용하는 자원 제약적인 장비에서 사용될 수 있다.

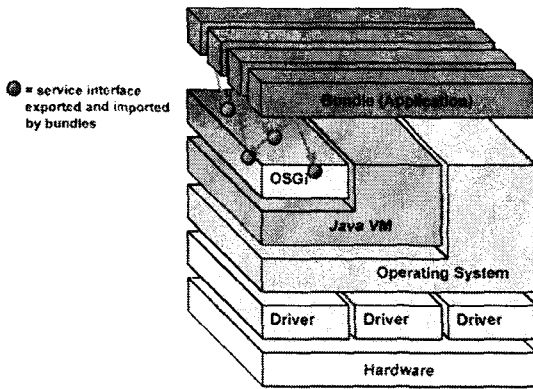
데이터베이스와 같이 많은 연산량을 요구하는 서비스나 영상 데이터나 멀티미디어 데이터의 처리 같은 작업은 자원 제약적인 장비에서는 적합하지 않다. 이러한 경우, 클라이언트 코드는 OSGi 번들로서 OSGi 임베디드 머신에 등록하고, 컴퓨팅 능력이 크게 요구되는 서버 코드는 서버 시스템과 관련된 곳에 위치시킬 수 있다.

번들을 사용하는 서비스의 관점에서는 등록된 서비스는 단지 일반적인 OSGi 번들로 보여지며 번들 개발자의 관점에서는 모든 것이 로컬 시스템에서 실행되기 때문에 분산 처리에 대한 추가적인 작업이 필요하지 않다. 또한 OSGi 시스템과의 통신뿐만 아니라 이기종과의 통신도 가능하게 되므로, 이전의 OSGi 아키텍처를 이기종 시스템과의 상호연동이 가능한 분산 컴퓨팅 환경으로 확장시킬 수 있다.

2. 관련 연구

2.1 OSGi(Open Service Gateway Initiative)

OSGi는 네트워크 환경의 다양한 부가 서비스들의 관리문제를 다루는 산업 컨소시엄이다. 표준 자바 기술을 사용하여, 서비스 생명 주기 관리를 위한 일반적인 프레임워크를 제공한다[3].



[그림 1] OSGi 프레임워크의 아키텍처

그림1은 OSGi 프레임워크의 아키텍처를 나타내며, 윗 부분에 나타난 번들은 배치와 관리의 기본 단위이다. 번들은 서비스들의 모음이며, 그 각각의 서비스들은 어플리케이션의 특정한 기능을 수행한다. 또한, 전체적인 플랫폼이나 서비스의 수행을 멈추지 않고도 각각의 번들을 제거하거나 멈추게 할 수 있으며, 특정 시간에 쉽게 활성화되거나 설치되는 번들의 동적인 환경을 정의한다. 그러므로 각각 상호작용 하는 서비스들에 대해 OSGi 프레임워크는 그들 각각을 발견할 수 있고 협력하는 번들들 사이의 서비스 의존도 관리를 통하여 서비스 레지스트리라 불리는 표준 서비스 발견 매커니즘을 제공한다. 서비스 레지스트리를 사용하여, 번들들은 다른 서비스들에 대한 의존이나 그들의 요구사항을 등록할 수 있다.

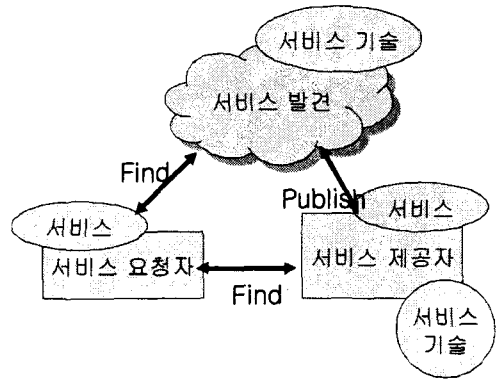
본 논문에서는 웹서비스를 이용하여 이 서비스 레지스트리를 확장시키고, 이를 통해 외부 시스템들이 지원하는 서비스들에 대한 접근을 가지는 확장 서비스 레지스트리를 제안한다.

2.2 웹 서비스

웹 서비스는 그림2와 같이 크게 4단계로 구성된 개발 단계를 가지고 있으며, 각 개발단계의 자원을 위한 관련 표준들을 이용하고 있다[4].

- 서비스 작성과 기술
- 서비스 등록
- 서비스 발견
- 서비스 호출

현재 웹 서비스에서 이용하고 있는 표준으로는 SOAP,



[그림 2] 웹 서비스 개념도

WSDL, UDDI가 있다[5][6][7].

SOAP(Simple Object Access Protocol)은 Microsoft에서 제안하여 현재 W3C에서 표준화를 진행하고 있는 XML 기반 프로토콜로서, 웹 서비스의 요청 및 응답에서 사용되는 메시지 형식을 정의하고 있다. WSDL(Web Services Description Language)은 웹 서비스 이용에 필요한 인터페이스와 입/출력 메시지의 형식을 기술하기 위해 이용되며, UDDI(Universal Description, Discovery and Integration)는 웹 서비스에 대한 디렉토리 서비스를 지원하기 위해 개발된 분산 레지스트리 표준으로 웹 서비스를 등록하고, 검색/바인딩하기 위한 메커니즘을 제공한다. 현재의 웹 서비스는 이 세가지 기술을 기반으로 구현되어 있으며, 이는 UDDI 레지스트리가 SOA의 기본 구조에서의 서비스 레지스트리로서 동작하며, 이 때 서비스 간에 교환되어지는 메시지는 SOAP을, 서비스의 바인딩과 호출을 위해 필요한 인터페이스의 명세는 WSDL을 이용하도록 되어 있다. 이들 표준들은 모두 XML로 작성되었으며, 또한 기존의 Web 환경에 그대로 이식할 수 있다는 점 때문에, 적은 이식 비용과 플랫폼에 독립적이라는 특징을 가지게 된다.

이러한 플랫폼 독립적인 특징을 이용하여 기존의 OSGi시스템이 이웃 OSGi 시스템들이나 이기종 시스템들과의 연동이 가능하게 된다.

2.3 홈네트워크 미들웨어와 OSGi 시스템간의 연동

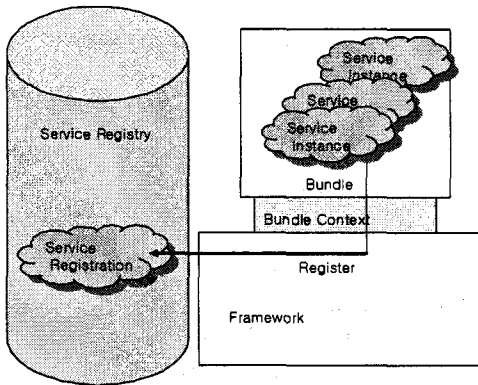
홈네트워크에서의 연구들은 [8][9][10]과 같이 주로 UPnP나 Jini, HAVi등과 같은 홈네트워크 미들웨어들간의 연동에 관한 것이나 [14]와 같은 Jini 와 OSGi 프레임워크의 연동, 혹은 [15]와 같은 UPnP와 OSGi 프레임워크간의 연동과 같이 홈네트워크 미들웨어와 OSGi 프레임워크와의 연동, 그리고 OSGi 프레임워크 상에서의 홈

네트워크 미들웨어들간의 연동에 관한 연구들이 주로 진행되어 왔다[11][12][13].

최근들어 연구의 다각화로 OSGi 프레임워크간의 연동에 관한 연구도 진행되고 있다.[16] 하지만 여전히 OSGi와 이기종 시스템간의 연동에 관한 연구는 부족한 상황이다.

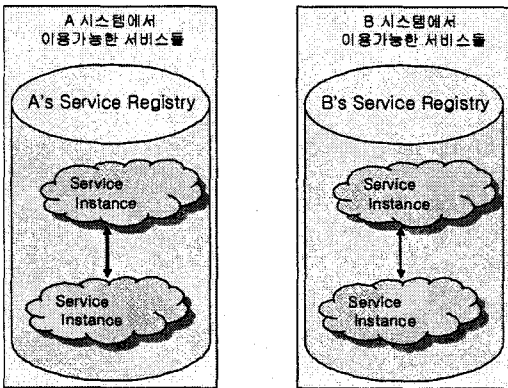
3. 웹 서비스를 이용한 확장 서비스 레지스트리의 아키텍처

3.1 기존 OSGi 시스템의 서비스 레지스트리의 아키텍처



[그림 3] 기존의 서비스 레지스트리에 기반한 서비스 오퍼레이션

그림 3은 기존의 OSGi 시스템이 가지는 중앙 집중화된 형태의 서비스 레지스트리에 기반한 OSGi 시스템의 서비스 수행을 나타내고 있다[17]. 이 시스템은 오직 로컬 OSGi 시스템의 서비스 레지스트리에 등록된 서비스들만을 사용할 수 있으며, 이 시스템의 로컬에 존재하는 각 번들들은 서로 이용 및 참조가 가능하다.



[그림 4] 현재 OSGi의 서비스 레지스트리

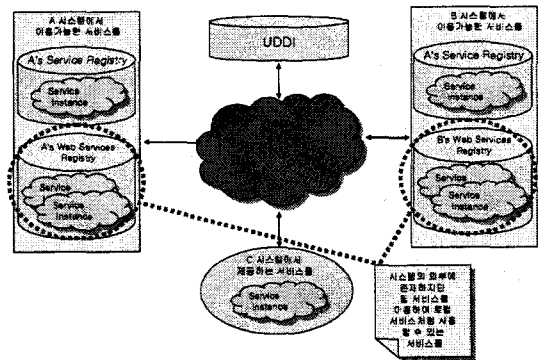
그림4는 각각의 서비스 레지스트리들이 이용 가능한 서비스들을 나타내는 아키텍처를 나타내고 있다[16]. 시스템 A와 B는 단지 자신들의 레지스트리에 가지는 서비스들만을 이용할 수 있다. 이러한 한계 때문에 이 시스템은 외부 네트워크에 존재하는 OSGi 시스템들이나 이기종의 시스템과는 연동할 수 없다.

3.2 확장 서비스 레지스트리

확장 서비스 레지스트리의 아키텍처는 크게 다음과 같은 특징을 가지게 된다.

- 확장 서비스 레지스트리의 서비스 액세스 방법이 기존의 레지스트리의 것과 동일하기 때문에 OSGi 시스템의 수정이 필요 없어 다른 번들들과도 자유롭게 상호 연동이 가능하다.
- 로컬 서비스를 이용하는 경우가 아닌 외부 시스템의 서비스를 이용하게 될 경우 번들 내부에서는 웹 서비스를 이용하기 때문에 SOAP 프로토콜을 이용하게 된다. SOAP은 HTTP 포트를 사용하기 때문에 외부 네트워크와의 통신에서 생길 수 있는 방화벽에 대한 문제를 고려하지 않아도 된다.
- 작업에 대한 시스템의 처리 능력이나 리소스를 고려한 작업의 분배가 가능하여 시스템들을 좀 더 효율적으로 이용, 관리할 수 있다.
- 시스템이 UDDI를 이용함으로써 현재 사용하고 있는 서비스들 외에 추가적으로 배포되는 서비스들에 대한 검색 및 사용이 용이하다.

OSGi 시스템들과 이기종 시스템들은 이 아키텍처를 통하여, 각자가 가지는 서비스들을 다른 시스템들에게 공유가 가능하며, 위와 같은 특징들을 가지는 확장 서비스 레지스트리의 아키텍처는 그림 5와 같이 구성된다.

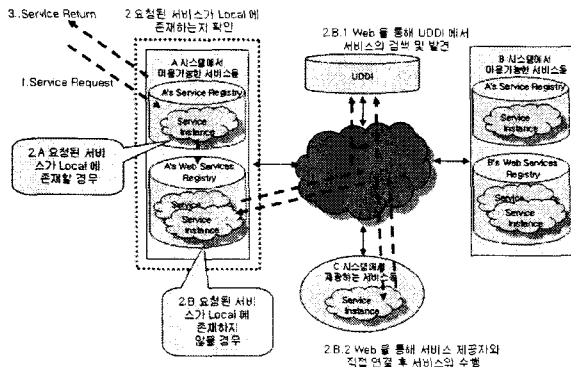


[그림 5] 웹 서비스를 이용한 확장 서비스 레지스트리

그림5는 OSGi 시스템 A, B와, 이기종 시스템 C로 구성된다. A 시스템은 로컬 서비스 레지스트리를 가지며, 이와 함께 확장 레지스트리에서는 UDDI를 통한 서비스 접근을 위해 얻어온 정보를 통해 만들어진 서비스 번들이 등록되어 있어 외부 OSGi 시스템 B와 이기종 시스템 C가 배포한 서비스들을 로컬 시스템에서 가지는 서비스 사용법과 다르지 않게 이용할 수 있으며, 또한 시스템 A가 가지는 서비스를 UDDI를 통해 배포하여 시스템 B 나 C가 이용할 수 있게 된다.

3.3. 수행 시나리오

그림 6은 확장 서비스 레지스트리의 시스템으로부터의 서비스 요청에 대한 서비스 발견 수행 과정을 묘사하고 있으며, 아래와 같은 순서로 수행한다.



[그림 6] 확장 서비스 레지스트리의 동작 시나리오

1. 시스템으로부터 서비스요청에 대한 메시지를 받는다.
2. 서비스 레지스트리는 자신이 가지는 서비스들인지 여부를 검사한다.

시스템은 2가지의 시나리오로 분기 되며, 첫 번째 시나리오는 다음과 같다.

- 2.A 요청된 서비스가 로컬에 존재 할 경우
- 3.A 해당 서비스를 시스템에게 리턴한다.

두 번째 시나리오는 다음과 같다.

- 2.B.1 요청된 서비스를 로컬 서비스 레지스트리에서 찾지 못하였기 때문에 확장 서비스 레지스트리에서 찾는다.
- 2.B.2 웹을 통해 UDDI에서 발견한 서비스를 가지는 서비스 제공자와 직접 연결 후 서비스를 수행한다.
- 3.B 서비스를 시스템에게 리턴한다.

3.4 구현 이슈

OSGi 아키텍처는 자바기반으로 동작하기 때문에 자바 웹서비스 기술을 이용한다. 자바 웹서비스를 이용하기 위해 SOAP 프로토콜이 이용하는 포맷인 XML 파일의 파싱을 위한 파서가 필요하다. 대표적인 파서로 JAXP(Java API for XML Processing)가 있으며, OSGi 프레임워크가 이것을 표준 서비스로서 포함하고 있으므로 이 서비스를 이용하여 SOAP 메시지의 파싱에 이용할 수 있다[18][19].

4. 결론 및 추후 과제

본 논문에서는 OSGi 시스템이 가지는 서비스 레지스트리의 분산 환경 컴퓨팅의 지원과 이기종 시스템과의 연동을 위한 확장 서비스 레지스트리 아키텍처를 제시하였다. 이 아이디어는 OSGi 시스템에서 웹 서비스 기술을 이용함으로써 가능하게 된다.

이를 통하여 이 시스템은 다음과 같은 장점을 가진다.

- 분산 컴퓨팅의 이점으로 작업의 효율적 분배와 시스템의 효율적 이용
- 웹 서비스를 이용한 이기종 시스템 및 이웃 OSGi 시스템과의 연동
- 기존의 시스템의 수정 없이 확장 레지스트리의 구성 및 이용
- SOAP 을 이용하기 때문에 연동하는 시스템들과의 통신에서 방화벽을 고려할 필요 없음
- 웹 서비스의 UDDI를 사용함으로써 서비스의 배포 및 사용의 용이

이를 통해 리소스가 많이 필요한 작업의 경우 제한된 리소스를 가지는 임베디드 시스템보다는 좀 더 풍부한 리소스를 가지는 시스템에게 맡겨 처리하도록 하여 리소스 제약에 대한 문제를 해결 할 수 있다.

추후 과제로서 만약의 경우 발생 될 수 있는 시스템의 연결 실패와 같은 상황에 대한 결함허용(Fault tolerance) 처리 매커니즘이 필요하다. 또한 웹 서비스를 이용함으로써 생기는 XML 문서 파싱의 오버헤드를 피하기 위하여, OSGi 시스템 상호간의 통신을 RMI 통신으로 대체하여 시스템의 성능을 향상 시키고자하는 대체 통신 작업 전환 등과 같은 자동화가 필요하다[20].

참고문헌

- [1] Roy w. Schulte Yefim V. Natis, "Service Oriented Architecture," Gartner Group, SSA Research Note SPA-401-068, 1996.
- [2] Open Service Gateway initiative (OSGi), OSGi alliance, <http://www.osgi.org>
- [3] ICROSS TECHNOLOGY Homepage, <http://www.icrosstech.com/~icross/default.html>
- [4] W3C Web Services WG, "Web Services Architecture," <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>, W3C Working Group Note 11 February 2004
- [5] 이경하, 이규철, "SOA와 Web Services," October 2-004
- [6] Nilo Mitra, "SOAP Version 1.2 Part 0:Primer," W-3C Recommendation, <http://www.w3.org/TR2003/REC-soap12-part0-20030624/>, June 2003.
- [7] Roberto chinnici, "Web Services Description Language (WSDL) Version 2.0 part 1: Core Language," <http://www.w3.org/TR/2004/WD-wsd120-200040803/>, W3C Working Draft 3 August 2004
- [8] "UDDI Technical White Paper", http://uddi.org/pubs/lru_UDDI_Technical_White_Paper.pdf, UDDI.org, September 2000.
- [9] 배유석, 손영성, 오봉진, 문경덕, "정보가전 제어 미들웨어 통합에 관한 연구", 한국정보처리학회, November 2003.
- [10] 한상숙, 은성배, 김철민, "UPnP-to-Jini 서비스의 설계 및 구현", 한국정보처리학회 논문지 A, February 2004
- [11] Allard, J., Chinta, V, Gundala, S, Richard III, G.G., "Jini meets UPnP: an architecture for Jini/UPnP interoperability", Applications and the Internet, 2003. Proceedings. 2003 Symposium on Publisher IEEE, 2003
- [12] The Universal Plug and Play Forum ,<http://www.upnp.org/>
- [13] Jini , <http://www.jini.org>
- [14] HAVi, <http://www.havi.org/>
- [15] 양동준, 김태형, "개방형 서비스 게이트웨이(OSGi) 환경에서 Jini 서비스의 통합", October, 2001.
- [16] Dong-Oh Kang, Kyuchang Kang, Sung-Gi Choi, Jeunwoo Lee, "UPnP AV architectural multimedia system with a home gateway powered by the OSGi platform", Consumer Electronics, 2005. ICCE. 2005 Digest of Technical Papers. International Conference on , 2005
- [17] Kyuchang Kang, Jeunwoo Lee, Hoon Choi, "Extended Service Registry for Distributed Computing Support in OSGi Architecture", Feb 20-22, 2006
- [18] 서대영, "OSGi Tutorial", May 23, 2005
- [19] Java API for XML Processing (JAXP), "<http://java.sun.com/webservices/jaxp/>"
- [20] Java, "<http://java.sun.com>"
- [21] Java Remote Method Invocation(RMI) <http://java.sun.com/j2se/1.3/docs/guide/rmi/index.html>