2002 년 한국정보처리학회 춘계학술발표논문집 제 9 권 제 1 호

Java 컴포넌트 소프트웨어 동적통합

이금용**
**영산대학교 정보통신공학부
e-mail : office@java-tech.com

Dynamic Integration of Java Component Softwares

Geum-Yong Lee**
**Dept. of Information and Communication, Young-San University

요 약

자바의 Jini 기술을 이용하여 네트웍에 분산된 Java 컴포넌트 소프트웨어를 동적으로 통합하기 위한 방법론을 제안한다. 통합대상 개별 컴포넌트는 Jini 인터페이스(interface)를 구현하도록 설계하거나 혹은 기존 컴포넌트를 포장한 것으로서 네트웍에 구성된 Jini Federation 에 Jini Service 오브젝트로서 공개된 것이다. 통합의 주체는 ServiceDiscoveryManager 와 같은 특별한 Jini 헬퍼클래스 기반으로 작성된 클라이언트 프로그램이다. 클라이언트는 지정된 조건 (Entry 오브젝트)을 충족하는 컴포넌트 소프트웨어를 네트웍으로부터 ServiceTemplate 를 이용하여 검색, 다운로드 한 후 단일한 GUI 를 가지는 프로그램에 통합한다. 필요한 컴포넌트만을 필요할 때 사용할 수 있고, 영구 설치의 필요성이 없으므로 소프트웨어 개발과 배포에 대한 새로운 관점을 제시할 것으로 판단된다.

1. 서론

컴퓨터 소프트웨어는 급변하는 신기술 개발 혹은 신수요 반영의 결과로서 매우 짧은 수명주기 및 갱신 주기를 가지고 있다는 특징이 있다. 그렇다면, 사용중인 소프트에어를 기능적, 경제적인 측면에서 항상 최적의 상태로 유지하기 위해서는 어떻게 해야 할까 ? 대부분의 오피스 제품에서 보는 바와 같이 번들 형태로 출시되는 대용량 소프트웨어에는 수명주기동안 한 번도 사용되지 않는 불필요한 컴포넌트가 포함되어 가격을 올리고 메모리 소비를 유도하고 있는 동안에도, 인터넷에는 매일 새롭고 유용한 Freeware 와 Shareware 가 발표되고 있는 것이다.

위와 같은 필요성, 즉, 자신이 필요한 소프트웨어만을 항상 최적의 상태로 유지하기 위한 방법의 하나로 본 논문에서는, 인터넷에 분산되어 있는 컴포넌트 소프트웨어의 검색 및 형식 인식, 다운로드, 단일 클라이언트 어플리케이션으로서의 동적 통합이라는 개념을 제시하고 Java 에 근거한 Jini 기술을 이용하여 구체화 하기 위한 방법론을 논한다.

인터넷을 통한 종래의 컴포넌트 소프트웨어(모듈) 통합은, 사전 지정된 URL 을 통해 한정된 형식을 가지는 모듈을 주기적으로 업데이트 하는 식으로 이루어졌다[1]. 본 연구에서는, 네트웍의 어떤 서버에서든 새로운 모듈이 발표되기만 하면 그 모듈의 제작언어와 형식에 구애됨이 없이 비동기적으로 검색, 자동으로 현재 사용중인 클라이언트 어플리케이션에 통합시키는 것을 목적으로 한다. 모듈별로 별도로 설치되는 것이 아니고, 단 하나의 GUI 를 가지는 어플리케이션에 컴포넌트로서 통합되는 것이다.

개발자가 해당 컴포넌트 소프트웨어를 네트웍에 배포하는 형식과 내용은 별도의 연구와 상업적 고려에 의해 결정되는 문제이지만, 전체 팩키지가 아니라 원하는 컴포넌트만을 신속하게 입수하여 활용할 수 있다는 측면에서 컴퓨터 소프트웨어에 관한 새로운 관점을 제공할 것으로 판단된다.

제 2 절에서는 Jini 기술에 대한 개요를 소개하고, 제 3 절에서는 본 논문에서 제시되는 컴포넌트 소프트웨어의 동적통합 모델을 설명한다.

2. Jini 기술 개요

오늘날의 정보통신 기술은 어떠한 정보장치라도 일정한 프로세서와 약간의 메모리, 그리고 네트웍 카드만 있다면 그 외부적인 모양과는 관계없이 네트웍의 활성 노드로 참여 시킬 수가 있다. 네트웍화 된 컴퓨터는 이제 동적 분산형태로 우리 곁에 상존하는 '환경'이 되어 있는 것이다. 이러한 상황을 인간의 개입

을 최소화 한 상태에서 구성하고 운영할 수 있는 완벽한 솔루션은 아직까지 존재하지 않는다고 볼 수 있다. 마이크로소프트사의 Millennium Edition, 썬 마이크로시스템즈의 Jini, 휴렛 팩커드사의 e-Speak 정도가 일정한 솔루션을 제공하고자 하는 비젼을 가지고 있다.

2.1 Jini 로 무엇을 할 수 있는가 ?

Jini 는 언제, 어디서든 그리고 무엇이든 기본적인 하드웨어 컴포넌트 (Java Virtual Machine 을 탑재할 만한 프로세서, 메모리, 통신카드)만 갖추었다면 네트웍에 접속하여 원하는 컴뮤니티(= Jini Federation)를 발견, 참여, 작업수행이 가능하게 할 목적으로 개발되었다. 뿐만 아니라, 컴뮤니티에 참여하는 모든 Jini 오브젝트 (이것을 Jini Service 라고 하며, Java Interface 에 의해 포장된 컴포넌트 소프트웨어)는 설치와 환경설정의 과정이 자동화되며, 하드웨어 장치라 할지라도 단일한 Jini Service 로서 추상화함으로써 컴뮤니티 구성과 운영시 동일한 프로토콜에 기반하여 접근할 수 있도록 해준다. 부분적 네트웍 파손에 자동 대처하며, 필요한 Jini Service 의 자기복구가 가능하다.

2.2 Jini 의 시스템 구조

Jini 시스템은 3 개의 큰 구조로 구성되어 있다고 말할 수 있다. 기반구조(Base Architecture), 프로그래밍 구조 (Programming Architecture), 그리고 서비스 구조 (Service Architecture)가 그것이다. 이 구조들은 단일 JVM(자바 가상기계)들의 네트웍 연합체 (Jini Federation)를 형성한다. 단일 JVM 이 제공하는 다양한 API 기술을 네트웍 전체로 확장하는 것이다.

(1) Jini Base Architecture

Jini 시스템에 의해서 구현되는 네트웍 솔루션을 형성하기 위한 기반을 제공하는 구조로서 Discovery/Join Protocol (Jini 서비스 오브젝트가 네트웍에 존재하는 Jini Federation 을 발견하고, 거기에 참여하며, 자기 자신의 존재를 Jini Federation 의 멤버에게 공지하기 위한 방법을 규정), RMI (Remote Method Invocation; 서비스 오브젝트간 분산 메써드 호출이 가능하도록 하는 기술), 분산 보안모델 (Java 보안모델을 분산환경에 확장), 그리고 Lookup Service (Jini Federation 에 존재하는 개별 Jini 서비스 오브젝트에 대한 레지스트리이며, 서비스 오브젝트간 상호 식별할 수 있는 시스템 서비스)로 구성된다.

(2) Jini Programming Architecture

네트웍에 접속된 각 노드간 자바 오브젝트의 이동을 지원하기 위한 구조이다. Lease interface (Jini Service 및 보조 오브젝트들에 대한 네트웍 자원관리를 지원; 모든 자원은 지정된 시간동안만 유효하며, 해당 자원의 소유주체가 유효기간 연장을 요청하지 않을 경우 사용되는 자원은 해제된다), Event interface (JavaBeans 의 위임모델을 네트웍 분산 오브젝트로 확장한 것이며, Jini Federation 의 각 멤버들에 대한 비동기 이벤트 통지를 가능하게 한다), 그리고 Transaction interface (Jini Federation 에 관계된 각 프로세스간 ACID 트랜잭션 서비스를 지원한다)로 구성되어 있다.

2.3 Jini Services Architecture

Jini Service 는 Jini Federation 을 구성하는 멤버로서, 네트웍에 접속할 수 있는 하드웨어, 소프트웨어 등을 총괄하여 표현한다. Jini Service 는 기본적으로 자바 오브젝트이며, 자바 이외의 언어로 작성된 경우 CORBA IDL 을 이용하여 Java 오브젝트로 전환할 수 있다. 각 Jini Service 가 가지는 자바 인터페이스를 통하여 해당 Service 를 식별, 다운로드, 실행할 수 있다.

앞서 기술한 Lookup Service 이외에, 모든 사용자 정의 Service, JavaSapces Service (가상 공간상에 정의되는 공유메모리 모델 및 리모트 이벤트 모델에 근거하여 Jini Federation 에 존재하는 다른 Jini Service 에 대하여 데이터 보존 서비스를 제공) 등이 여기에 속한다.

2.4 Jini Federation 의 실행 알고리즘

Lookup Service 는 네트웍에 Jini Federation 을 형성하고, 그 안에 참여하는 Jini Service 에 대한 레지스트리를 관리하고 Jini Service 간 통신을 중개한다. Jini Federation 의 실행 알고리즘을 간략히 설명함으로써 Jini 기술에 대한 소개를 마치고자 한다. 보다 자세한 내용은 참고문헌[2], [3], [4]을 참조하기 바란다.

① 네트웍에 퍼블리시 되는 Jini Service 는 Discovery Packet 을 통하여 하나 이상의 Lookup Service 를 발견하게 된다. Discovery Packet 은 Jini Service 자신에 대한 참조를 포함하고 있다.

② Jini Federation 에 존재하는 Lookup Service 는 지정된 포트에서 Discovery Packet 을 감지하고, 해당 Jini Service 에 대하여 적절히 응답한다.

③ 응답을 수신한 Jini Service 는 해당 Lookup Service 에 자기 자신의 속성, 형태 및 설명사항을 Proxy (역시 자바 오브젝트임) 형태로서 Upload 한다. 이후, Jini Federation 의 모든 멤버는 주어진 Jini Service 를 활용할 수 있다.

④ Jini Service 를 사용하고자 하는 클라이언트는 Discovery Protocol 를 이용하여 Jini Federation 에 참여(Join)하며, 이때 하나 이상의 Lookup Service 를 발견한다.

⑤ Lookup Service 를 발견한 클라이언트는 지정된 서비스 타입 (ServiceTemplate 오브젝트)에 의해 원하는 Jini Service 를 검색할 수 있다. ServiceTemplate 는 서비스 속성 (Entry 오브젝트)을 이용하여 상세하게 정의할 수 있다.

⑥ 검색, 선택한 Jini Service 의 프락시가 클라이언트의 컴퓨터로 Download 됨으로써, 클라이언트는 Jini Service 오브젝트의 어떠한 공개 메써드라도 호출할 수 있다. 필요시, codebase 속성값에 의해서 지정된 별도 URL 의 웹서버로부터 자바 클래스를 위한 메타데이터 등을 다운로드 한다.

3. Java 컴포넌트 소프트웨어의 동적 통합모델

Jini Service 오브젝트로 제작되거나, 비 Java 소프트웨어를 포장 (wrapping)하여 Jini Service 로 변환된 컴포넌트 소프트웨어를 Jini Federation 에서 자동 검색, 다운로드, 사용자 정의 클라이언트 어플리케이션으로의 동적 통합을 위한 시스템 모델을 제안한다. 논의의 구체성을 위하여 오피스 소프트웨어를 예로 한다.

3.1 컴포넌트 소프트웨어 만들기

본 연구에서 정의하는 Jini Service 는 네트웍 접속 장치간 이동성 (Mobility)를 보장하기 위해 Serializable 인터페이스를 구현하고, 클라이언트에게 추상화된 단일 Service 프로토콜을 지원하기 위하여 공개된 이름 OfficeComponent 을 사용하여 제작된다. 클라이언트 어플리케이션에서 GUI 로 장착되어야 하므로 JPanel 오브젝트로 하였다. 사례는 다음과 같다.

```
public class myJiniService extends JPanel
implements Serializable, OfficeComponent
{ ...//본문 }
```

클래스 본문에는 RMI, Lease, Transaction, Remote Event 를 위한 내부 클래스와 메써드를 정의한다. 외부 클래스에서 이들 작업에 대한 정의를 할 수도 있다.

3.2 컴포넌트 소프트웨어 서비스 속성 요건

Jini Service 는 자신의 속성을 정의하는 오브젝트와 함께 Jini Federation 에 공개된다 (Jini Service 의 속성은 net.jini.entry 팩키지의 AbstractEntry 클래스를 확장하여 정의됨). 클라이언트 어플리케이션에 구현된 GUI 와 효과적이고 표준적인 통합을 도모하기 위하여, 본 연구에서는 모든 컴포넌트 서비스 속성을 JavaBeans 와 GUI 를 겸하도록 요건 ("GUI Beans"라고 하자)화 하였다. 다음에 그 예를 든다.

```
public class ComponentCategory extends
AbstractEntry
implements SeriviceControlled { ...//본문 }

public class ComponentCategoryBean
extends JPanel
implements EntryBean, Serializable
{ ...//본문 ComponentCategory 의 멤버변수에 대
하여 get, set 메써드로서 접근 지원}
```

위와 같이 하면, 클라이언트에 다운로드된 컴포넌트 소프트웨어의 관련 정보를 GUI 로 쉽게 표현할 수 있다. ServiceControlled 는 서비스 속성을 클라이언트가 수정할 수 없게 하기 위함이며, EntryBean 은 JavaBeans 타입의 속성을 제작할 수 있게 한다.

본 연구에서 통합대상 컴포넌트 소프웨어의 속성으로서 지정한 사항은 다음과 같다.

- 컴포넌트의 캐티고리 (ComponentCategory) : 예시 WordProcessor, Presentator, SpreadSheet, ImageViewer, MediaPlayer, HTTPBrowser, FTPClient, MailClient
- 컴포넌트의 실행 클래스명 (ComponentClass)
- 컴포넌트의 도움말 파일 (ComponentHelp)
- 컴포넌트의 개발자, 버전, 발표연월일, 연락처 등 일반정보 (ComponentRights)
- 컴포넌트의 평가정보 (ComponentInfo): 다운로드 횟수, 사용자 평가 포인트, 유료 컴포넌트의 경우 가격 등 해당 컴포넌트를 클라이언트가 채택, 사용하는 것을 결정하기 위한 의사결정 정보

클라이언트 어플리케이션은 위와 같은 속성 정보를 근거로 하여, 자신의 어플리케이션에 통합하기를 원하는 컴포넌트 소프트웨어를 Jini Federation 에서 검색하는데 사용되는 ServiceTemplate 오브젝트를 만든다.

3.3 컴포넌트 소프트웨어 통합 플랫폼

다시 말하여, 클라이언트 어플리케이션 (여기서는 편의상 "**jOffice**"라고 칭한다)은 통합대상 컴포넌트 소프트웨어에 대한 속성 정의 (상기 3.2 절 참조, ServiceTempate 사용), Lookup Service 에 대한 검색 (Lookup), 서비스 클래스의 프락시 오브젝트 및 RMI Stub, 기타 보조 Serializable 오브젝트들을 codebase URL Web Server 로부터 다운로드 받기 및 설치하기, GUI Beans 형태의 Entry 오브젝트의 정보 판독 및 이를 jOffice 에 통합하기 위한 렌더링 리프레쉬(Refresh), 그리고 마우스 혹은 키보드에 의한 오피스 컴포넌트 호출 실행기능을 제공하는 Java 플랫폼이다.

그림 1. 에 jOffice 의 거동을 UML Statechart Diagram 으로 보였다. 중요한 Use Case 는 다음과 같다.

- Prompt for new construction : Jini Federation 을 검색하여 새로운 컴포넌트를 jOffice 에 통합할 것인가 여부를 Consumer(클라이언트 어플리케이션의 사용자)에게 묻는 윈도우를 띄운다.
- ShowServiceTemplateBox : ServiceTemplate 의 정보를 편집할 수 있는 윈도우를 띄운다.
- Perform service lookup : Lookup Service 에 대하여 ServiceTemplate 에 부합하는 컴포넌트 소프트웨어를 검색한다.
- LoadComponents : 이미 사용자의 컴퓨터에 저장된 컴포넌트를 jOffice 가 실행되는 JVM 에 로딩한다.
- Fix RemoteException : 네트웍에서 발생할 수 있는 여러 예외적 상황을 분석하고 적절한 대응 및 수정조치를 취한다.
- GetComponents : RMI Stubs, EntryBean 등 컴포넌트 클래스를 구성하고 실행하는 데 필요한 메타데이터를 별도로 지정된 codebase URL Web Server (HTTP)에서 다운로드 한다.
- InstallComponents : 컴포넌트를 설치하는 작업이다. 예를 들면, jOffice 는 서비스 속성중 Component Category 의 문자열을 주메뉴로 설정하고 있으며, 기타 속성정보는 해당 부메뉴로 표시하는데, 설치 과정중 이러한 내용을 반영하기 위한 프로세스가 실행되는 것이다.

- GetComponentInfo : 컴포넌트의 서비스 속성을 저장하는 ServiceAttributes 오브젝트를 얻어낸다.
- ParseComponentInfo : 오피스 컴포넌트의 상세정보를 판독한다.
- Update jOffice Environs : 판독된 오피스 컴포넌트 상세정보를 반영하여 jOffice 실행 환경설정 파일을 고쳐 쓴다.
- Draw jOffice : 구성된 어플리케이션의 GUI를 렌더링하는 작업이다.

그 밖에 플랫폼에 적용된 설계기준은 다음과 같다.

- 사용자에 의한 컴포넌트 신규 통합요청을 접수하거나, 플랫폼의 최초 실행시 제 2.4 절에 소개된 Jini Federation 실행모델에 따라 네트웍 접속을 진행한다.
- jOffice 는 JPanel, Panel, JFrame 을 기반으로 제작되어야 한다. JFrame 이외의 경우에는 EJB[5]등 웹 어플리케이션과의 통합을 위해 사용된다.
- 환경정보파일을 이용하여 GUI 및 기능을 제어한다. 다운로드된 컴포넌트의 서비스 속성을 분석한 후 필요한 제어정보는 환경정보파일에 저장한다.
- 컴포넌트의 캐티고리 정보 (ComponentCategory)는 종류별로 jOffice 의 메뉴에 정리되며, 해당 메뉴에 대하여 마우스 클릭 혹은 단축키를 누름으로써 컴포넌트 소프트웨어를 실행한다.

4. 결론

Jini 는 동일한 복수의 Jini Federation 에 대하여 UDP 기반 멀티캐스트 및 TCP 기반 유니캐스트 Lookup 을 지원하고 있으므로, 모든 네트웍에 등록된 컴포넌트 소프트웨어를 검색하고 통합할 수 있다.

본 연구에서 개발한 jOffice 는 기본적으로 public 도메인에 있는 Jini Federation 에 참여하도록 하고 있으나, 필요시 지정된 명칭의 Jini Federation 에 대하여도 가능하다.

C++, C, LISP 등 Java 이외의 컴포넌트에 소프트웨어에 대하여는 CORBA IDL 을 사용하여 자바와의 인터페이스를 만든 후 (IDL-TO-Java 컴파일러 사용), 이를 Serializable 를 구현하고 UnicastRemoteObject 를 확장하도록 함으로써 Jini Service 로 만들 수 있었다.

참고문헌

[1] Forte for Java, Ver. 3.0, Enterprise Edition, Sun Microsystems, 2001
[2] W. Keith Edwards, "Core Jini", 2nd Ed., Prentice Hall, 2001
[3] Sing Li, et al., "Professional Jini", Wrox Press, 2000
[4] Robert Flenner, "Jini and JavaSpaces Application Development", Sams Publishing, 2002
[5] Richard Monson-Haefel, "Enterprise Java Beans", 3rd Ed., O'Reilly & Associates, 2001

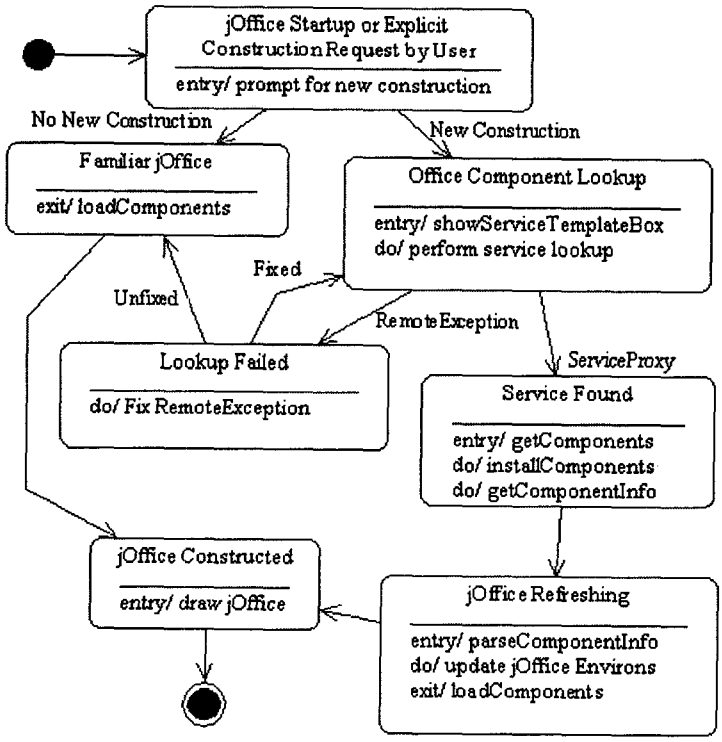

그림 1. Jini 기술을 이용한 컴포넌트 소프트웨어 동적 통합과정