UI4GSD: 글로버스 툴킷 4 기반 그리드 서비스 개발을 위한 사용자 인터페이스의 설계 및 구현☆

UI4GSD: Design and Implementation of User Interface for Grid Service Development Based on Globus Toolkit 4

김 혁 호* 김 양 우** 이 필 우***
HyukHo Kim Yangwoo Kim Pil-Woo, Lee

요 약

본 논문에서는 그리드 미들웨어인 글로버스 툴킷 4 상에서 그리드 서비스 개발자들이 효율적으로 서비스를 개발할 수 있는 사용자 인터페이스(UI4GSD, User Interface for Grid Service Development)를 제시하고자 한다. 일반적으로 그리드 서비스를 개발하기 위해서는 개발에 필요한 전문 지식이 요구되며, 서비스 개발 및 개발된 서비스를 글로버스 컨테이너에 배치하고 테스트하기 위해서는 많은 시간이 필요하기 때문에 서비스 개발의 효율성이 매우 떨어진다. 그러나 UI4GSD는 GUI를 통해 개발자로부터 서비스 개발에 필요한 정보를 입력 받아 처리함으로써 그리드 서비스 개발에 필요한 서비스 인터페이스 파일, 빌드 파일, 서비스 클래스들과 클라이언트 클래스를 자동으로 생성한다. UI4GSD에서는 정형화된 5단계의 개발 프로세스에 따라 그리드 서비스 개발이 이루어지고, 단계별로 입력된 데이터를 기반으로 쉽게 서비스를 개발할 수 있다. 결과적으로 UI4GSD는 그리드 서비스 개발을 위한 쉽고 편리한 작업 환경을 제공함으로써 서비스 개발의 편리성과 효율성을 증가시킬 수 있다.

Abstract

This paper presents UI4GSD (User Interface for Grid Service Development) for grid service developers that provides an efficient and friendly development environment based on Globus Toolkit 4. Normally, implementing grid service requires special expert knowledge for Grid as well as programming. Moreover, grid service development as well as testing of the deployed service in the Globus container takes a long time, which makes the grid service implementation very inefficient. However, UI4GSD can automatically generate a grid service interface file, a build file, and grid service class files as well as a client class file, using the information supplied by developers through GUI. In UI4GSD, a grid service is developed easily based on typical five step processes with required input data fed at each step. As a result, UI4GSD can provide an easy and convenient development environment thereby increasing the efficiency and convenience in developing grid services.

☞ keywords: Grid, Grid Service, Globus Toolkit 4

1. 서 론

그리드는 주로 하이퍼텍스트 형태의 정보만을 공유하는 웹과는 달리 지리적으로 분산된 고성능 컴퓨터, 대용량 데이터베이스 및 첨단 장비 등 다양한 컴퓨팅 자원을 초고속 네트워크로 연동시켜 준다. 또한 이를 통하여 고속 연산, 대량의 데이터 처리, 첨단 장비의 상호 공유 등을 가능하게 하고, 가상공간에서 협업 연구나 작업을 가능하게 해주는 새로운 개념의 정보통신 인프라 및 서비스를 통칭하는 기술[1, 2, 3]이다. 이러한 그리드

* 준 회 원 : 동국대학교 대학원 정보통신공학과 재학(박사)
hulegea@dongguk.edu
** 정 회 원 : 동국대학교 정보통신공학과 교수
ywkim@dongguk.edu(교신/책임저자)
*** 정 회 원 : 한국과학기술정보연구원(KISTI)
그리드컴퓨팅연구팀 팀장(책임연구원)
pwlee@kisti.re.kr
[2007/06/01 투고 - 2007/06/07 심사 - 2007/08/07 심사완료]
☆ 이 연구는 2007년도 정부(과학기술부)의 재원으로 한국과학재단의 지원을 받아 일부 수행된 연구임(No. F01-2007-000-10032-0).

환경을 구축하는데 사용되는 대표적인 미들웨어가 바로 글로버스 툴킷(Globus Toolkit)이다.

글로버스 툴킷[4, 5, 6]은 그리드 컴퓨팅 환경을 구축하는데 사용되는 대표적인 구축 도구로 그리드 컴퓨팅에서 사실상의 표준으로 인정받고 있다. 글로버스 툴킷은 그리드와 그리드 어플리케이션을 지원하는 서비스 및 소프트웨어 라이브러리로서 보안, 서비스 발견, 자원 관리, 데이터 관리, 통신 오류 감지, 이식성 등 그리드에서 필요로 하는 서비스들을 독립적으로 제공한다. 그러나 글로버스 툴킷을 이용하여 많은 그리드 어플리케이션과 시스템들이 개발되었고 그에 따라 글로버스 툴킷이 사실상의 그리드 표준 미들웨어가 되었지만 글로버스 툴킷은 몇 가지 단점을 갖고 있다. 첫째, 글로버스 툴킷 설치상의 문제점이다. 글로버스 툴킷은 설치 시 필요한 요구사항이 너무 많고 이에 따라 설치에 소요되는 시간이 길다는 단점이 있다. 둘째, 글로버스 툴킷을 이용한 시스템 구성과 환경 설정 그리고 보안 설정이 매우 까다롭다. 셋째, 개발자가 글로버스 툴킷을 기반으로 그리드 서비스를 구현하기 위해서는 웹 서비스, 그리드 서비스, 자바 프로그래밍 등등의 전문 지식이 요구된다. 넷째, 글로버스 툴킷에서는 컨테이너에 배치된 서비스에 대한 정보를 제공하지 않기 때문에 컨테이너에 배치된 서비스의 테스트가 어렵다. 글로버스 툴킷에는 이와 같은 문제점뿐만 아니라 운영체제간의 호환성 문제와 기타 작은 버그 등의 문제점이 존재한다.

본 논문에서는 위의 문제점들 가운데 그리드 서비스 개발 시 발생하는 문제점을 해결하기 위해서 그리드 서비스 개발을 위한 사용자 인터페이스(UI4GDS)를 제안하고자 한다. UI4GSD는 정형화된 5단계의 개발 프로세스를 거쳐 서비스를 개발하며, 개발된 서비스는 바로 글로버스 컨테이너에 배치된다. 이렇게 글로버스 컨테이너에 배치된 서비스는 클라이언트 프로그램을 통해 즉시 테스트 할 수 있다. UI4GSD는 GUI를 통해 입력받은 정보를 기반으로 서비스 클래스, 클라이언트 클래스, 그리고 빌드 파일 등등의 서비스 개발에 필요한 파일들을 자동으로 생성한다. 따라서 UI4GSD는 그리드 서비스 개발자들에게 작업의 편리성과 시간 단축의 효과를 제공함으로써 개발의 효율성을 증가시킬 수 있다.

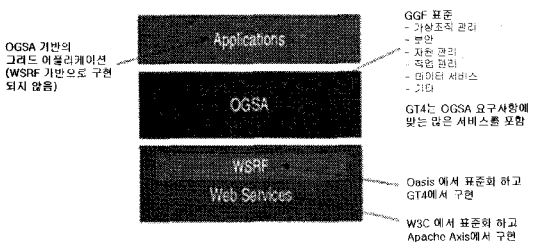
본 논문의 구성은 다음과 같다. 2장에서는 글로버스 툴킷의 발전 과정과 그리드 서비스 개발 방법에 대해 설명하고, UI4GSD와 기존의 다른 서비스 개발 환경과의 비교를 통해 장단점을 알아본다. 3장에서는 제안된 UI4GSD에 대해 기술하며, 4장은 본 논문에서 제안한 UI4GSD의 구현 결과와 이를 이용한 서비스 개발에 대해서 설명한다. 마지막으로 5장 결론 및 향후 연구로 구성되어 있다.

2. 관련 연구

2.1 글로버스 툴킷

글로버스 툴킷[4, 5, 6]은 대표적인 그리드 미들웨어로서, 여러 개의 소프트웨어 컴포넌트들로 구성되어 있다. 글로버스 툴킷은 크게 보안, 데이터 관리, 실행 관리, 정보 서비스, 런타임으로 분류할 수 있으며, 현재 글로버스 툴킷은 웹 서비스 표준을 기반으로 효율적인 자원 관리를 위해 확장된 표준안인 WSRF(Web Services Resource Framework)[8]를 구현한 글로버스 툴킷 4(GT4)가 발표된 상태이다. 그림 1에서 볼 수 있듯이, 글로버스 툴킷 4는 기존의 웹 서비스 인프라에 사용할 수 있게 보완된 OGSI(Open Grid Service Infrastructure)[9, 10]를 WS-Resource 개념[11]을 도입하여 WSRF 형태로 재구성(refactoring)하면서 웹 서비스 스펙으로 바뀌었다. 이를 통해 OGSA (Open Grid Service Architecture)[12, 13]는 웹 서비스 위에 통합되어 올려지게 되었고 실제적으로 그리드는 웹 서비스를 기반으로 구축되고 동작하게 되었다.

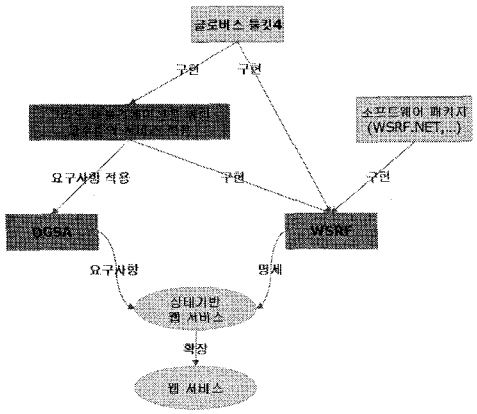
또한 현재 글로버스 툴킷은 글로버스 툴킷 3를



(그림 1) OGSA, GT4, WSRF 와 웹 서비스간의 관계[15]

거쳐 글로버스 툴킷 4로 발전하면서 Java WS Core, C WS Core, 그리고 Python WS Core, 기존의 globus_io를 대체해 보다 많은 개발 API를 제공하는 XIO, 신뢰할 수 있는 파일전송을 위한 RFT(Reliable File Transfer) API와 프로토콜을 상호운용 할 수 있는 새로운 버전의 GridFTP 서버 등이 기본 서비스 모듈로 추가되었다.

그리드 서비스[14] 개발자들은 글로버스 툴킷 4를 이용해 그리드 애플리케이션에 필요한 서비스들을 OGSA의 요구사항에 맞추어 개발하는데, 글로버스 툴킷 4에 구현된 WSRF 위에 올려지게 된다. 이 때 OGSA가 요구하는 웹 서비스들은 자원의 상태 정보를 유지해야 하며, 이것은 WSRF를 통해 정의된다. 그리고 이러한 상태 정보를 유지하고 있는 웹 서비스들은 보다 일반적인 웹 서비스들로 확장된다.



(그림 2) OGSA, 글로버스 툴킷 4, WSRF와 웹 서비스의 관계

그리드 서비스는 전형적으로 다음의 5 단계를 따라 개발할 수 있다[15, 16].

- 1 단계: 인터페이스 정의(WSDL Language)
- 2 단계: 서비스 구현(Java, C, Python Programming Language)
- 3 단계: 배치 설정(WSDD, JNDI)
- 4 단계: GAR파일 생성(Ant Tool)
- 5 단계: 서비스 배치(GT4 Tool)

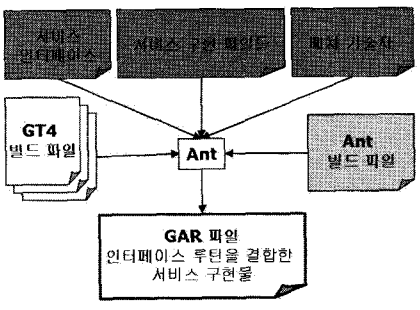
그리드 서비스 개발의 첫 단계는 서비스 상태 유지를 위해 WSRF를 이용하여 개발할 서비스의 인터페이스를 정의하는 것이다. 서비스 인터페이스는 개발할 서비스가 어떤 서비스인지를 사용자가 알 수 있도록 하는 것으로 WSDL을 통해 작성된다. 이때 서비스의 내부 동작(어떤 알고리즘을 사용하는지 또는 다른 시스템들과 어떻게 상호작용하는지)에 대해서는 관계할 필요가 없고, 개발자는 단지 어떤 기능이 사용자에 의해 활용 가능한지만 알면 된다. 웹 서비스 용어상에서 서비스 인터페이스는 통상적으로 포트 타입(Port Type)이라 불리며, 보통 portType이라 쓴다.

서비스 인터페이스를 정의한 후 두 번째 단계는 정의한 인터페이스를 그리드 서비스로 구현하는 단계이다. 실제로 서비스의 구현 부분은 QName 인터페이스 부분과 서비스 구현 부분으로 나누어 볼 수 있다. 그 중 하나인 QName 인터페이스 클래스는 개발하는 서비스 상 사용된 모든 네임스페이스를 정의하기 위해 편리한 관리 인터페이스를 제공한다. 서비스에서 사용되는 값과 자원 속성들은 서비스의 값이 변경될 때 자원으로부터 호출되며 이 클래스에 변경된 사항이 등록된다. 여기서 말하는 자원(Resource)은 서비스에 의해 생성된 객체를 말하며 서비스와 관련된 값들과 상태를 유지한다. 나머지 하나는 서비스와 자원에 관한 정보를 가진 자바 클래스이다. 서비스 상에서 작성된 메서드는 개발자로 하여금 자원에 접근하거나 값을 변경할 수 있도록 하는 인

터페이스 역할을 한다. 이는 서비스에 대한 게이트웨이 또는 대표 인터페이스로 간주된다.

다음 세 번째 단계는 실제로 작성된 서비스를 글로버스 컨테이너를 통해 이용 가능하게 만드는 것이다. 이 작업은 WSDD(Web Service Deployment Descriptor)[17]와 JNDI(Java Naming and Directory Interface)[18] 배치 파일을 통해 이루어진다. WSDD는 배치 기술자(Deployment Descriptor)로 불리는 파일로서 배치 작업의 중요 컴포넌트 중의 하나이다. 이 파일은 글로버스 컨테이너가 개발된 서비스를 사용자에게 어떻게 공개하는지를 알려준다. 예를 들면, 사용자는 개발된 서비스의 URI를 배치 기술자 파일을 통해 알 수 있다. 배치 기술자 파일은 WSDD 포맷에 맞게 작성된다. 그리고 JNDI는 XML로 작성된 설정 파일로써 WSDD 파일과 같은 위치에 저장되며 이 파일은 서비스의 자원 참조를 용이하게 하기 위한 특정 정보를 제공한다.

다음으로 네 번째 단계는 지금까지 작성된 파일들을 글로버스 컨테이너에 배치하기 위한 작업이다. 이 단계에서는 서비스 배치에 필요한 파일(WSDL로 작성된 서비스 인터페이스 파일, 서비스 구현 파일, WSDD와 JNDI로 작성된 배치 기술 파일)들을 묶어 'Grid Archive' 또는 'GAR' 파일을 생성한다. GAR 파일은 지금까지 작성한 모든 파일들과 구현한 서비스의 배치를 위한 웹 서비스 컨테이너의 정보를 담고 있는 파일로써 사용자들이 배치된 서비스를 활용할 수 있도록 해준다.



(그림 3) Ant를 이용한 GAR 파일 생성

그림 3에서 볼 수 있듯이, Ant는 지금까지 작성된 파일을 이용하여 GAR 파일을 생성하는데 이때 다음과 같은 복잡한 과정을 거쳐 GAR 파일이 생성된다[6].

- WSDL 파일 처리(예, binding)
- WSDL 파일로부터 스텁(Stub) 클래스 생성
- 스텁 클래스 컴파일
- 구현 서비스 클래스 컴파일
- 특정 디렉토리에 서비스 배치

위와 같은 여러 가지 작업을 통해서 GAR 파일을 생성할 수도 있지만 Ant[19]라는 Java Build Tool을 이용하면 간단히 이 작업을 수행할 수 있다.

마지막 단계는 생성된 GAR 파일을 글로버스 컨테이너에 배치시키는 작업이다. 글로버스 컨테이너에 서비스를 등록하고 해지하는 작업은 글로버스 툴킷에서 제공되는 명령어(globus-deploy-gar/globus-undeploy-gar)를 통해 쉽게 처리할 수 있다.

2.2 관련 연구와 비교

표 1과 같이 본 논문에서 제안한 UI4GSD의 기능과 특징을 다른 개발 툴들 중 가장 대표적인 이클립스 기반의 개발 환경과 비교 평가해 보았다. 그리드 서비스 개발 툴들은 크게 서버 측(즉, 서비스) 개발과 클라이언트 측 개발로 나누어 볼 수 있으며, 현재 글로버스 툴킷 4를 기반으로 한 그리드 서비스 개발 툴들은 대부분 이클립스 기반의 플러그인(Plug-in) 형식으로 되어 있다. 이러한 개발 툴들에는 Globus Alliance의 GT4IDE[21], 그리고 University of Marbug에서는 GDT[22]와 Java Annotation을 사용하여 그리드 서비스를 개발할 수 있는 방법을 제공하고 있으며, IBM에서는 이클립스 IDE[23] 방식의 개발 방법을 제공하고 있다. 그리고 클라이언트 측 개발은 Java CoG Kit[24]가 활용되고 있는 상황이다.

(표 1) 다른 개발 환경과의 비교

이클립스 기반의 개발환경(그리드 서비스 개발 플러그인)	UI4GSD 개발 환경
일반적 자바 개발 환경에 추가적으로 이클립스를 설치하고 그리드 서비스 용 플러그인 설치가 필요함	어떤 자바 개발 환경에서도 그리드 서비스 개발 가능
그리드 서비스 구현만 용이	그리드 서비스와 클라이언트 모두 구현 용이
그리드 서비스 개발 관련 파일(서비스 관련 파일을 개발자가 수동적으로 생성해야 함)	그리드 서비스 개발 관련 파일(서비스 관련 파일을 UI4GSD에서 자동으로 생성해 줌)
각 환경에 맞는 그리드 서비스 개발 방법 존재(추가적으로 각 개발환경에 맞는 개발 방법을 학습해야 함)	일반적인 구현 방법을 따름(전형적인 그리드 서비스 개발방법을 따르기 때문에 추가적인 개발 방법이 없음)
플러그인 설치 및 이클립스 환경 구성 시간이 오래 걸림	개발 환경 구성이 용이함
개발 환경(운영체제, 이클립스 버전, JDK 버전)에 민감함	개발 환경에 민감하지 않음
그리드 서비스 개발 및 테스트 시간이 오래 걸림	그리드 서비스 개발 및 테스트 시간의 단축

우선 Globus Alliance의 GT4IDE는 이클립스 플러그인 기반이지만 모든 이클립스 버전에서 개발 환경을 제공하는 것이 아니다. 즉, 이클립스의 버전이 바뀌면 새로운 이클립스 개발 툴을 다운받아 설치해야 하며 그리드 서비스 개발 플러그인 또한 변경해야 한다. 그리고 모든 이클립스 버전에 맞게 GT4IDE 플러그인을 제공하는 것 또한 아니다. 이는 사용자가 개발 환경을 구축하는데 시간이 많이 소비되며 매우 번거로운 작업을 초래할 수 있다.

다음 Marbug의 GDT는 이클립스에서 개발 환경을 제공하지만 GDT는 자신만의 개발 방법을 갖고 있다. 즉, GDT상에서 그리드 서비스를 개발하기 위해선 특정 개발 방법을 습득한 후 사용해야 한다. 또한 GDT는 자바 컴파일러 버전에 민감하다. 이는 모든 컴파일러 버전을 지원하는 것이 아니기 때문에 GDT와 호환되는 자바 컴파일러를 확인한 후 사용해야 하는 문제점이 존재한다.

그리고 Java Annotation 개발 방법 또한 자신만의 개발 방법을 갖고 있으며, 서비스 개발에 필요한 자바 프로그래밍 문법과 Java Annotation에서 제공되는 프로그래밍 문법까지 습득해야 하는 단점이 있다. 이와 같은 사항은 개발자로 하여금 서비스 개발에 필요한 지식뿐만 아니라 각각의 개발 툴에서만 제공하는 개발 방법과 설정에 필요한 요구사항까지 모두 숙지해야 하는 불필요한 상황을 발생시킬 수도 있다.

마지막으로 IBM에서 제공한 개발방법도 역시 이클립스를 기반으로 하고 있으나 그리드 서비스 개발 환경 구성을 위한 여러 가지 추가적인 플러그인을 설치해야 하는 번거로움이 존재하고 운영체제에 독립적이지 못한 문제점이 있다.

이와 더불어 이클립스 플러그인 기반 개발 툴들은 주로 이클립스 환경을 바탕으로 그리드 서비스를 개발하기 위한 개발 환경에만 치중되어 있으며, 글로버스 컨테이너에 배치된 서비스에 대한 접근과 테스트를 위한 클라이언트 프로그램의 작성에 대해서는 다소 미흡하다.

위와 같은 서버 측 개발 툴들에 비해 Java Cog Kit는 주로 클라이언트 측 기능과 컴포넌트들을 제공하지만 그리드 서비스 개발에 필요한 서버 측 고려사항을 다루지 않고 있다. 또한 Java Cog Kit는 자바 프로그래밍 방법뿐만 아니라 Cog Kit에서 제공하는 클래스와 컴포넌트들의 사용방법까지 알아야 하며, Java CoG Kit는 그 자체만으로도 꽤 복잡한 프레임워크를 갖고 있어서 그 기능을 활용할 수 있기 전까지 사용자는 상당한 시간

을 투자해야 한다는 단점이 있다.

이에 반해 UI4GSD는 개발환경 구축에 필요한 패키지들의 추가적인 설치가 필요 없으며, 그리드 서비스 개발에 필요한 전문지식의 습득 또한 요구하지 않는다. 그리고 UI4GSD는 임의의 추가적인 개발 설정이 필요 없이 단순히 글로버스 툴킷이 설치된 곳에 단 한 번의 설치로써 서비스 개발을 수행할 수 있다는 특징을 갖고 있다. 또한 서비스 개발 후 배치된 서비스를 테스트할 수 있는 클라이언트 프로그램을 쉽게 작성할 수 있으므로 매우 편리하며 효율적인 작업 환경을 제공하고 이에 따라 그리드 서비스 개발에 높은 생산성을 지원한다.

3. UI4GSD (User Interface for Grid Service Development)

본 논문에서는 글로버스 툴킷 4 상에서 그리드 서비스를 개발하는 개발자들에게 보다 쉽고 효율적으로 작업을 수행할 수 있는 개발 환경을 제공하고자 한다. 제안한 UI4GSD는 크게 세 가지 인터페이스로 나눌 수 있다. 첫 번째는 개발자로부터 서비스 구현에 필요한 정보를 입력 받는 인터페이스, 두 번째는 입력 받은 정보를 통해 서비스 구현에 관련된 파일을 생성하는 인터페이스, 그리고 마지막은 생성된 서비스의 배치에 관한 인터페이스이다.

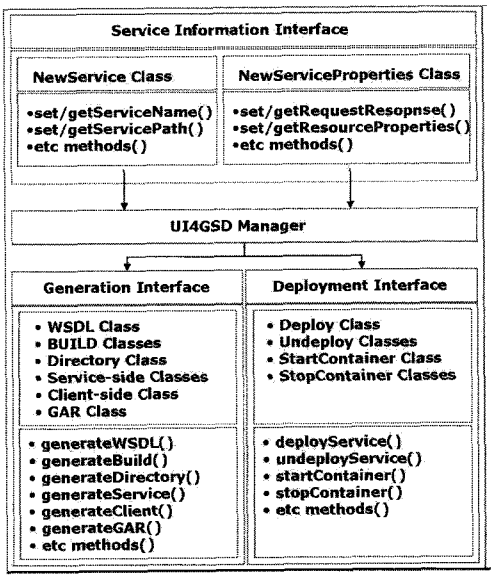
3.1 UI4GSD의 서비스 구성 요소

UI4GSD의 전체적인 서비스 구성은 그림 4와 같다. UI4GSD의 내부 구성은 세 가지 인터페이스로 나누어 볼 수 있다. 첫 번째인 서비스 정보 인터페이스는 주요 클래스로 NewService 클래스와 NewServiceProperties 클래스가 있으며, 다음과 같은 기능을 갖는다.

- set/getServiceName: 구현할 서비스의 이름 정의
- set/getServicePath: 구현할 서비스의 작업 위치 정의
- set/getResourceProperties: 구현할 서비스의 자원 속성 정의
- set/getRequestResponse: 구현할 서비스의 요청 응답 속성 정의

두 번째인 클래스 생성 인터페이스는 서비스 정보 인터페이스를 통해 입력 받은 정보를 처리하는 인터페이스로써, WSDL 클래스, BUILD 클래스, Directory 클래스, Service-side 클래스, Client-side 클래스, GAR 클래스 등이 존재하며 다음과 같은 주요 기능을 갖는다.

- generateWSDL: 구현할 서비스의 WSDL인터페이스 파일 생성
- generateBuild: 빌드 파일 생성
- generateDirectory: 각 서비스 파일들이 위치할 디렉토리 생성
- generateService: 구현할 서비스 파일 생성
- generateClient: 클라이언트 파일 생성
- generateGAR: 컨테이너에 배치할 GAR파일 생성



(그림 4) UI4GSD의 전체적인 서비스 구성도

세 번째 배치 인터페이스는 생성된 서비스의 배치/해제 그리고 글로버스 컨테이너의 실행/정지에 관한 작업을 수행한다. 주요 클래스로는 Deploy/Undeploy 클래스와 Start/StopContainer 클래스가 존재하고, 주요 기능은 다음과 같다.

- deployService: 생성된 GAR파일을 컨테이너에 배치 수행
- undeployService: 컨테이너에 배치된 서비스 파일의 해지 수행
- startContainer: 글로버스 컨테이너의 시작 기능 수행
- stopContainer: 글로버스 컨테이너의 정지 기능 수행

개발자는 위의 세 가지 인터페이스를 통해 서비스를 구현할 수 있고, 구현된 서비스를 컨테이너에 배치하고 난 후 글로버스 컨테이너를 구동시켜 배치된 서비스를 확인할 수 있다. 또한 배치된 서비스는 UI4GSD에서 클라이언트 프로그램을 작성해 바로 테스트할 수 있다.

3.2 사용자 인터페이스

사용자 인터페이스[20]는 개발자가 그리드 서비스를 보다 쉽고 편리하게 개발할 수 있도록 기능 정보의 그룹화와 기능의 동기화 그리고 개발자를 위한 기능 정보를 검색할 수 있도록 설계되어야 한다.

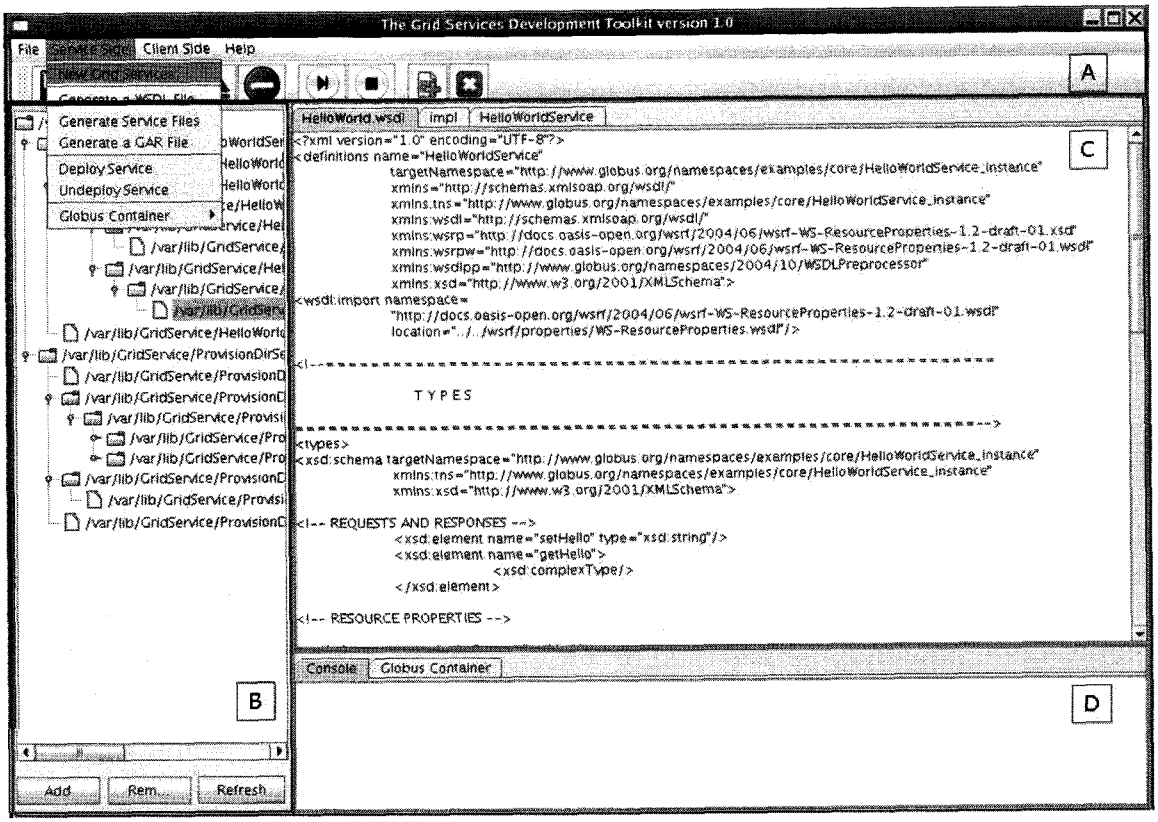
사용자 인터페이스의 전체 화면은 그림 5에서 볼 수 있다. 사용자 인터페이스의 화면 구성은 크게 네 부분으로 나누어 볼 수 있는데, 우선 메뉴와 툴바 인터페이스를 볼 수 있다. 메뉴의 주요 구성은 서비스 개발의 편의성을 위해 기능별로 메뉴를 그룹화하여 구성하였고, 사용된 메뉴의 이름 또한 개발자가 이해하기 쉽게 구성하였다. 그리고 툴바 역시 각 기능마다 툴팁을 설정해 놓아서 개발자가 쉽게 사용할 수 있도록 하였다.

"Service Side" 메뉴를 드래그하면 개발자는 그림 5의 A부분에서 볼 수 있듯이 하위 메뉴 리스트를 확인 할 수 있다. 메뉴 리스트의 순서는 서비스 개발 프로세스의 순서에 따라 구성되어 있고, 따라서 서비스 개발자는 메뉴 순서에 맞게 순차적으로 서비스를 개발하면 된다. 이는 서비스 개발자가 보다 쉽고 편리하게 서비스를 개발할 수 있도록 개발 프로세스 정보를 제공하는 효과도 기대할 수 있다. 서비스 개발 중에 발생하는 문제를 해결하기 위한 "Help" 메뉴는 개발자들에게 서비스 개발 프로세스의 설명과 힌트를 제공한다. 그래서 현재 작업 중인 부분과 남은 작업이 무엇인지 확인할 수 있도록 하였다.

그림 5의 B 영역은 서비스 네비게이터(Navigator) 부분이다. 이 네비게이터 부분은 생성된 서비스 파일들과 디렉토리 구조를 트리 형식으로 보여준다. 따라서 작업 중에 지금까지 작성된 파일과 디렉토리가 무엇인지 쉽게 확인할 수 있다. 또한 서비스 파일의 내용 편집이 필요한 경우 해당 파일을 더블 클릭하면 바로 파일을 열어서 편집 작업을 수행할 수 있다.

그림 5의 C 영역은 작업 윈도우 부분이다. 여러 개의 파일을 한 번에 열어 작업을 처리할 수 있도록 멀티 탭 기능을 지원한다. 각 탭의 이름은 작업하려는 파일의 이름으로 되어 있어 현재 열려있는 파일이 어떤 파일인지 쉽게 파악할 수 있다.

마지막으로 그림 5의 D영역은 두 개의 탭으로 구성되어 있다. 그 중 하나는 파일 및 디렉토리 생성 결과와 서비스 빌드 수행 결과 등등의 서비스 구현 중에 발생하는 작업의 결과를 보여주는 콘솔(Console) 윈도우이고, 나머지 하나는 글로버스 컨테이너를 위한 콘솔 윈도우이다. 글로버스 컨테이너의 시작 및 정지 결과를 확인할 수 있는 윈도우이다. 또한 구현한 서비스를 컨테이너에 배치했을 경우 그 결과를 컨테이너 상에서 확인할 수 있으며 클라이언트 프로그램을 통해서 배치된 서비스를 테스트할 수도 있다.



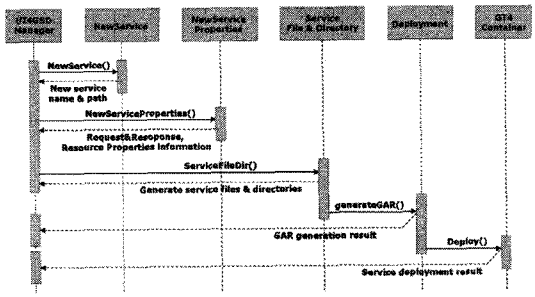
(그림 5) UI4GSD의 GUI 화면 구성

4. UI4GSD의 구현

4.1 서비스 개발 프로세스

이미 언급했듯이 그리드 서비스는 정형화된 5 단계의 개발 프로세스를 통해 개발된다. 첫 단계로 WSDL을 통해 서비스 인터페이스를 정의하고, 두 번째 단계로 정의된 인터페이스를 통해 구현하고자 하는 서비스를 구현한다. 세 번째 단계로 서비스 구현이 끝나면 서비스 배치를 위해 서비스 배치 스크립터 파일을 생성한다. 다음 네 번째 단계로 자바 빌드 툴인 Ant를 통해 구현한 서비스의 GAR(Grid Archive) 파일을 생성한다. 마지막 단계로 GAR파일이 생성되면 글로버스 툴킷에서 제공되는 배치 명령어를 통해 구현된 서비스를 글로버스 컨테이너에 배치한다.

그림 6은 UI4GSD를 통한 서비스 개발 프로세스를 보여준다.



(그림 6) 서비스 개발 프로세스

서비스 구현은 UI4GSD Manager에서 GUI 방식의 사용자 인터페이스를 통해 NewService 클래스

를 호출하면서 시작된다. NewService 클래스는 구현하고자 하는 서비스의 이름과 저장 위치를 개발자로부터 입력 받고 이 정보를 다시 UI4GSD Manager로 반환한다. 서비스 이름과 저장 위치가 정의되면, UI4GSD Manager는 NewServiceProperties 클래스를 호출하여 개발 프로세스의 1단계인 WSDL로 서비스 인터페이스를 정의하기 위한 속성 값들을 입력 받는다. 이 때 서비스 자원의 속성 값들이 입력된다.

서비스 자원의 속성 값 입력이 완료되면 UI4GSD Manager는 NewService 클래스와 NewServiceProperties 클래스에서 입력 받은 정보들을 다시 ServiceFileDir 클래스로 전송한다. ServiceFileDir 클래스는 전송 받은 정보를 이용하여 그리드 서비스를 빌드하기 위한 빌드 파일, 구현하고자 하는 서비스 파일 그리고 서비스 배치를 위한 배치 스크립터 파일 등을 생성한다. 또한 ServiceFileDir 클래스는 구현된 서비스를 테스트하기 위해 클라이언트 클래스와 각각의 파일들이 위치할 디렉토리를 생성한다. 이 단계가 그리드 서비스 개발 프로세스의 2번째와 3번째 단계이다.

다음으로 서비스 관련 클래스들과 디렉토리가 생성되면, UI4GSD Manager는 바로 Deployment 클래스를 호출하여 지금까지 생성된 서비스 파일들을 묶어 구현하려는 서비스의 GAR 파일을 생성한다. 이 과정은 Ant를 이용하여 처리되며 이 단계가 서비스 개발 프로세스의 4번째 단계이다.

마지막 단계로 GAR 파일이 생성되면 UI4GSD Manager는 Container 클래스를 호출하여, 생성된 GAR파일을 글로버스 컨테이너에 배치하고 해제할 수 있다. 이 과정은 글로버스 툴킷에서 제공되는 배치 명령어(globus-deploy-gar/globus-undeploy-gar)를 통해 처리되며 본 단계가 서비스 개발의 마지막 단계인 서비스 배치 및 해제 단계이다.

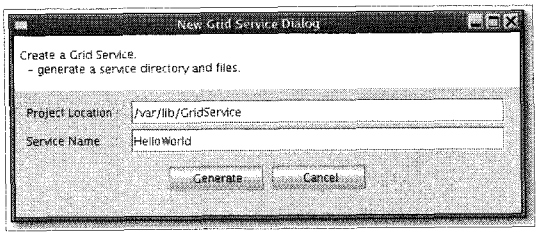
이렇게 5단계를 통해 서비스가 구현되고 글로버스 컨테이너에 배치되면, UI4GSD에서 바로 클라이언트 프로그램을 작성하여 배치된 서비스를 테스트할 수 있다.

4.2 서비스 개발

전형적인 그리드 서비스 개발 프로세스는 5 단계를 걸쳐 이루어지지만, UI4GSD는 단지 개발하려는 서비스에 대한 정보를 입력함으로써 모든 단계마다 개발자들이 수작업으로 처리할 필요 없이 서비스 개발을 가능하게 한다. 개발자는 단순히 개발 프로세스에 맞게 개발하려는 서비스에 대한 정확한 정보만을 입력하면 된다. 따라서 개발자는 서비스 개발의 모든 프로세스를 UI4GSD를 통해 쉽고 효율적으로 대치할 수 있다.

• 서비스 기본 정보 설정

UI4GSD를 실행한 후, 'Service Side' 메뉴를 클릭하면 가장 위에 'New Grid Service' 메뉴가 존재한다. 이 메뉴를 클릭하게 되면 그림 7과 같이 개발하려는 서비스의 이름과 저장위치를 입력 받을 수 있는 사용자 인터페이스 다이얼로그 윈도우가 뜬다.



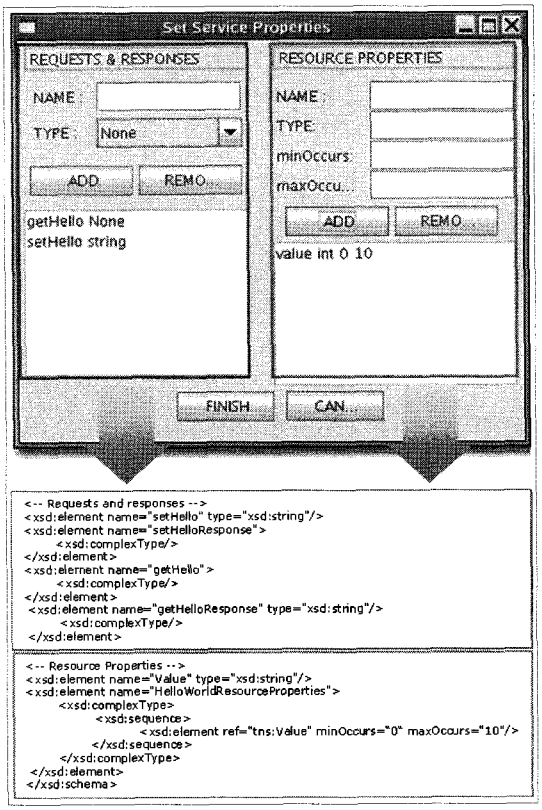
(그림 7) 새로운 서비스 생성 윈도우

그림 7의 인터페이스를 통해 서비스에 대한 기본 정보를 입력 받으며, UI4GSD Manager는 입력 받은 정보를 통해서 서비스 개발에 필요한 기본 설정을 하게 된다.

• 서비스 인터페이스 생성

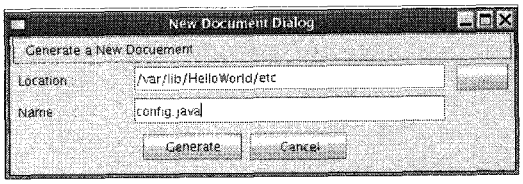
구현하려는 서비스의 기본 정보를 입력 받으면, 서비스 개발 프로세스의 첫 단계인 서비스 인터페이스를 통해 WSDL 파일을 생성해야 한다. 이 작업은 같은 메뉴 아래의 'Generate a WSDL'

메뉴를 통해 이루어진다. 이 메뉴를 클릭하게 되면 그림 8과 같은 다이얼로그 윈도우가 생성된다.



(그림 8) 서비스 속성 정보 입력 윈도우

그림 8에서 위의 사용자 인터페이스는 서비스의 요청과 응답 방법 그리고 자원 속성들을 정의한다. 그림 8에서 보는 것과 같이 인터페이스 화면 왼쪽은 서비스 요청 및 응답 방법을 정의하는 부분으로 이름과 유형을 입력 받는다. 유형의 종류는 Boolean, Byte, Short, Int, Long, float, Double, String, Array 등등이 있다. 그리고 인터페이스의 오른쪽은 자원 속성을 정의하는 부분이다. 사용자 인터페이스를 통해 'REQUEST & RESPONSE'의 이름과 타입 그리고 'RESOURCE PROPERTIES'의 입력 값들이 전달되면 그림 8의 아래 부분과 같이 WSDL 파일이 작성된다. 또한 추가로 파일 생성이 필요할 경우를 고려하여 'File' 메뉴에 'New Document' 메뉴를 구성해 놓았다. 'New Document' 메뉴를 클릭하면 그림 9와 같은 다이얼로그 윈도우가 생성된다.

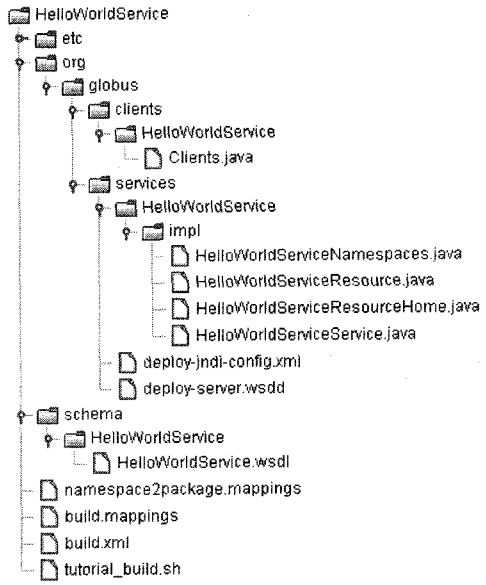


(그림 9) 새로운 문서 생성 윈도우

그림 9에서 볼 수 있듯이 사용자는 생성하려는 문서의 필요한 정보(파일 이름, 저장위치)만을 입력하면 새로운 문서는 자동으로 생성된다.

• 서비스 파일과 디렉토리 구성

서비스 구현에 중요한 WSDL 파일이 작성되면, 개발자는 다음으로 구현할 서비스 파일과 디렉토리를 구성해야 한다.



(그림 10) 생성된 서비스 파일과 디렉토리 구조

서비스 파일과 디렉토리 구성 작업은 'Service Side' 메뉴의 'Generate Service & Dir' 메뉴를 클

릭함으로써 수행된다. 이 단계는 그리드 서비스 개발 프로세스의 2단계와 3단계의 작업을 동시에 처리하게 된다. 개발자는 한 번의 클릭을 통해 서비스 개발에 필요한 모든 파일과 디렉토리를 생성하게 되는데, 그 결과는 그림 10을 통해 확인할 수 있다. 이 파일 구조를 살펴보면 가장 상위에 서비스 이름 'HelloWorldService'가 존재하고 그 아래로 글로버스 컨테이너에 배치된 서비스를 테스트하기 위한 클라이언트 파일, 구현할 서비스 파일들, 그리고 서비스 배치에 필요한 'deploy-jndi-config.xml' 파일과 'deploy-server.wsdd' 파일이 위치하며 그 아래로 서비스 인터페이스를 통해 생성된 WSDL 파일과 서비스 빌드를 위한 파일들이 위치하게 된다.

• 서비스 파일 생성 및 빌드

서비스 개발에 필요한 파일과 디렉토리가 생성될 때 이미 서비스 파일에는 서비스 구현에 필요한 기본 메서드들이 작성되어 있다. 그림 11은 WSDL 파일을 바탕으로 작성된 서비스 파일의 일부를 보여준다.

```
<xsd:element name="setHello" type="xsd:string"/>
<xsd:element name="setHelloResponse">
    <xsd:complexType/>
</xsd:element>
```

```
public SetHelloResponse setHello(String s) throws RemoteException
{
    // TODO Auto-generated method stub
}
```

(그림 11) 서비스 파일 생성

다음으로 개발자는 작성된 파일 내용을 바탕으로 필요한 내용을 추가 삽입하여 서비스 개발을 수행하면 된다. 이렇게 모든 서비스 내용 작성이 끝나면 지금까지 생성된 파일의 빌드 작업을 수행해야 한다. 이 작업은 앞에서도 언급했지만, 모든 처리 작업을 각각 개별로 나누어 처리하면 작업량과 시간이 많이 소요되므로 본 논문에서도 Ant를 이용하여 이 작업을 처리한다. 생성된 파일 구조를 보면 Ant를 위한 'build.xml'파일이 존재하는 것을 확인할 수 있다. 빌드 작업 수행과 GAR 파일 생성 작업은 'Generate GAR File' 메뉴를 통해 이루어진다.

• 서비스 배치 및 해제

서비스의 GAR 파일이 생성되면 마지막 작업으로 서비스를 글로버스 컨테이너에 배치하고 테스트 하면 된다. 서비스 배치 및 해제와 관련된 작업은 'Service Side' 메뉴의 'Deploy Service'와 'Undeploy Service' 메뉴를 통해 이루어지며, 글로버스 컨테이너 구동과 정지 또한 'Start GT4 Container' 메뉴와 'Stop GT4 Container' 메뉴를 통해 이루어진다. 이렇게 하여 그리드 서비스를 구현하고 글로버스 컨테이너에 배치하였다면, 마지막으로 배치된 서비스를 클라이언트 프로그램을 통해 테스트함으로써 정상적인 실행 동작을 확인할 수 있다.

5. 결론 및 향후 연구

본 논문에서는 그리드 미들웨어인 글로버스 툴킷 4상에서 개발자가 쉽고 효율적으로 그리드 서비스를 개발할 수 있는 사용자 인터페이스인 UI4GSD를 제시하였다. 일반적인 그리드 서비스의 개발 프로세스는 정형화된 5단계를 걸쳐 이루어진다. 개발자는 서비스 개발을 위해서 각 단계별로 필요한 전문 지식이 요구되며 단계마다 처리해야 할 작업이 많고 복잡하다. 하지만 UI4GSD는 각 단계마다 간단한 GUI를 통하여 개발자로부터 서비스 개발에 필요한 정보만을 입력받아 개발 작업을 처리할 수 있다. 간단한 예를 들어, 서비스 개발 툴을 사용하지 않고 명령어를 통해 작성된 서비스 파일들의 빌드 작업과 클라이언트 프로그램의 컴파일과 실행 처리를 수행하려면 다음과 같이 긴 명령어를 작성해야 한다.

• BUILD: ./globus-build-service.sh -d <SERVICE

WORKING DIRECTORY> -s <SERVICE INTERFACE DIRECTORY>/SERVICE INTERFACE.wsdl

- COMPILE STEP 1: source $GLOBUS_LOCATION/etc/globus-devel-env.sh
- COMPILE STEP 2: javac -classpath ./build/stubs/classes/:$CLASSPATH <CLIENT PROGRAM LOCATION>/CLIENT.java
- RUN: java -classpath ./build/stubs/classes/:$CLASSPATH org.globus.clients. SERVICE_INSTANCE. CLIENT http://127.0.0.1:8080/wsrf/services/examples/<SERVICE NAME>

하지만 UI4GSD에서는 위와 같은 작업을 단순히 메뉴에서 제공하는 기능을 통해 한 번의 클릭으로 모두 처리할 수 있다. 따라서 다음과 같은 작업을 수행하지 않아도 되기 때문에 개발에 익숙하지 않은 사용자들 또한 쉽게 서비스를 개발하고 테스트해볼 수 있는 장점을 갖는다. 그리고 각 단계마다 개별적으로 작업을 처리하지 않고 동시에 작업을 처리하기 때문에 개발 시간의 단축 효과도 얻을 수 있다. 또한 개발한 서비스를 UI4GSD에서 글로버스 컨테이너에 바로 배치하고 테스트할 수 있기 때문에 개발자들이 보다 쉽고 효율적으로 서비스 개발 작업을 수행할 수 있으며 이에 따라 높은 생산성을 기대할 수 있다.

이와 더불어 UI4GSD는 이클립스 기반의 툴들과는 달리 개발환경 구축에 필요한 패키지들의 추가적인 설치가 필요 없으며, 그리드 서비스 개발에 필요한 기본적인 지식 외의 개발 환경 구축을 위한 추가적인 전문지식의 습득 또한 요구하지 않는다. 그리고 UI4GSD는 임의의 추가적인 개발 설정이 필요 없이 단순히 글로버스 툴킷이 설치된 곳에 단 한 번의 설치로써 서비스 개발을 수행할 수 있다는 장점을 갖는다.

본 논문에서 제안한 UI4GSD는 위에서 언급한 여러 가지 장점들이 있지만 몇 가지 향후 연구과제들이 존재한다. 첫째, 본 논문에서 제안한 UI4GSD는 서비스 개발 시 자바 프로그램 언어만을 통해 작업이 이루어진다. 물론 자바 언어가 가장 보편적으로 많이 쓰이고는 있지만 현재 글로버스 툴킷은 C 웹 서비스 코어, Python 웹 서비스 코어와 같은 언어들도 지원하고 있기 때문에 추후 다양한 언어를 지원 가능하도록 구성할 필요가 있다. 둘째, 글로버스 툴킷은 RedHat 리눅스 환경을 기본으로 지원하여 왔지만 현재는 Apple Mac OS, Debian Linux, Fedora Core, FreeBSD, HP/UX, IBM AIX, Sun Solaris, SUSE Linux, Windows까지 다양한 플랫폼을 지원한다. 따라서 추후 여러 다양한 환경에서 서비스 개발을 지원할 수 있도록 확장하는 것을 고려해 볼 수 있다. 이와 같은 사항을 추가적으로 지원한다면 그리드 서비스 개발 작업을 수행하는 개발자들에게 보다 쉽고 편리한 작업 환경을 제공하여 서비스 개발의 효율성과 생산성을 증대시킬 것으로 기대되며, 더 나아가 그리드 기술과 산업의 확산에 기여할 수 있을 것으로 예상된다.

본 논문에서 제안하는 그리드 서비스 개발을 위한 사용자 인터페이스인 UI4GSD는 조만간 개선된 버전을 그리드 포럼 코리아 사이트에 올려 그리드 기술 개발에 관심 있는 산학연의 관계자들이 자유로이 사용해 볼 수 있도록 할 예정이다.

Acknowledgement

이 연구는 2007년도 정부(과학기술부)의 재원으로 한국과학재단의 지원을 받아 일부 수행된 연구임(No. F01-2007-000-10032-0).

참 고 문 헌

[1] I. Foster, "What is the Grid? A Three Point Checklist", GRIDToday, July 20, 2002.

[2] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of Grid Enabling Scalable Virtual Organizations", International J. Supercomputer Applications 15(3), November 5, 2003.

[3] I. Foster and C. Kesselman, The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kaufmnn, 2ndedition, November 18, 2003.

[4] The Globus Toolkit. http://www.globus.org/toolkit/.

[5] The Globus Alliance. http://www.globus.org/.

[6] Ian Foster, A Globus Toolkit Primer (Draft), ht tp://www.globus.org/ toolkit/docs/4.0/key/GT4_Pr imer_0.6.pdf, April 26, 2005.

[7] Fran Berman, Geoffrey Fox, and Anthony J.G. Hey, Grid Computing: Making The Global Infrastructure a Reality, John Wiley & Sons, April 8, 2003.

[8] Web Services Resource Framework, http://docs. oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-o s.pdf, 1 April, 2006.

[9] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, P. Vanderbilt, D. Snelling, Open Grid Services Infrastructure (OGSI) Version 1.0, Global Grid Forum Draft Recommendation, June 27, 2003.

[10] K. Czajkowski, D. Ferguson, I. Foster, J. Frey, S. Graham, T. Maguire, D. Snelling, S. Tuecke, "From Open Grid Services Infrastructure to WS-Resource Framework: Refactoring & Evolution", March 5, 2004.

[11] K. Czajkowski, D. F. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, W. Vambenepe, The WS-Resource Framework, March 5, 2004.

[12] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich, The Open Grid Services Architecture, Version 1.0, Global Grid Forum (GGF), January 29, 2005.

[13] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Service Architecture for Distributed Systems Integration", Open Grid Service Infrastructure WG. Global Grid Forum, June 22, 2002.

[14] Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke, "Grid Services for Distributed System Integration", IEEE, June 6, 2002.

[15] Borja Sotomayor and Lisa Childers, GLOBUS TOOLKIT 4 Programming JAVA Services, Morgan Kaufmann, November 1, 2005.

[16] Borja Sotomayor, The Globus Toolkit 4 Progra mmer's Tutorial, http://www.casa-sotomayor.net/ gt3-tut orial/index.html.

[17] Web Service Deployment Descriptor, http://ws.a pache.org/axis/java/reference.html.

[18] Java Naming and Directory Interface (JNDI), http://java.sun.com/products/jndi/.

[19] Apache Ant, http://ant.apache.org/.

[20] Scott W. Ambler, "User Interface Design: Tips and Techniques", Cambridge University Press, October 26, 2000.

[21] GT4IDE of Globus Alliance, http://gsbt.sourcefo rge.net/content/view/12/29/.

[22] The Grid Development Tools for Eclipse, http://ds.informatik.uni-marburg.de /MAGE/gdt/.

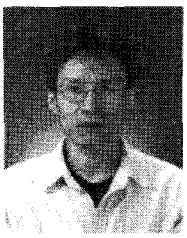
[23] The IDE for Grid Service Development using Eclipse, http://www.ibm.com/developer Works.

[24] A Java Commodity Grid Kit, http://www.cogkit. org/.

◐ 저 자 소 개 ◑

김 혁 호(HyukHo Kim)
2003년 대전대학교 정보통신공학과 졸업(학사)
2005년 동국대학교 대학원 정보통신공학과 졸업(석사)
2005년~현재 동국대학교 대학원 정보통신공학과 재학(박사)
관심분야 : 그리드 컴퓨팅, P2P 컴퓨팅, 유비쿼터스 컴퓨팅
E-mail : hulegea@dongguk.edu

김 양 우(Yangwoo Kim)
1984년 연세대학교 전자공학과(공학사)
1986년 Syracuse Univ. 컴퓨터공학전공(공학석사)
1992년 Syracuse Univ. 컴퓨터공학전공(공학박사)
1992년~1996년 한국전자통신연구원 선임연구원
1996년~현재 동국대학교 정보통신공학과 교수
관심분야 : 분산 그리드 컴퓨팅 시스템, 컴퓨터구조
e-mail : ywkim@dongguk.edu



이 필 우(Pil-Woo, Lee)
1991년 Univ. of Tsukuba 이공학연구과 컴퓨터공학전공, 공학석사
1997년 Univ. of Tsukuba 공학연구과 컴퓨터공학전공, 공학박사
현재 한국과학기술정보연구원(KISTI) 그리드컴퓨팅연구팀 팀장(책임연구원)
관심분야 : 그리드컴퓨팅, 분산컴퓨팅, 프로그래밍언어
E-mail : pwlee@kisti.re.kr