품질 기반 장애 극복을 지원하는 웹 서비스 시스템

이 용 표[†] · 신 재 동[††] · 한 상 용[†††]

요 약

웹을 통한 분산 컴퓨팅 환경을 제공하는 웹 서비스의 사용은 날로 증가하고 있다. 웹 서비스는 재사용 가능한 소프트웨어 컴포넌트를 제공하여 하나의 웹 서비스를 여러 사용자들이 사용하거나, 한 사용자가 여러 개의 웹 서비스들을 사용할 수 있게 한다. 이에 따라 웹 서비스의 신뢰성이 더욱 중요해지고 있지만, 기존의 장애 극복 방법은 극복을 위해 기본 클라이언트 측 웹 서비스 엔진의 변형이나 애플리케이션에서 별도 구현을 필요로 하며, 응답 시간 등의 품질 요소를 확장하여 고려하지 못하고 있다. 본 논문에서는 이를 개선하여 품질 요소를 지원하는 장애 극복 시스템과 여기에 필요한 기술 언어를 제안한다.

키워드 : 웹 서비스, Fault-Tolerant, QoS

Web Services System Supporting Fault-Tolerance based on the Quality

Yongpyo Lee[†] · Jaedong Shin[††] · Sangyong Han[†††]

ABSTRACT

Recently web services are being used to provide environments of distributed computing. Web services provide reusable software component. So, one web service can be used by many users, and one user can use different web services. For reliable use of web services, in these cases, it is important to be fault-tolerance. Existing fault-tolerant methods in web services need a kind of client modification and cannot consider extensible factors like quality. This study suggests the system architecture and description language for the system which can improve some of these problems.

Key Words : Web Services, Fault-Tolerant, QoS(Quality of Service)

1. 서 론

소프트웨어를 개발하고 운영하는 환경은 시대가 흘러 컴퓨팅 환경이 변화함에 따라 함께 변화하고 있다. 특히 네트워크의 확산은 폐쇄적이고 독립적인 환경 내에서만 수행되던 소프트웨어들을 분산 컴퓨팅 환경에서 수행 가능하도록 하였다. 웹 서비스는 XML 기술을 바탕으로 기존의 웹 환경을 이용하여 분산 컴퓨팅을 가능하게 하는 새로운 기술이다. 웹 서비스는 사용자가 언제든지 필요한 때에 원격지에서 인터넷으로 전달받아 사용하도록 하는 서비스 모듈(service module)이라 할 수 있으며, 사용자는 각기 다른 서비스 모듈들을 연결하여 자신의 새로운 서비스나 애플리케이션을 만들 수 있다. 또한 하나의 웹 서비스를 여러 사용자들이 함께 사용할 수도 있다. 많은 IT관련 기관이나 업체들은 웹 서비스를 미래의 컴퓨팅 환경 패러다임의 기반 인프라 기술로 여기고 이에 대한 리더십을 확보하려 애쓰고 있다[1, 2].

한편 웹 서비스가 사용하는 인터넷 환경은 여러 가지 장애 발생 가능성을 가지고 있다. 네트워크상에서의 오류 가능성뿐만 아니라 늘어나는 해킹이나 바이러스에 의한 공격 보고들은 인터넷의 장애가 언제든지 발생할 수 있음을 보여준다[3]. 따라서 웹 서비스를 이용한 애플리케이션의 안정성 있는 실행을 위해서는 웹 서비스에 대한 장애 극복 방법이 필요하다. 웹 서비스 장애 극복의 기본적인 방법은, 장애가 발생한 웹 서비스와 동일한 웹 서비스로 전환 연결을 하는 것이다. 그런데 웹 서비스 장애 극복을 위한 기존의 방법은 사용자 애플리케이션에서 일일이 장애 상황을 처리하는 등 사용자 측에서의 변형이 필요하며, 서비스 전환을 위해 동일 서비스를 찾는 방법도 사용자에 의존적이다. 또한 웹 서비스의 장애 상황뿐만 아니라 품질 요소 등 다른 요소들을 확장하여 고려할 수 없는 상황이다.

본 논문에서는 장애가 발생한 경우에 대해서만 웹 서비스의 전환 처리가 되던 것에, 장애를 포함한 서비스 품질 요소 지원을 가능하게 하여, 원하는 서비스 품질에 만족하지 못할 경우 다른 서비스로 전환 연결을 지원할 수 있도록 하

※ 이 논문은 2004년도 중앙대학교 학술연구비(교수연구년 연구비) 지원에 의한 것임.
† 정 회 원 : 중앙대학교 대학원 컴퓨터공학과
†† 준 회 원 : 중앙대학교 대학원 컴퓨터공학과 석사과정
††† 종신회원 : 중앙대학교 컴퓨터공학과 교수
논문접수 : 2005년 1월 6일, 심사완료 : 2005년 10월 17일

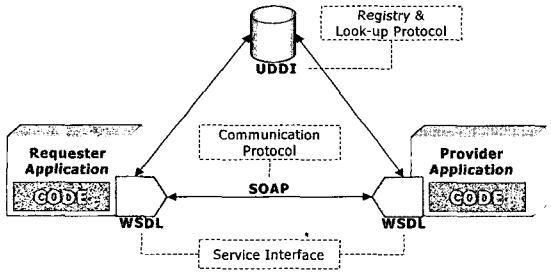
였으며, 시스템을 위해 웹 서비스에서의 효율적인 서비스 전환 방법과 동일 웹 서비스 검색 방안도 함께 제안하고 있다.

본 논문의 구성은 다음과 같다. 우선 2장에서는 기존 웹 서비스의 기본 구조와 장애 극복 방법에 대해 살펴본다. 3장에서는 기존 장애 극복 방법들의 문제점을 개선한 시스템의 구조와 필요한 기술 언어에 대해 논의한다. 4장에서는 3장의 내용을 바탕으로 구현 및 성능 평가를 하고, 마지막 5장에서는 결론 및 향후 연구 과제에 대해서 기술한다.

2. 관련 연구

2.1 웹 서비스

웹의 표준화 기구인 월드 와이드 웹 컨소시엄(W3C)은 웹 서비스를 서로 다른 플랫폼을 가진 기계들도 해석할 수 있는 포맷의 인터페이스를 통해 XML로 구조화된 데이터를 HTTP 등 기존 웹 기술을 이용하여 전송함으로써, 네트워크를 통한 상호운용성을 지원할 수 있도록 디자인된 소프트웨어 시스템이라고 정의하고 있다[4]. 웹 서비스는 (그림 1)과 같이 크게 세 가지 요소로 이루어져 있는데, 서비스 제공자와 서비스 사용자, 서비스 레지스트리가 그것이다[5]. 이들 사이에서 웹 서비스가 이뤄지기 위해서는 약속된 메시지, 서비스 정보, 검색 방법이 필요하며, 이에 대응하는 XML 기반의 표준이 바로 SOAP[6], WSDL[7], UDDI이다. (그림 2)는 이러한 표준들의 상호 관계를 보여준다.



(그림 1) 웹 서비스의 구조

서비스 등록과 검색	UDDI
서비스 기술	WSDL
XML 메시징	SOAP
전송 네트워크	HTTP/SMTP/FTP

(그림 2) 웹 서비스 스택

이러한 웹 서비스 기본 구조에서는 서비스 제공자로부터 서비스를 받는데 문제가 발생했을 경우 문제를 해결하는 방법이 정해져 있지 않다.

2.2 웹 서비스에서의 장애 극복

장애 극복을 하는 방법들[8, 9]은 서버의 장애를 감지하고, 이후의 클라이언트 요청은 백업 서버로 보내는 방식을 취하고 있다. 웹 서비스 기본 구조에서는 클라이언트 측에서 개별적으로 서비스 장애를 탐지하고 새로운 서비스를 찾아 해당 서비스로 재 호출하는 역할을 클라이언트 측에서 일일이 구현해야 한다.

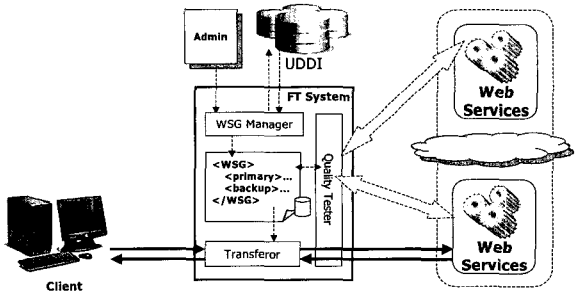
이중 웹 서비스의 장애 극복에 관한 연구[9]는 SOAP 메시지를 확장하여 연결 가능한 서비스 정보를 클라이언트에 제공해 주는데, 이 방법에서도 장애 극복 서비스를 이용하기 위해서 클라이언트 측에 확장된 SOAP 메시지를 처리하는 엔진이 필요하여 추가적인 변형이 요구된다.

이렇듯 기존 방법에서는 장애 극복을 사용하기 위해 클라이언트에서 어떠한 형태로든 수정이 필요하다. 또한 서비스 장애 상황 외에 서비스 품질이 원하는 수준에 미치지 못한 경우에 대하여 극복 방안을 고려하고 있지 못하며, 백업 서비스로서의 동일한 웹 서비스를 검색하는 방안에 대해서도 서비스 제공자와 사용자 모두에게 편리한 등록, 자동 검색, 연결을 제공해주고 있지 못하다.

3. 품질 기반 장애 극복 웹 서비스 시스템의 설계

3.1 시스템의 구조

이 절에서는 품질 기반 장애 극복 서비스를 위한 시스템의 구조를 설명한다. 시스템은 3.2절에서 살펴볼 세 가지 기본 기능들을 담당하는 오브젝트들로 구성되어 있으며, 이는 (그림 3)과 같다.



(그림 3) 품질 기반 장애 극복 시스템의 구조

초기 설정 단계에서 사용자는 Admin 애플리케이션을 이용하여 웹 서비스 등록을 한다. Admin은 WSDL을 등록하여 동일 웹 서비스 그룹을 설정하는 애플리케이션이며, 사용자는 Admin을 통해 직접 동일 웹 서비스를 등록할 수도 있다. 각 사용자는 IP 주소를 통해 식별되며, 사용자별로 동일 웹 서비스가 관리된다. 시스템은 WSG(Web Service Group) Manager를 통해 등록된 웹 서비스 마다 3.2절의 UDDI를 이용한 동일 서비스 검색 방법을 이용하여 동일 서비스들을 검색, 추가 하여 적당한 형태로 기술한다. 기술하는 내용의 구조는 3.3절에서 설명한다.

동일 웹 서비스 그룹을 완성하면, Quality Tester가 동일 웹 서비스 그룹에 속한 웹 서비스들 각각에 테스트 메시지를 보내 품질을 측정하여 메인 웹 서비스를 결정한다. 시스

템은 자신의 주소로 변경한 WSDL을 공개하여 클라이언트가 이를 통해 시스템에 연결할 수 있도록 한다.

이렇게 초기 설정을 마치면 서비스 사용자는 이 시스템을 통해 웹 서비스를 이용할 수 있게 된다. 실행 시간 단계에서, 사용자가 보낸 서비스 요청 메시지는 메인 웹 서비스로 전송되고 응답 메시지를 돌려받게 된다. Quality Tester는 메인 웹 서비스에 대한 품질 측정을 계속 하면서, 측적한 품질을 기록한다.

웹 서비스의 품질은 대표적으로 응답시간을 예로 들 수 있다. 응답시간은 요청을 보내고 응답을 받는데 걸린 시간을 의미한다.

메인 웹 서비스의 품질이 기준을 만족하지 못할 경우 Quality Tester는 3.2절에서처럼 동일 웹 서비스 그룹 내의 웹 서비스들에 다시 메시지를 보내 새로운 메인 웹 서비스를 정하게 된다.

Transferor는 클라이언트에서 클라이언트가 보내온 메시지를 메인 웹 서비스로 전달해 주는 역할을 한다. 이 때 클라이언트는 어떤 수정도 필요하지 않다. Transferor는 기술된 내용으로부터 메인 웹 서비스를 알아내어 메시지를 주고받는다.

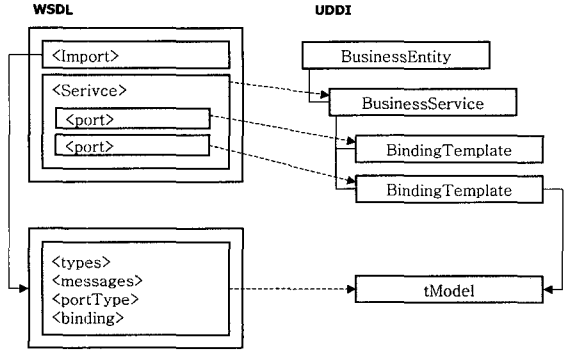
3.2 시스템의 기능

3.2.1 동일한 서비스 관리

백업용 동일 서비스를 찾고 연결해 주기 위해서 UDDI의 tModel을 이용한다. 서비스 등록기인 UDDI는 businessEntity, businessService, bindingService, tModel로 구성되며, 이 중 tModel은 서비스나 밸류 셋(value set)에 대한 명세(specification)를 기술하여, 웹 서비스가 어떤 협약을 따르는지, 무슨 스펙이나 표준을 따르는지를 나타낸다.

UDDI는 (그림 4)에서처럼 WSDL 문서의 내용을 반영하는데, WSDL의 인터페이스 부분은 tModel로 정의한다[11]. tModel은 키 값을 통해 서비스 인터페이스에 대한 고유한 값을 부여할 수 있게 된다. 따라서 tModel의 키 값을 알고 있다면, 이를 검색하여 동일한 인터페이스를 가진 웹 서비스들을 찾아낼 수가 있다.

그러나 tModel의 키 값을 UDDI 검색을 통해 얻을 수 있기는 해도, 사용자가 일일이 tModel의 키 값을 알고 있기는 쉽지 않다. 그래서 본 시스템은 tModel과 매핑되는 WSDL



(그림 4) WSDL과 UDDI의 매핑[10]

〈표 1〉 tModel 작성요령[12]

UDDI tModel	WSDL Service Interface	Required
name	targetNamespace attribute for definitions element	Yes
description	documentation element within the definitions element	No
overviewURL	[Service interface document URL and binding specification]	Yes
categoryBag	[Not applicable]	Yes

정보를 이용한다. 매핑 관계는 <표 1>에 정리되어 있으며, 이에 따라 서비스 사용자는 기존에 사용하고 있던 웹 서비스 WSDL의 targetNamespace 값을 통해 tModel 이름(name)을 검색, 동일 웹 서비스를 찾게 된다.

서비스 제공자가 백업 시스템을 추가할 때도 단순히 UDDI에 동일한 tModel 등록을 함으로써 백업 시스템을 알릴 수 있게 되어, 서비스 신뢰성을 높일 수 있다. 단, UDDI는 자유롭게 서비스에 대한 메타데이터 저장을 할 수 있는 레지스트리이므로, 직접 동일 웹 서비스를 등록할 수 있는 방법도 필요하다.

3.2.2 품질 요소들에 대한 관리

시스템이 장애를 포함한 웹 서비스의 품질을 관리하기 위해서는 동일 웹 서비스들의 품질을 언제 검사할지 결정해야 한다. 서비스에 대한 최신의 정보를 얻기 위해서는 매번 모든 서비스를 호출하여 품질을 측정하는 것이 좋지만, 효율이 떨어지게 된다. 본 논문이 제안하는 시스템은 아래와 같은 방법을 사용한다.

먼저, 설정된 동일 웹 서비스 그룹의 모든 웹 서비스들에 테스트 메시지를 보내서 장애 여부를 비롯한 웹 서비스 품질 요소들을 측정한다. 이 결과를 바탕으로 메인(main) 웹 서비스를 선택하게 된다.

이후부터 시스템은 사용자가 보낸 요청 메시지를 메인 웹 서비스로 전송하게 된다. 메인 웹 서비스에서 돌아온 응답 메시지들은 그 때마다 측정되어 통계 자료로 활용한다.

만일 메인 웹 서비스가 장애 등 품질 조건에 만족하지 않으면, 그때 다시 동일 서비스 그룹 내 모든 웹 서비스들에게 메시지를 보내어, 새로운 메인 웹 서비스를 설정한다. 품질 조건에 만족하지 않는 경우를 세부적으로 살펴보면, 장애 발생 시에는 메시지를 재전송하지만, 품질 기준을 넘은 경우에는 일단 가장 좋은 결과를 반환하고 다음 요청부터 새로운 메인 웹 서비스로 연결한다.

3.2.3 복구 메커니즘

장애 극복을 위해서는 사용자가 WSDL을 보고 보내온 SOAP 요청 메시지를 동일 서비스 그룹의 다른 웹 서비스 제공자에게 전달하고 그 SOAP 응답 메시지를 해당 사용자에게 돌려주는 것이 필요하다.

SOAP의 구조에서 SOAP 자체는 시스템의 주소를 표현

하는 부분이 없으며 따라서 복수 개의 동일한 서비스에 보내는 SOAP 메시지 자체에는 차이가 없다. SOAP은 HTTP 등 다른 프로토콜에 실려 전달되는데, 주소는 이러한 프로토콜에서 담당한다. 따라서 본 시스템에서는 서비스 전환시에 SOAP 메시지는 변경 없이 프로토콜에서 주소를 변경하여 SOAP 메시지를 보내도록 하였다.

본 논문에서 제안하는 장애 극복 시스템은 동일 서비스 그룹을 설정하면, 시스템은 연결 주소 부분만 자신의 것으로 바꾼 WSDL을 제공한다. WSDL의 웹 서비스 인터페이스를 기술하는 부분은 같은 서비스를 제공하는 경우에 동일하며, 시스템의 위치와 연관이 있는 부분은 service의 port 엘리먼트이다.

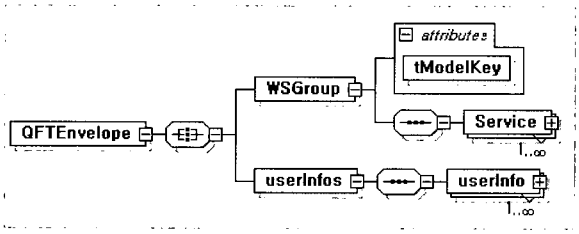
클라이언트는 장애 극복 시스템의 WSDL을 보고서 보통의 웹 서비스를 이용하듯이 SOAP 메시지를 생성하여 보낸다. 시스템은 SOAP 메시지를 받아서 적당한 시스템으로 중계를 하게 된다.

3.3 시스템을 위한 기술 언어

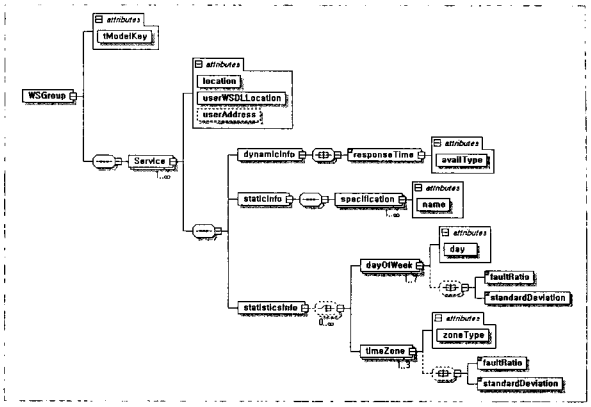
품질 기반 장애 극복 웹 서비스 시스템을 위한 정보는 크게 세 가지로 나누어 볼 수 있다. 먼저, 대체할 수 있는 동일한 서비스들을 기술한 정보, 그리고 각 웹 서비스의 품질 정보를 측정하여서 이를 기술한 정보, 마지막으로 클라이언트의 정보와 요구하는 품질 을 기술한 정보가 필요하다. 이러한 정보를 기술하는 방법은 여러 가지가 있으나, 여기에서는 사용이 쉽고 확장성이 뛰어난 XML을 기반으로 정보들을 기술하고자 한다. 본 절에서는 이러한 내용들을 기술하는 기술 언어를 설계 제안한다. 이 설계 언어의 네임스페이스는 "http://ec.cse.cau.ac.kr/qfs"를 사용한다.

장애 극복 시스템에서 웹 서비스의 품질을 나타내는 요소 중 가장 대표적인 것은 장애(fault) 여부와 응답 시간(response time)이다. 여기에서는 이러한 동적 정보뿐 아니라 일정 기간 동안의 응답 시간 편차나, 장애 발생 빈도 등의 통계적 품질 요소와, 다양한 웹 서비스관련 표준 지원 여부 등의 정적 품질 요소를 네임스페이스 검사를 통해 사용하도록 하였다.

필요한 정보들은 동일 웹 서비스 그룹을 중심으로 QFT Envelope 엘리먼트를 사용해서 기술하였으며, 루트 엘리먼트인 QFTEnvelope 엘리먼트는 (그림 5)과 같이 크게 동일 서비스를 관리하는 WSGroup 엘리먼트와 클라이언트의 정보를 관리하는 userInfos 엘리먼트 부분으로 나뉜다.



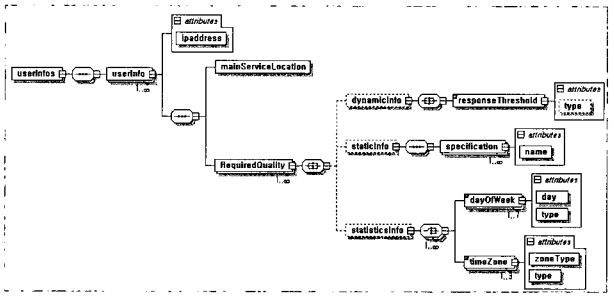
(그림 5) 품질 기반 장애 극복 시스템을 위한 기술 언어의 스키마 다이어그램



(그림 6) WSGroup 엘리먼트의 스키마 다이어그램

WSGroup 엘리먼트는 동일 웹 서비스들을 하나의 그룹으로 묶고[9] 각각의 품질 정보들을 기술하기 위한 엘리먼트이다. WSGroup 엘리먼트 스키마는 (그림 6)과 같다.

동일 웹 서비스 그룹은 같은 웹 서비스 인터페이스를 가지므로 이를 나타내는 tModel key 속성 값을 대표로 하여 웹 서비스 그룹을 나타낸다. WSGroup 엘리먼트는 동일 웹 서비스로 묶인 각각의 웹 서비스들의 정보를 가지는 Service 엘리먼트들을 자식 엘리먼트로 갖는다. Service 엘리먼트는 서비스의 주소를 나타내는 location 속성 값을 통해 구분된다.



(그림 7) userInfos 엘리먼트의 스키마 다이어그램

userInfos 엘리먼트는 해당 웹 서비스 그룹을 사용하는 사용자들에 대한 클라이언트 측의 정보를 기술하기 위한 엘리먼트이다. userInfos 엘리먼트 스키마는 (그림 7)과 같다.

userInfos 엘리먼트는 userInfo 엘리먼트를 자식 엘리먼트로 가지며, 각각의 userInfo 엘리먼트는 ipaddress 속성 값을 통해 사용자를 구분한다. userInfo 엘리먼트는 mainServiceLocation 엘리먼트와 RequiredQuality 엘리먼트를 갖는데, 여기에 메인 웹 서비스와 원하는 품질 기준을 기술하게 된다.

4. 시스템의 구현 및 평가

4.1 품질 기반 장애 극복시스템의 구현

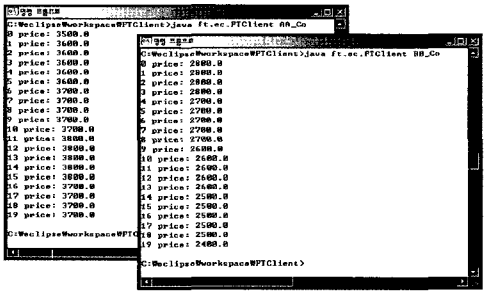
본 품질 기반 장애 극복 시스템은 Java 언어로 구현하였으며, 웹 서비스 제공자나 클라이언트 측에 필요한 웹 서비

스 엔진으로는 Apache Axis를 사용하였다.

실험을 위해서 'double getStockPrice(String)'의 형태를 가지는 메소드를 만들어 웹 서비스로 공개 하였으며, 이는 10초마다 일정 범위에서 사인 곡선 형태로 ±100씩 변하는 값을 반환한다. 4.2절의 실험을 예로 들면, 3200에서 3800까지 10초마다 100씩 증가하고 3800에서 3200까지 10초마다 100씩 감소하는 값을 가진다.

서비스 제공자는 각기 다른 서버에 동일하게 배치(deploy)하였으며, LAN 환경에서 실험을 하는 관계로 각각의 웹 서비스들은 sleep() 명령의 인자 값을 외부 파일에서 읽어 들여, 지연(delay) 시간을 동적으로 설정, 응답 시간의 변화를 줄 수 있도록 하였다.

클라이언트는 설정한 시간 간격과 횟수대로 위의 웹 서비스를 호출하여 결과값을 화면에 출력한다.



(그림 8) 클라이언트 수행 화면

4.2 성능 및 기능 평가

평가는 크게 기능 평가와 성능 평가 두 부분으로 나누어 진행하였다.

시스템의 기능 평가에서 클라이언트는 5초마다 한 번씩 서비스를 호출한다. <표 2>는 본 시스템을 사용하지 않고 바로 웹 서비스를 호출할 때의 결과로, 중간에 11000ms의 지연시간을 설정하자 100 이상으로 값이 변화하는 구간이 발생하고 있다.

<표 3>과 <표 4>는 본 시스템을 사용했을 때의 결과로, 시스템에 설정한 응답시간의 임계값(threshold)은 5000ms이다. <표 3>은 위에서처럼 11000ms의 지연시간을 준 경우이고, <표 4>는 LAN선을 제거하여 장애 상황을 만든 경우이다.

<표 3>과 <표 4>의 결과는 중간에 응답 시간이 5000ms를 초과하면 3000ms의 응답시간을 가지는 다른 동일한 웹 서비스로 전환되는 것을 보여준다.

〈표 2〉 극복 시스템 사용 안한 경우(응답 시간 초과)

횟수	1	2	3	4	5
결과 값	3600	3600	3700	3700	3800
응답시간	30	40	20	10	11016
횟수	6	7	8	9	10
결과 값	3800	3600	3400	3300	3300
응답시간	11026	11026	11026	11026	11026

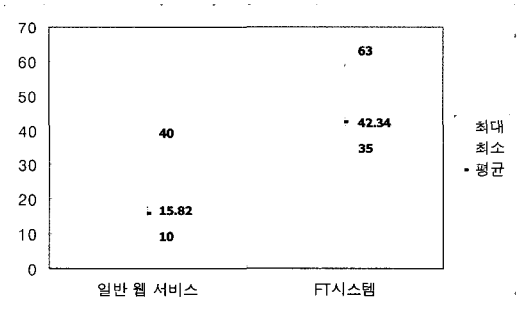
〈표 3〉 극복 시스템을 사용한 경우(응답 시간 초과)

횟수	1	2	3	4	5
결과 값	3400	3500	3500	3600	3700
응답시간	62	50	48	42	5140
횟수	6	7	8	9	10
결과 값	3700	3800	3700	3700	3600
응답시간	3050	3045	3043	3038	3040

〈표 4〉 극복 시스템을 사용한 경우(장애 발생)

횟수	1	2	3	4	5
결과 값	3200	3300	3300	3400	3500
응답시간	52	48	42	38	5118
횟수	6	7	8	9	10
결과 값	3500	3600	3700	3700	3800
응답시간	3040	3042	3038	3036	3036

성능 평가는 가끔 발생하게 되는 장애 상황을 위한 시스템이 평소 서비스 이용에 부담을 주어서는 안되므로, 극복 시스템을 거치지 않고 웹 서비스를 호출하는데 걸리는 시간과 시스템을 거쳐서 웹 서비스를 호출할 때 걸리는 시간을 측정 비교한다. 각각에 대해 클라이언트에서 2초마다 50번 호출을 하여 응답 시간의 최대치와 최소치, 그리고 평균을 구하였다. (그림 9)는 그 결과이다. 단위는 ms(millisecond)이며, 둘 사이에는 25ms 정도의 차이를 보이고 있다. 즉 중간에 극복 시스템을 거쳐 가는데 25ms 정도의 시간이 걸리게 된다.



(그림 9) 극복 시스템 사용시 응답 시간 비교

5. 결론 및 향후 연구과제

웹 서비스의 사용은 점차 확대되고 있으며, 이에 따라 웹 서비스의 장애 발생 등을 극복할 수 있는 방안이 필요하게 되었다. 본 논문은 기존의 장애 극복 방법을 개선하여 클라이언트의 수정 없이 품질 요소들을 추가 반영할 수 있으며 동일 서비스 검색의 불편을 개선한 극복 시스템의 구조를 제안하고 평가하였다.

극복 시스템을 이용하여 서비스 제공자는 동일한 서비스

를 지원하는 백업 시스템을 같은 도메인 아래 두지 않고서도 사용할 수 있다. 이는 본 시스템과 백업 시스템들이 동시에 문제가 생길 확률을 낮출 수 있다.

성능 측정 결과에서 본 시스템을 사용한 경우와 그렇지 않은 경우는 25ms 정도의 차이를 보이고 있다. 이는 클라이언트가 본 시스템을 거치지 않고 직접 웹 서비스를 호출하는 경우에도 각각의 호출에서 일어나는 편차 정도의 시간이다. 본 논문의 구조에서는 클라이언트 측에 수정을 하지 않고 있어, 기존의 SOAP 엔진을 수정한 것과 비교하여 돌아가는 시간이 조금 더 걸리는 것은 감수하여야 한다. 한편, 서버-클라이언트 구조 기반의 본 시스템에서 시스템 자체에 문제가 생기는 부담을 줄이기 위해 시스템의 Transferor 부분을 백업용으로 더 두어서 heartbeat 방식[8]으로 장애를 확인, 문제가 생긴 경우 대신하여 메시지를 전송하도록 하는 방안을 고려할 수 있다.

본 연구는 품질 요소를 고려한 장애 극복 시스템에 필요한 부분과 구조를 살펴본 것으로, 사용자 증대 등 필요에 따라 다른 기능을 확장하거나 최적화할 여지는 남아있다. 또한, 응답 시간을 예상하여 서비스를 분산하는 등 품질을 좀 더 확실히 보장하는 방안에 대한 연구가 향후 필요하다.

참 고 문 헌

[1] 김진태, 김은정, 박수용, "웹 서비스를 위한 소프트웨어 공학", 정보처리학회 제9권 4호, 2002.7.

[2] 이경하, 이규철, "웹 서비스의 향후 발전 방향", 정보처리학회 제9권 4호, 2002. 7.

[3] KrCERT 인터넷 침해사고 대응지원센터, 해킹 및 바이러스 통계보고서, "http://www.krcert.or.kr/"

[4] Web Service Architecture, "http://www.w3.org/TR/ws-archi/" W3C Working Group Note, 2004. 2. 11.

[5] 이한수, 웹 서비스 실전프로그래밍, 한빛미디어, 2003.

[6] Simple Object Access Protocol (SOAP) 1.1, "http://www.w3.org/TR/2000/NOTE-SOAP-20000508/"

[7] Web Services Description Language (WSDL) 1.1, "http://www.w3.org/TR/wsdl"

[8] Navid Aghdaie, Yuval Tamir, "Client-Transparent Fault-Tolerant Web Service", Performance, Computing, and Communications, 2001. IEEE International Conference on. 4-6 April 2001, pp.209-216.

[9] Deron Liang, Chen-Liang Fang, Chyouhwa Chen, Fengyi Lin, "Fault tolerant web service", Proceedings of the Tenth Asia-Pacific Software Engineering Conference 2003 (APSEC'03)

[10] 정지훈, 웹 서비스, 한빛미디어, 2002.

[11] UDDI V3, "http://uddi.org/pubs/uddi-v3.htm"

[12] Understanding WSDL in a UDDI registry, "http:// www-106.ibm.com/developerworks/webservices/library/ws-wsdl/", IBM developerWorks, 2002. 9.

이 용 표

e-mail : drgnyp@archi.cse.cau.ac.kr

2003년 중앙대학교 컴퓨터공학과(공학사)

2005년 중앙대학교 대학원 컴퓨터공학과 (공학석사)

관심분야 : 웹 서비스, ebXML, mobile

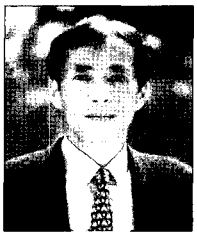
신 재 동

e-mail : mulli2@archi.cse.cau.ac.kr

2005년 중앙대학교 컴퓨터공학과(공학사)

2005년~현재 중앙대학교 대학원 컴퓨터공학과 석사과정

관심분야 : 웹 서비스, AI(Artificial Intelligence)



한 상 용

e-mail : hansy@cau.ac.kr

1975년 서울대학교 공과대학(공학사)

1984년 Minnesota 공과대학(공학박사)

1984년~1995년 IBM 책임연구원

1995년~현재 중앙대학교 컴퓨터공학과 교수

관심분야 : Web Engineering(Web Services, etc.), Information Retrieval, eCommerce Technologies