Extended Virtual Synchrony를 지원하는 자바 그룹통신 시스템

(A Java Group Communication System supporting Extended Virtual Synchrony)

문 남 두 † 이 명 준 ††
(Nam-Doo Moon) (Myung-Joon Lee)

요 약 인터넷의 성장과 함께 자바 네트워크 응용서비스가 빠른 속도로 증대되고 있다. 이러한 응용서비스는 일시적으로 네트워크가 분할되거나, 또는 특정 프로세스가 실패하더라도 투명하고도 안정적이며, 지속적으로 제공되는 것이 바람직하다. 이러한 요구사항을 만족시키기 위하여 다수의 그룹통신 시스템이 개발되어 왔다. 그러나 기존의 자바 그룹통신 시스템은 Extended Virtual Synchrony를 지원하지 못하거나 선입선출(FIFO), 인과(causal), 전체(total) 그리고 안전(safe) 순서 전달서비스와 같은 다양한 메시지 전달방식을 지원하고 있지 않다.

본 논문에서는 그룹 멤버간의 다양한 메시지 전달방식과 Extended Virtual Synchrony 모델을 지원하는 JACE 자바 그룹통신 시스템의 설계와 구현에 관하여 기술한다. JACE 시스템은 다양한 방식으로 쌓을 수 있는 다수의 프로토콜 모듈로 구성되어 있다. 개발된 JACE 시스템을 이용하여 웹 서비스에 대한 정보를 등록하고 발견할 수 있는 UDDI 레지스트리가 자바를 이용하여 실험적으로 구현되었다.

키워드 : 네트워크 분할, 자바 그룹통신 시스템, Extended Virtual Synchrony, UDDI 레지스트리

Abstract Important Java network application services have been rapidly increased along with the growth of the Internet. So, it is desirable for such applications to serve transparently, continuously and safely even if the network is temporally partitioned or certain hosts running those services are crashed down. To satisfy such requirements, many group communication systems have been developed. However, existing Java-based group communication systems do not support both the extended virtual synchrony and various types of message delivery such as FIFO, causal, total and safe delivery service.

In this paper, we present the design and implementation of a Java group communication system, named JACE, supporting various types of message delivery between group members and the extended virtual synchrony model. The JACE system consists of a number of protocol modules which can be stacked on top of each other in a variety of ways. In addition, using the JACE system, we have developed an experimental UDDI registry for discovering and publishing information about Web services.

Key words : Network Partition, Java Group Communication System, Extended Virtual Synchrony, UDDI Registry

1. 서 론

인터넷의 급속한 성장과 정보통신 관련 기술의 발전으로 오늘날 다양한 분야에서 자바 네트워크 응용서비스 개발이 활발하게 이루어지고 있다. 이러한 응용서비스는 동일한 서비스를 제공하는 다수의 프로세스가 통신 네트워크를 통하여 상호 연결된 다수의 컴퓨터상에서 일관된 서비스를 제공할 때 더욱 유용하다. 유용한 서비스를 다수의 프로세스에 복제하여 제공함으로써 가용성을 높이고 특정 프로세스가 실패하더라도 안정적이고 지속적인 서비스를 제공할 수 있다. 하지만, 통신 네트워크가 일시적으로 분할되고 프로세스가 실패하는 상황에서 프로세스들 간에 복제되는 정보를 일관성 있게 유지하는 분산 어플리케이션 개발은 쉽지 않다.

† 학생회원 : 울산대학교 컴퓨터정보통신공학과
dooya@mail.ulsan.ac.kr
†† 종신회원 : 울산대학교 컴퓨터정보통신공학과 교수
mjlee@uou.ulsan.ac.kr
논문접수 : 2003년 5월 16일
심사완료 : 2003년 10월 27일

프로세스 그룹[1]은 분산 시스템의 신뢰성과 성능향상을 제공하기 위하여 프로세스의 상태를 중복하여 유지함으로써 어느 한 프로세스가 실패하더라도 복제된 다른 프로세스를 통하여 지속적으로 서비스를 제공하는 기법이다. 이러한 프로세스 그룹을 지원하는 다수의 *그룹통신 시스템*(group communication system)이 개발되어 왔다. 기존의 개발된 시스템으로는 Isis[2], Horus [3], Transis[4, 5], Totem[6], JavaGroups[7], 그리고 Jgroup[8] 시스템 등이 있다. 개발자는 이러한 시스템을 이용함으로써 보다 쉽게 견고한 분산 어플리케이션을 개발할 수 있다.

그룹통신 시스템 모델은 크게 VS(Virtual Synchrony)[9,10] 모델과 EVS(Extended Virtual Synchrony) [11,12] 모델로 구별될 수 있다. VS 모델은 그룹의 멤버들간에 송수신 되는 일반 메시지와 그룹 멤버십 변경 메시지를 순서화(ordering) 한다. VS 모델을 지원하는 그룹통신 시스템은 응용프로세스가 실행되고 있는 호스트가 실패하거나 일시적으로 네트워크의 분할이 발생되어 그룹이 상호 통신할 수 없는 여러 *서브그룹*(subgroup)으로 분리된 후 재결합될 때 이들 사이에 일관성을 유지하지 못하는 문제점을 가지고 있다. EVS 모델은 하나의 프로세스 그룹이 일시적인 네트워크 단절로 상호 통신할 수 없는 두 개 이상의 서브그룹으로 분리되고 이후 다시 네트워크가 복원되어 상호통신이 가능하게 될 때, 다시 이 서브그룹들이 하나의 그룹으로 동작할 수 있도록 프로세스 멤버들간의 메시지 전달을 일관성 있게 유지한다.

본 논문에서는 JACE(Java Advanced Communication Environment)라 명명한 그룹통신 시스템의 설계와 구현에 대하여 소개한다. 기존의 개발된 시스템들이 특정 플랫폼만을 지원하는 반면 JACE 시스템은 자바로 개발되었으며 분산 자바 어플리케이션의 가용성과 신뢰성 제공을 목적으로 한다. JACE 시스템은 EVS를 지원하며 그룹통신의 유연성을 제공하는 프로토콜 스택 개념과 메시지의 선입선출(FIFO), 인과(casual), 전체(total)와 안전(safe) 순서 전달서비스를 지원한다. 또한 JACE 시스템에서는 프로세스 그룹에 참여하는 멤버들간의 복제되는 상태정보에 관련하여 메시지의 유형을 크게 세 가지 유형으로 구분하는 히스토리(history) 계층을 지원한다.

본 논문의 구성은 다음과 같다. 2장에서는 그룹통신 시스템의 일반적인 기능과 관련연구에 대하여 소개한다. 3장에서는 JACE 시스템 구조와 JACE 프로토콜 계층간의 통신방법을 서술한다. 4장에서는 JACE 시스템이 제공하는 서비스와 성능에 관하여 살펴본다. 5장에서는 JACE 시스템을 이용하여 개발한 그룹으로 동작하는 *UDDI 2.0 레지스트리*에 대하여 설명한다. 마지막으로 6장에서 결론을 맺는다.

2. 관련 연구

최근 몇 년간 다양한 그룹통신 시스템이 개발되어 왔다. 기존의 개발된 시스템으로는 Isis, Transis, Totem, Horus, JavaGroups 그리고 Jgroup 등이 있다. 그룹통신 시스템은 기본적으로 메시지의 순서화 기능, 신뢰성 있는 멀티캐스팅 기능과 그룹의 멤버십 관리 기능을 제공한다.

Isis 시스템은 Cornell 대학에서 개발한 그룹통신 시스템이다. Isis는 기본적인 프로세스 그룹의 기능을 지원하고, 그룹의 모든 멤버들간에 일관성 있고 순서화된 메시지 전달을 보장한다. Isis 시스템은 VS 모델을 지원하는 범용 목적의 시스템이며 그룹통신 시스템의 시초라고 말할 수 있다. 그리고, Isis는 프로세스 그룹 개념을 적용하여 다양한 응용프로그램을 개발할 수 있는 개발자 인터페이스를 Toolkit의 형태로 제공하고 있으며, 프로세스 그룹의 구성원들간에 일관성을 유지하기 위한 상태전이 기능도 제공하고 있다. 프로세서의 복원(recovery)과 네트워크 분할과 재결합되는 상황에 관한 지원은 없으며 하부통신은 점대점(point-to-point) 방식으로 구현되었다.

Transis 시스템은 Lansis와 Toto 프로토콜을 사용하여 순서화된 메시지의 신뢰성 있는 전달과 멤버십 관리를 지원한다. Lansis 프로토콜은 Trans 프로토콜에서 유래되었으며 메시지를 부분 순서(partial order)로 전달하기 위해 사용된다. Lansis 프로토콜을 실행하는 프로세스는 멀티캐스트 메시지들이 인과 순서(causal order)가 만족되어 전달될 수 있을 때까지 인정신호(ack)를 기다린다. Toto 프로토콜은 결함-포용적이지 못하므로 호스트 실패 발견과 재구성에 관한 기법은 Lansis 프로토콜에 의해 제공된다. 실패를 발견하는 기법으로는 타임아웃을 사용하며, 새 멤버십에 대한 동의를 얻기 위한 과정에서 여러 번의 메시지 전달을 필요로 한다. 멤버십이 재구성되는 동안에도 메시지는 멀티캐스트 될 수 있지만 Toto 전체 순서화(total ordering) 프로토콜은 중지된다.

Totem 시스템은 다수의 LAN으로 연결된 광범위한 네트워크에서도 신뢰성 있는 멀티캐스트와 멤버십 서비스를 제공한다. Totem 시스템은 계층적으로 두 개의 프로토콜을 가지고 구성되었다. 하위 계층은 브로드캐스트 도메인(broadcast domain)에서 신뢰성 있는 멀티캐스트와 프로세서 멤버십을 제공하기 위한 단일-링 프로토콜이며, 상위 계층은 네트워크의 전체에 걸쳐 신뢰성 있는 전달과 순서화를 제공하는 다중-링 프로토콜이

다. 게이트웨이(gateways)들은 브로드캐스트 도메인들 사이에서 일반 메시지와 상호 연결된 프로세서 멤버십의 변경에 관한 정보를 넘겨주기 위한 역할을 수행한다. 각 게이트웨이는 두 개의 브로드캐스트 도메인을 상호 연결하고 각 도메인의 단일-링 프로토콜에 참여한다. 각 도메인은 여러 개의 다른 도메인과 연결하는 다수의 게이트웨이를 포함할 수 있다.

Horus 시스템은 Isis 시스템의 후속 버전이며 분산 어플리케이션 작성을 위한 유연성 있는 그룹통신의 기능을 제공한다. 단일 시스템으로 동작하는 프로세스 그룹이 요구하는 정도에 따라 그룹통신의 프로토콜 계층을 구성할 수 있다. 그룹통신의 형태는 Lego 블록처럼 쌓인 프로토콜 스택에 따라 매우 다양하게 표현될 수 있다. 그룹통신 시스템 개발자는 새로운 프로토콜 계층을 추가함으로써 Horus 시스템을 쉽게 확장할 수 있다.

Jgroup 시스템은 분산 객체 기술과 그룹 기술이 통합된 형태이다. Jgroup은 복제를 통하여 신뢰성 있고 고가용적인 서비스 개발을 지원하는 *객체 그룹* 프로그래밍 패러다임을 지원한다. Jgroup은 자바 분산 객체 모델에 기반을 두고 있고, 자바로 구현되었다.

기존의 개발된 그룹통신 시스템과 JACE 시스템을 비교해보면 표 1과 같다. 표 1에서 보는 바와 같이 다수의 그룹통신 시스템들이 C언어로 작성되었으며 특정 플랫폼만을 지원하고 있다. Jgroup과 JACE 시스템은 Java 언어로 개발되어 플랫폼에 독립적이다. Horus와 JACE 시스템의 구조는 기능별로 분리된 프로토콜 계층의 형태로 구현됨으로써 새로운 기능을 추가하거나 기존의 구현을 새로운 구현으로 대체하는데 비교적 용이하다. 반면 다른 그룹통신 시스템들의 경우에는 모든 기능들이 단일 혹은 소수의 통합된 계층으로 구현됨으로써 시스템 확장이나 수정이 비교적 복잡하다.

JACE 시스템은 네트워크의 분할로 인해 프로세스 그룹이 두 개 이상의 서브그룹들로 나뉘어 지더라도 지속적인 고가용성(high availability)의 지원을 위해 히스토리(history) 계층을 지원한다. 프로세스 그룹에 참여하는 멤버들 간의 복제되는 정보는 각 멤버들에게 전달되어 처리되는 메시지에 의해 그 상태가 결정된다고 가정한다. 히스토리 계층은 각 멤버들에 의해 복제되는 상태정보의 특징에 따라 메시지의 유형을 3가지로 구별한다. 첫째로, *history-sensitive* 메시지는 프로세스의 상태가 처리된 메시지의 집합과 처리순서에 의해 결정되는 상태정보에 적용된다. 둘째로, *semi history-free* 메시지는 최종적으로 전달된 메시지에 의해 그 상태가 결정되는 정보에 적용된다. 셋째로, *history-free* 메시지는 프로세스의 상태를 변경하지 않는 메시지에 적용된다. 네트워크 복원으로 인하여 다수의 서브그룹이 통합되거나 새로운 멤버가 그룹에 참여할 때 history-sensitive 메시지의 경우 동일한 메시지 집합에 대해 동일한 순서로 교환되며, semi history-free 메시지의 경우는 최종의 메시지만 교환된다. history-free 메시지의 경우는 메시지 교환을 필요로 하지 않는다.

표 2는 그룹통신 시스템을 이용하여 개발한 응용사례를 보여준다.

표 1 그룹통신 시스템 비교

그룹통신 시스템	지원모델	메시지 전달방식	개발언어 및 OS	시스템 구조	지원 메시지종류
Transis	EVS 프로세스 그룹지원	· FIFO · Causal · Total · Safe	C언어 UNIX	통합된 단일 계층 구조 (시스템 확장이 비교적 복잡함)	· *history-sensitive*
Horus	EVS 프로세스 그룹지원	· FIFO · Causal · Total · Safe	C언어 UNIX	기능별로 분리된 프로토콜 스택구조 (시스템 확장이 비교적 단순함)	· *history-sensitive*
Totem	EVS 프로세스 그룹지원	· Total · Safe	C언어 UNIX	통합된 단일 계층구조 (시스템 확장이 비교적 복잡함)	· *history-sensitive*
Jgroup	VS 객체그룹 지원	· Total	Java언어 플랫폼 독립적	통합된 단일 계층구조 (시스템 확장이 비교적 복잡함)	· *history-sensitive*
JACE	EVS 프로세스 그룹지원	· FIFO · Causal · Total · Safe	Java언어 플랫폼 독립적	기능별로 분리된 프로토콜 스택구조 (시스템 확장이 비교적 단순함)	· *history-sensitive* · *semi history-free* · *history-free*

표 2 그룹통신 시스템 응용사례

그룹통신 시스템	응용사례
Transis	· SNMP 기반 네트워크 관리 툴킷 · 공유(shared) 화이트 보드 · 협력(collaboratoin) 작업 시스템의 메시징 기능과 멤버십 기능 · 멀티미디어 멀티캐스트 서비스 등
Horus	· CORBA request broker · 결함포용(fault-tolerant) 멀티미디어 툴킷 · 공동 문서 편집기 등
Totem	· 항공 트래픽 제어 시스템 (데모 버전) · Eternal System 등
Jgroup	· JINI 확장 등
JACE	· Replicated Object Space · LDAP Server and LDAP Service Provider · private UDDI 2.0 registry 등

3. JACE 시스템 구조

JACE 시스템은 동일한 서비스를 제공하는 자바 응용 프로세스를 그룹으로 동작될 수 있도록 지원하고 그룹 멤버간에 일관성을 유지하는 자바 그룹통신 시스템이다. 3장에서는 JACE 시스템의 구조에 대해 살펴본다.

3.1 JACE의 프로토콜 스택

JACE 시스템은 프로세스 그룹 내에서 발생한 메시지들에 대한 순서화(ordering) 기능, 신뢰성 있는 멀티캐스트 기능 그리고 멤버십 관리 기능 등을 지원한다. JACE 시스템은 메시지의 분실, 네트워크 분할, 프로세스의 실패(fail)와 같은 결함이 발생하는 상황에서 신뢰성 있는 메시지 전달을 보장한다.

JACE 시스템 구조는 그림 1에서 보는 바와 같이 다수의 프로토콜 계층으로 구성되었다. 이러한 JACE의 구조는 응용서비스의 요구사항에 적합한 서로 다른 프로토콜 스택을 구성할 수 있도록 한다. 그룹통신 시스템

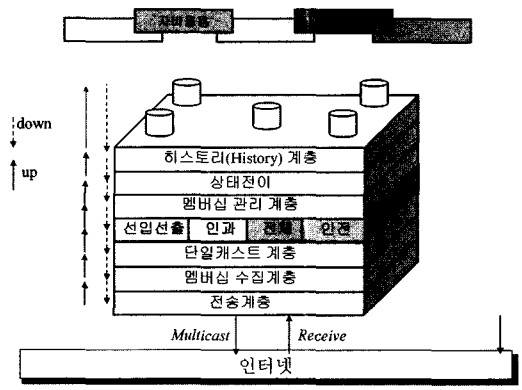

그림 1 JACE 시스템 구조

개발자는 JACE 시스템에 새로운 프로토콜 계층을 추가함으로써 용이하게 시스템을 확장할 수 있다. 또는 기존의 구현된 계층을 새로운 알고리즘을 적용하여 새로운 구현으로 수정할 수 있다.

- 전송 계층 : 전송 계층은 UDP 프로토콜을 사용하여 메시지를 송수신한다. 메시지는 단일캐스트 메시지와 그룹내의 모든 멤버를 대상으로 하는 멀티캐스트 메시지로 구분될 수 있다. 이 계층에서 전달되는 메시지는 분실되거나 중복될 수 있다.
- 멤버십 수집 계층 : 프로세스 그룹에 참여하고자 하는 새로운 멤버가 기존의 프로세스 그룹을 발견하고 멤버십을 수집하기 위한 계층이다.
- 단일캐스트 계층 : 단일캐스트 계층은 그룹의 멤버에게 단일캐스트 메시지를 송신하고 메시지의 수신자로부터 수신확인(ack) 응답 메시지를 기다린다. 일정시간(타임아웃) 동안 메시지를 수신하였다는 응답이 없으면, 메시지를 재전송 한다. 메시지를 재전송을 하였지만 여전히 수신확인 응답이 없으면 네트워크 단절이나 멤버 프로세스의 실패(crash)로 판단하고 상위 계층으로 멤버십 변경을 통보한다.
- 선입선출 순서화 계층 : 메시지 전달순서에 있어서 송신자의 선입선출(FIFO) 순서가 만족되어 상위 응용계층에 전달될 수 있도록 지원한다. 이 계층은 메시지에 대한 확인응답 기법을 사용하여 구현되었다.
- 인과 순서화 계층 : 메시지 전달순서에 있어서 송신자의 선입선출(FIFO) 순서를 보장하며 추가적으로 인과(causal) 관계에 있는 메시지의 전달순서를 보장하여 상위 응용계층에 전달될 수 있도록 지원한다.
- 전체 순서화 계층 : 그룹의 멤버들간에 통신하는 메시지들에 대하여 전체적으로 일관된 메시지 순서를 부여한다. 이를 위해 그룹에 참여하는 프로세스 멤버들을 대상으로 가상 링(virtual ring)을 형성하고, 가상 토큰을 순환시킨다. 토큰을 받은 멤버만이 새로운 메시지에 시퀀스 번호를 지정하고 멤버들에게 멀티캐스트 할 수 있다. 이를 통하여 전체적으로 유일한 일련의 메시지 전달 순서를 만든다.
- 안전 순서화 계층 : 이 계층은 메시지의 전체 순서화 요구사항을 만족시키며 추가적으로 프로세스 그룹에 참여하는 모든 멤버가 해당 메시지를 수신하였다고 확인된 경우에만 상위 응용 계층으로 전달되도록 보장한다. 안전 순서화 계층은 전체 순서화 계층과 마찬가지로 토큰 기반의 계층이다. 연속된 2번의 토큰 순환동안 모두 수신하였다고 확인된 메시지는 안전 순서(safe order) 전달의 요구사항을 만족하므로 상위계층으로 전달될 수 있다.
- 멤버십 관리 계층 : 프로세스 그룹에 참여하는 멤버

들에 대한 멤버십을 관리하는 계층이다. 멤버십 관리 계층은 하위 계층으로부터 전달받은 실패발견 정보를 기초로 새로운 멤버십을 형성하며, 새로운 멤버가 그룹에 참여하거나 탈퇴할 때 이를 하위 계층으로 통지하고 멤버십 알고리즘을 수행한다.

- 상태전이 계층 : 새로운 멤버가 그룹에 참여하거나 네트워크 단절로 상호통신 할 수 없었던 멤버가 네트워크 복원으로 다시 동일 그룹으로 참여할 때 프로세스 멤버들간의 상태정보를 동일하게 맞추기 위해 상태복원 알고리즘을 수행한다.
- 히스토리 계층 : 히스토리 계층은 JACE 시스템만의 특징으로 히스토리 모델을 지원하는 계층이다. 히스토리 모델은 프로세스 그룹에 참여하는 멤버들간의 통신하는 메시지를 그 성격에 따라 3가지로 구분하고 이러한 메시지들에 대해 일관된 처리를 보장한다. 멤버들간의 복제되는 정보는 처리되는 메시지에 의해 그 상태가 결정될 수 있다. 히스토리 모델은 멤버들간의 통신하는 메시지를 (1) 전달된 메시지의 집합과 전달순서에 프로세스의 상태가 결정되는 *history-sensitive* 메시지, (2) 최근 전달된 메시지에 의해 그 상태가 결정되는 *semi history-free* 메시지 그리고 (3) 프로세스의 상태를 변경하지 않는 *history-free* 메시지로 구분한다. JACE 시스템은 기존의 그룹통신 시스템과는 다르게 메시지의 유형을 3가지로 확장하여 구별한다. 이로써 JACE 시스템은 프로세스 그룹이 다수의 서브그룹으로 분할될 때 응용프로그램의 성격에 따라 분할된 서브그룹에서 지속적으로 수행할 수 있는 서비스의 형태를 다양화하였다.

3.2 프로토콜 계층간의 상호통신

JACE의 각 프로토콜 계층은 그림 2에서 보는 바와 같이 UpHandler Thread와 DownHandler Thread를 갖는다. 프로토콜 계층간의 상호통신은 이들 두 개의 Thread에 의해 이루어진다. 각각의 프로토콜 계층은 자신과 직접적으로 연결된 상위 프로토콜 계층과 하위 프로토콜 계층에 대한 참조(reference)를 가진다. 상위 프로토콜 계층에 대한 참조는 UpHandler Thread가 메시지를 상위로 전달하기 위해 사용되며, 하위 프로토콜 계층에 대한 참조는 DownHandler Thread가 메시지를 하위로 전달하기 위해 사용된다. 각 프로토콜 계층은 하위 프로토콜 계층으로부터 전달받은 메시지의 헤더 정보를 이용하여 메시지를 처리하고 상위 프로토콜 계층으로 전달될 필요가 있는 메시지는 UpQueue에 임시 보관한다. 각 프로토콜 계층은 상위 프로토콜 계층으로부터 전달받은 메시지에 해당 프로토콜 계층의 헤더 정보를 메시지에 추가한다. 헤더 정보가 추가된 메시지는 하위 프로토콜 계층으로 전달되기 위해 DownQueue에 임시 보관된다. UpQueue와 DownQueue 큐는 단일 판독자와 기록자를 위한 동기화 큐로 구현되었다.

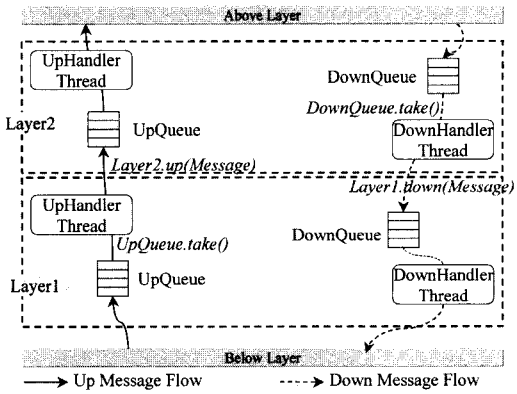

그림 2 JACE 프로토콜 Layer 상호간의 통신

4. JACE 서비스

4.1 메시지 순서화(Message ordering)

JACE 시스템은 프로세스 그룹에 참여하는 응용프로세스의 요구사항에 따라 다양한 메시지 전달방식을 지원한다.

- 선입선출(FIFO) 순서화 : 송신자의 메시지 전달순서가 보장된다. 응답확인 기법을 이용하여 구현되었다. 송신자의 선입선출 순서가 만족된 메시지는 상위 응용계층으로 전달된다.
- 인과(causal) 순서화 : 송신자의 선입선출(FIFO) 전달순서를 만족하면서 추가적으로 임의의 메시지에 대한 응답 메시지가 결코 선행 메시지보다 먼저 전달되지 않는 것을 보장한다. 메시지의 인과(causal) 관계를 표현하는 Vector Time 알고리즘을 사용하여 구현되었다.
- 전체(total) 순서화 : 그룹에 참여하는 모든 멤버에게 동일한 순서로 메시지가 전달된다. 그룹에 참여하는 멤버들을 대상으로 가상 링을 구성하고, 논리적인 토큰을 순환시킨다. 토큰을 전달받은 멤버만이 새로운 메시지에 일련번호를 지정하고 멀티캐스트 할 수 있다. 이러한 방법으로 그룹내의 전체 순서(total order)를 만든다.
- 안전(safe) 순서화 : 전체 순서화 요구사항을 만족하면서 그룹 내의 모든 멤버가 수신하였다는 확인이 있을 때 응용계층으로 메시지를 전달한다. 연속된 2번의 토큰순환동안 모든 멤버로부터 수신이 확인된 메시지는 안전 순서의 요구사항을 만족하므로 상위 계층으로 전달될 수 있다.

4.2 Extended Virtual Synchrony의 지원

VS 모델은 그룹의 멤버들간에 송수신 되는 일반 메시지와 그룹 멤버십 변경 메시지를 순서화(ordering) 한다. VS 모델은 프로세스 P, Q가 멤버십 V1에서 멤버십 V2로 진행해 나갈 때 그 사이에 발생한 메시지들을 동일한 순서로 응용 어플리케이션에 전달할 수 있도록 보장한다. 그러나 VS 모델을 지원하는 그룹통신 시스템은 응용프로세스가 실행되고 있는 호스트가 실패하거나 일시적으로 네트워크의 분할이 발생되어 그룹이 상호 통신할 수 없는 여러 서브그룹으로 분리된 후 재결합될 때 이들 사이에 일관성을 유지하지 못하는 문제점을 가지고 있다.

JACE 시스템은 EVS 모델을 지원한다. EVS 모델은 Isis 시스템의 VS 모델을 확장하여 분할된 모든 서브그룹에서 지속적인 실행을 지원한다. EVS는 일시적으로 네트워크가 분할되고 재결합되는 상황, 프로세스가 실패하고 재참여하는 상황에서도 그룹 멤버간에 일관성 있는 메시지의 전달과 멤버십 관리를 보장하기 위한 모델이다. JACE 시스템은 EVS 지원을 위해 시스템의 상태를 두 가지 유형의 뷰(View) 상태로 정의한다. 이러한 뷰 상태정보는 그룹에 참여하는 프로세스의 멤버십과 식별자(unique identifier)로 표현된다.

(1) 정규 뷰(Regular View) : 정규 뷰 상태에서는 클라이언트의 서비스 요청 메시지나 응용서비스의 응답 메시지와 같은 일반적인 메시지가 송수신된다.

(2) 과도기 뷰(Transitional View) : 과도기 뷰 상태는 네트워크의 단절로 인하여 그룹이 상호 통신할 수 없는 여러 분할영역으로 나뉘어지더라도 안전순서 메시지의 요구사항을 만족시키면서 메시지를 올바르게 전달하기 위한 것이다. 과도기 뷰가 포함하는 멤버십 정보는 이전의 정규 뷰 상태에서 새롭게 형성될 정규 뷰 상태로 옮겨가는 프로세스 멤버십 정보로 구성된다.

표 3의 뷰 변경 메시지는 가상 링을 구성하는 프로세스의 멤버십에 관한 정보를 포함한다. 뷰 변경 메시지는 정규 뷰 변경 메시지와 과도기 뷰 변경 메시지로 구분되며 이전의 정규 뷰에서 과도기 뷰로의 변경이나 과도기 뷰에서 새로운 정규 뷰로의 변경을 나타낸다. 히스토리 프로토콜 계층은 두 종류의 뷰 변경 메시지를 이용하여 최종적으로 그룹 멤버에 전달될 메시지의 순서를 결정한다. 멤버십관리 프로토콜 계층은 뷰 변경 메시지를 이용하여 그룹에 참여하는 프로세스의 멤버십을 관리한다. 뷰 변경 메시지의 구조는 표 3과 같다.

프로세스 그룹에 참여하는 멤버의 상태는 그림 3과 같이 6개의 상태로 정의된다.

- 초기 상태(init state, 그룹참여전 상태) : 프로세스 그룹에 참여하려는 멤버가 처음 구동되면 초기 상태에 놓이게 된다.
- 정규 상태(normal state) : 멤버십의 변화가 없으며 메시지 순서화 알고리즘을 수행한다.
- 멤버십 수집 상태(gather state) : 일정 시간동안 멤버십 정보를 교환한다.
- 재구성 상태(reformation state) : 새 링의 멤버십에 대한 합의를 시도하는 상태이다.
- 동의 상태(consensus state) : 새 링의 토큰 순환경로가 결정되며 새 링의 멤버십에 관한 정보가 교환되는 상태이다.
- EVS 상태(EVS state) : 정규 상태로 옮겨가기 전에 멤버들간의 메시지 복원과 새로운 멤버십을 인스톨하는 작업을 수행하는 상태이다.

네트워크 통신 링크의 장애로 프로세스 그룹이 상호 통신 할 수 없는 서브그룹으로 분할될 때 멤버십 관리 계층은 그림 3과 같이 다수의 서로 다른 상태로 이동한다. 그림 3에서 보여주는 각각의 단계는 최종적으로 멤버들간에 새로운 멤버십에 대한 동의를 구하고 멤버들간의 상태를 일관되게 유지하는데 있다. 정규 상태에서는 가상 링을 대상으로 논리적인 토큰을 순환되며 메시지 순서화 기능이 수행된다. 네트워크 통신 링크의 장애로 발생한 네트워크 분할은 순환되는 토큰의 분실로 발견될 수 있으며, 일정 시간동안 토큰을 수신하지 못한 각 프로세스는 멤버십 수집 상태로 이동한다. 멤버십 수집 상태에서 각 멤버는 일정시간 동안 자신이 알고 있

표 3 뷰 변경 메시지의 구조

regularViewID	정규 뷰 변경 메시지의 경우 : 새롭게 형성되는 정규 뷰의 ID 과도기 뷰 변경 메시지의 경우 : 이전 정규 뷰의 ID
msgSeqNumber	정규 뷰 변경 메시지의 경우 : 0 과도기 뷰 변경 메시지의 경우 : 이전 정규 뷰에서 전송된 메시지의 최대 일련번호
transitionalViewID	정규 뷰 변경 메시지의 경우 : 이전 과도기 뷰 ID 과도기 뷰 메시지의 경우 : 새롭게 형성될 정규 뷰 직전의 과도기 뷰 ID
membership	뷰 변경 메시지가 개시하는 뷰의 멤버십 정보
same_state_membs	현재 동일한 상태에 있는 멤버들에 대한 멤버십 정보를 포함한다. 상태전이 계층에서 상태 교환시 same_state_membs 멤버중 한 멤버만이 대표로 자신의 상태를 포함한 상태메시지를 전달한다.

는 멤버십 정보를 교환한다. 일정 시간동안 멤버십 정보를 교환한 후 재구성 상태로 이동한다. 재구성 상태에서 각 멤버는 새로운 멤버십에 대한 동의를 구하게 된다. 새로운 멤버십에 대해 동의가 이루어지면 가장 작은 프로세스 식별자를 갖는 멤버가 프로세스 그룹의 대표자가 되어 동의 상태로 이동한다. 새로운 멤버십이 형성되는 과정에서 동의 토큰은 새 링의 대표자에 의해 생성되어 새 링의 멤버십을 대상으로 두 번 순환한다. 동의 토큰의 첫 번째 순환에서는 제안된 새 링의 멤버들에 의해 동의 토큰이 포함하고 있는 정보들이 변경된다. 변경되는 정보로는 각 멤버들을 대상으로 이전의 링에 관한 정보 및 이전의 링에서 마지막으로 상위 어플리케이션 계층에 전달한 메시지의 시퀀스 번호 등의 정보가 동의 토큰에 추가된다. 두 번째 동의 토큰의 순환에서는 첫 순환에서 모아진 정보들을 각 멤버들이 획득하게 된다. 동의 토큰을 두 번 수신한 멤버는 EVS 상태로 이동되며 EVS 상태에서는 멤버들간의 일관된 상태를 유지하기 위해 EVS 알고리즘을 수행한다.

새로운 멤버가 기존 프로세스 그룹에 참여하거나 네트워크 분할과 재결합으로 멤버십에 변화가 발생할 때 멤버십 관리 계층은 새로운 멤버십에 대한 동의를 구한 후, EVS 상태에서 EVS 알고리즘을 수행한다. EVS 알고리즘을 수행하는 프로세스는 다음의 과정을 처리한다.

(1) 이전에 같은 링의 멤버였던 다른 프로세스들과 아직 전달하지 않은 이전 링에서의 멀티캐스트된 메시지에 대해 같은 메시지 집합을 유지하기 위해 메시지를 재 전송한다.

(2) 재전송 받은 메시지와 이전 링에서의 메시지들을 모두 포함하여 전체순서나 안전순서의 전달 요구사항에 따라 이전 링의 정규 뷰에서 전달될 수 있는 메시지들을 상위 계층에 전달한다.

(3) 과도기적 뷰를 시작하는 뷰 변경 메시지를 전달한다.

(4) 이전 링의 정규 뷰에서는 전체순서나 안전순서의 요구조건을 만족시키지 못해 상위 계층으로 전달될 수 없었던 메시지들에 대해 과도기적 뷰 안에서 전체순서나 안전순서의 요구조건을 만족시킨다면 이러한 메시지들을 상위 어플리케이션 계층에 전달한다.

(5) 새 정규 뷰를 시작하는 두 번째 뷰 변경 메시지를 전달한다.

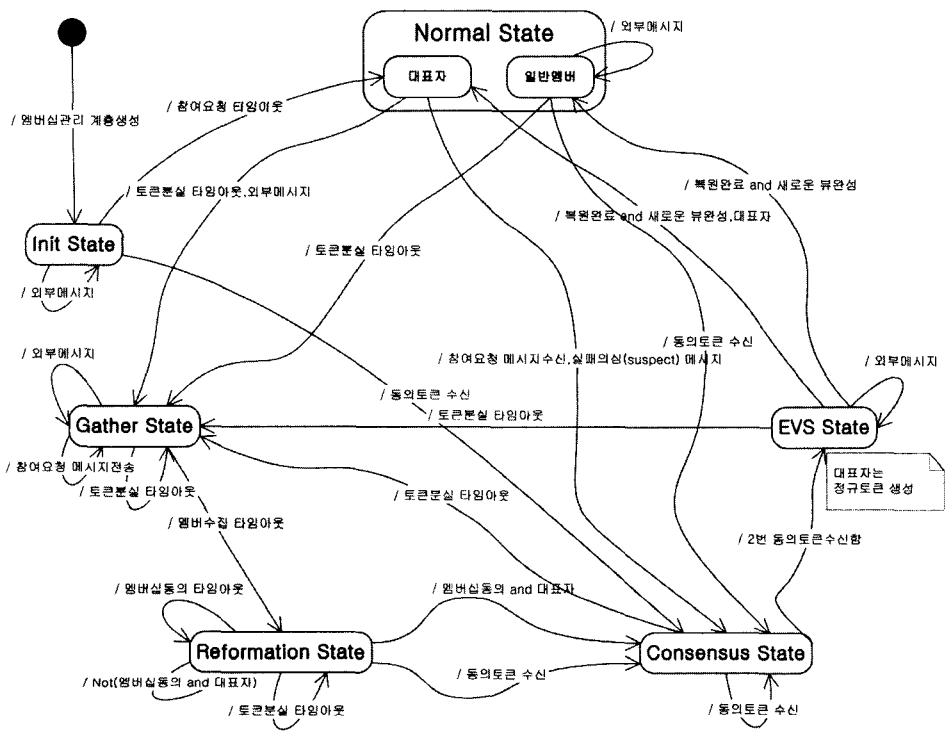

그림 3 멤버십 관리 계층의 상태도

(6) 정규 뷰 상태로 전환한다.

단계2에서 단계6까지는 다른 프로세스와의 통신에 관련되지 않고 지역적으로 수행되며 하나의 액션처럼 수행된다.

4.3 전체 순서화와 안전 순서화

전체 순서화는 그룹 내의 모든 메시지가 그룹 구성원 모두에게 동일한 순서로 전달되는 것을 요구한다. (즉, 프로세스 그룹의 멤버 P에서 메시지 m1을 메시지 m2보다 먼저 전달받았다면, 그룹 내의 다른 모든 멤버들도 이와 마찬가지로 m2 이전에 m1을 전달받는다.) 전체 순서화 프로토콜 계층과 안전 순서화 프로토콜 계층은 그룹에 참여하는 멤버들로 구성된 가상 링(ring)을 형성하고 가상 토큰을 순환시킨다. 토큰은 가상 링을 순환하며, 토큰을 전달받은 멤버만이 새로운 메시지에 순서를 부여하고 멀티캐스트 할 수 있다. 가상 링 상에서 멀티캐스트 되는 메시지는 토큰내의 정보를 이용하여 고유의 연속된 번호(sequence number)를 부여받는다. 토큰을 수신한 멤버는 자신이 가지고 있는 새로운 메시지를 멀티캐스트 하거나 다른 멤버가 재전송을 요청한 메시지에 대해 재전송을 수행한다.

토큰 기반의 순서화 프로토콜은 메시지의 순서화 기능을 제공하면서 동시에 프로세스 그룹에 참여하는 멤버십 관리를 수행할 수 있는 장점이 있다. 토큰 분실이 발생하게 되면 멤버의 실패나 네트워크의 분할로 간주하고 새로운 멤버십을 형성하기 위해 멤버십 관리 알고리즘을 수행시킨다. 토큰 기반의 순서화 프로토콜은 멤버가 자신이 참여하는 그룹의 멤버들에게 메시지를 멀티캐스트 하기 위해서는 토큰을 수신할 때까지의 지연시간이 있고 지속적으로 토큰을 순환시켜야 하는 오버헤드가 발생한다는 단점이 있다.

새로운 멤버는 프로세스 그룹에 참여하기 위해 LAN 상에 초기 멤버십 요청 메시지를 멀티캐스트 한다. 기존 그룹이 발견되지 않는다면 자신만을 멤버십으로 가상 링을 형성하게 된다. 기존 그룹이 발견된 경우에는 기존 그룹에 참여하는 멤버들에 대한 응답을 통해 초기 멤버십 정보를 일정 시간동안 수집한다. 새로운 멤버는 수집된 정보에서 기존 프로세스 그룹의 대표자 정보를 구하고 기존 대표자에게 그룹참여 요청 메시지를 전송한다. 그룹참여 요청을 수신한 대표자는 새로운 멤버를 포함한 멤버십을 형성하고 토큰을 순환시킨다.

가상 링을 순환하는 토큰은 다음의 정보를 포함한다.

- *type* : 정규 토큰 혹은 동은 토큰으로 지정될 수 있다. 정규 토큰은 일반 메시지의 순서화에 사용되며, 동의 토큰은 멤버십이 변경될 때 그룹내의 멤버들간에 메시지 복원 알고리즘 수행을 위해 사용된다.
- *tokenSeq* : 각 멤버는 토큰을 수신하고 다음의 멤버로 토큰을 전달하기 전에 이 필드의 값을 1 증가시킨다.
- *seq* : 가상 링 상에 멀티캐스트된 메시지들 중에서 최대 일련 번호
- *viewId* : 토큰이 순환하고 있는 가상 링의 식별자
- *aru* : 그룹내의 모든 멤버가 수신하였다고 알려진 메시지의 최대 일련번호, 안전 순서의 요구조건을 만족시키는지 확인하기 위해 필요하다.
- *retransmitRequestList* : 그룹내의 임의의 멤버에 의해 재전송이 요구되는 메시지에 대한 일련번호
- *aruId* : 토큰내의 seq 값보다 작은 값으로 aru 값을 지정한 멤버의 식별자
- *multicastCount* : 최근 1번의 토큰순환 동안 멀티캐스트된 메시지의 수
- *backlog* : 현재 큐에 보관되어 있으며 멀티캐스트될 필요가 있는 메시지의 수

프로세스 그룹에 참여하는 각 멤버는 프로토콜 스택에 전체 순서화 혹은 안전 순서화 프로토콜 계층을 포함할 수 있으며 이들 프로토콜 계층은 다음의 정보를 관리한다.

- *myAru* : 자신이 수신한 메시지들 중에서 전체 순서를 만족시키는 최대 메시지 일련번호
- *lastRoundTokenAruSeen* : 이전 토큰에서 구한 aru 값
- *newMessageQueue* : 멀티캐스트 될 필요가 있는 메시지를 임시 보관하는 큐
- *receivedMessageQueue* : 자신이 수신한 메시지를 보관하는 큐. 전체 순서를 만족하는 메시지만을 보관한다.
- *receivedPendingMessageQueue* : 자신이 수신한 메시지중 전체 순서를 만족하지 않는 메시지만을 보관하는 큐. receivedPendingMessageQueue에 보관된 메시지는 후에 전체 순서를 만족하게 되면 receivedMessageQueue로 이동된다.

안전 순서화 프로토콜 계층은 위의 정보 외에 추가적으로 다음의 정보를 관리한다.

- *myLastSafeDeliverdSeq* : 이전의 토큰순환에서 안전 순서의 요구조건을 만족시켜서 상위 계층으로 전달된 메시지의 최대 일련번호
- *discardUptill* : 자신을 포함한 모든 멤버가 discardUptill 필드의 값보다 작거나 같은 일련번호를 갖는 메시지를 모두 수신했음을 나타낸다.

연속된 2번의 토큰 순환동안 모든 그룹내의 멤버가 수신하였다고 확인된 메시지는 안전 순서의 요구조건을 만족한다. 메시지 전달서비스들 중에서 전체(total) 순서화와 안전(safe) 순서화 기능은 응용프로그램 개발시에

유용하게 이용될 수 있다. 안전 순서화는 기본적으로 전체 순서화를 요구하며 추가적으로 프로세스 그룹에 참여하는 모든 멤버가 수신하였다는 확인이 있은 후 상위 응용계층으로 메시지를 전달하게 된다.

JACE 시스템의 전체 순서화와 안전 순서화에 관한 처리성능은 10 Mb Ethernet 으로 연결된 4대의 PC (1.2 Ghz CPU, 256 MB) 컴퓨터를 이용하여 측정되었다. 그림 4은 한 호스트 상에서 그룹에 참여하는 멤버 수에 따라 전체 순서화와 안전 순서화의 처리 성능을 보여준다. 그림 4에 보여진 바와 같이 한 호스트상의 그룹에 참여하는 멤버수가 많아짐에 따라 성능이 저하되는 것을 볼 수 있다. 그림 5는 서로 다른 호스트 상에서 그룹에 참여하는 멤버 수에 따라 전체 순서화와 안전 순서화에 대한 처리 성능을 보여준다. 그림 5에서 보는 바와 같이 서로 다른 호스트 상에서 그룹에 참여하는 경우 멤버의 수에 상관없이 일정한 성능을 보여주고 있다. 성능측정에서 메시지의 크기는 1024 바이트를 사용하였다. 따라서 JACE 시스템의 경우 프로세스 그룹의 형성에 있어 한 호스트 상에 다수의 멤버가 그룹에 참여하는 것보다 서로 다른 호스트 상에 위치한 멤버들을 대상으로 프로세스 그룹을 형성하는 것이 보다 나은 메시지 처리성능을 제공할 수 있다.

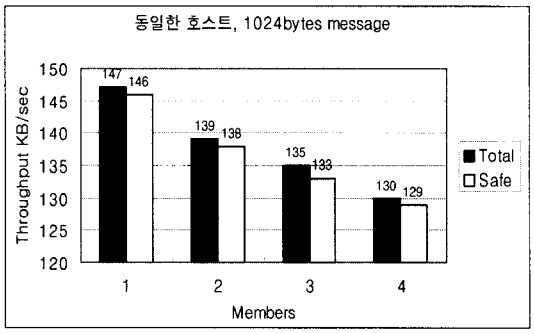

그림 4 1 호스트, 1024 bytes 메시지

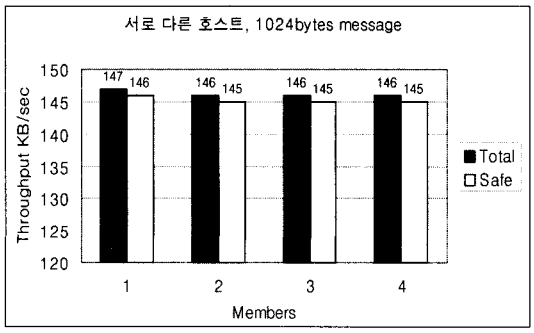

그림 5 4대 호스트, 1024 bytes 메시지

5. JACE를 이용한 UDDI 레지스트리

웹 서비스는 WWW의 차세대 기술로서 프로그램을 웹 사이트에 구축하는 방법을 통해 다른 응용들이 이를 접근, 이용할 수 있는 기능을 제공하는 분산 컴퓨팅 기술이다. UDDI(Universal Description, Discovery and Integration Service)는 웹 서비스를 위한 웹 기반 분산 레지스트리의 표준이며, 비즈니스들이 제공하는 웹 서비스를 등록, 검색, 발견하기 위한 메커니즘을 제공한다. UDDI.org는 UDDI에 관한 명세[13, 14]를 제안하였다. 또한 UDDI.org는 UDDI 서비스에 참여하는 개별 노드들의 레지스트리 변경을 모든 노드에서 인지하고, 노드간의 레지스트리 상태를 일관성 있게 유지시킬 목적으로 UDDI 복제 명세[15]를 제안하였다. UDDI 서비스는 다수의 노드에 복제되어 서비스를 제공할 때 더욱 유용하다.

JACE 시스템을 이용한 UDDI 2.0 레지스트리에 대하여 살펴본다. UDDI 서비스 그룹에 참여하는 멤버가 그룹통신 시스템을 이용하여 정보를 복제하는 처리 과정을 살펴본다(그림 6의 단계 ①에서 ⑧단계).

단계 ① : UDDI 레지스트와 상호작용하는 클라이언트는 UDDI API를 이용하기 위해 UDDI 스키마에 따른 SOAP 메시지를 작성해야 한다. SOAP 메시지는 IBM UDDI4J와 같은 클라이언트 패키지를 통해 생성할 수 있으며 서버쪽 서블릿 RPCRouter로 SOAP 메시지를 전달한다.

단계 ② : RPCRouter에서 클라이언트에서 보낸 SOAP 메시지에서 API 이름과 API에 따른 파라미터 정보들을 추출해 UddiService로 전달한다. 즉, RPC-Router는 호출될 UDDI API의 종류를 분별하기 위한 선처리 프로세서의 역할을 수행하며 요청 서비스에 대한 응답 메시지를 생성해 다시 클라이언트로 보내는 일종의 라우터 역할을 담당하게 된다.

단계 ③-④ : RPCRouter는 클라이언트의 SOAP 메시지를 JACE를 통하여 멀티캐스트한다.

단계 ⑤-⑥ : 다른 호스트 상에서 동작하는 JACE 시스템은 멀티캐스트 메시지를 수신하고, 자신과 연결된 RPCRouter로 SOAP 메시지를 전달한다.

단계 ⑦-⑧ : 호스트B의 RPCRouter는 수신한 SOAP 메시지에서 API 이름과 API에 따른 파라미터 정보들을 추출해 UddiService로 전달한다. 이를 통해 UDDI 개체가 복제된다.

그림 7은 클라이언트의 요청을 수신하고 이를 UddiService 전달하며 그룹통신을 이용하여 그룹에 참여하는 다른 멤버에게 메시지를 전달하는 RPCRouter 서블릿 클래스를 보여준다. 응용서비스의 신뢰성은 그룹

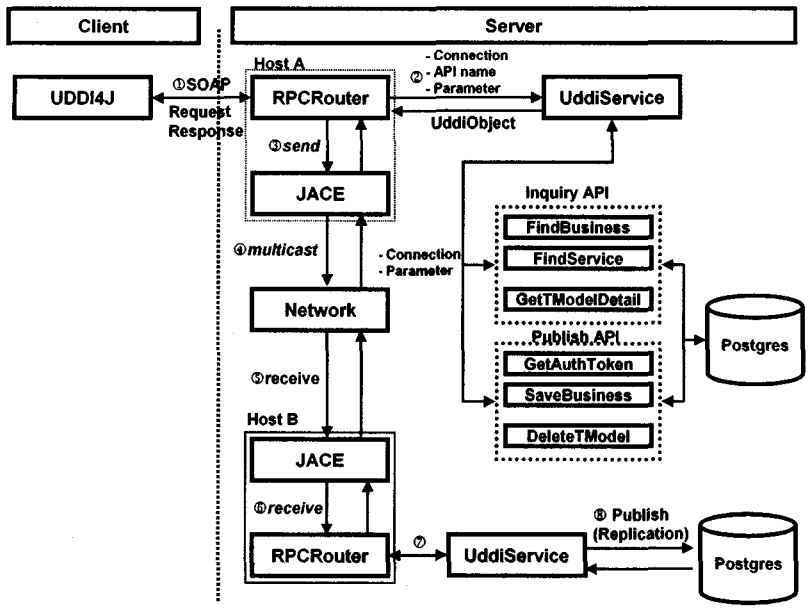

그림 6 그룹통신을 이용한 UDDI 레지스트리간의 통신

```
public class RPCRouter extends HttpServlet implements MessageListener, MembershipListener {

    private String                mGroupName="MyUDDI";
    private GroupCommunication    mGC = null;
    private ReceiverThread        mReceiverThread = null;
    private DOMBuilder            mBuilder = null;
    private DOMOutputter          mOutputter = null;
    private String mGCProperty="COM:FIND:FD:UNICAST:SAFE:MEMBERSHIP";
    public void init(ServletConfig config) throws ServletException {
          super.init(config);
       try{

          (1) mGC = new GroupCommunication( mGCProperty );
          (2) mGC.connect( mGroupName );
          } // ...
          mReceiverThread = new ReceiverThread();
          mReceiverThread.start();
    }
    public void processRequest(String anApiName, UddiObject aParam){
        org.w3c.dom.Document doc = aParam.getDocument();
        org.jdom.Document jdoc = mBuilder.build( doc );
        GroupUDDIRequest uddiReq = new GroupUDDIRequest(anApiName, jdoc);
        try{
          (3) mGC.send(new Message(null, null, uddiReq));
          }catch(Exception e){ System.err.println(e); }
    }
    public void doPost (HttpServletRequest req, HttpServletResponse res)
                                                throws ServletException, IOException {
        // ...
        UddiService uService = new UddiService( con );
        UddiObject obj = uService.invokeAppropriateApi( apiName, param);
        con.close();
```

```
        processRequest( apiName, param );
    }
    public void receive( Message aMsg ) { /* ..... */ }
    public void viewChanged( View aView ) { /* ..... */ }
}
```

그림 7 RPCRouter class

UDDI 2.0 레지스트리의 구현은 JDK 1.4 개발환경, Tomcat 4.0.6 웹 서버, UDDI 개체의 저장과 관리를 위한 PostgreSQL를 이용하였다. 클라이언트와 서버간의 SOAP 메시지 통신에서 클라이언트의 요청 메시지는 IBM에서 개발된 UDDI4J[16]를 사용하여 작성하고, 클라이언트와 서버간의 SOAP 메시지 전송은 Apache SOAP 2.2 를 사용하였으며, JACE 시스템을 통하여 다른 노드의 멤버에게 XML 요청을 전달하기 위해 JDOM-b8[17]이 사용되었다.

6. 결 론

응용서비스를 다수의 프로세스에 복제하여 제공함으로써 가용성을 높이고 특정 프로세스가 실패하더라도 안정적이고 지속적인 서비스를 제공할 수 있다. 하지만, 통신 네트워크의 일시적인 단절과 프로세스의 실패가 발생하는 상황에서 프로세스들간에 복제되는 서비스를 일관성 있게 유지하는 분산 어플리케이션 개발은 쉽지 않다. 그룹통신 시스템은 프로세스 그룹을 지원하며 그룹멤버들간의 일관성을 보장한다.

EVS 모델은 VS 모델을 확장하여 네트워크 분할과 재결합이 발생되더라도 그룹 멤버들간의 메시지 전달을 일관성 있게 보장한다. 본 논문에서는 응용서비스의 요구에 따른 다양한 메시지 전달방식과 EVS 모델을 지원하는 JACE 그룹통신 시스템에 대하여 기술하였다. JACE 시스템은 다수의 프로토콜 계층으로 구성되었다. 이러한 프로토콜 계층에는 UDP 메시지를 전달하는 전송계층, 프로세스 그룹의 멤버십을 수집하기 위한 멤버십 수집 계층, 신뢰성 있는 메시지의 전달을 위한 단일 캐스트 계층, 메시지 전달서비스에 관련된 선입선출 순서화 계층, 인과 순서화 계층, 전체 순서화 계층 그리고 안전 순서화 계층 등이 있다. 또한 프로세스 그룹의 멤버십 관리를 위한 멤버십관리 계층, 서브그룹들이 병합되었을 때 상태교환을 위한 상태전이 계층 그리고 프로세스의 복제되는 정보의 특성에 따라 그에 적합한 3가지 유형의 메시지를 지원하는 히스토리 계층 등이 있다. 개발자는 응용서버의 특성에 따라 이들 그룹통신 계층을 조합함으로써 원하는 서비스의 질을 제공하는 그룹통신을 이용할 수 있다.

JACE 그룹통신 시스템의 특징은 다음과 같다.

(1) JACE 시스템은 특정 플랫폼에 독립적으로 실행될 수 있다.

(2) 다양한 분야에서 제공되는 자바 응용서비스의 견고성을 지원하는 그룹통신 시스템이다.

(3) 히스토리 계층을 지원한다.
(*history-sensitive* 메시지, *semi history-free* 메시지, *history-free* 메시지)

(4) Extended Virtual Synchrony 모델을 지원하다.

(5) 다양한 메시지 전달방식을 지원한다. (선입선출, 인과, 전체, 안전순서 전달방식)

(6) 개발자는 응용서비스의 요구사항에 따라 JACE 프로토콜 스택을 구성할 수 있고, 새로운 프로토콜 계층을 추가함으로써 쉽게 JACE 시스템을 확장할 수 있다.

개발된 JACE 시스템을 이용하여 웹 서비스에 대한 정보를 등록하고, 발견할 수 있는 UDDI 레지스트리(registry)가 구현되었다. JACE 시스템을 이용한 UDDI 서비스는 UDDI 서비스에 참여하는 개별 노드들의 레지스트리 변경을 모든 노드에서 실시간에 인지하고, 노드간의 레지스트리 상태를 일관성 있게 유지한다. 현재 JACE 시스템은 단일 LAN 환경에서 프로세스 그룹의 형성을 지원하고 있으며 향후 인터넷 환경으로 확장할 계획이다.

참 고 문 헌

[1] Kenneth P. Birman, *Building Secure and Reliable Network Applications*, Manning Publications Co., 1996.

[2] Kenneth P. Birman and Robbert van Renesse, *Reliable Distributed Computing with the ISIS Toolkit*, Los Alamitos, CA., IEEE Computer Society Press, 1994.

[3] Robbert van Renesse, Kenneth P. Birman, and Silvano Maffeis, *Horus : A Flexible Group Communication System*, Communications of the ACM, Vol. 39, No. 4, pp.75~83, 1996.

[4] Amir, Y., Dolev, D., Kramer, S., and Malki, D., *Membership algorithms in broadcast domains*, In Proceedings of the 6th International Workshop on Distributed Algorithms(Haifa, Israel), Springer-Verlag, Berlin, Germany, pp.292~312, 1992.

[5] Amir, Y., Dolev, D., Kramer, S., and Malki, D.,

Transis: A communication subsystem for high availability, In Proceedings of the IEEE 22nd Annual International Symposium on Fault-Tolerant Computing(Boston, MA), IEEE, New York, pp.76~84, 1992.

[6] Yair Amir, Louise E. Moser, P. Michael Melliar-Smith, Deborah A. Agarwal and Paul Ciarfella, *The Totem Single-Ring Ordering and Membership Protocol*, ACM Transactions On Computer Systems, 13(4), pp.311~342, November 1995.

[7] Bela Ban, *Design and Implementation of a Reliable Group Communication Toolkit for Java.*

[8] Alberto Montresor, *The Jgroup Reliable Distributed Object Model*, In Proceedings of the Second IFIP WG 6.1 International Working Conference on Distributed Applications and Interoperable Systems(Dais '99), Helsinki, Finland, June 1999. Also appears as Technical Report UBLCS 1998-12, December 1998.

[9] K. Birman and T. Joseph, *Exploiting Virtual Synchrony in Distributed Systems*, In Proceeding of the ACM Symposium on Operating Systems Principles, pp.123~138, November 1987.

[10] K. P. Birman, *Virtual Synchrony Model*, In Reliable Distributed Computing with the Isis Toolkit, IEEE press.

[11] L. E. Moser, Y. Amir, P. M. Melliar-Smith and D. A. Agarwal, *Extended Virtual Synchrony*, In Proceeding of the 14th International Conference on Distributed Computing Systems, pp.56~65, June 1994.

[12] O. Babaoglu, A. Bartoli and G. Dini, *Enriched View Synchrony: A Paradigm for Programming Dependable Applications in Partitionable Distributed Systems*, In IEEE Transactions on Computers, 46(6), pp.642~658, June 1997.

[13] UDDI Version 2.04 API, Published Specification, Dated 19 July 2002.

[14] UDDI Version 2.03, Data Structure Reference, Published Specification, Dated 19 July 2002.

[15] UDDI Version 2.03, Replication Specification, Published Specification, Dated 19 July 2002.

[16] http://oss.software.ibm.com/developerworks/projects/uddi4j

[17] http://www.jdom.org/

문 남 두

1997년 2월 울산대학교 전자계산학과 졸업(학사). 1999년 2월~울산대학교 컴퓨터정보통신 공학과 졸업(공학석사). 2003년 8월~울산대학교 컴퓨터정보통신 공학과 졸업(공학박사). 관심분야는 그룹통신 시스템, 분산객체, CSCW, 모바일 프로그래밍 등

이 명 준

1980년 2월 서울대학교 수학과 졸업(학사). 1982년 2월 한국과학기술원 전산학과 졸업(석사). 1991년 8월 한국과학기술원 전산학과 졸업(박사). 1982년 3월~현 울산대학교 컴퓨터정보통신 공학부(교수) 1993년 8월~1994년 7월 미국 버지니아대학 교환교수. 관심분야는 프로그래밍언어, 분산 객체 프로그래밍 시스템, 병행 실시간 컴퓨팅, 인터넷 프로그래밍시스템, 바이오인포매틱스 등