에이전트 기반의 고장허용 객체그룹 모델 구축

정회원 강 명 석*, 김 학 배*

Construction of an Agent-based Fault-Tolerant Object Group Model

Myungseok Kang*, Hagbae Kim* ***Regular Members***

요 약

본 논문에서는 효율적인 객체 관리와 고장 회복을 위해 에이전트 기술과 중복 메커니즘을 이용한 고장허용 객체그룹(Fault Tolerant Object Group, FTOG)를 기반으로 에이전트 기반의 고장허용 객체그룹 모델을 제안한다. 고장허용 객체그룹의 확장된 기능으로 다섯 가지의 에이전트 - 내부처리 에이전트, 등록 에이전트, 상태처리 에이전트, 사용자인터페이스 에이전트, 서비스 에이전트를 정의하였다. 제안된 모델에서 에이전트들의 역할은 분산된 객체들의 상호작용을 줄이고 보다 효과적인 서비스를 제공하는데 있다. 에이전트 기반의 고장허용 객체그룹 모델의 효율성을 검증하기 위해 홈네트워크 서비스를 제공하는 가상의 지능형 홈네트워크 시뮬레이터를 구현하였다. 시뮬레이션을 통하여 제안한 모델은 객체들 간의 상호작용을 줄이고(부하감소) 효율적인 고장 회복 등 안정적이고 신뢰성 있는 서비스를 제공함을 검증하였다.

Key Words : Fault Tolerant Object Group, Agent, Intelligent Object Management Services

ABSTRACT

We propose an Agent-based Fault Tolerant Object Group model based on the agent technology and FTOG model with replication mechanism for effective object management and fault recovery. We define the five kind of agents - internal processing agent, registration agent, state handling agent, user interface agent, and service agent - that extend the functions of the FTOG model. The roles of the agents in the proposed model are to reduce the remote interactions between distributed objects and provide more effective service execution. To verify the effectiveness of the proposed model, we implemented the Intelligent Home Network Simulator (IHNS) which virtually provides general home networking services. Through the simulations, it is validated that the proposed model decreases the interactions of the object components and supports the effective fault recovery, while providing more stable and reliable services.

Ⅰ. 서 론

오늘날 컴퓨팅 환경과 정보 시스템은 끊임없는 발전을 거듭하며 과거 중앙 집중 처리 방식의 환경에서 네트워크 기반의 분산 처리 시스템 환경으로 변화하고 있다. 또한 디지털 가전의 보급으로 네트워크 환경이 정보가전 기기로 확산되면서 홈네트워크[1-2]와 같은 새로운 분야에 대한 관심이 높아지고 있다. 이러한 시스템에서는 분산 환경에 기초한 서비스들이 대두되어 원하는 서비스의 목적에 따라 환경 변화를 인지하고 계획을 세워 그 결과를 사용자에게 제시해줄 수 있는 새로운 인터페이스 기술

* 연세대학교 전기전자공학과 디지털정보처리 연구실(mskang@yonsei.ac.kr), (hbkim@yonsei.ac.kr)
논문번호 : KICS2008-01-047, 접수일자 : 2008년 1월 25일, 최종논문접수일자 : 2008년 11월 27일

을 필요로 한다. 이러한 인터페이스를 일반적으로 에이전트(Agent)라고 부른다. 다양한 분야에서 여러 가지 응용 에이전트들이 개발되고 있는데 그중에서도 이동 에이전트에 대한 관심이 커져가고 있다. 이동 에이전트는 이질적인 망에서 자신의 제어로 호스트를 옮겨 다니며, 다른 호스트의 에이전트 서버나 에이전트와 상호 작용하면서 사용자의 작업을 수행하는 프로그램이다. 이동 에이전트 기술은 1994년 이후 중앙 집중형 관리 방식의 단점을 극복하기 위한 모델로서 급속히 발전을 시작하여, Telescript, Aglet Workbench, Odyssey, Agent-Tcl 등 많은 이동 에이전트 시스템[3-4]이 개발되었다.

에이전트 기술과 함께 분산 처리 환경에서는 시스템을 구성하는 많은 객체들을 효율적으로 관리 할 수 있는 객체관리 기법들이 제공되어야 한다. 이를 위해 분산 객체 컴포넌트를 이용한 시스템 모델에 관한 연구가 90년대 후반이후 OMG(Object Management Group)의 CORBA 객체 모델, TINA-C의 TINA 객체 모델, Microsoft의 DCOM (Distributed Component Object Model) 등을 통해 이루어져 왔다[5-7]. 또한 최근 유비쿼터스 환경에서는 위의 객체관리 기술을 기반으로 Gaia, Aura, RCSM 등의 연구가 진행 중이다[8-13]. 그러나 이와 같은 모델들은 하부 플랫폼에 종속적이며, 사용자에게 신뢰성과 투명성을 위한 고장발견 및 회복기술이 부재하고, 자율적인 상호운용성이 부족한 제약사항들을 가지고 있다.

이를 해결하기 위해 본 연구에서는 특정 ORB(Object Request Broker)에 종속되지 않고 분산 객체 관리의 편의성과 일관성을 제공하고, 객체의 중복 기법을 통하여 고장 발생 시 신뢰성과 가용성을 높이는 기법과 에이전트 기법을 사용하여 컴퓨팅 객체 간 자율적이고 효율적인 상호운용성 등의 장점을 가지는 에이전트 기반의 고장허용 객체그룹 모델을 제안한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 분산 컴퓨팅 환경에서 제안된 관련연구를 기술하며, Ⅲ장에서는 에이전트 기반의 고장허용 객체그룹 모델을 정의하고 설계한다. Ⅳ장에서는 제안한 모델을 적용하여 가상의 홈네트워크 환경을 구성하고 시뮬레이션 결과를 분석한다. 마지막으로 Ⅴ장에서는 연구에 대한 결론을 맺는다.

Ⅱ. 관련 연구

최근 분산 컴퓨팅 환경에서 주요 서비스 기능들을 구현하고, 다양한 서비스들을 사용자의 간섭을 최소화하면서 효율적으로 관리/사용할 수 있는 기술들이 연구되고 있다. 본 연구에서는 두 개의 대표적인 연구(Gaia, RCSM)와 본 모델을 위해 개발하였던 고장허용 객체그룹(FTOG)[13-16]에 대해 살펴본다.

2.1 Gaia와 RCSM

일리노이 대학 어바나 샴페인 캠퍼스에서 개발한 Gaia는 인간이 생활하는 물리적인 공간에 필요한 정보를 요청하고 이용 가능한 다양한 리소스를 통해 원하는 작업을 쉽게 수행할 수 있는 공간인 액티브 공간을 실현하는 것을 목표로 하고 있다. Gaia 커널은 컴포넌트 관리 코어와 5가지의 기본 서비스(이벤트 관리, 컨텍스트 서비스, presence 서비스, space repository, 컨텍스트 파일 시스템)들을 제공한다. 이들 기본 서비스들과 응용은 컴포넌트 기반 분산 객체로 이루어져 있다. 컴포넌트 관리 코어는 이러한 Gaia 컴포넌트들과 애플리케이션을 동적으로 시스템에 로드, 언로드하거나 전송, 생성, 삭제하는 역할을 담당하며 CORBA를 근간으로 하고 있다. 이벤트 관리자는 CORBA 이벤트 서비스에 기초하여 액티브 공간상에서 발생하는 이벤트들을 다른 컴포넌트에 분배하고, 컨텍스트 서비스는 응용이 특정 컨텍스트를 등록할 수 있도록 한다. 새로운 자원이나 사람이 시스템에 추가되거나 삭제되면 presence 서비스는 이를 인식하고 이벤트를 발생시킨다. 이러한 이벤트에는 인식된 자원이나 사람에 대한 정보를 포함하고 있다. presence 서비스에 의해 발생되는 이벤트는 space repository 서비스가 받게 되며 인식된 개체에 대한 설명을 담은 XML 정보를 저장관리한다. 컨텍스트 파일 시스템은 개인적인 데이터 정보를 사용자의 존재 여부에 따라 자동으로 마운트/언마운트 하여 응용에 이용 가능하도록 해준다.

Gaia는 CORBA를 기반으로 하는 모델의 특징을 가지고 있다. 물론 컨텍스트를 사용하여 지능적인 서비스를 제공하는 장점을 가지고 있지만 특정 하부구조에 독립적이지 못하며, 자유로운 확장성에서 많은 제약사항을 가지고 있다.

RCSM(Reconfigurable Context-Sensitive Middleware)은 애리조나 주립대학에서 연구 개발되고 있으며 컨텍스트와 서비스와의 능동적 결합을 주요 목표로 하고 있다. RCSM은 객체지향 개발 환경을 지향하고 있으며 CORBA와 매우 밀접한 관계를 가지고 있다. RCSM 응용들은 컨텍스트 적응형 객체들로 이루어

지는데 이러한 객체들은 컨텍스트 적응형 인터페이스 부분과 서비스 구현 부분으로 구분된다. 컨텍스트 적응형 인터페이스는 컨텍스트와 이에 적용될 메소드의 조합으로 구성되며 컨텍스트 인지 인터페이스 정의 언어로 기술된다. 이는 인터페이스 컴파일러에 의해 컴파일 되어 적응형 객체 컨테이너(ADC)로 만들어진다. ADC는 획득된 컨텍스트 정보 중 관심 있는 컨텍스트를 발견하면 객체의 메소드를 실행한다. 또한 ORB를 수정하여 컨텍스트 적응형 RCSM ORB로 확장하여 응용 소프트웨어에 투명한 통신 환경을 제공하고 객체들을 대신하여 디바이스 및 서비스의 검색을 수행한다. RCSM은 컨텍스트 적응형 객체 뿐만 아니라 기존의 CORBA, COM 객체들을 함께 혼용하여 사용할 수 있는 구조를 가지고 있다.

그러나 RCSM은 특정 ORB에 제한적이지 않고, 지능적인 서비스를 제공하지만 서비스의 안정성과 지속성을 위한 고장회복에 대한 방안들이 없다.

2.2 고장허용 객체그룹(Fault Tolerant Object Group, FTOG)

고장허용 객체그룹[13]은 분산되어 있는 객체들을 관리하고 객체들의 고장 발생시 이를 회복하고 지속적인 서비스를 제공하기 위한 모델로, 임의의 서비스를 수행하기 위한 객체들의 논리적인 집합이다. 그림 1과 같이 고장허용 객체그룹은 객체그룹관리자, 보안객체, 정보저장소, 중복관리자, 고장발견자 등의 구성요소로 구현될 수 있다. 이러한 구성요소들은 다음과 같이 설명된다. 객체그룹관리자(Object Group Manager, OGM)는 고장허용 객체그룹의 전반적인 관리를 책임지며 객체그룹과 사용자 간의 인터페이스를 제공한다. 또한 임의의 객체를 특정 고장허용 객체그룹에 등록시키거나 탈퇴시키는 기능을 가지며, 보안객체(Security Object, SeO)에게 객체 접근권한 검사를 요청하고, 접근이 허가된 요청에 대해 정보저장소(Information Repository Object, IRO)에 서비스를 수행할 객체 레퍼런스를 요청한다. 보안객체는 사용자 및 서비스 객체의 접근 권한을 검사한다. 정보저장소는 서비스명 및 레퍼런스 정보 등 서비스를 수행할 고장허용 객체그룹의 정보가 저장된 객체리스트를 가진다. 또한 객체그룹관리자로부터 새로운 그룹 등록이나 탈퇴 요청을 받아 정보를 갱신한다. 고장발견자(Fault Detector, FD)는 주기적으로 서비스 객체의 상태를 판단한다. 서비스 객체의 주기적인 신호에 대한 응답을 이용하여 고장을 찾고, 중복관리자에 이를 보고하는 기능을 갖

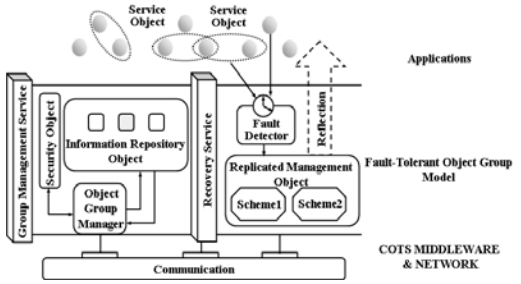


그림 1. 고장허용 객체그룹 구조

는다. 서비스 객체의 상태정보 전달 주기 동안 응답하지 않은 서비스 객체는 오류가 있다고 판단하게 된다. 중복관리자(Replication Management Object, RMO)는 사용자의 서비스 요청에 대하여 중복된 서비스 객체(Service Object, SO)에 메시지를 보낸다. 또한 다른 사용자들의 서비스 요청에 대한 스케줄링을 제공하며 서비스 객체의 실행결과를 사용자에게 보낸다.

고장허용 객체그룹은 기본적으로 두 가지 서비스를 제공한다. 하나는 객체 관리의 편의성을 제공하기 위한 그룹관리 서비스(Group Management Service)이다. 그룹관리 서비스는 객체의 등록, 레퍼런스 정보 변경 등 고장허용 객체그룹의 관리를 수행하며 객체 관리에 대한 일관성을 제공한다. 다른 하나는 중복객체를 사용하여 고장이 발생한 경우에도 지속적으로 서비스를 수행할 수 있는 회복 서비스(Recovery Service)이다. 회복 서비스를 위해 두 가지의 고장회복 방안을 제공한다. 첫번째로 사용자의 서비스 요청에 대하여 하나의 서비스 객체를 동작시키며 고장이 발생한 경우 대기(Standby) 상태의 다른 서비스 객체를 동작시키는 방법이다. 두번째 방안은 첫번째 방안에 체크포인트(Checkpoint) 방법을 첨가한 것이다.

고장허용 객체그룹은 분산환경에서 분산되어 있는 객체를 관리하고 안정적으로 지속적인 서비스를 제공하는 점에서 장점을 가지고 있지만 객체간의 빈번한 상호작용으로 인해 많은 부하와 효율성 저하를 초래하기 쉽다. 따라서 본 논문에서는 에이전트 메커니즘을 사용하여 에이전트 기반의 고장허용 객체그룹을 제안한다.

Ⅲ. 에이전트 기반의 고장허용 객체그룹 모델

본 장에서는 에이전트 기반의 고장허용 객체그룹 모델을 설명한다. 분산 네트워크 환경은 불확실성

하에서 예측하지 못한 여러 가지 문제를 만날 수 있고, 이로 인하여 사용자가 요청한 서비스 및 작업을 수행 할 수 없게 되거나, 불필요한 자원 소모를 가져올 수 있다. 이러한 상황에 부딪쳤을 때 에이전트 기반의 고장허용 객체그룹은 높은 효율성을 가질 수 있다. 또한 객체간 빈번한 원격 통신으로 인한 객체 관리 부하를 감소시킬 수 있다. 본 모델은 에이전트의 자율성과 이동성을 이용하여 원격지의 객체를 등록하고 상태를 감시하며 그 결과를 보여주게 된다.

3.1 고장허용 객체그룹에서 에이전트 생성

고장허용 객체그룹에서 에이전트 기반의 서비스를 제공하기 위해 다섯 가지의 에이전트를 정의한다. 사용자는 사용자인터페이스 에이전트를 이용하여 사용자 등록 및 서비스를 요청할 수 있다. 서비스 객체는 등록을 위하여 에이전트를 생성하고 이를 OGM에 전달하여 고장허용 객체그룹에 등록한다. FD는 주기적으로 메시지를 전달하는 대신 상태처리 에이전트를 생성하고 서비스 객체를 차례로 방문하여 상태를 감시하게 된다. RMO는 서비스 에이전트를 이용하여 서비스 객체에 작업을 할당하고 실행시킨다. 또한 고장허용 객체그룹 구성요소 간의 메시지 처리는 내부처리 에이전트가 담당하게 된다. 표 1은 각 에이전트에 대한 오퍼레이션을 나타낸다.

표 1. 에이전트 오퍼레이션

Agent Class	Description
내부처리 에이전트 (Internal Processing Agent)	에이전트 기반 고장허용 객체그룹의 구성 요소 간 메시지 전달 및 처리
등록 에이전트 (Registration Agent)	객체의 등록, 탈퇴 등 그룹관리 서비스 제공
상태처리 에이전트 (State Handling Agent)	객체의 고장 검사, 서비스 상태 전달 등 회복 서비스 제공
사용자인터페이스 에이전트 (User Interface Agent)	사용자와 에이전트 기반의 고장허용 객체그룹 간의 인터페이스 제공
서비스 에이전트 (Service Agent)	서비스 객체로 사용자 요청 전달 및 실행을 담당

3.1.1 내부처리 에이전트

객체의 인증, 상태 보고 등 고장허용 객체그룹 구성요소 간의 프로세싱을 위한 에이전트이다. 기본적으로 고장허용 객체그룹의 각 구성요소들로부터 메시지를 전달받아 이를 실행하는 요소에 전달하고 그 결과를 되돌리는 구조로 되어 있다. 인증 요청, 객체 상태 보고, 레퍼런스 정보 전달 등이 이에 해당한다.

3.1.2 등록 에이전트

고장허용 객체그룹에 등록하기 위하여 서비스 객체에서 생성되는 이동 에이전트이다. 생성되는 등록 에이전트에는 포트 번호 및 객체의 IP 등 물리적 주소 정보와 객체 이름, 제공되는 서비스, 등록되어질 가상의 그룹 정보 등이 포함된다. 생성된 에이전트는 객체를 이동하여 OGM에 서비스 객체의 정보를 전달한다.

3.1.3 상태처리 에이전트

서비스 객체의 상태 검사, 서비스 상태 전달을 위하여 FD에서 생성되는 이동 에이전트이다. 생성되는 에이전트는 거쳐야 할 서비스 객체의 위치 정보 및 체크포인트 방법을 위한 각 서비스 객체의 실행 상태를 저장할 저장소를 가지고 있다.

3.1.4 사용자인터페이스 에이전트

사용자의 등록, 서비스 요청 등 사용자와 고장허용 객체그룹 간의 메시지 전달 및 사용자의 편의성을 위해 제공되는 에이전트이다.

3.1.5 서비스 에이전트

사용자의 서비스 요청에 대하여 이를 서비스 객체로 전달하기 위한 에이전트이다. 여러 서비스를 통합적으로 제공할 수 있는 구조를 가지고 있다.

3.2 에이전트 기반 고장허용 객체그룹 모델 구조

에이전트를 고장허용 객체그룹 모델에 적용할 경우, 본 장에서 설명한 것처럼 여러 가지의 이득을 얻을 수 있다. 그림 2는 에이전트 기반의 고장허용 객체그룹 모델의 구조를 나타낸다. 그림에서 볼 수

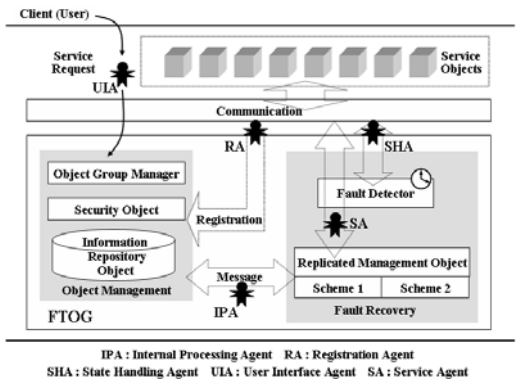


그림 2. 에이전트 기반 고장허용 객체그룹 모델 구조

있듯이 고장허용 객체그룹에서 객체의 관리 및 상태 감시 등을 에이전트가 수행하는 구조를 가지게 된다. 이 경우 작업이 고장허용 객체그룹 구성요소를 주체로 하여 객체간의 통신을 통하여 이루어지는 것이 아니라 에이전트가 이동하여 해당 객체에서 수행되므로, 트래픽 양 감소 및 고장허용 객체그룹 구성요소의 작업 부하를 줄일 수 있다.

사용자와 고장허용 객체그룹 간에는 사용자인터페이스 에이전트가 존재하여 서비스 요청에 대한 편의성을 제공하는 구조를 가지고 있다. RMO는 서비스 에이전트를 이용하여 요청된 서비스에 대한 작업을 서비스 객체로 전달한다. 요청된 서비스의 형태는 단일 작업을 수행할 수도 있고, 여러 작업을 묶어 통합 서비스 형태로 제공할 수도 있다. 통합 서비스가 제공되어야 하는 경우, 기존 모델에서는 RMO에서 서비스 객체로 작업 메시지를 계속하여 전달해야 한다. 이 경우 개별적인 작업 전달 대신 필요한 작업들을 담아 서비스 에이전트를 사용하여 전달하면 메시지 수를 감소시킬 수 있다. 또한 상태처리 에이전트를 활용하여 서비스 객체의 상태 검사를 수행하며 고장허용 객체그룹의 구성요소 간 메시지 전달 및 처리는 내부처리 에이전트가 담당한다. 표 2는 구성요소 간의 통신을 위하여 내부처리 에이전트로 전달되는 메시지 리스트를 나타낸다.

표 2. 내부처리 에이전트와 고장허용 객체그룹 구성요소의 통신 인터페이스

FTOG	Outgoing Messages	Incoming Messages
OGM	request_authentication(SeO) handle_obj_info(IRO)	return_access_right(SeO) return_handle_result(IRO)
SeO	send_access_right(OGM)	request_authentication(OGM)
IRO	send_handle_result(OGM) send_ref_info(RMO) send_SO_info(FD)	handle_obj_info(OGM) request_ref_info(RMO) request_SO_info(FD)
RMO	request_ref_info(IRO)	return_ref_info(IRO) report_obj_state(FD)
FD	report_obj_state(RMO) request_SO_info(IRO)	return_SO_info(IRO)

3.3 에이전트 기반 고장허용 객체그룹의 서비스

에이전트 기반의 고장허용 객체그룹 모델은 중복 객체들을 관리하고 서비스를 실행하기 위하여 그룹 관리 서비스 및 회복 서비스를 제공한다. 본 절에서는 등록 에이전트 및 상태처리 에이전트를 이용한 두 가지 서비스의 과정을 설명한다. 이것은 이동 에이전트의 특징인 이동성을 활용한 것으로 각 객체로 차례로 이동하여 해당 작업을 수행한 후 다시 돌아와 결과를 보고하도록 하는 것이다. 구체적인 과정은 다음과 같다.

3.3.1 에이전트 기반의 그룹관리 서비스

고장허용 객체그룹에서 서비스를 제공하기 위한 서비스 객체들은 OGM에 등록되어야 하며 등록되지 않은 서비스 객체들은 사용자의 요청을 수용할 수 없다. 이 경우 등록을 요청하는 서비스 객체들은 OGM이 위치한 장소를 정확히 알아야 한다는 단점이 있다. 이러한 단점을 극복하기 위한 방법으로 이동 에이전트를 활용할 수 있다. 이동 에이전트는 방문한 객체에서 얻은 정보에 따라 자율적으로 위치를 이동해 갈 수 있다. 특정 고장허용 객체그룹에 등록되고자 하는 서비스 객체들은 자신의 레퍼런스 정보를 가진 등록 에이전트를 생성하고, 네트워크를 이동하면서 원하는 그룹에 등록되어질 수 있다.

그림 3은 서비스 객체의 레퍼런스 정보를 가진 이동 에이전트가 특정 위치에 있는 고장허용 객체그룹에 등록되어지는 과정을 나타낸다. 이미 고장허용 객체그룹에 등록되어 있는 서비스 객체들은 자신이 등록된 그룹의 정보를 저장하고 있다. 등록되어질 서비스 객체는 제공할 수 있는 서비스 및 네트워크상의 위치 등 자신의 레퍼런스 정보를 저장하고 있는 등록 에이전트를 생성한다. 생성된 에이전트는 주위의 객체 노드로 이동하여 등록되고자 하는 고장허용 객체그룹의 정보와 비교한다. 만약 목적지 객체의 등록된 고장허용 객체그룹 정보가 등록을 원하는 고장허용 객체그룹의 정보와 일치한다면 위치 정보를 받아서 원하는 위치로 이동하게 된다. 또한 일치하지 않는 정보를 가지고 있다면 등록 에이전트는 다른 객체 노드로 이동하게 된다. 이러한 과정을 반복하여 등록되어질 고장허용 객체그룹을 찾아가게 된다.

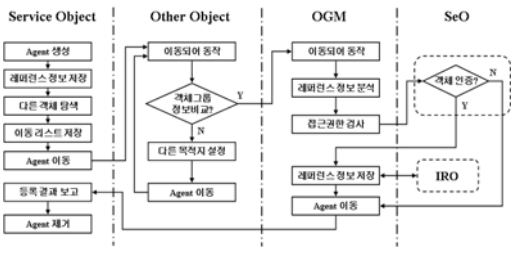


그림 3. 에이전트 기반 그룹관리 서비스 과정

등록을 원하는 고장허용 객체그룹의 위치를 찾은 경우 등록 에이전트를 전송받은 OGM은 에이전트에 저장된 서비스 객체의 레퍼런스 정보를 이용하여 일련의 등록 과정을 거치게 된다. 등록 과정이 완료되면 에이전트는 서비스 객체로 돌아가 OGM에 등록되었음을 알리게 된다. 또한 서비스 객체의 탈퇴 시에는 자신의 객체 이름을 담은 에이전트를 전송하여 고장허용 객체그룹에 저장된 정보들을 삭제하게 된다.

3.3.2 에이전트 기반의 회복 서비스

고장허용 객체그룹 모델의 FD는 메시지를 보내고 이에 대한 응답을 이용하여 서비스 객체의 상태를 파악하게 된다. 이 경우 등록된 서비스 객체의 수가 증가할수록 전송되어야 하는 메시지 수 또한 비례적으로 증가해야 한다는 단점이 있다. 또한 체크포인트를 이용하는 방안 2의 경우에는 같은 기능의 서비스 객체 및 RMO에 실행 상태를 전달하게 되며, 이 경우 전체적인 메시지 수 또한 크게 증가한다. 결과적으로 많은 서비스 객체가 등록되어진 경우에는 메시지 수가 크게 증가하여 부하가 커지게 된다. 이러한 단점을 극복하기 위하여 에이전트를 활용하여 서비스 객체의 고장을 검사하고 객체의 서비스 실행 상태를 전달한다. 이 경우 고장을 검사하기 위한 작업이 서비스 객체로 이동한 에이전트에서 수행되기 때문에 이를 담당하는 FD의 부하가 줄어들며, 주기적인 메시지 교환으로 인한 트래픽 양을 감소시킬 수 있다.

그림 4는 이러한 에이전트의 회복 서비스 과정을 나타낸다. 이동 에이전트를 활용하여 서비스 객체의 고장을 감시하며, 이동 에이전트는 여러 서비스 객체들에게 차례대로 이동하여 작업을 수행한 후 FD로 돌아와 결과를 보고한다. 구체적인 과정은 다음과 같다.

FD는 서비스 객체의 상태를 감시하기 위한 상태처리 에이전트를 생성한다. 생성되는 에이전트는 IRO로부터 고장을 감시해야 할 서비스 객체의 리스트를 전달받고 이동을 시작한다. 상태처리 에이전트가 서비스 객체에 도착하면 객체의 상태를 검사하고 상태정보를 저장한 후 다음 서비스 객체로 이동하게 된다. 에이전트가 다음 서비스 객체로 이동할 수 없거나 이동 후 서비스 객체의 오류를 발견한 경우에는 고장 상태임을 판단하게 된다. 마지막 서비스 객체의 검사를 완료하였을 경우에는 다시 FD로 돌아가 서비스 객체의 리스트를 갱신하고 객체의 상태정보를 보고하는 과정을 거치게 된다.

단계 1. 등록 에이전트 생성
- 1.1 서비스 객체는 등록을 위한 에이전트를 생성한다. 등록 에이전트에는 서비스 목록, 물리적 객체 정보, 객체 위치 및 등록되고자 하는 가상의 객체그룹 정보가 저장된다.
- 1.2 생성된 에이전트는 이동을 위하여 주위의 다른 객체를 탐색하고 이동리스트를 저장한다.

단계 2. 등록 에이전트 이동
- 2.1 등록 에이전트는 설정된 목적지 객체로 이동한다.
- 2.2 도착 후 목적지 객체의 레퍼런스 정보를 이용하여 다음 목적지를 판단한다.
  - 2.2.1 등록되고자 하는 객체 정보와 일치하면 다음 이동 대상 목적지로 설정한다. (단계 3.1)
  - 2.2.2 일치하지 않으면 다른 임의의 객체를 목적지로 설정한다. (단계 2.1)
- 2.3 설정된 목적지 객체로 이동한다.

단계 3. 객체의 등록
- 3.1 에이전트는 최종적으로 등록되고자 하는 객체그룹에 도착한다.
- 3.2 OGM은 에이전트에 저장된 레퍼런스 정보를 기반으로 등록 과정을 수행한다.
  - 3.2.1 SeO는 요청한 서비스 객체를 인증하고 접근권한을 설정한다.
  - 3.2.2 IRO는 서비스 객체의 레퍼런스 정보를 저장한다.
- 3.3 등록이 끝나면 서비스 객체로 이동한다. (단계 4.1)

단계 4. 에이전트 제거
- 4.1 등록 에이전트는 처음의 출발지 객체로 이동하여 결과를 객체에 알린다.
- 4.2 모든 과정이 끝나면 등록에이전트는 제거됨.

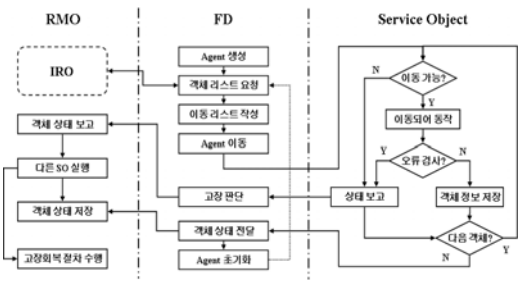


그림 4. 에이전트 기반 회복 서비스 과정

Ⅳ. 가상의 지능형 홈네트워크 시뮬레이션

본 장에서는 에이전트 기반의 고장허용 객체그룹 모델을 바탕으로, 사용자의 요청에 의한 가정 내 기기들의 서비스를 실행하고 관리할 수 있는 가상의 홈네트워크 시뮬레이터를 설계하여 본 모델의 정확한 서비스 수행능력을 평가한다. 시뮬레이터는 에이전트 기반 고장허용 객체그룹 모델에서 제공하는 그룹관리 서비스와 회복 서비스 및 홈오토메이션 서비스를 제공하게 된다.

4.1 지능형 홈네트워크 시뮬레이션 구성

본 절에서는 지능형 홈네트워크 시뮬레이터(Intelligent Home Network Simulator, IHNS)의 구성요소 및 에이전트 기반 고장허용 객체그룹 모델과의 구성 관계를 설명한다. IHNS는 서비스 요청에 대해 홈어플라이언스들을 제어하게 되는 가상의 홈네트워크 시뮬레이터이다. 사용자가 어떠한 서비스를 요청하였을 경우 IHNS는 요청된 서비스가 수행되는 일련의 과정을 보여주게 된다.

그림 5는 IHNS의 구성요소들을 나타낸다. 홈네트워크 시뮬레이션은 3장에서 제시한 에이전트 기반 고장허용 객체그룹의 구성 컴포넌트들로 구성된다. 즉, 고장허용 객체그룹 구성 컴포넌트, 본 논문에서 제안한 에이전트 객체, 홈네트워크 구성에 필요한 컴포넌트들로 구성된다. 고장허용 객체그룹은 OGM, SeO, IRO, RMO, FD로 구성되어 있으며, 에이전트 객체에는 UIA, SHA, RA, SA, IPA가 있다. 서버 그룹은 서비스 제어 서버(Service Control Server, SCS), 관리서버(Management Server), 서비스 게이트웨이(Service Gateway)로 이루어져 있으며, 홈어플라이언스에는 에어컨, 선풍기, 보일러, 홈

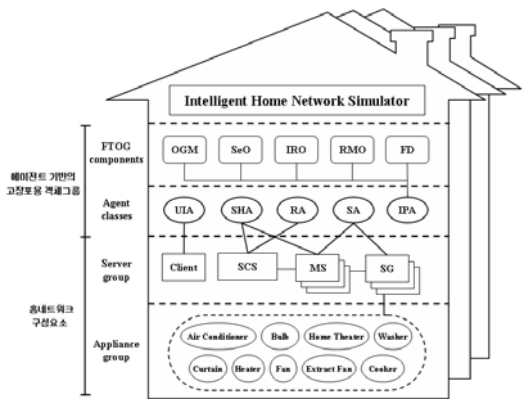


그림 5. IHNS의 구성요소

단계 1. 상태처리 에이전트 생성
1.1 FD는 객체의 상태 감시를 위한 에이전트를 생성한다.
1.1.1 상태처리 에이전트를 초기화하고 IRO에 객체 정보를 요청한다.
1.1.2 IRO는 동록된 서비스 객체의 물리적 정보를 찾아 리스트를 전달한다.
1.2 상태처리 에이전트는 전달된 객체 리스트를 기반으로 이동 리스트를 저장한다.
단계 2. 객체 상태 감시
2.1 상태처리 에이전트는 설정된 목적지 객체로 이동한다.
2.1.1 만약 설정된 목적지로 이동할 수 없다면 고장으로 판단하고 다음 목적지로 이동한다. (단계 3.1)
2.2 목적지에 도착하면 객체 상태를 검사한다.
2.2.1 객체의 오류를 검사한다.
(오류 발생시 단계 3.1)
2.2.2 실행중인 서비스 및 상태 정보를 저장한다.
2.2.3 다른 서비스 객체의 서비스 실행 정보를 저장한다.
2.3 다음 목적지 객체로 이동한다. (단계 2.1)
2.3.1 다음 목적지가 존재하지 않는다면 FD로 이동한다. (단계 4.1)
단계 3. 고장 회복
3.1 FD는 상태처리 에이전트로부터 전달된 객체 정보를 이용하여 객체의 고장을 판단하고 RMO에 보고한다.
3.2 RMO는 다른 서비스 객체에 서비스 에이전트를 전달한다.
3.3 RMO는 고장이 발생한 서비스 객체를 회복하는 절차를 수행한다.
단계 4. 에이전트 초기화
4.1 객체 검사가 끝나면 상태처리 에이전트는 FD로 이동한다.
4.2 FD는 에이전트에 저장된 객체 정보를 RMO로 전달한다.
4.3 상태처리 에이전트를 초기화한다.(단계 1.1.1)

시어터, 전기밥솥, 전구, 커튼, 세탁기 등이 있다. 서비스 제어 서버는 객체의 등록, 탈퇴 및 접근 권한 관리와 관리 서버의 고장 감시 등 전반적인 관리를 책임지게 된다[18-19]. 관리서버는 홈오토메이션을 위하여 서비스 게이트웨이와 홈어플라이언스들을 관리한다.IHNS 에서 사용자 및 다른 구성요소들은

표 3. 에이전트 기반 고장허용 객체그룹 모델과 IHNS의 구성 관계

Agent-based FTOG components	Policy	Functions in IHNS
OGM IRO	객체 등록	- 관리 서버, 서비스 게이트웨이 및 홈 어플라이언스의 등록과 레퍼런스 정보 처리
SeO	보안	- 사용자의 서비스 요청에 대한 접근 권한 검사
RMO FD	고장 허용	- IHNS 구성요소들의 고장 검사 - 구성요소들의 고장 발생시 회복 방안에 따른 고장 회복 수행

OGM, SeO, IRO를 통하여 등록되어 진다. RMO는 사용자의 서비스 요청에 대하여 관리서버에 서비스 에이전트를 전송한다. 관리서버는 서비스 게이트웨이를 통하여 홈어플라이언스들을 제어한다. FD는 상태처리 에이전트를 이용하여 관리서버, 서비스 게이트웨이의 상태를 감시한다.

표 3은 에이전트 기반의 고장허용 객체그룹 모델과 IHNS의 구성요소들 간의 관계를 나타낸다. IHNS는 에이전트 기반의 고장허용 객체그룹 모델에서 제공하는 서비스에 의해 사용자의 요청을 수행하게 된다. <표 3>로부터 에이전트 기반 고장허용 객체그룹 모델은 객체그룹 구성요소 간의 정책 적용을 통하여 적절한 서비스를 제공할 수 있음을 볼 수 있다.

4.2 IHNS에서 제공하는 서비스

IHNS에서는 에이전트 기반 고장허용 객체그룹에서 제공하는 서비스 및 홈네트워크 환경에서 제공될 수 있는 통합적인 홈오토메이션 서비스를 제공하게 된다. 표 4는 IHNS에서 제공되는 서비스를 나타낸 것이다.

표 4. IHNS의 서비스

Services in IHNS	Description
그룹관리 서비스	- 서비스 게이트웨이, 관리서버, 홈어플라이언스의 등록 - 사용자의 서비스를 위한 접속
회복 서비스	- 관리서버, 서비스 게이트웨이 및 홈어플라이언스들의 고장 판단 - 고장 발생시 서비스 센터로 오류 보고
홈오토메이션 서비스	- 홈어플라이언스의 통합 서비스 제공

4.2.1 IHNS의 그룹관리 서비스

사용자 및 관리서버 등 서비스 객체들은 서비스 제어 서버에 등록되어 진다. 사용자의 접속은 외부에서 가정 내 서비스 기기들의 제어를 위해 접근을 요청하는 경우이다. 관리서버 및 서비스 게이트웨이는 등록을 위한 에이전트를 생성한다. 생성되는 에이전트에는 서비스 객체의 위치, 제공하는 서비스 등의 정보를 포함한다. 관리서버의 경우 네트워크상의 위치 및 서비스를 제공해야 할 게이트웨이 리스트 정보가 포함된다. 서비스 게이트웨이의 경우에는 우선 소속되어야 할 관리서버 및 게이트웨이의 위치와 가정 내에서 서비스되는 어플라이언스 정보를 포함한다. 생성된 등록 에이전트는 객체 노드를 이동하면서 최종적으로 서비스 제어 서버에 도착하게 된다. 서비스 제어 서버는 에이전트에 포함되어진 객체의 정보를 OGM으로 전달한다. OGM은 등록 요청에 따라 SeO에 접근 권한 검사를 요청한다. 등록을 요청한 서비스 객체가 접근 권한을 가지고 있다면 서비스 제어 서버에 연결되었음을 알리고 IRO에서는 해당 서비스 객체의 레퍼런스 정보를 관리하게 된다. 접근권한이 없을 경우에는 연결될 수 없으며 탈퇴를 요청한 경우에는 IRO에 저장된 레퍼런스 정보는 지워진다.

4.2.2 IHNS의 회복 서비스

IHNS의 회복 서비스는 크게 에이전트 구조를 이용한 관리서버 등 관리를 위한 기기의 회복과 사용자에게 직접적인 서비스 제공을 위한 기기의 고장 발생시 회복시키는 방법으로 나누어진다. 우선 에이전트를 활용한 관리 기기의 회복을 살펴보면, 서비스 제어 서버는 관리서버의 상태 정보를 감시하게 된다. 에이전트가 오류가 발생한 관리서버를 발견한 경우에는 이를 서비스 제어 서버로 알린다. 또한 서비스 제어 서버는 해당 관리서버에서 수행중인 서비스를 다른 관리서버에서 수행되도록 하며, 관리서버의 오류를 회복하기 위하여 서비스 센터로 상태를 전달한다.

서비스 기기들의 회복 서비스에서는 고장 발생시 같은 역할을 하는 다른 서비스 객체들을 실행시켜 요청된 사용자의 서비스를 만족시키고, 고장이 발생한 홈어플라이언스의 정보를 서비스 센터로 전달하여 회복을 요청한다. 예를 들어 사용자가 100(lux)의 실내 조도량을 요청하였을 때 전구가 고장이 난 경우, 다른 전구를 자동으로 동작시키거나 작동중인 전구의 조도를 높여서 요청된 조도를 유지한다.

4.2.3 IHNS의 홈오토메이션 서비스

IHNS는 에이전트 기반 고장허용 객체그룹에서 제공하는 서비스 이외에 홈네트워크를 위한 서비스

를 에이전트를 활용하여 수행하게 된다. 사용자가 서비스를 요청하면 서비스 제어 서버는 사용자에 해당하는 관리서버와 서비스 게이트웨이 정보를 이용하여 서비스 에이전트를 생성한다. 서비스 에이전트는 해당 관리서버로 이동하게 되며, 에이전트를 전송받은 관리서버는 요청된 서비스를 분석하고 서비스 게이트웨이를 통하여 관련된 어플라이언스를 제어한다. 이때 요청되는 서비스는 다수의 어플라이언스를 제어하는 형태가 될 수 있다. 예를 들어, 사용자가 요청한 작업이 TV를 작동시키는 것이라면 이에 관련된 제어만 하면 된다. 하지만 귀가 시간에 맞추어 온수를 준비하고 에어컨이나 보일러 작동, 환기 등 다수의 어플라이언스를 작동시켜야 할 필요가 있는 서비스를 요청한 경우는 이에 관련된 제어 메시지를 각각 전달하는 것이 아닌 에이전트를 활용하여 전달할 수 있다. 즉, 관리서버는 다수의 제어 메시지를 담아 서비스 에이전트를 이용하여 전체적인 서비스 관리를 수행할 수 있다.

4.3 시뮬레이션 수행 결과

4.3.1 전체 메시지 수 비교

에이전트 기반 고장허용 객체그룹(Agent-based Fault-Tolerant Object Group)과 고장허용 객체그룹(Fault-Tolerant Object Group)의 메시지 수를 비교한다. 에이전트 기반 고장허용 객체그룹의 경우 서비스를 수행하기 위한 전체적인 메시지 수를 감소시킬 수 있다. 예를 들어, 등록된 서비스 객체의 수를 n, 요청된 서비스들에 대한 작업수를 k 라고 한다면 고장허용 객체그룹 모델에서 고장을 감시하기 위해 필요한 메시지는 2n 이다. 또한 고장허용 객체그룹 모델에서는 요청된 서비스를 수행하기 위하여 각 작업당 계속해서 메시지를 전달해야 하므로 k 개의 메시지가 더 필요하다. 따라서 전달해야 하는 총 메시지 수는 2n+k 이다.

반면에 에이전트 기반 고장허용 객체그룹 모델의 경우에는 서비스 객체의 고장을 감시하기 위하여 상태처리 에이전트를 전달하게 되므로 n+1 번의 이동이 필요하다. 또한 서비스 에이전트를 전달하여 전체 작업을 전달하게 된다. 작업수가 서비스 객체의 수보다 작은 경우($n \geq k$), 할당된 서비스 객체로 k+1 번의 이동이 있게 된다. 이때 전체 메시지 수는 n+k+2 이다. 반대로 작업수가 더 많은 경우에는 ($n<k$), 서비스 객체에 다수의 작업이 할당되어야 하므로 최대 n+1 번의 이동만 있으면 된다. 따라서

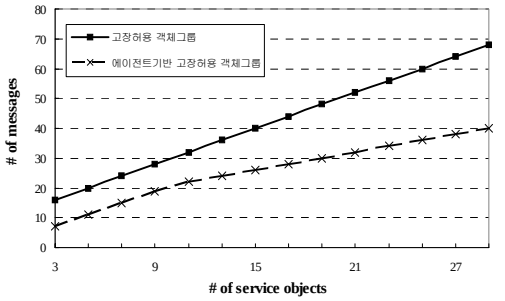


그림 6. 메시지 수 비교(작업수: 10)

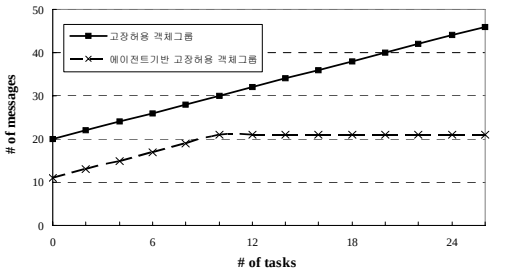


그림 7. 메시지 수 비교(서비스 객체 수: 10)

전체 메시지 수는 2n+2 이다.

그림 6과 그림 7은 고장허용 객체그룹과 에이전트 기반 고장허용 객체그룹 모델의 경우 메시지수를 비교하여 나타낸 것이다. 그림 6은 수행해야 할 작업수를 10 으로 한 후 등록된 서비스 객체의 수에 따른 메시지 수의 변화를 보여준다. 이와는 달리, 그림 7은 서비스 객체의 수를 10으로 고정한 후 대기중인 작업수에 대한 메시지 수의 변화를 보여준다. 그림에서 볼 수 있듯이 에이전트를 활용하면 등록된 서비스 객체의 수가 많을수록, 수행해야 할 작업수가 많을수록 전달해야 할 메시지 수가 감소하게 된다. 이는 객체간 통신으로 인한 관리 부하를 감소시키고 효율적으로 서비스를 수행할 수 있음을 나타낸다.

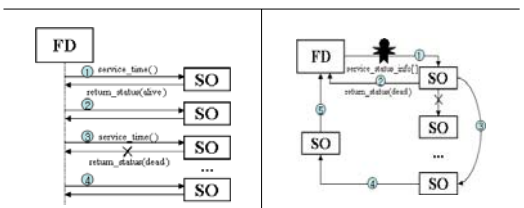
4.3.2 평균 고장감지 시간 비교

서비스 객체의 고장 발생시 이를 감지할 수 있는 방안에 대한 평균 고장감지 시간을 비교한다. 고장감지를 위한 방안에는 고장허용 객체그룹(Fault-Tolerant Object Group)에서 제공하는 주기적인 메시지에 대한 응답을 이용하는 방법과, 본 논문에서 제안한 상태처리 에이전트를 이용하여 고장발견 즉시 FD로 보고하는 에이전트 기반의 고장허용 객체그룹(Agent-based Fault-Tolerant Object

Group) 방법이 있다. 시뮬레이션은 다음과 같은 가정을 기본으로 한다.

- 개별 서비스 객체의 고장은 독립적이다.
- 각 서비스 객체는 충분한 용량의 저장장치를 가지고 있다.
- 각 구성요소(FD 및 서비스 객체) 간의 거리는 같다. 즉 구성요소 간 동일한 전달 시간(Transfer Time)을 가진다.
- 고장 발생률(Fault Rate)은 Poisson random process를 따른다.
- 객체 간 이동시 에이전트는 손실되지 않는다. 즉 목적지로 완전히 이동된다.
- 객체 노드에서의 프로세스 수행시간은 같다.

그림 9와 그림 10은 평균 고장감지 시간을 비교한 것이다. 시뮬레이션을 위하여 고장 발생률을 0.005로 하였다. 그림 9는 고장허용 객체그룹 모델의 고장 감시주기를 7로 하고 서비스 객체 수의 변화에 따른 시간을 나타내며, 그림 10은 서비스 객체 수를 5로 하고 고장허용 객체그룹 모델의 고장 감시주기의 변화에 따라 평균 고장감지 시간을 측정한 것이다. 그림에서는 서비스 객체의 수가 많을수록 고장감지 시간은 비교적 크며, 두 시뮬레이션 구조의 시간차가 작아진다. 이는 객체가 증가할수록 에이전트의 이동에 걸리는 시간이 더 길어지기 때문이다. 또한 에이전트를 활용한 경우 고장 감시 주



고장허용 객체그룹에서의 고장감지 | 제안한 에이전트 기반의 고장허용 객체그룹에서의 고장감지

그림 8. 시뮬레이션 구조

그림 9. 평균 고장감지 시간 비교(감시주기 : 7)

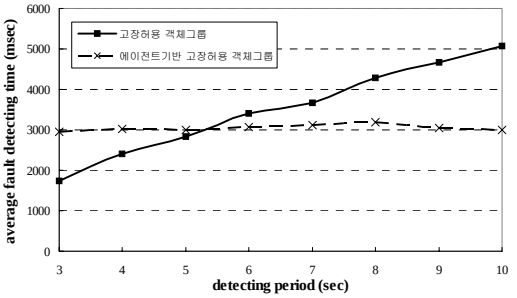


그림 10. 평균 고장감지 시간 비교(서비스 객체 수 : 5)

기가 클수록 유리하다. 반면에, 고장허용 객체그룹 모델은 감시주기가 작을수록 에이전트를 사용하는 경우보다 빠른 고장 감지를 이룰 수 있음을 알 수 있다. 그러나 감시주기가 작거나 객체의 수가 증가할수록 기존 모델은 많은 메시지를 전달해야 하므로, 에이전트를 활용하여 객체의 상태를 감시하는 경우 객체간 메시지 교환에 비해 적은 부하를 사용하고 서비스를 수행할 수 있음을 보여준다.

V. 결 론

본 논문에서는 서비스 실행의 효율성을 높이고 객체 관리 부하를 감소시키기 위한 방안으로 에이전트 기반의 고장허용 객체그룹 모델을 설계하였다. 에이전트 기반의 고장허용 객체그룹 모델은 분산 환경에서 객체 관리의 편의성을 제공하고, 고장 발생시에도 안정된 서비스를 제공할 수 있는 모델이다. 본 모델에서는 요청된 서비스를 수행하고 객체 등록 및 객체의 고장을 회복하기 위한 다섯 가지의 에이전트를 정의하였다. 등록 에이전트를 이용하여 서비스 객체를 특정 고장허용 객체그룹에 등록하고, 서비스 에이전트를 이용하여 사용자에게 통합 서비스를 제공할 수 있다. 사용자인터페이스 에이전트는 서비스 요청, 결과 전송 등 사용자에게 편의성을 제공하게 된다. 또한 상태처리 에이전트를 활용하여 편리한 고장 회복 방안을 제공하며, 내부처리 에이전트는 고장허용 객체그룹 구성요소 간의 프로세싱을 담당한다.

제공되는 각각의 서비스들을 검증하기 위한 방법으로, 에이전트 기반 고장허용 객체그룹 모델을 기반으로 하여 가상의 홈네트워킹 서비스를 제공하기 위한 지능형 홈네트워크 시뮬레이션(IHNS) 환경을 구현하였다. IHNS는 관리서버, 홈어플라이언스 등 분산된 객체를 관리하고 사용자의 서비스 요청을

수행하는 일련의 과정을 보여준다. IHNS에서는 에이전트 기반 고장허용 객체그룹 모델의 그룹관리 서비스 및 회복 서비스를 제공하며, 사용자의 요청에 대한 홈네트워킹 서비스를 수행할 수 있다. 시뮬레이션 결과에서 객체의 고장 발생시 이를 감지하고 회복하기 위한 방안으로 에이전트를 활용할 수 있음을 알 수 있다. 또한 관리서버 등의 객체 고장 시에도 지속적인 서비스를 제공하여 사용자의 요청을 효율적으로 만족시킬 수 있으며, 에이전트를 활용할 경우 객체간의 빈번한 메시지 교환에 의한 객체 관리 부하를 감소시킬 수 있음을 보여준다. 즉, 에이전트 기반 고장허용 객체그룹 모델은 객체의 등록 및 탈퇴 등 객체의 관리에서 유연한 확장성을 가질 수 있으며 효율적인 고장 회복 및 서비스 실행을 지원할 수 있다.

참 고 문 헌

[1] H. Schulzrinne, W. Xiaotao, S. Sidiroglou and S. Berger, "Ubiquitous computing in home networks", Communications Magazine, IEEE, pp.128-135, November 2003.

[2] Dimitar Valtchev and Ivailo Frankov, "Service Gateway Architecture for a Smart Home", IEEE Communications Magazine, pp.126-132, April 2002.

[3] T. Komiya, H. Ohsida and M. Takizawa, "Mobile agent model for distributed object systems," Proc. 5th IEEE International Symposium on Object-Oriented Real-time Distributed Computing, 2002.

[4] F. Zhang, M. Deng, Z. Qin and M. Zhou, "Establish the modeling moving object agents in distributed system," Proc. 2003 IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, pp.290-294, October 2003.

[5] Michiharu Takemoto and Takayuki Nakamura, "Performance Evaluation of a Fault-Tolerant Mechanism Based on Replicated Distributed Objects for CORBA", ISORC2001, pp.95-102, 2001.

[6] P. Narasimhan and P.M. Mellier-Smith, "Strong Replica Consistency for Fault- Tolerant CORBA Applications", Sixth IEEE International Workshop on Object-oriented Real-time Dependable Systems, pp.16-23, January 2001.

[7] L.C. Lung, J. da Silva Fraga, J.M. Farines, M. Ogg, and A. Ricciardi, "CosNamingFT - A Fault-Tolerant CORBA Naming Service", Proc. 18th International Symposium on Reliable Distributed Systems, pp.254-262, 1999.

[8] M. Roman, C. Hess, R. Cerqueira, A. Ranganat, R.H. Campbell, and K. Nahrstedt, "Gaia: A Middleware Infrastructure to Enable Active Spaces", IEEE Pervasive Computing, pp.74-83, 2002.

[9] Ellick Chan, Jim Bresler and Roy Campbell, "Gaia Microserver: An Extendable Mobile Middleware Platform", Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications, pp.309-313, 2005.

[10] Juan C. Garcia-Ojeda, Jose de J. Perez-Alcazar and Alvaro E. Arenas, "Extending the Gaia Methodology with Agent-UML", Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, pp.1456-1457, 2004.

[11] S. Stephen, Y. Wang, B. Wang and K.S. Sandeep, "Reconfigurable Context-Sensitive Middleware for Pervasive Computing", IEEE Pervasive Computing, pp.33-40, July 2002.

[12] S. Stephen and Fariaz Karim, "An Adaptive Middleware for Context-Sensitive Communications for Real-Time Applications in Ubiquitous Computing Environments", Real-Time Systems, pp.29-61, January 2004.

[13] Myungseok Kang, Jaeyun Jung, Younghoon Whang, Younyong Kim and Hagbae Kim, "FTOG-Based Management and Recovery Services", IEICE Trans. Information and Systems, Vol.E88-D, No.11, November 2005.

[14] Myungseok Kang, Jaeyun Jung and Hagbae Kim, "Construction of a Fault-Tolerant Object Group Framework and Its Execution Analysis Using Home-Network Simulations", IEICE transactions on Communication, Vol.E89-B. No.12, pp.3446-3449, 2006.

[15] 김태욱, 강명석, 김학배, “TMR 시스템 기반의 Checkpointing 기법에 관한 연구”, 2003 한국정보처리학회 추계 학술발표논문집, Vol.10, No.2, pp.397-400, 2003

[16] 정재윤, 강명석, 김학배, “효율적 객체 관리 및 부하 분산을 위한 고장포용 객체그룹 프레임워크 설계”, 한국통신학회, Vol.32, No.1, pp.22-30, 2007

강 명 석 (Myungseok Kang) 정회원

2001년 2월 원광대학교 컴퓨터공학과 졸업

2003년 2월 원광대학교 컴퓨터공학과 석사

2003년9월~현재 연세대학교 전기전자공학과 박사과정

<관심분야> 실시간 고장포용 시스템, 지능형 홈네트워크

김 학 배 (Hagbae Kim) 정회원

1988년 2월 서울대학교 전자공학과 졸업

1990년 2월 미국 미시간대학교 전기 및 컴퓨터공학과 석사

1994년 2월 미국 미시간대학교 전기 및 컴퓨터공학과 박사

1996~현재 연세대학교 전기전자공학과 교수

<관심분야> 실시간 시스템, 인터넷 웹서버 기술, 디지털시스템 고장포용 및 신뢰도 평가분야