효율적 객체 관리 및 부하 분산을 위한 고장포용 객체그룹 프레임워크 설계

준회원 강 명 석*, 준회원 정 재 윤*, 정회원 김 학 배**

Design and Analysis of Fault-Tolerant Object Group Framework for Effective Object Management and Load Distribution

Myungseok Kang*, Jaeyun Jung* *Associate Members*, Hagbae Kim** *Regular Member*

요 약

본 논문에서는 분산 객체 관리의 편의성을 제공하고, 고장 발생시에도 안정된 서비스를 가능케 하는 고장포용 객체그룹(Fault-Tolerant Object Group, FTOG) 프레임워크를 제안한다. FTOG 프레임워크는 서비스 우선순위와 체크포인트 방법을 이용하여 두 가지의 고장 회복 방안을 제공하며 퍼지 기반의 부하 추론 과정을 통한 부하 분산을 수행하여 서비스 실행에서의 효율성을 높인다. 또한 가상의 홈네트워크 환경을 구성하고 FTOG 프레임워크를 적용하여 시뮬레이션을 통해 분산되어 있는 객체들의 관리 및 부하 분산 등 본 모델의 안정성 및 신뢰성을 검증하였다.

Key Words : FTOG(Fault-Tolerant Object Group), Group Management Service, Load Scheduling Service

ABSTRACT

In this paper, to achieve consistency maintenance as well as stable service execution, we build a Fault-Tolerant Object Group framework that provides both of the group management service and the load scheduling service. The group management service supports the object management such as registration and authentication, and provides two schemes for failure recovery using the service priority and the checkpointing. In the load scheduling service, we improve the effectiveness of service execution through the reasoning process of object loads based on the ANFIS architecture. The effectiveness in the performance of the developed framework is validated through a virtual home-network simulation based on the FTOG framework.

Ⅰ. 서 론

인터넷과 같은 컴퓨터 네트워크 분야의 급격한 발전으로 인해 중앙 집중 처리 방식의 환경에 비하여 상대적으로 처리능력과 처리효율이 높고 위험을 분산시킬 수 있는 분산 처리 시스템 환경으로 변화하고 있다. 분산 처리 기술 중 객체 컴포넌트 기술은 분산 시스템의 확장성 구현과 신뢰성 보장을 위해 핵심적인 요소이다. 분산 객체 컴포넌트를 이용한 시스템 모델에 관한 연구로 90년대 후반 이후 OMG(Object Management Group)의 CORBA 객체 모델, TINA-C의 TINA 객체 모델, Microsoft의 DCOM(Distributed Component Object Model)등의 객체 모델의 연구가 진행되고 있다. 이와 같은 객체

※ 본 연구는 2006년도 교육인적자원부 BK21 사업의 일환인 연세대학교 전기전자공학부 TMS 사업단의 지원을 받아 연구되었음.
* 연세대학교 전기전자공학과 디지털정보처리 연구실 (mskang, jaeyun@yonsei.ac.kr),
** 연세대학교 전기전자공학과 디지털정보처리 연구실 (hbkim@yonsei.ac.kr)
논문번호 : KICS2005-08-330, 접수일자 : 2005년 8월 10일, 최종논문접수일자 : 2006년 11월 14일

모델을 통한 분산 시스템의 가장 주요한 문제 중 하나는 분산 객체 컴포넌트의 네트워크 통신에서의 결함이 발생하거나 객체 컴포넌트의 고장이 발생할 경우에도 계속적인 서비스 제공이 이루어져야 한다는 것이다. 분산 시스템에서 객체 컴포넌트의 결함은 전체 시스템의 고장을 일으키며, 모든 사용자의 서비스 수행에 영향을 미치게 된다.

분산 시스템에 결함이 존재할 때, 신뢰성(reliability), 가용성(availability)을 증가시키는 방법 중 가장 대표적인 방법은 중복 객체(replicated object)를 이용하는 방법이다. 이러한 분산 시스템을 기반으로 중복 객체를 이용하여 고장 포용 서비스를 제공하려는 많은 연구들이 있다[1-11]. 우선, 통합(integration) 방법이 있다. 이 방법은 중복관리 방안이 ORB (Object Request Broker)에 포함되어 있는 방안이다[5-8]. 또한, 서비스 형태로 중복객체를 관리하는 방법이 있다. 이것은 중복관리 방안이 서비스 객체의 형태로 ORB에 제공되는 형태이다[9-11]. 또, ORB의 메시지를 운영체제(OS)상에서 가로채어(intercept) 모든 중복객체로 전송(multicast)하는 방안이 있다[12]. 마지막으로 Object Management Group(OMG)의 FT-CORBA가 있다[2]. 이러한 방법들은 아직까지 신뢰성 있는 어플리케이션을 구현하는데 어려움을 가지고 있다[3]. 이와 함께 고장포용 범위(Fault-Tolerance Domain)에 있는 모든 호스트들은 같은 고장포용 조직(Fault-Tolerance Infrastructure)을 지원하는 ORB를 사용해야만 하는 제약이 있다. 또한, 서비스 객체의 부하 분산을 위한 방안의 부재로 인하여 효율적으로 사용자의 서비스 요청을 만족시킬 수 없다는 단점이 있다.

위의 연구들의 제약사항을 극복하고자 본 논문에서는 분산 객체 관리의 편의성을 제공하고, 고장 발생시에도 안정된 서비스를 가능케 하는 고장포용 객체그룹(Fault-Tolerant Object Group, FTOG) 프레임워크를 제안한다. FTOG 프레임워크는 두 가지의 서비스를 제공하는데 하나는 그룹관리 서비스이고, 다른 하나는 부하분배 서비스이다. 그룹관리 서비스에서는 전반적으로 FTOG 프레임워크를 관리하고 고장 발생시에도 지속적인 서비스를 제공할 수 있는 방안을 제공한다. 부하분배 서비스에서는 서비스 객체의 부하를 판단하여 사용자의 요청을 적절히 분산시켜 서비스의 효율성을 향상시키는 부하분산 서비스를 제공한다.

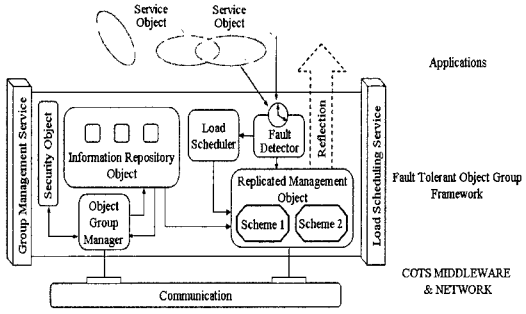


그림 1. FTOG 프레임워크

Ⅱ. 고장포용 객체그룹 프레임워크

고장포용 객체그룹 프레임워크는 분산되어 있는 객체들의 관리 및 회복, 부하 분산 서비스를 지원하는 모델로, 임의의 서비스를 수행하기 위한 객체들의 논리적인 집합이다. 그림 1과 같이 FTOG 모델은 객체그룹관리자, 보안객체, 정보저장소, 중복관리자, 부하분배자, 고장발견자 등의 구성요소로 구현될 수 있다. 이러한 구성요소들은 다음과 같이 설명된다.

객체그룹관리자(Object Group Manager, OGM)는 FTOG의 전반적인 관리를 책임지며 객체 그룹과 사용자 간의 인터페이스를 제공한다. 또한 임의의 객체를 특정 FTOG에 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 객체 리스트를 가지고 있으며 객체그룹관리자의 요청에 따라 정보를 변경한다.

고장발견자(Fault Detector, FD)는 주기적으로 서비스 객체의 상태를 판단한다. 서비스 객체의 상태 정보 전달 주기 동안 보고하지 않은 서비스 객체는 오류가 있다고 판단하게 된다. 또한 고장발견자는 서비스 객체로부터 전달되는 객체의 자원정보를 부하분배자(Load Scheduler, LS)로 전달한다. 부하분배자는 퍼지 논리를 기반으로 서비스 객체의 부하를 추론하여 서비스 우선순위(Service Priority, SP)를 결정하고 이 정보를 중복관리자에 전달한다. 중복관리자(Replication Management Object, RMO)는 사용자의 서비스 요청에 대하여 중복된 서비스 객

체(Service Object, SO)에 메시지를 보낸다. 또한, 다른 사용자들의 서비스 요청에 대한 스케줄링을 제공하며 서비스 객체의 실행결과를 사용자에게 보낸다.

Ⅲ. FTOG 프레임워크의 서비스

제안하는 FTOG 프레임워크는 기본적으로 두 가지 서비스를 제공한다. 하나는 객체 관리의 편의성을 제공하고 중복객체의 투명성을 보장하기 위한 그룹관리 서비스이다. 다른 하나는 부하분배 서비스이다. FTOG 프레임워크는 부하분배 서비스를 통하여 서비스 객체의 부하에 따라 클라이언트의 서비스 요청을 분산 수행하는 부하 분산 방안을 제공한다.

3.1 그룹관리 서비스

그룹관리 서비스는 객체의 등록, 레퍼런스 정보 변경 등 FTOG 프레임워크의 관리를 수행하며 객체 관리에 대한 일관성을 제공한다. 한편, 서비스 객체의 고장이 발생한 경우 이를 감지하고 수행중인 서비스가 지속될 수 있도록 회복 방안을 제공하여 객체의 상태에 대한 투명성을 제공한다.

3.1.1 객체 관리 방안

FTOG 프레임워크는 기본적으로 OGM, SeO, IRO를 이용하여 객체들의 등록 및 탈퇴, 사용자 인증, 권한 부여, 정보관리 등 전체적인 관리 기능을 제공한다. 객체를 등록할 경우, OGM에 등록 메시지를 보내면 SeO에서는 접근 허가에 대한 인증 및 권한부여를 한다. SeO에서 수행하는 사용자 인증 절차는 사용자의 서비스 요청에 대한 권한 여부를 확인하는 과정이다. IRO에서는 관련된 정보를 관리한다. IRO에서 수행하는 정보 관리 기능에는 객체 정보 생성, 삭제, 검색 기능을 포함한다. 그림 2는 객체의 등록 요청시 FTOG 프레임워크에서 수행하는 일련의 과정들을 보여준다.

3.1.2 고장 회복 방안

사용자의 요청에 의해 서비스 수행 중 고장이 발생한 경우, 진행중인 서비스는 중단될 수 있으며 효율적인 서비스를 제공할 수 없다. 본 절에서는 중복객체를 사용하여 고장이 발생한 경우에도 지속적으로 서비스를 수행할 수 있는 방안을 제안한다. FTOG 프레임워크는 두 가지의 고장 회복 방안을 제공한다. 첫번째로 사용자의 서비스 요청에 대하여

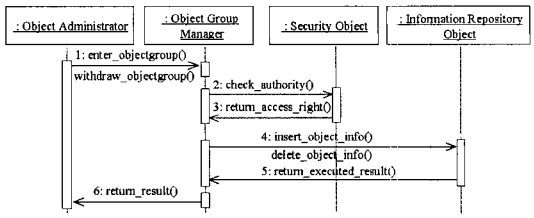


그림 2. 객체 등록 수행 과정

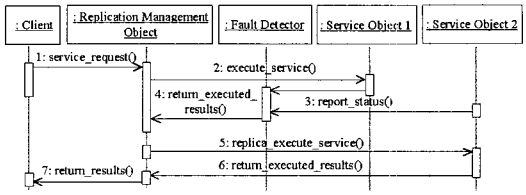


그림 3. 고장 회복 수행 과정

하나의 서비스 객체를 동작시키며 고장이 발생한 경우 가장 높은 서비스 우선순위를 가지는 다른 서비스 객체를 동작시키는 방법이다. 두번째 방안은 첫번째 방안에 체크포인트 방법을 첨가한 것이다. 그림 3은 객체 관리에 의해 사용자에게 서비스 객체의 레퍼런스 정보가 전달된 후 고장이 발생한 경우의 회복 과정을 보여준다.

■ 중복객체를 통한 고장 회복 (방안 1)

그림 4에서 보여지듯이 방안 1에서는 사용자가 요청한 서비스는 RMO를 통하여 서비스 객체에 보내진다. 여기서 스케줄링 메커니즘을 통해 가장 높은 우선순위를 가진 서비스 객체에 서비스가 할당된다. FD는 서비스 객체의 상태를 주기적으로 검사하고 고장이 발생한 서비스 객체를 RMO에 알린다. RMO는 가장 작은 부하를 가진 서비스 객체를 실행시키고 고장이 발생한 서비스 객체를 회복하는 과정을 수행한다.

■ 중복객체 및 체크포인트 방법을 통한 고장 회복 (방안 2)

체크포인트 방법을 이용하는 방안 2에서는 주기적인 체크포인트를 통하여 서비스 객체의 실행 상태 정보를 RMO및 다른 서비스 객체들에게 알려주게 된다[13]. 그림 5는 서비스 객체에서 고장이 발생하였을 때 체크포인트를 이용한 고장회복 과정을 보여준다. 방안 1에서와 같이 실행중인 서비스 객체의 고장이 발견되었을 경우에 RMO는 다른 서비스 객체 중 하나를 실행하게 된다. 이때 고장이 발생한

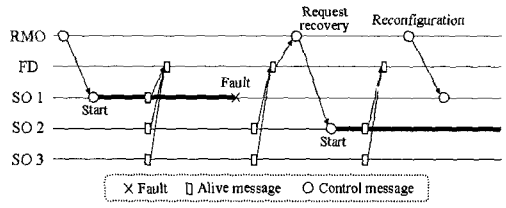


그림 4. FTOG에서의 고장 회복 (방안 1)

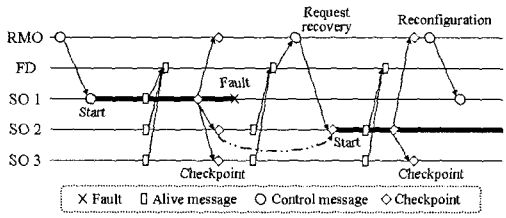


그림 5. FTOG에서의 고장 회복 (방안 2)

서비스 객체의 이전 서비스 실행 상태 정보를 알고 있으므로 이미 처리된 서비스는 수행되지 않고 저장되어 있는 체크포인트에서 재시작한다. 따라서 사용자의 요청을 더 빠른 시간 안에 수행할 수 있다.

3.2 부하분배 서비스

부하분배 서비스는 서비스 객체의 부하 정보를 이용하여 클라이언트의 서비스 요청 및 고장이 발생한 경우의 부하가 적은 객체에 서비스를 할당 함으로서 더 효과적인 서비스를 수행하기 위한 방법이다. 스케줄링 우선순위를 결정하기 위하여 기본적으로 퍼지 시스템 기반의 ANFIS 모델[14]을 이용한 서비스 객체의 상태 모델링을 수행하게 된다. 서비스 객체의 정확한 상태를 얻기 위하여 퍼지 논리를 적용하고 퍼지 규칙에 근거하여 서비스 요청에 대한 부하 배치를 결정한다. 모델링의 결과인 객체의 부하를 정량화하고 그 값을 바탕으로 우선순위를 결정하여 효율적인 부하 분산을 수행한다. 그림 6은 FTOG 프레임워크에서 제공하는 부하분배 과정을 나타낸다.

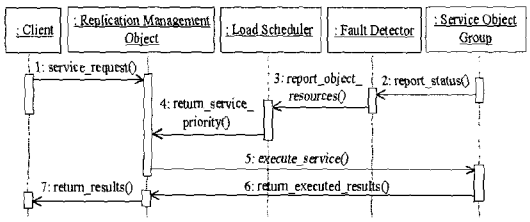


그림 6. 부하분배 수행 과정

3.2.1 부하 추론을 위한 ANFIS 모델링

효과적인 부하배치를 위해서는 서비스 객체의 정확한 상태를 바탕으로 부하 분산을 하는 것이 중요하다. 또한 서비스 객체는 정확한 자원상태 정보를 보고 할 수 있어야 한다. 네트워크 환경은 많은 불확실성을 가진 변수들이 존재하며 변수들 사이에 비선형적인 관계가 많은 컴퓨터 네트워크 시스템을 다루기 위해서 수학적 모델링은 많은 제한이 따른다. 결국 이러한 제한이 컴퓨터 네트워크 시스템의 불확실성을 반영하기 어렵게 만든다. 따라서 이러한 문제를 극복하고 효과적인 부하배치를 효과적으로 수행하기 위하여 객체의 자원정보의 ANFIS 모델링을 통하여 서비스 객체의 상태를 정확하게 추론하여 부하 분산을 수행한다.

퍼지 멤버십 함수는 가장 간단한 삼각형 멤버십 함수를 사용한다. 퍼지 입력 멤버십 함수를 결정하기 위해서, 본 논문에서는 CPU 사용량을 위한 멤버십 함수를 CPU로 정의 하고, 메모리를 MEM, 부하 평균을 LOAD로 표기하였다. CPU, MEM 및 LOAD를 위한 퍼지 집합 정의는 각각 {*Very Small(VS), Small, Medium, Big, Very Big(VB)*}로 정의한다. 그리고 퍼지 출력 멤버십 함수로, 5개의 멤버십 함수를 사용하며, 네트워크 처리량 NT(Network Throughput)를 위한 퍼지 집합은 {*Very Small(VS), Low, Medium, High, Very High(VH)*}이다. NT가 높을수록 처리 부하가 많은 것으로 클라이언트의 요청을 받아들이지 않거나, 받아들이더라도 서비스 지연이나 실패가 발생할 확률이 높다는 것을 의미한다. 즉, 이것은 클라이언트 요청을 처리하기 위해서 서비스 객체의 자원들이 많이 사용되고 있는 상태를 말한다.

모델링에 있어서 보다 정확한 추론을 하기 위해 출력오차를 줄이기 위해서 초기 소속 함수에 대한 학습이 필요하다. 일반적으로 멤버쉽 함수 $\mu_{A_i}(x)$는 식 (1)과 같이 최대치 1 과 최소치 0을 가지는 종형이 된다. 여기서 $\{ a_i , b_i , c_i \}$ 을 전건부 파라미터(premise parameter)라고 하며 이 값의 변화에 따라 종형 함수의 모양이 결정된다[14][15].

$$\mu_{A_i}(x) = \exp\left\{ - \left[\left(\frac{x - c_i}{a_i} \right)^2 \right]^{b_i} \right\} \quad (1)$$

퍼지 규칙은 ANFIS 모델링에 적합한 식 (2)과 같은 Sugeno 형태의 IF-THEN 규칙을 사용하고 퍼지 소속 함수는 종형 함수를 사용한다. 여기서 $\{ a_0^i,$

a_1^i , a_2^i , a_3^i } 는 ANFIS 학습 후 얻어지는 후건부 파라미터(consequent parameter) 이다.

IF CR is RL and MR is RM and LAV is RH,

THEN $f_i = a_0^i x_1 + a_1^i x_2 + a_2^i x_3 + a_3^i$ (2)

CR: CPU Consumption Range, LAV: Load Average,
MR: Memory Consumption Range, RL: Resource Low,
RM: Resource Medium, RH: Resource High

3.2.2 서비스 객체의 부하 분산

그림 7은 FTOG 프레임워크의 부하분산 수행 과정을 나타낸 그림이다. 서비스 객체는 주기적으로 상태 정보 및 시스템 자원 정보를 FD로 전달한다. FD는 전달된 서비스 객체의 상태를 점검하고 시스템 자원 정보를 LS로 전달한다. LS는 ANFIS 모델을 기반으로 하여 객체의 부하를 추론하고 이 정보를 기반으로 서비스 우선순위를 결정한다. 또한 RMO는 결정된 우선순위 정보를 이용하여 클라이언트의 서비스 요청 및 고장 발생시 실행할 서비스 객체를 결정하게 된다. 우선순위 결정시 서비스 객체의 수를 N이라고 한다면 가용성이 가장 큰 객체를 1, 가장 적은 가용성을 가진 객체를 N으로 결정한다. 추론된 객체의 NT가 높을수록 우선순위는 낮아지며 우선순위가 높을수록 RMO가 부하 배치를 결정할 때 선택할 확률이 높아진다는 것을 의미한다.

사용자의 서비스 요청이 서비스 객체에게 전달될 때 서비스 우선순위를 결정하는 방법을 예를 들어 설명한다. 서비스 객체들은 같은 역할을 수행하는 중복객체이며 그룹관리 서비스에 의해 등록되어 서비스를 수행하고 있다고 가정한다. 서비스 객체들의 CPU, MEM 및 네트워크 출력값은 서비스 객체들의 실제 자원 정보를 측정하여 사용한다. 클라이언트들이 차례대로 서비스를 요청하였을 경우, LS는 ANFIS 모델링을 통하여 객체의 부하 정보를 추론한다. 이 정보를 기반으로 우선 순위가 결정되고 사용자들의 서비스 요청이 분산 수행된다.

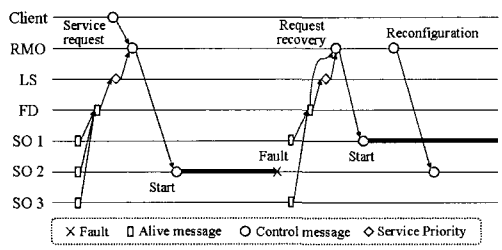


그림 7. FTOG 프레임워크의 부하분산

표 1.사용자의 요청에 따른 서비스 객체 우선순위

Service Object 1				
CN	CPU	MEM	NT	SP
C1	21.70	76.99	17.40	2
C2	**15.90**	**77.81**	**18.22**	**1**
C3	22.00	77.00	27.46	3
C4	22.00	77.09	32.83	3
C5	**24.94**	**77.10**	**34.70**	**1**
C6	41.93	77.65	43.14	3
C7	49.81	77.67	39.77	3
C8	34.80	77.27	44.11	2
C9	**44.80**	**77.28**	**48.06**	**1**

Service Object 2				
CN	CPU	MEM	NT	SP
C1	11.03	72.52	18.71	3
C2	15.09	75.63	18.82	2
C3	**7.12**	**72.56**	**16.34**	**1**
C4	**11.85**	**72.58**	**20.85**	**1**
C5	15.88	72.70	36.15	3
C6	27.27	73.09	38.44	2
C7	**24.51**	**75.64**	**32.20**	**1**
C8	24.73	72.73	48.56	3
C9	37.61	72.85	53.18	2

Service Object 3				
CN	CPU	MEM	NT	SP
C1	**12.84**	**38.80**	**15.67**	**1**
C2	13.92	38.98	24.14	3
C3	13.00	38.84	24.37	2
C4	13.96	39.65	27.94	2
C5	17.84	39.70	35.83	2
C6	**19.28**	**43.33**	**30.16**	**1**
C7	22.83	43.33	35.26	2
C8	**24.72**	**40.94**	**41.25**	**1**
C9	30.76	41.00	55.43	3

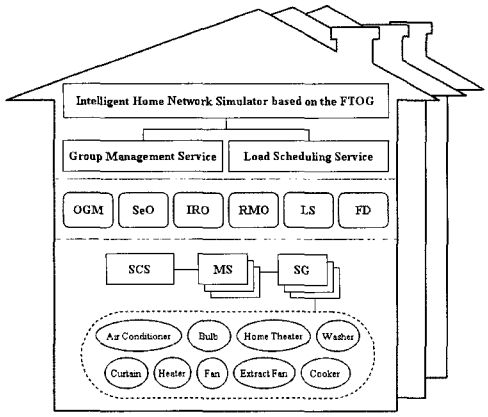
CN: client name, CPU: CPU utilization, MEM: memory usage
NT: network throughput, SP: service priority

예를 들어, 표 1과 같은 자원을 가지고 있는 서비스 객체가 있다고 하면 ANFIS 추론의 결과에 따라 서비스 객체들은 1부터 3까지의 우선순위를 갖게 된다. Client 1이 서비스를 요청한 경우 LS에서는 각 서비스 객체의 부하 정보인 네트워크 출력값을 추론한다. 이 경우 서비스 객체 3의 네트워크 출력값(15.67)이 서비스 객체 1(17.40) 및 서비스 객체 2(18.71)에 비해 상대적으로 낮으므로 가장 높은 우선순위를 가지게 된다. 이 결과를 이용하여 RMO는 서비스 객체 3에 Client 1의 요청을 할당한다. Client 2가 서비스를 요청하였을 경우 서비스 객체 1이 가장 낮은 네트워크 출력값(18.22)를 가져서 높은 우선순위를 갖게 되어 서비스를 실행하게 된다.

Ⅳ. FTOG 프레임워크 기반의 가상 홈네트워크 시뮬레이터

FTOG 프레임워크를 기반으로, 사용자의 요청에 의한 가정 내 기기들의 서비스를 실행하고 관리할 수 있는 홈네트워크 시뮬레이터 (Intelligent Home-Network Simulator, IHNS)를 설계하여 본 모델의 정확한 서비스 수행능력을 평가한다. IHNS는 서비스 요청에 대해 홈어플라이언스들을 제어하게 되는 가상의 홈네트워크 시뮬레이터이다. 사전에 정의된 서비스 중에서 사용자가 임의의 서비스를 요청하였을 경우, IHNS는 요청된 서비스가 수행되는 과정을 보여주게 된다.

그림 8은 IHNS의 구성요소들을 나타낸다. 홈네트워크 시뮬레이션은 2장에서 제시한 FTOG의 구성 컴포넌트들과 함께, 홈네트워크 구성에 필요한 컴포넌트들로 구성된다. IHNS의 구성요소에는 서비스



OGM Object Group Manager SeO Security Object IRO Information Repository Object
RMO Replicated Management Object LS Load Scheduler FD Fault Detector
SCS Service Control Server MS Management Server SG Service Gateway

그림 8. IHNS의 구성요소

제어 서버(Service Control Server, SCS), 관리서버(Management Server, MS), 서비스 게이트웨이(Service Gateway) 및 다양한 홈어플라이언스들로 구성된다. 서비스 제어 서버는 객체의 등록, 탈퇴 및 접근 권한 관리와 관리 서버의 고장 감시 등 전반적인 관리를 책임지게 된다. 관리 서버는 홈오토메이션을 위하여 서비스 게이트웨이와 홈어플라이언스들을 관리한다. 홈어플라이언스들은 에어컨, 선풍기, 보일러, 홈시어터, 전기밥솥, 전구, 커튼, 세탁기로 이루어져 있다.

IHNS에서 사용자 및 다른 구성요소들은 OGM, SeO, IRO를 통하여 등록되어 진다. RMO는 사용자의 서비스 요청에 대하여 관리 서버에 서비스 메시지를 보낸다. 관리 서버는 서비스 게이트웨이를 통하여 홈어플라이언스들을 제어한다. FD는 관리 서버, 서비스 게이트웨이 및 홈어플라이언스들의 고장을 검사한다. LS는 관리 서버의 시스템 자원 정보를 이용하여 네트워크 출력값을 추론하고 서비스 요청에 따른 부하를 분산하는 역할을 수행한다.

표 2는 FTOG 프레임워크와 IHNS의 구성요소들 간의 관계를 나타낸다. IHNS는 FTOG 프레임워크에서 제공하는 서비스에 의해 사용자의 요청을 수행하게 된다. 표 2로부터 FTOG 프레임워크는 객체그룹 구성요소 간의 정책 적용을 통하여 적절한 서비스를 제공할 수 있음을 볼 수 있다.

V. FTOG 프레임워크의 성능 분석

FTOG 프레임워크에서는 적절한 부하 분산을 수행하기 위해서 서비스 객체의 부하를 정확히 예측

표 2. FTOG 프레임워크와 IHNS의 구성 관계

FTOG 구성요소	정책	IHNS의 수행 기능
객체그룹관리자(OGM) 정보저장소(IRO)	객체등록	- 관리 서버, 서비스 게이트웨이 및 홈어플라이언스들의 등록
보안객체(SeO)	보안	- 서비스 요청에 대한 접근 권한 검사
중복관리자(RMO) 고장발견자(FD)	고장포용	- 관리 서버 및 홈어플라이언스의 고장 검사 - 관리 서버 및 홈어플라이언스의 고장 발생시 회복 방안에 따른 고장 회복 수행
부하분배자(LS)	부하분산	- 관리 서버의 자원정보를 활용하여 서비스 우선순위 결정 - 요청된 서비스 수행 및 고장 발생시 적절한 부하 분산 수행

하는 것이 중요하다. 또한 서비스 객체들의 신뢰성을 높일 수 있는 고장 회복 방안이 필요하다. 본 장에서는 가상 홈네트워크 시뮬레이션을 기반으로 LS에서 수행하는 ANFIS 모델링에 따른 서비스 객체의 부하예측 결과와 고장 회복 방안들에 대한 평균 수행 시간을 분석한다.

5.1 ANFIS 모델링 결과

ANFIS 모델링을 위하여 CPU 사용률, 메모리 사용량, 부하 평균 및 네트워크 출력값 등 4가지 데이터를 측정하였다. 그림 9는 서비스 객체의 CPU 사용량을 나타낸다. CPU 사용률에는 네트워크 접속 및 서비스 요청에 대한 작업량이 반영되어 있다. 그러므로 CPU 사용률은 객체의 상태를 잘 반영하여 중요한 성능 분석 요소가 될 수 있다. 그림 10과 11은 각각 메모리 사용량과 부하 평균을 보여준다. 부하 평균은 사용자의 요청된 서비스를 제공하기 위하여 서비스 객체에서 동작중인 전체 프로그램과 관련되어 있다. 그림 12는 요청에 대한 서비스 객체의 네트워크 출력값을 나타내며 초당 패킷수는 부하 분산에서 중요한 요소 중 하나이다. 작업수가 높아질수록 출력값은 증가하게 되고, 서비스가 수행되면 점차적으로 감소하게 되는 것을 알 수 있다.

측정한 네 가지 요소에 대한 ANFIS 모델링 결과는 그림 13과 같다. 입력으로 CPU 사용률, 메모리 사용량, 부하 평균을 사용하며 출력은 네트워크 출력값이다. 모델링 결과에서 세 가지의 입력에 대하여 출력인 네트워크 출력값을 잘 추론하고 있음을 알 수 있다. 즉 서비스 객체의 시스템 자원 상태 정보를 측정할 수 있다면 서비스 요청에 대하여 실행 우선순위를 결정할 수 있으며 그에 따라 부하 분산을 수행할 수 있음을 알 수 있다.

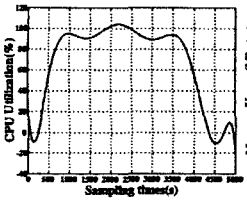


그림 9. CPU 사용률

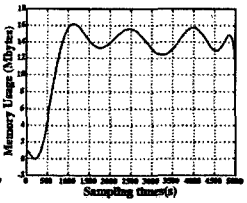


그림 10. 메모리 사용률

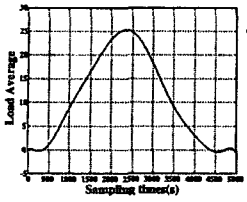


그림 11. 시스템부하평균

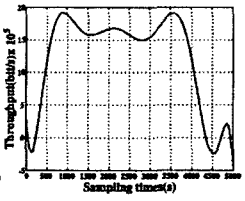


그림 12. 네트워크출력값

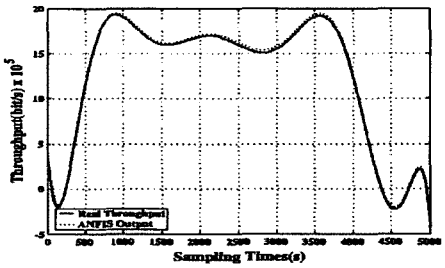


그림 13. 네트워크 출력값에 대한 ANFIS 모델링 결과

5.2 평균 수행 시간 결과

본 절에서는 IHNS를 통하여 사용자의 요청이 실행되는 평균 시간을 비교한다. 각 파라미터에 따른 시뮬레이션 구성은 그림 14와 같으며 표 3을 기초로 하여 파라미터의 수치를 변화시키면서 방안 1 및 2의 전체 평균 수행시간을 측정하였다. 또한 시뮬레이션은 다음과 같은 가정을 기초로 하여 이루어진다.

- 개별객체의 고장은 독립적이다.
- 각각의 객체들은 충분한 용량의 저장장치를 가지고 있다.
- 서비스 요청(service request), 고장 발생(fault rate), 서비스 수행(service rate)은 Poisson 확률분포를 따른다.
- 방안 2에서 checkpoint는 고장상황 없이 이루어진다.

표 3. 시뮬레이션 파라미터

Fault Rate	0.001
Service Rate	0.1
Request Rate	0.05
Fault Detect Period	10

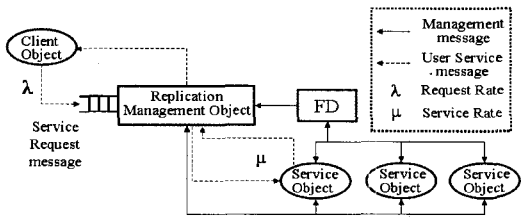


그림 14. 시뮬레이션 구성

그림 15는 고장 발생률의 변화에 따른 방안 1, 방안 2의 평균 수행시간을 비교한 것이다. 또한 표 4는 서비스 요청률 및 서비스 수행률의 변화에 따른 평균 수행시간을 나타낸다. 방안 1은 전체 평균 수행시간이 방안 2에 비하여 크게 나타났다. 하지만 고장 발생률이 작아질수록 방안에 따른 차이가 적어진다. 따라서 방안 2는 고장이 빈번하게 일어나는 경우 방안 1에 비하여 유리하며 빠른 복구가 가능함을 알 수 있다. 일반적으로 고장이 빈번하게 발생하는 경우, 체크포인트 방법에 의하여 신속한 회복을 달성할 수 있다

표 4. 평균 수행시간

Request Rate	**0.04**	**0.045**	**0.05**
Scheme 1	22.476	22.791	26.544
Scheme 2	21.837	22.236	25.763
Service Rate	**0.1**	**0.2**	**0.3**
Scheme 1	26.544	9.87	6.64
Scheme 2	25.763	9.16	5.95

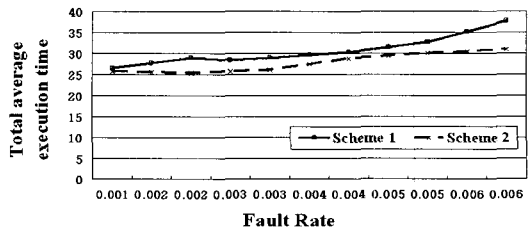


그림 15. 고장 발생률 변화에 따른 각 방안별 평균 수행시간

Ⅵ. 결 론

본 논문에서는 분산 환경에서 동작하는 서비스의 신뢰성, 안정성을 증가시키기 위하여 그룹관리 서비스 및 부하분배 서비스를 제공하는 FTOG 프레임워크를 제안하였다. FTOG 프레임워크에서 그룹관리

서비스는 OGM, SeO 및 IRO를 이용하여 객체 관리의 일관성 및 편의성을 제공한다. 또한, FTOG 프레임워크는 RMO 및 FD를 이용하여 고장을 감지하고 이를 회복하는 방안을 제공한다. 본 논문에서는 고장 회복을 위하여 서비스 우선순위 및 체크포인트 방법을 이용한 두 개의 회복 방안을 제안하였다. 부하분배 서비스에서는 LS를 이용하여 퍼지 이론 기반의 서비스 객체 상태 모델링을 수행하고 서비스 우선순위를 결정한다. 이 결과를 이용하여 사용자의 서비스 요청에 대한 부하 분산을 수행하며 서비스 실행의 효율성을 높일 수 있다.

FTOG 프레임워크에서 제공하는 서비스는 가상의 홈네트워크 시뮬레이션을 통해 검증하였다. 시뮬레이션에서는 관리 서버, 서비스 게이트웨이 등 홈네트워크 구성에 필요한 요소들을 통하여 FTOG 프레임워크를 적용하고 결과를 분석하였다. 시뮬레이션 결과는 서비스 객체의 네트워크 부하를 정확히 예측하여 서비스 요청을 분산시킬 수 있음을 보여준다. 또한 관리 서버 등의 서비스 객체 고장시에도 지속적인 서비스를 제공하여 사용자의 요청을 만족시킬수 있음을 알 수 있다. 고장 발생률이 높은 경우에는 체크포인트 방법에 의해 방안 2가 방안 1보다 더 빠른 평균 수행시간을 가지게 된다.

참 고 문 헌

[1] M. Takemoto, "Fault-tolerant Object on Network-wide Distributed Object-oriented Systems for Future Telecommunications Applications," IEEE, pp.139-146, April 1997.

[2] Object Management Group, "Fault Tolerant CORBA (Final Adopted Specification)," OMG Technical Committee Document formal/01-12-29, Dec. 2001.

[3] P. Felber, P. Narasimhan, "Experiences, Strategies, and Challenges in Building Fault-Tolerant CORBA Systems", IEEE Trans. Computers, Vol.53, no.5, pp.497-511, May 2004.

[4] H.S. Kam, "Fault tolerant cluster computing through replication," IEEE Proceedings of the 1997 International Conference on Par- allel and Distributed Systems, pp.756-761, February 1997.

[5] S. Maeis, "Adding group communication and fault tolerance to CORBA", Proceedings of the 1995 USENIX Conference on Object-Oriented Technologies, June 1995.

[6] S. Maffeis, "A Flexible System Design to Support Object Groups and Object-Oriented Distributed Programming", Proceedings of ECOOP'93, Lecture Notes in Computer Science 791, 1994.

[7] A. Vaysburd, "Building Reliable Inter- operable Distributed Objects with the Maestro Tools", Technical Report TR98 -1678, Department of Computer Science, Cornell University, May 1998.

[8] M. Cukier, J. Ren, C. Sabnis, W.H. Sanders, D.E. Bakken, M.E. Berman, D.A. Karr, and R. Schantz, "AQuA: An Adaptive Architecture that Provides Dependable Distributed Objects", Proc. IEEE 17th Symposium on Reliable Distributed Systems, pp.245-253, October 1998.

[9] P. Felber, B. Garbinato, and R. Guerraoui, "The Design of a CORBA Group Comm- unication Service", Proc. of the 15th Symposium on Reliable Distributed Systems, pp.140-149, October 1996.

[10] D. Liang, C.L. Fang, S.M. Yuan, C. Chen, and G.E. Jan, "A Fault-Tolerant Object Service on CORBA", The Journal of Systems and Software, 48, 1996.

[11] 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.

[12] P. Narasimhan and L.E. Moser 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.

[13] 김태욱, 강명석, 김학배, "TMR 시스템 기반의 Checkpointing 기법에 관한 연구", 한국정보처리학회 추계 학술발표논문집, Vol.10, no.2, pp.397-400, 2003.

[14] J. Shing R. Jang, "ANFIS:Adaptive Network -Based Fuzzy Inference System", IEEE Trans. System, Man and Cybernetics, Vol.23, no.3, pp.665-685, May 1993.

[15] T. Takagi, M. Sugeno, "Fuzzy identification of system and its application to modeling and control", IEEE Trans. Systems, Man and Cybernetics, Vol.15, no.1, pp.166-132, 1985.

강 명 석 (Myungseok Kang) 준회원

2001년 2월 원광대학교 컴퓨터공학과 졸업
2003년 2월 원광대학교 컴퓨터공학과 석사
2003년9월~현재 연세대학교 전기전자공학과 박사과정
<관심분야> 실시간 고장포용 시스템, 지능형 홈네트워크

정 재 윤 (Jaeyun Jung) 준회원

2004년 2월 연세대학교 전기전자공학과 졸업
2006년 2월 연세대학교 전기전자공학과 석사
2006년~현재 삼성전자 정보통신총괄 무선사업부 연구원
<관심분야> 홈네트워크, RTOS 기술

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