실시간 CORBA 시스템 구현

한대만[0], 한윤기, 최만억, 구용완
수원대학교 컴퓨터학과
han38[0]@hanmail.net, yunkihan@mail.suwon.ac.kr, argos12@mail.suwon.ac.kr, ywkoo@mail.suwon.ac.kr

Real-Time CORBA System Implementation

Dae-Man Han[0], Yun-Ki Han, Man-Uk Choi, Yong-Wan Koo
Dept. of Computer, The University of Suwon

요 약

본 논문에서는 효율적으로 객체를 관리하기 위해 객체 그룹의 개념을 도입하여 클라이언트/서버 형태의 분산 처리 시스템 내에서 특정 시간에 서비스가 지원될 수 있도록 실시간 처리를 할 수 있는 CORBA 시스템을 구현한다.

1. 서 론

본 논문에서는 효율적으로 객체를 관리하기 위해 객체 그룹의 개념을 도입한다. 또한, 현재 클라이언트/서버 형태의 분산 처리 시스템 내에서 클라이언트내의 응용 시스템들이 서버에게 특정 시간 안에 서비스가 지원될 수 있도록, 객체 그룹을 이용하여 실시간 처리를 할 수 있는 CORBA 기반 분산 실시간 시스템을 구현한다.

2. 관련연구

2.1. 객체 그룹의 정의

분산 컴퓨팅 환경에서 서비스나 시스템은 다수의 노드에 분산되어 있는 객체들을 분산객체(Distribute Object)라 하며, 이러한 분산객체들로 구성된다. 그러나 분산된 객체들은 응용 프로그램 개발자들에게 복잡한 설계와 관리의 어려움을 주고 있다. 그래서 분산 객체들의 복잡한 설계와 관리의 어려움을 해결하기 위해 객체들을 하나의 집합단위로 구성하는 객체그룹(Object Group)에 대한 개념이 도입되어, 객체그룹마다 관리를 할 수 있게 되었다.

객체그룹은 분산된 객체들의 집합으로 이루어져 있으며, 그룹내의 객체들을 관리하는 그룹 관리자(Group Manager)를 두어 그룹단위로 관리 및 유지되고 있다.[O97]

객체그룹은 분산 컴퓨팅 환경의 위치에 관계없이 다른 객체들간이나 객체그룹간의 접속이 가능해야하고, 가용성 및 신뢰성 등을 지원하고 있다.

객체그룹은 외부의 다른 객체집합과의 인터페이스를 위해 각각 관리적 측면에서 접근할 수 있는 그룹 관리자(Group Manager)와 그룹을 생성하거나 삭제할 수 있는 객체그룹(Object Group)과 객체그룹내의 메시지를 송수신할 수 있는 메시지 전송자(Message Sender)와 메시지 수신자(Message Receiver), 실제 처리를 담당하는 그룹 서비스(Group Service),그리고 객체의 정보와 상태를 나타내기 위한 그룹정보(Group Information)로 구성된다.

다음은 객체그룹의 특성을 나타낸다.

- 객체그룹은 캡슐화되어 구현된다.
- 객체그룹은 분산객체의 집합으로 구성된다.
- 객체그룹은 그룹내에 하나의 그룹 관리자를 포함한다.
- 객체그룹의 객체들은 그룹 관리자를 통하여 그룹에 가입, 탈퇴한다.
- 객체그룹 내의 각 객체들은 외부의 객체와 인터페이스를 위해 그룹 관리자를 호출한다.

2.2. 실시간 스케줄링

실시간 스케줄링은 한정된 컴퓨터 자원을 극대화 할 수 있는 기법이다. 일반적인 운영체제의 스케줄링 방식은 FCFS(Fist Come First Served), SJF(Shortest job First), Priority Scheduling, Round Robin Scheduling, Mutilevel Queue

Scheduling, Mutilevel Feedback Queue Scheduling이 있다.[AP94] 하지만 이와같은 스케줄링은 범용 운영체제를 위해 개발된 스케줄링 방법이므로 엄격한 시간 제약이 있는 실시간 시스템에 적용하기에는 무리가 있다. 따라서 실시간 시스템을 위한 별도의 스케줄링 기법이 필요하다. 실시간 스케줄링 기법은 시간 구동 스케줄링과 우선순위 구동 스케줄링의 두 가지로 구분할 수 있다.

3. 실시간 CORBA

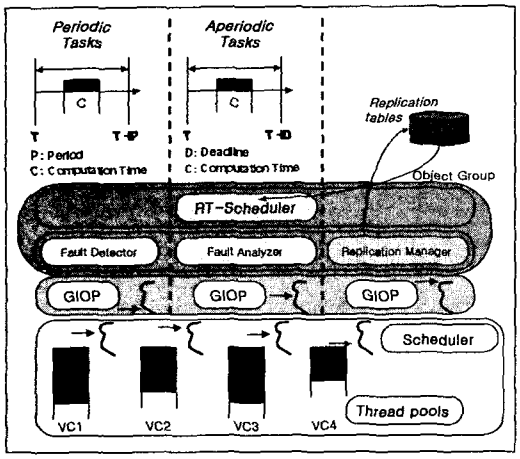
3.1. 실시간 CORBA ORB 구조

실시간 CORBA 구조는 고정 우선순위(fixed priority) 스케줄링을 사용하고 CORBA 시스템에서 예측가능성(predictability)를 지원하도록 표준이 설계 되어있다. 이 표준은 기존의 CORBA 표준안에 채택된 OMG Messageing Specification을 확장하여 사용하고 있으며, RT-CORBA 사양은 Messageing Specification의 QoS 정책을 활용하여 사용하고 있다. 실시간 코바 모듈은 우선순위 모델 및 정책, 스레드 관리, 공유 자원 보호, 클라이언트와 서버의 연결, 프로토콜 정책 등의 대하여 인터페이스를 정의하여 사용자로 하여금 실시간 CORBA 모듈을 활용할 수 있도록 하였다.

3.2. 실시간 CORBA 시스템의 설계

본 논문에서 제안한 실시간 시스템의 구조는 기존의 CORBA에 객체 그룹을 정의하고, 이러한 그룹 통신 위에서 실시간성을 부여하기 위해 RM 스케줄링 방식으로 주기 타스크를 스케줄링 하게 하였다.

[그림3-1]은 본 논문에서 제안한 실시간 시스템 구조이다



[그림3-1] 제안한 실시간 시스템 구조

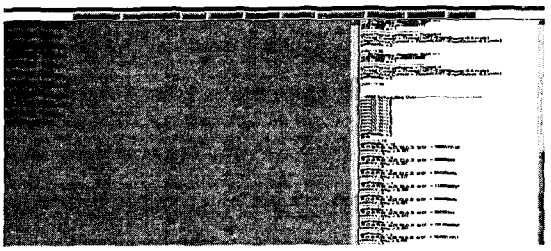
이 방법은 주기 타스크들을 RM 알고리즘으로 스케줄할 때, 스케줄의 각 시점에서 비주기 타스크를 위해 얼만큼의 시간을 사용할 수 있는지 하는 정보를 미리 계산해 두고 복구블록의 우선순위를 결정하는데 이용한다. 주기타스크에는 고정우선순위를, 복구할때에는 동적우선순위를 부여하는 혼합적인 우선순위 부여 방식이다.

```
while( ∀j∈hp(i) == TRUE){

  P_i(t) = jC_i + Σ_{k=1}^{i-1} ( ⌈ t/T_k ⌉ × C_k ) ;

Current_Table[i][Current_Time ] = P_i(t) ;

// i 레벨의 우선순위에 현재의 시간을
        표시하는 테이블 위치에 저장

// 우선순위가 i 인 주기 타스크와
    i 이상의 주기 타스크 집합의 수행시간 계산. }

return P_i(t) ; // 주기 타스크의 수행시간 반환
```

[알고리즘1] 주기 타스크의 수행시간 계산 알고리즘의 의사코드

4. 성능분석 및 향후연구 과제

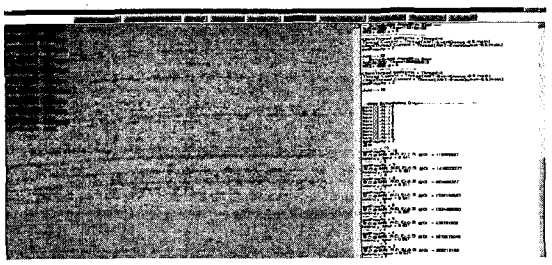
4.1 성능분석



[그림4-4] 주기 : RM , 비주기 : EDF

[그림4-4]는 주기 타스크는 RM으로 스케줄링하고, 비주기 타스크에 대하여 마감시간이 짧은 타스크에 높은 우선 순위를 주는 EDF에 대한 화면이다.

[그림4-5]는 주기 타스크는 RM기법으로 스케줄링하며, 비주기 타스크는 슬랙 스틸링기법을 이용하여 우선순위를 부여한 화면이다. 위의 구현에서 결함이 발생하지 않은 타스크에 대해서는 RM으로



[그림4-5] 주기 : RM , 비주기 : SS

스케줄링 하고, 결함이 발생한 타스크에 대하여 EDF, SS, RM으로 스케줄링 한 이유는 복구블럭 내의 타스크들에게 각각의 스케줄링 알고리즘을 적용하여 마감시간내의 응답시간을 비교하기 위해서이다.

본 실험에서 비교하려는 세 개의 알고리즘은 모두 RM 알고리즘을 기반으로 하고 있으며 이 RM 알고리즘은 주기의 크기에 의해 우선순위를 정하므로 주기들의 조합에 가장 민감하다. 본 논문에서 제안한 타스크의 수행시간을 고려하여 복구블록의 주기를 정의하므로 수행시간의 형태에도 영향을 받는다.

논 논문의 성능 분석결과는 다음과 같다. 결함이 발생하지 않았을 경우의 응답성을 측정해 본 결과 0.3 ~0.55의 빠른 응답성을 보였지만, 결함이 3% 발생했을 경우, EDF, RM, SS의 응답성은 각각 0.75~0.9, 0.9~1.23, 0.9~1.23을 보였다. 또한, 결함이 5% 발생했을 경우, EDF, RM, SS의 응답성은 각각 1.1~1.17, 1.2~1.5, 1.3~1.55을 보였고, 결함이 10% 발생한 경우, EDF, RM, SS의 응답성은 각각 2.0~2.3, 2.1~2.8, 2.3~3을 보였다. 본 실험 결과에서 알 수 있듯이 결함 허용성을 부여한 스케줄링의 경우에는 EDF, RM, SS 순으로 응답성을 보였으며, 결함이 발생하지 않았을 때와 결함이 발생했을 경우의 차는 결함의 발생 빈도에 의해 응답성이 떨어짐을 알 수 있다.

4.2. 향후연구 과제

추후 연구 과제로는 다양한 방법의 스케줄링 정책을 지원할 수 있도록 스케줄러의 구조를 보완하고, 단순한 시간적 제약을 가지는 입력값이 아닌 멀티미디어 데어터의 QoS 데이터, 병행처리 정보, 실제 클라이언트의 중요도에 따라 데이터를 적용할 수 있도록 설계한다.

[참고문헌]

[1] Abraham Silberchatz, and Peter B. Galvin, Operating System Concepts, Addison-Wesley, 1994.

[2] B. Sprunt, L. Sha, and J. P. Lehoczky, "Aperiodic Task Scheduling for Hard Real_Time Systems", Real-Time Systems, Vol. 1, pp. 27-69.

[3] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment", Journal of the ACM, vol. 20, no. 1, pp.46-61, 1973.

[4] Douglas C. Schmidt,et al. " The Design of the TAO Real-Time Object Request Broker", Computer Communication Vol 21. No 4, pp294 ~ 324, 1998.

[5] David L. Levine, Christopher D. Gill, and Duuglas C. Schmidt, "Dynamic Scheduling Strategies for Avioics Systems Conferences, 1998.

[6] Douglas C. Schmidt, Fred Kuhns, " An Overview of the Real-Time CORBA Specification ", IEEE Computer special issue on Object-Oriented Real-Time Distributed Computing, 2000. 6

[7] Felice Balarin, Luciano Lavagno, Praveen Murthy, and Alberto Sangiovanni-Vincentelli, "Scheduling for Embedded Real-Time System", IEEE Design and Test of computer, Vol. 15, No,pp71-82, January 1998.

[8] R. I. Davis, "Approximate Slack Stealing Algorithm for Fixed Priority Pre-emptive System", University of York, England.

[9] S. Eun, S. R. Maeng, and J. W. Cho, "A new scheduling problem in responsive systems," Submitted to IEICE trans. INF. & Sys.

[10] S. R. Thuel and J. P. Lehoczky, "On-line Scheduling of Hard Deadline Aperiodic Tasks in Fixed-Priority System", In Processing of the 14th Real-Time Systems Symposium.