# 분산 실시간 환경을 위한 신뢰성있는 그룹 관리 기법

남광현[O] 홍영식
동국대학교 컴퓨터공학과
{rhkdgus, hongys}@dgu.ac.kr

# A Reliable Group management for the Distributed Real-time Environment

Kwang-Hyun Nam[O] Young-Sik Hong
Dept. of Computer Engineering, Dongguk Univ.

## 요 약

인터넷의 대중화로 컴퓨터 네트워크의 사용이 보편화 되었고, 그룹통신 기법이 화상회의등 많은 응용분야에서 활용되고 있다. 분산 실시간 시스템에서 그룹 통신의 신뢰성을 보장하는 일은 매우 중요하다. 또, 그룹 통신에서 효과적으로 그룹을 관리할 필요가 있고, 현재 실시간 환경에서의 그룹통신은 네트워크의 부하 때문에 생기는 패킷의 손실과 노드의 고장 같은 여러 가지 문제점들을 가지고 있다. 본 논문에서는 이러한 문제점들을 개선시켜 보고자 고장이 발생했을 때에도 그룹 변경을 가능하게 하는 분산 실시간 환경을 위한 신뢰성 있는 그룹관리기법을 제안하고, 실시간 시뮬레이션을 통해 제안된 기법을 평가한다.

## 1. 서 론

최근 몇 년간 활발하게 컴퓨터 네트워크의 사용이 대중화되면서 서로 다른 두 대의 컴퓨터 자료 교환이 대부분이었다. 그러나 이미 방대해진 인터넷 환경에서 두 대의 컴퓨터사이의 자료 교환은 많은 제약이 생겨나게 되었고 더 많은 컴퓨터와의 자료교환을 필요로 하게 되었다. 성능 면에서나 활용의 정도에 있어서, 이러한 한계를 극복하기 위하여 여러 가지 방법론이 제시되어 왔고, 더 많은 컴퓨터와 효율적으로 통신을 하기 위해서 그룹을 지정해서 그룹 내에 있는 컴퓨터와 통신을 하는 그룹 통신이 다양하게 사용되고 있다.

일반적으로 그룹통신 시스템은 분산 응용프로그램들의 오류를 대비하기 위해 메시지 블로킹(Message blocking)을 하며 블로킹을 하는 동안 다른 작업의 간섭으로 인한 오류를 방지하고 그룹을 쉽게 만들 수 있도록 한다. 그룹 통신 시스템은 그룹을 관리하는 서비스와 믿을 수 있는 멀티캐스팅 서비스를 제공하며, 그룹 관리 서비스는 그룹의 멤버가 바뀔 때마다 그룹의 뷰(View)를 유지하고 이 정보를 멤버들에게 전송한다.[1] 신뢰성 있는 그룹 통신을 하기 위해서는 뷰의 유지가 중요한 부분이다. 본 논문에서는 뷰의 관리와 유지를 하기 위해 기존의 그룹관리를 기초로 하여 분산 실시간 환경을 위한 그룹 관리 방법을 제시하고 시뮬레이션하였다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 본 연구와 관련된 연구를 정리하고, 3장에서는 제안된 그룹관리 기법을 설명하고 4장에서는 실험 결과를 평가하고 분석한다. 마지막으로 5장에서는 결론 및 향후 연구 방향을 제시한다.

## 2. 관련연구

LAN 환경에서의 그룹 통신 시스템으로는 초창기의 ISIS로부터 최근의 Transis, Horus, Totem, RMP와 같은 시스템들이 있다. 이 시스템들은 뷰 동기(View Synchrony)와 확장된 가상동기(Extended Virtual Synchrony)와 같은 그룹통신 모델에 대해 연구를 했다. 뷰 동기(View Synchrony)방식을 사용한 그룹통신 시스템은 수년간 많은 발전을 거듭해 왔다. WAN환경에서는 Client-Server구조를 가진 Transis시스템의 FIFO queue에 변경된 View를 유지하고 동의 알고리즘인 Fast Agreement와 Slow Agreement 통해 뷰 변경을 하는 그룹 통신시스템이 연구되었고[2][3], View의 갱신을 긍정적으로 예측하여 적용하고 틀렸을 때는 Roll-back 알고리즘을 적용하여 기존의 가상동기를 효율적으로 적용하는 Optimistic Virtual Synchrony[6]가 연구되었다.

## 3. 그룹 관리 기법

실시간 분산 그룹통신 시스템에서 신뢰성있는 그룹통신을 하기 위해서는 뷰의 동기화가 필요하다. 그룹 관리를 위해서는 그룹을 생성하고, 가입(join), 탈퇴(leave)를 관장하는 노드가 필요하며, 이러한 작업을 그룹관리기 노드가 담당한다. [그림 1]은 그룹관리기 노드가 뷰를 변경할 때 뷰 변경에 대한 블로킹(blocking)하는 과정을 보인다. 블로킹을 하는 이유는 뷰 변경이 일어나는 도중에 다른 뷰 변경 요청을 방지하여 변경에 대한 간섭을 줄이기 위해서다. 그룹내의 모든 멤버들은 메시지

수신에 대한 FIFO 큐를 유지하여 고장등으로 인한 수신된 뷰가 유효하지 않을 때를 대비한다.

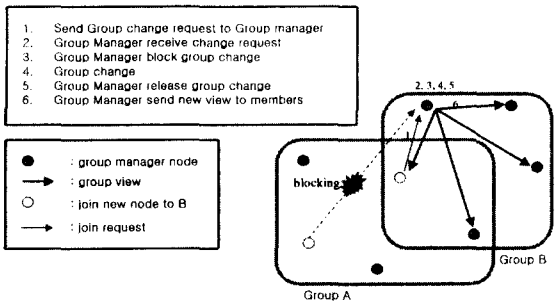


그림 1. 뷰 변경후 전송

그룹 변경의 유형은 가입, 탈퇴, 생성, 삭제 4가지 경우를 고려하였고, 본 연구에서는 노드의 고장에 관한 메시지가 있는 경우에도 그룹 변경이 가능하도록 했다.

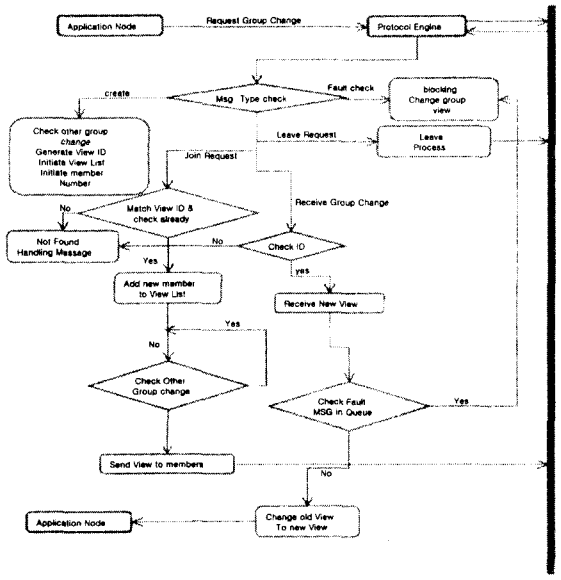


그림 2. 뷰 변경 흐름도

[그림 2]는 그룹 변경이 이루어지는 과정을 보이는 순서도이다. 그룹 변경 요청이 들어오면 그룹관리기는 메시지 타입을 확인하고 그룹의 ID를 체크한다. 가입이 발생했을 때 그룹관리기는 가입 요청을 한 노드가 이미 뷰에 속해있는 노드인지 확인하고 뷰에 추가한다. 탈퇴가 발생하면 그룹관리기는 뷰에서 요청을 한 노드를 삭제하고 뷰를 재구성하고 변경된 뷰를 멤버들에게 전달한다. 이 과정에서 그룹관리기는 탈퇴가 성공적으로 이루어 졌다는 메시지를 요청한 노드에게 보낸다.

임의의 노드에서 고장이 발생했을 때 고장이 발생한 노드에 대해서 그룹관리기는 고장난 노드에 대해 뷰 변경을 실시하여야만 잘못된 뷰로 인한 오동작을 방지할 수 있다. 그래서 고장난 노드에 대해서 뷰에서 삭제를 하여 뷰를 갱신한다. 고장난 노드는 외부에서 메시지 형태로 그룹관리기가 수신한다고 가정한다. 노드가 고장났다는 메시지를 수신한 그룹관리기는 자신이 관리하고 있는 모든 멤버에게 취소(Abort) 메시지를 전송한다. 즉, 전에 보낸 뷰가 유효하지 않다는 메시지를 보내는 것이다. 그룹에 속해있는 모든 노드는 큐에 수신된 뷰에 대하여 일정시간동안 보관하고 있다가 그룹관리기로부터 취소(Abort)메시지를 수신하지 않으면 유효하다고 판단하여 새로운 뷰로 자신의 뷰를 갱신한다.

## 4. 실험 및 분석

제안된 그룹관리 기법의 시뮬레이션을 위해 본 연구에서는 TMO(time-triggerd message-triggerd object) 시스템을 활용하여 실험하였다.[4][5] [그림 3]에서 보는 것처럼 SpM (Spontaneous Method), SvM(Service Method), ODSS(object data store) 3개 부분으로 구성된 객체이다. SpM은 시간조건인 AAC(Automatic Activation Condition)가 만족되면 자동으로 호출되는 특징을 가지고 있고, SvM은 외부로부터 메시지등 이벤트에 의해 반응하는 메소드이다. 또한 ODSS는 SpM과 SvM 사이에서 데이터 공유 및 동기화를 위해 쓰인다.

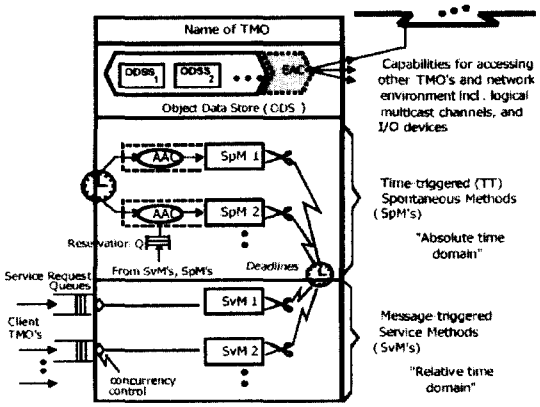


그림 3. TMO의 구성[5]

제안된 그룹 관리 기법을 TMO환경에서 제공된 API를 사용한 C++ 프로그램으로 작성하여 3대의 Pentium III PC 및 Window 2000시스템에서 실험하였다. [4] [5]
그룹 변경 과정은 [그림 2]에서 나온 순서도에 따라 구현하였고 뷰 변경이 보장되는 지 알아보기 위하여 25초 동안 실험하였으며, 25번의 그룹 변경을 요청하여 모든 그룹 멤버의 뷰가 모두 일치하는 지 비교하였고 변경 중 10%의 고장 발생 메시지를 수신했을 때 그룹 뷰 변경이 이루어지는 가를 검토하였다. [표 1]은 실험에 사용된 파라미터를 기술한 것이다.

[표 1] 실험 파라미터

| Parameter | Value |
|---|---|
| Node | 4, 6, 8 |
| Message Size | 180bytes |
| Release Time (Queue에 유지하는 시간) | 200ms |

[표 2]는 그룹관리기가 변경 요청을 받아 변경을 완료하고 변경된 뷰를 그룹의 멤버에게 전달했을 때 모두 같은 뷰를 가지는 시간을 측정한 값과 고장이 발생했다는 메시지를 받았을 때 그룹관리기에서 뷰의 무효화를 위해 취소(abort) 메시지를 전달하는 시간의 최대값을 측정하였다.

각 그룹멤버들이 뷰를 수신하여 큐에 유지하는 시간은 200ms로 하였고, 노드의 수를 4개에서 8개로 증가시키며 실험하였다.

[표 2] 그룹 변경의 전송 지연시간

| 노드수 | 그룹 변경 시간(고장시) | 무효화 성공시간 |
|---|---|---|
| 4 | 평균 210ms(평균 621ms) | 최대 41ms |
| 6 | 평균 210ms(평균 600ms) | 최대 60ms |
| 8 | 평균 210ms(평균 710ms) | 최대 61ms |

실험결과에서 그룹 변경이 완료되는 시간은 평균 210ms이고, 뷰를 큐에 유지하는 시간 200ms와 전송시간 10ms 임을 보인다. 뷰 변경을 위해 큐에 뷰를 유지하는 시간이 무효화 성공시간보다 크면 변경에 대한 취소가 이루어진다.

고장 사실을 받아 뷰를 취소시키는 것은 그룹 관리의 신뢰성을 유지하는 중요한 의미를 갖는다. 왜냐하면 뷰를 모두 변경하거나 취소하기 때문에 일관된 특성을 가진다.

즉, 본 연구에서 제안한 그룹 관리 기법은 고장이 발생한 경우에도 뷰를 일관성 있게 변경한다.

## 5. 결론 및 향후과제

본 논문에서는 분산 실시간 시스템을 위한 그룹 관리기법을 실험하고 결과를 분석하였다. 실험 목적인 제안한 그룹 관리기법이 실시간 환경에서 사용 가능한지 검토해보고, 뷰의 일관성과 신뢰성이 보장되는지 실험을 하였다. 본 논문에서 제안한 그룹관리기법은 고장이 발생했을 때에도 신뢰성 있는 그룹 관리를 보장한다.

향후 과제로는 비동기 시스템에서 많이 사용되어지는 기존의 뷰 동기(View Synchrony)를 사용한 그룹관리 기법과 그룹변경 지연시간을 비교실험, 구현을 하여 성능평가를 하고 그룹관리 기법을 콤포넌트 형태로 구현하여 멀티캐스팅 시스템이나 화상회의 시스템에 적용할 계획이다.

## 6. 참고문헌


[1] O. Babaoglu, A. Bartoli and G. Dini, "Group Membership and View Synchrony in Partitionable Asynchronous Distributed Systems", IEEE Transactions on Computers, vol 46, No. 6, pp.642-658, June 1997

[2] Idit Keidar and Roger Khazan,"A Client-Server Approach to Virtually Synchronous Group Multicast: Specifications and Algorithms.",In the 20th International Conference on Distributed Computing Systems (ICDCS), pages 344-355, April 2000.

[3] Idit Keidar, Jeremy Sussman, Keith Marzullo, and Danny Dolev,"A Client-Server Oriented Algorithm for Virtually Synchronous Group Membership in WANs."In the 20th International Conference on Distributed Computing Systems (ICDCS), pages 356-365, April 2000.

[4]K. H. Kim, M.Ishida, J. Liu, "An Efficient Middleware Architecture Supporting Time-Triggered Message-Triggered Objects and an NT-based Implementatio n", Proc. 2nd IEEE CS International Symposium On Object-Oriented Real-time Distributed Computing(ISORC' 99), pp. 54-63., St. Malo, France, May, 1999.

[5] Eltefaat Shokri and Kane Kim,"TMO-Based Programming in COTS Software /Hardware Platforms: A Case study",Proc. ASSET'99(1999 IEEE Symp. on Application-Specific Systems and Softward Engineering & Technology), pp.88-94, Richardson, TX, March 1999.

[6] Jeremy Sussman, Idit Keidar, and Keith Marzullo, "Optimistic Virtual Synchrony. ", In the 19th IEEE Symposium on Reliable Distributed Systems (SRDS), pages 42-51, Nurnberg, Germany, October 2000.