작업 동기화를 위한 작업공간 트랜잭션 모델

오암석
동명정보대학교 멀티미디어공학과

Workspace Transaction Model for Job Synchronization

Am-Suk Oh
Department of Multimedia Engineering,
Tongmyong University of Information Technology

요 약

멀티미디어 응용들은 공동 작업을 위한 작업 동기화 지원이 필요하다. 이 논문은 데이터베이스를 이용한 멀티미디어 응용 개발시 한 트랜잭션의 처리가 긴 공동 작업 환경에서 작업 동기화를 지원하기 위한 새로운 공동 트랜잭션 모델을 제안한다. 이 논문에서 제시하는 모델은 기존의 로킹 기법을 사용하지 않고 여러 작업공간에서 수행되는 작업간의 동기화를 제공한다.

1. 서론

멀티미디어 응용들은 공동작업 환경에서 트랜잭션의 처리 시간이 길며 대화식(interactive) 처리가 요구된다. 또한 작업 공간(workspace)사이에는 오랜 기간 동안 서로 객체를 공유하며 공동으로 작업한다. 이 때, 한 트랜잭션이 공유 객체를 사용하는 동안 다른 트랜잭션들을 기다리게 하는 것은 동시성을 저하시키므로 사용자 그룹이 오랜 기간 동안 기다리지 않고 긴 트랜잭션들을 독립적으로 작업할 수 있도록 해야 한다. 그러나 잠금(locking)과 같은 기존의 병행성 제어 기법들은 너무 제한적인 동기화로 인하여 멀티미디어 응용에 적합하지 않다.

이 논문에서는 멀티미디어 응용들을 위한 공동 작업 환경에서 데이터베이스를 이용하여 작업 공간간 객체를 공유하며 공동작업할 때 작업의 동기화를 지원하는 작업공간 트랜잭션 모델(workspace transaction model)을 제시한다. 이 논문에서의 작업 환경은 서버에 공용 멀티미디어 데이타베이스가 있고 클라이언트는 여러 작업 공간들로 구성되어 있다고 가정한다. 이 모델에서 사용자는 긴 트랜잭션 처리를 위한 자신의 작업 공간을 가진다.

이 논문에서 제시하는 트랜잭션 모델은 작업공간에서 사용자가 독자적으로 변경을 수행하도록 하며 관련 트랜잭션간 공동 조정(coordination)은 사용자 정의 시점에서 처리된다. 즉, 관련 작업 공간간 직접적인 통신에 의해 공동 작업 따른 일관성 유지 및 변경 상충 문제를 처리한다.

2. 작업공간 트랜잭션의 특성

작업공간 트랜잭션(workspace transaction)은 각 작업공간에서 독자적으로 수행되는 작업단위이다. 작업공간 트랜잭션은 긴 시간이 소요되는 대화식 트랜잭션(interactive transaction)이다. 일반적으로 작업공간 트랜잭션은 다음과 같은 특성을 가진다. 대개 사용자들간의 대화가 요구되는 긴 작업이므로 이로 인하여 처리가 오랜기간 동안 계속된다. 따라서 작업공간 트랜잭션의 철회는 작업 손실면에서 기존의 트랜잭션보다 오버헤드 문제가 더 심각하

다. 둘째, 작업공간 트랜잭션은 여러 부태스크들의 집합으로 구성되며 사용자는 전체 트랜잭션을 포기할 필요 없이 부태스크만을 포기할 수 있다. 세째, 트랜잭션들이 대화적으로 처리되므로 한 트랜잭션의 포기가 다른 트랜잭션들의 포기를 가져오는 연쇄 복귀(cascading rollback)문제가 발생할 수 있다. 네째, 작업의 처리량이 사용자간 빠른 응답 시간으로 정의되며 처리량의 최적화를 위해 응답시간이 빨라야 한다. 다섯째, 기존의 동기화 제어 방법이 역 효과를 가져온다. 왜냐하면 작업공간 트랜잭션은 사용자들 간의 공동 조정이 요구되며 사용자들은 서로 작업을 진행해 가면서 자신들의 트랜잭션을 처리한다. 따라서 독자적으로 작업 계획을 미리 세워 진행하는 것이 어려우며 때로는 작업의 효율성을 떨어 뜨릴 수 있기 때문이다.

이 논문에서 제시하고자 하는 작업공간 트랜잭션 모델도 이와 같은 특성을 가지는 대화적 긴 트랜잭션(interactive long trnsaction)이다.

3. 작업공간 트랜잭션 모델링

이 논문에서 정의하고자 하는 작업공간 트랜잭션은 완료전까지 대화식으로 공동 작업을 수행하는 긴 트랜잭션이며 공유 객체 변경시 사용자 정의 시점에서 트랜잭션 연산 실행에 의해 작업 동기화를 수행한다.

```
Interactive Long Transaction{
    ILT: (TReadSet, TLong_UpdateSet, TSave-propagateSet, TCommitSet)
}
```

- TReadSet; 트랜잭션이 체크아웃하는 모듈들의 집합
- TLong_UpdateSet; 트랜잭션이 수정작업 중인 모듈들의 집합
- TSave-propagateSet; 트랜잭션이 저장-전파하는 모듈들의 집합
- TCommitSet; 트랜잭션이 체크인하는 모듈들의 집합

그림 1. 작업공간 트랜잭션

그림 1은 작업공간 트랜잭션을 정의한 것이다. 작업공간 트랜잭션은 트랜잭션 연산자들을 가지며 각 작업공간 마다 자신의 작업공간 트랜잭션들이 존재한다. 작업공간 트랜잭션은 공용 데이타베이스에서 객체를 복사할 수 있으며 자신의 작업공간에서 객체를 변경하여 변경된 객체를 공용 데이타베이스에 기록한다.

그림 1의 read 연산은 객체를 공용 데이타베이스에서 작업공간으로 복사하며 체크아웃(check out)의 의미이다. commit 연산은 변경한 객체를 공용 데이타베이스로 copy-back(역 복사)하기 위한 연산이며 체크인(check in) 의미이다.

작업공간 트랜잭션 처리는 많은 사용자들과의 공동 작업이 요구되며 트랜잭션이 완료되기 전에 commit되지 않은 데이타 즉, 일부 변경된 내용을 관련 트랜잭션들에게 통보하는 것이 바람직하다. save-propagate 연산은 트랜잭션이 작업을 수행하는 동안 트랜잭션의 중간 처리 결과를 자신의 작업공간에 저장하고 관련 트랜잭션으로 변경전파하기 위해 새로 정의한 연산자이다.

4. 구현

이 장에서는 실제 작업 동기화를 지원하는 작업공간 트랜잭션의 처리 과정을 GUI를 통하여 전반적으로 나타낸다.

그림 2는 작업공간 트랜잭션의 초기화면이다. 그림 2에서 왼쪽 부분은 공용데이타베이스를 관리하는 서버 프로세스(server process)를 나타내며 오른쪽 부분은 자신의 작업공간에서 독자적으로 작업을 하고 있는 클라이언트 프로세스(client process)를 나타낸다.

서버 프로세스의 상단에는 현재 서버 데이터베이스에 있는 모듈들을 예시하며 하단에는 클라이언트에서의 작업 공간들로 부터 수행되는 트랜잭션 연산들의 처리 과정을 예시한다. 클라이언트 프로세스는 여러 명령어 버튼으로 구성되어 있으며 모든 트랜잭션 연산들의 수행이 이 명령어 버튼의 클릭에 의해 동작한다.

명령어 버튼의 기능들은 다음과 같다:

- ***check_out***: 작업 공간의 트랜잭션이 서버에 있는 모듈을 복사한다.
- ***EDIT***: 서버에서 가져온 모듈에 대해 수정 작업한다.
- ***checkpoint:*** 일부 수정이 완료된 부분을 관련 작업 공간으로 변경 전파한다.
- ***Display server:*** 서버 DB에 있는 모듈들을 디스플레이한다.
- ***Display client:*** 작업 공간에 있는 모듈의 내용을 디스플레이한다.

■ *Check_in:* 작업 완료된 모듈을 서버로 역 복사 한다.

■ *END:* 작업을 종료한다.

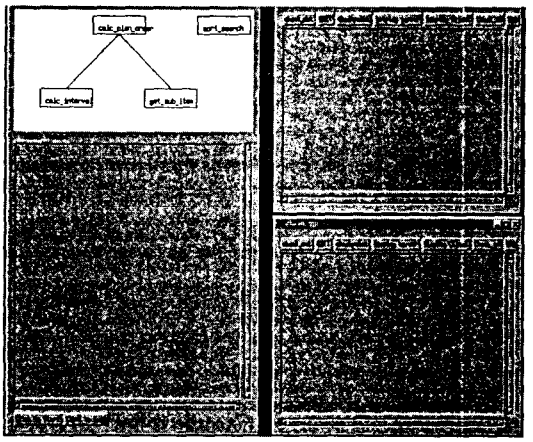

그림 2. 초기 작업 화면

그림 3은 작업 공간 A와 B가 서버 데이타베이스에서 모듈 sort_search를 복사하는 과정이다. 그림 4는 작업 공간 A가 서버에서 가져온 모듈 sort_search를 새로운 함수 search()를 에디터 모드 상에서 작업하는 것이며 그림 5에서 작업 공간 A의 작업 결과를 작업 스크린상에 나타내고 있다.

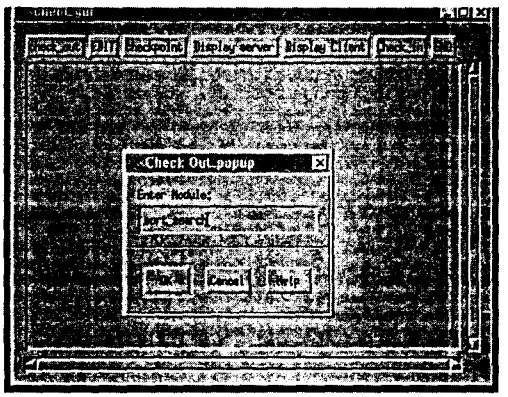

그림 3. 작업 공간 A와 B의 체크아웃

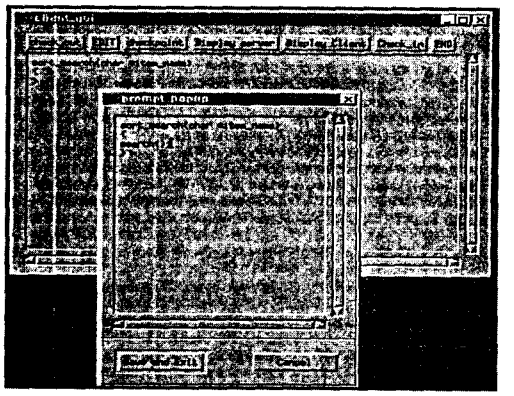

그림 4 작업 공간 A의 edit

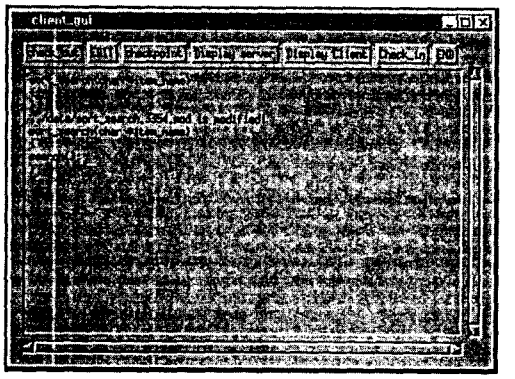

그림 5 작업 공간 A의 edit 결과

그림 6은 작업 공간 B가 서버에서 가져온 모듈 sort_search에 새로운 함수 sort()를 에디터 모드 상에서 작업하는 것이며 그림 7은 작업 공간 B의 작업 결과를 자신의 작업 스크린상에 나타내고 있다.

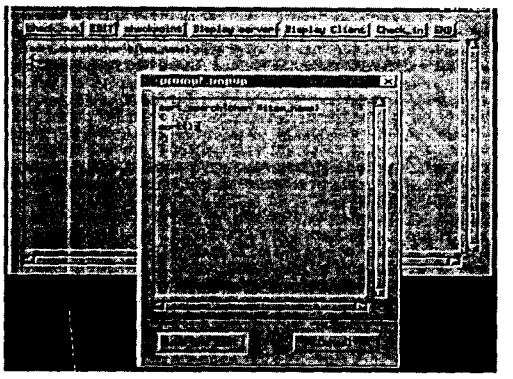

그림 6 작업 공간 B의 edit

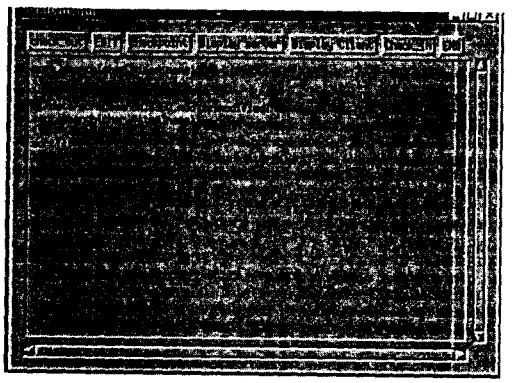

그림 7 작업 공간 B의 edit 결과

그림 8에서는 작업 공간 A와 B가 자신들이 일부 작업한 변경 내용을 변경 전파하기 위해 트랜잭션 연산인 save-propagate를 수행한 후 작업 공간 A와 B가 서로 변경 전파된 내용을 접수하고 다이얼로그 박스에 있는 메뉴를 통하여 내용을 검사한 후 (세번째 메뉴 선택) 수락 여부를 통지하는 것을 나타낸다.(수락시 네번째 메뉴, 거부시 다섯번째 메뉴 선택) 이 예에서는 수락한 것으로 가정한다.

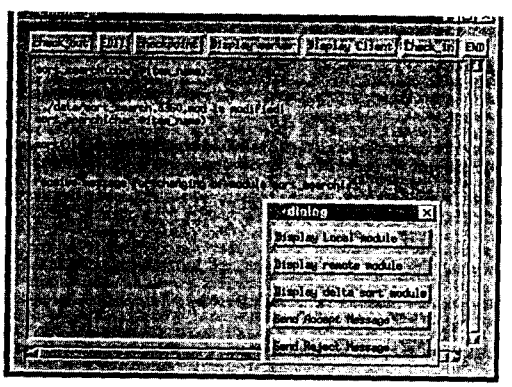

그림 8 작업 공간 A와 B의 변경 전파 접수 및 수락 여부 결정

그림 9는 지금까지의 작업 공간 A와 B의 트랜잭션 연산 수행에 따른 서버 프로세스의 처리 과정을 나타낸다.

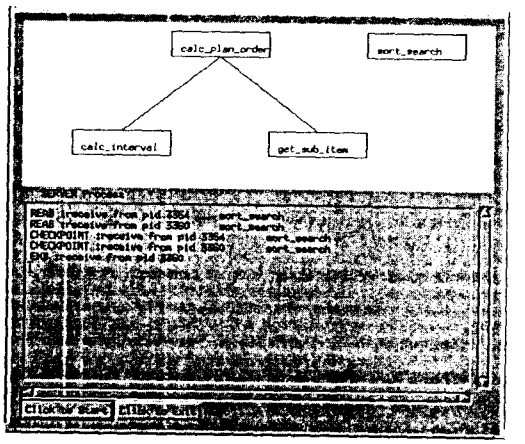

그림 9 서버 객체의 처리 결과

그림 10과 그림 11은 각각 작업 공간 A와 B의 처리 결과이다.

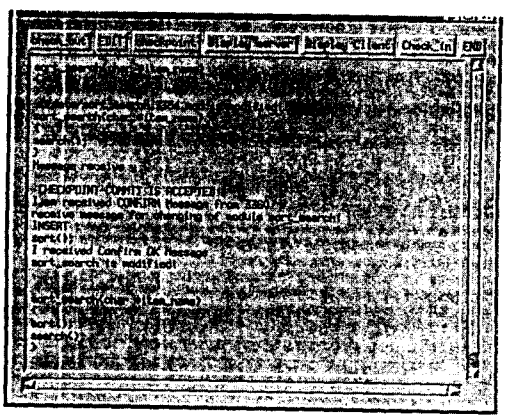

그림 10 작업 공간 A의 처리 결과

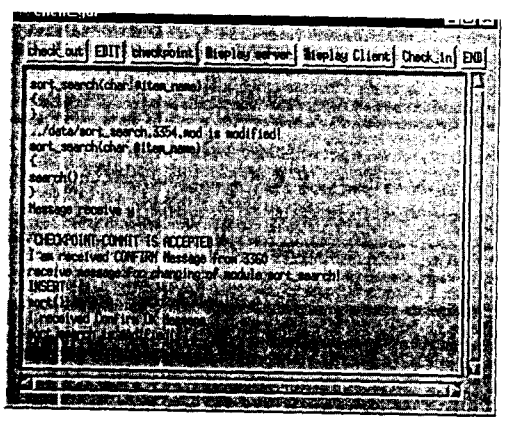

그림 11 작업 공간 B의 처리 결과

5. 결론

이 논문에서는 멀티미디어 응용의 작업 동기화를 위한 작업공간 트랜잭션 모델을 제시하였다. 이 트랜잭션 모델에서는 작업 동기화를 위해 사용자 정의 시점에서 트랜잭션들의 변경을 전파하고 수락/거부를 처리하기 위해 여러 트랜잭션 연산자들을 정의하였다. 특히 save-propagate 트랜잭션 연산자는 자신의 작업 내용을 사용자 시점에서 관련 작업공간으로 전파할 수 있도록 동기화를 제공해준다. 이러한 트랜잭션 연산자들을 기반으로 전반적인 공동 트랜잭션 모델을 설계하고 구현하였으며 기존의 로킹 방법을 사용하지 않고 대화적 작업공간 트랜잭션 환경에서 보다 유연한 작업 동기화가 가능함을 보였다.

참 고 문 헌

[1]Attie, P. C., et al, "Specifying and enforcing intertask dependencies", in Proc. Intl Conf. on Very Large Data Bases, 1993, pp. 134-145

[2]Barghouti, N. S. and Kaiser, G. E., "Modeling Concurrency in Rule-Based Development Environments", IEEE Expert, Vol.5, NO.6, Dec. 1990, pp. 15- 27

[3]Brown, D. R., et el., "Next-Cut: A Computational Framework for Concurrent Engineering", second Int. Symposium on Concurrent Engineering, Feb. 1990, Morgantown, West Virginia

[4]Chrysanthis, P. K., and Ramamritham, K.,"ACTA: A framework for specifying and reasoning about transaction structure and behavior", Proc. ACM SIGMOD Conf. Management of Data, ACM press, 1990, pp. 194-203

[5]Cohen, P., et el., "Multimodal Interaction for Distributed Applications", ACM International Multimedia Conference, 1997, pp. 31-40

[6]Cutkosky, M., et al., "PACT: An Experiment in Integrating Concurrent Engineering Systems", Computer Special Issue on Computer Supported Concurrent Engineering, IEEE. Jan., 1993, pp. 28-37

[7]Dayal, U., et el., "A Transactional Model for Long-Running Activities", in Proc. Intl Conf. on Very Large Data Bases, 1991, pp. 113-122

[8]Dittmann, J., et el., "Interactive Watermarking Environments", IEEE Multimedia Systems '98, 1998, pp. 286-294

[9]Geogakopolous, D., et el,"Specification and management of extended transactions in a programmable transaction environment", In Proc. of the 10th IEEE Int. Conference on Data Engineering, 1994, pp. 462-473

[10]Hoppe, H. U., Zhao, J., "C-TORI: An Interface for Cooperative Database Retrieval", 5th In Conf., DEXA '94, 1994, pp. 103-113

[11]Kaiser, G. E., "Interfacing Cooperative Transactions to Software Development Environments", Office Knowledge Engineering, Vol. 4, No. 1, 1991, pp. 56-78

[12]Kim, W., Modern Database systems: Cooperative Transactions for Multipleuser Environments, Addison-Wesley Publishing Company, 1995

[13]Nodine, M. H. and Zdonik, S. B., "Cooperative Transaction Hierarchies: A Transaction Model to Support Design Applications", in Proc. Intl Conf. on Very Large Data Bases, 1990, pp. 83-94

[14]Rusinkiewicz, M., et al., "Towards a Model for Multidatabase Transactions", International Journal of Intelligent and Cooperative Information Systems, Vol. 1, No. 3, 1992

[15]Rusinkiewicz, M., et al., "Towards a Cooperative Transaction Model - The Cooperative Activity Model-", in Proc. Intl Conf. on Very Large Data Bases, 1995, pp. 194-205

[16]Thomasian, A., "Checkpointing for Optimistic Concurrency Control Methods", IEEE Transactions on Knowledge and Data Engineering, Vol. 7, No. 2, 1995, pp. 332-339

[17]Yu, P. S., Dan, A., "Performance Analysis of Affinity Clustering on Transaction Processing Coupling Architecture", IEEE Transactions on Knowledge and Data Engineering, Vol. 6, No. 5, 1994, pp. 764-786