한국산학기술학회논문지
Vol. 5, No. 3, pp. 247-253, 2004

웹 기반의 신뢰성 있는 예약서비스를 위한 글로벌 트랜잭션 모델에 관한 연구

김수홍[1] · 윤용익[2]

A Study on Global-Transaction Model for the Web-Based Reliable Reservation Service

Soo-Hong Kim[1] and Yong-Ik Yoon[2]

요 약 본 논문에서는 웹 기반의 실시간 예약 서비스 시스템에서 신뢰성을 보장하기 위한 트랜잭션 모델에 대하여 주로 다룬다. 현재 기능별로 나누어져서 처리되고 있던 각각의 트랜잭션을 하나의 트랜잭션으로 묶어서 처리할 수 있는 웹 기반의 글로벌 트랜잭션 모델을 제시한다. 또한 트랜잭션 처리에서 발생되는 예외사항을 다루기 위하여 Fail-Check 프로세스를 제안한다. 그리고 시뮬레이션의 결과 제시한 모델의 타당성 및 가능성이 트랜잭션 처리 과정을 통하여 확인되었다.

Abstract This paper mainly deals with a transaction model for the reliability guarantee in a Web-Based Real-Time Reservation Service System. We suggest the Web-Based Global Transaction Model which binds up each transaction currently divided by a function into one transaction and deals with one transaction. Also, we propose a Fail-Check Process in order to deal with the exceptions to be happened in a transaction process. And the simulation shows us that the model which we suggest is feasible and possible through the transaction processing procedure.

Key Words :

1. 서 론

인터넷 환경의 빠른 보급으로 인터넷 기반 하에서의 전자상거래가 보편화되고 있다. 또한 공연이나 영화, 스포츠, 그리고 기타 교통수단의 이용에 있어서 예약 없이 사용하기가 힘들 정도로 예약 문화가 발달함에 따라 인터넷에서의 실시간 예약은 증가하게 되었다[1].

인터넷에서의 실시간 예약 시스템은 사용자들에게 편리한 예약 서비스를 제공한다. 일반적으로 예약 시스템은 일정과 예매가능 좌석을 조회하는 과정과 결제정보 입력 및 최종 예약을 하는 과정으로 이루어지게 된다 [2-8]. 이때 예매가능 좌석을 조회하는 절차와 최종 예약을 하는 절차 사이에는 필연적으로 사용자의 정보 입력 시간이 필요하기 때문에 일정시간의 시간이 소요된다. 따라서 일반적으로 예약의 단계를 예매가능 좌석을 조회하는 트랜잭션과 최종 예약을 하는 트랜잭션으로 나누게 되고, 사용자의 예매 확정은 최종 예약 트랜잭션에서 결정되는 모델을 택하고 있다. 즉 기존의 인터넷 예약 시스템은 각 종류별로 약간씩 다른 부분이 있지만, 일반적으로 좌석의 잠금(Lock) 없이 사용자가 결재 정보를 입력하고 결재가 완료된 후에야 선택한 좌석에 대하여 예약이 완료되는 모델이 사용되었다[6-8]. 그러나 이러한 모델은 사용자가 증가 할 때 예약을 하기위해 들어온 사용자의 순서에 따라 예약이 이루어지지 않고 사용자의 인터넷 사용 능력이나, 시스템이나 네트워크 등의 성능에 의존해 그 성능에 따라 예약 순서가 결정지어지는 결과를 초래하게 된다. 또한 예약 처음 시점의 잔여 좌석을 예매 종료 시까지 보장할 수 없어 예매 종료 시에 초기의 잔여 좌석의 수를 보장 할 수 없게 된다. 이 경우 인터넷 실시간 예약의 특성상 인기 있는 공연이나 명절 때의 운송수단 예매 같은 경우 다수의 사용자가 짧은 시간 안에 좌석을 선매(preemption)하기위해 동시에 접속할 때 예약 초기 시점의 정보가 예약 완료 시점까지 보장되지 않기 때문에 다수의 사용자는 좌석의 선매에 실패하게 되고 이는 신뢰성이 떨어지는 인터넷 예약의 모델이 될 수 있다. 만약 다원화되어 있는 다수의 트랜잭션을 하나의 트랜잭션으로 관리

[1]상명대학교, 컴퓨터정보통신공학부
[2]숙명여자대학교, 정보통신대학원

할 수 있다면 사용자는 초기의 좌석 정보를 최종 예약 단계까지 보장 받을 수 있기 때문에 인터넷 예약의 신뢰성을 증가시킬 수 있을 것이다. 따라서 인터넷 예약에서 예약의 신뢰성을 증가시키기 위한 방법으로 초기 조회시점부터 예매 좌석을 잠금 하고 이를 최종 예약 종료 트랜잭션까지 유지함으로써 인터넷 예약의 신뢰성을 보장할 수 있는 WBGT(Web-Based Global Transaction) 시스템을 제안하고자 한다.

WBGT 모델은 기존 여러 단계로 나누어져 있던 트랜잭션을 하나의 트랜잭션으로 묶어 처리하는 방식이다. 현재 웹에서의 예약 시스템은 예약 단계를 각각의 특성별로 여러 단계의 트랜잭션으로 구성하여 놓았다. 따라서 현 예약 시스템에서 여러 단계로 나누어져있던 트랜잭션을 하나의 WBGT 로 묶는다면 현 예약 시스템에서 사용자가 겪게 되는 불편을 절감 시킬 수 있으며, 신뢰성 또한 높일 수 있다.

본 논문에서는 예약 시, 처음단계에서의 좌석이 예약 종료 시까지 보장 받을 수 있는 예약 트랜잭션 모델에 대하여 제안하고자 한다. 또한 이로 인해 발생되는 문제에 대한 처리도 제안하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 우선 이와 관련된 연구로 현재 Web에서 서비스되어지고 있는 예약 시스템의 작업흐름도(Workflow) 및 트랜잭션 처리에 대하여 살펴보고, 이에 WBGT 모델 적용의 필요성을 살펴본다. 3장에서는 WBGT의 모델을 제시하고, 4장에서는 WBGT 모델의 시스템 구조 및 알고리즘을 설계한다. 5장에서는 3장과 4장을 토대로 간단한 예약 시스템을 만들어 구현해 보고, 6장에서는 본 논문이 제시한 시스템과 기존 시스템을 비교 분석한다. 마지막 7장에서는 결론 및 향후 연구방향을 제시하였다.

2. 현 예약시스템

2.1 현 예약시스템 분석

현재 서비스되고 있는 항공권, 철도, 고속버스, 공연/영화/스포츠 예약업무의 작업흐름도 및 예약 시스템의 트랜잭션 처리에 대하여 살펴보고자 한다.

각 업무별로 다른 작업흐름도와 트랜잭션 처리 형태를 갖고 있지만, 작업 흐름의 공통분모만을 뽑아서 항공편의 예를 들어 정리를 해 보고자 한다. 우선 작업흐름도를 간략히 정리해 보면, (일정선택 → 항공편 선택)의 R-Transaction과 (탑승자 정보입력 → 구매신청)의 W^1-Transaction, 마지막으로 (결제정보 입력 → 취소 및 구매완료)의 W_2-Transaction의 크게 세 단계의 Transaction으로 이루어져 있다. 이는 사용자가 맨 처음 일정 및 항공편을 선택하고 구매 신청을 하는 R-Transaction으로 첫 번째 단위 업무가 끝난 후, 탑승자 정보 입력과 함께 구매 신청을 하는 W_1-Transaction의 수행으로 두 번째 단위 업무가 끝나게 된다. 마지막으로 결재 정보 입력과 함께 구매 완료를 하는 W2-Transaction의 세 번째 단위 업무가 끝나야 예약이 성공적으로 끝나게 된다. 여기에서 W_1-Transaction의 두 번째 단위 업무가 끝나게 될 때, 시스템은 사용자에게 선택한 좌석에 대하여 예약 확인권을 주는 것과 같은 예약 번호를 부여하여 W_1-Transaction 수행이 끝나고 W_2-Transaction의 수행 전에도 좌석 예약에 대한 확신을 갖게 된다. 그 후 사용자는 그 좌석에 대하여 취소 및 예약 완료의 결정과 함께W_2-Transaction의 세 번째 단위 업무가 끝남으로써 최종적으로 예약 업무를 끝낼 수 가있다[4-5].

이러한 시스템의 트랜잭션 모델은 W_2-Transaction의 예약 완료 업무가 진행되기 전 W_1-Transaction 만으로도 사용자가 예약에 대한 확신 감을 줄 수 있는, 즉 사용자에게 신뢰감을 주는 방법으로는 적당한 트랜잭션 모델이다. 그러나 이 트랜잭션 모델은 사용자가 많을 경우나 사용자의 네트워크 등의 시스템적 상황이 좋지 못 할 경우에는 신뢰성을 떨어뜨리는 모델이 될 수 있다[15].

예를 들어서 위의 그림 1과 같이 현재 10석의 좌석이 남아있을 경우 10명 이상의 사용자들까지도 W_1-Transaction의 수행까지 보장을 받고 마지막 세 번째 단계인 W_2-Transaction에서 차례대로 10좌석까지만 예약이 끝나고 그 이후의 트랜잭션 처리 요청은 모두 실패로 끝나게 된다. 사용자들은 선착순 10명의 사용자가 탑승자 정보와 함께 구매 신청을 하는 W_1-Transaction의 수행까지 가게 되면, 그 후의 W_2-Transaction의 수행에 대해 보장 받는 업무 처리라는 착각을 할 수 있다. 따라서 사용자들이 사용상의 지연이나 시스템 속도 등의 지연으로 W_2-Transaction의 수행을 늦게 할 경우 10

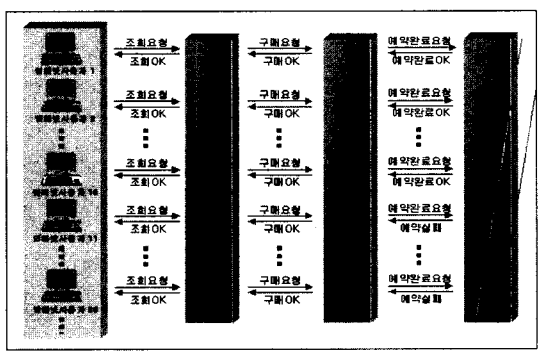

그림 1. 업무별 트랜잭션 처리 응답

좌석 이상의 예약 수행은 이루어지지 않아 W_2-Transaction의 수행은 실패로 끝나고 만약 W_1-Transaction 처리까지 갔더라도 예약 업무에 대한 완전한 업무처리는 할 수 없게 된다.

즉, R-Transaction, W_1-Transaction의 수행이 W_2-Transaction의 수행을 보장할 수 없으므로 예약 트랜잭션의 모델로 적합하지 못하다.

2.2 현 예약시스템의 문제점과 연구방향

현재 서비스 되고 있는 예약 시스템의 트랜잭션 형태는 사용자가 예약을 시작하여 종료할 때까지 트랜잭션을 기능별로 분리하여 처리하는 형태이며, 그 형태를 정리하면 다음 표 1과 같다.

표 1에서 보면 사용자가 예약을 하기 시작한 시점부터 종료 시까지 여러 단계의 트랜잭션으로 구성되어 있는 것을 알 수 있다. 이것은 예약이라는 하나의 트랜잭션을 여러 단계로 나누어 놓은 상태, 즉 조회시점 부터가 아니라 마지막 단계의 W_2-Transaction 부분에서만 좌석에 대한 잠금을 해놓은 상태이기 때문에, 사용자가 처음 수행하였던 R_1-Transaction의 결과로 예측하였던 예약의 결과는 맨 마지막 단계인 W_2-Transaction의 업무까지 완벽히 끝나야 확답을 받을 수 있게 된다. 또한 이러한 트랜잭션 형태는 사용자가 예약을 하기위해 들어온 순서대로 예약이 되지않고 인터넷 활용 능력이 뛰어난 사용자 또는 컴퓨터나 네트워크 등의 환경이 좋은 사용자 순으로 예약이 이루어지는 상황을 발생시키기도 한다. 이러한 트랜잭션 모델은 예약이라는 하나의 트랜잭션을 수행 할 때 앞의 트랜잭션의 ACID 특성 중 원자성과 내구성은 만족시킬 수는 있으나 일관성과 독립성은 항상 만족 시킬 수 없다[10,14,16].

표 1. 현 예약시스템의 트랜잭션 형태

예약 시스템	Transaction Flow
항공권	R-Transaction → W_1-Transaction → W_2-Transaction
철도	R_1-Transaction → R_2-Transaction → W_1-Transaction → W_2-Transaction
고속 버스	W_1-Transaction → R-Transaction → W_2-Transaction
문화	R_1-Transaction → W_1-Transaction → R_2-Transaction → W_2-Transaction

R-Transaction : Read-Transaction

W-Transaction : Write-Transaction

웹에서의 예약 문화가 자리 잡아가고 점차 현재보다 더 많은 웹에서의 예약 건수들이 증가 할 것으로 예상되는 가운데, 사용자들에게 예약 트랜잭션의 신뢰성 보장은 필수 요소가 된다. 그러므로 현재 서비스되고 있는 웹에서의 예약 시스템 트랜잭션 모델에 대한 새로운 논의가 요구된다[11].

3. WBGT 모델

3.1 시스템 모델

아래 그림 2는 시스템 모델로, 인터넷 예약자의 요청을 받아 바로 서버내의 예약 업무를 담당 하는 어플리케이션(Application) 부분으로 가지 않고, 일반적인 예약 시스템의 모델과 달리 그 사이에 WBGT 모델을 거쳐서 가는 형태이다. 또한 WBGT 모델의 안정성을 보장 하기 위한 FailCheck 프로세스가 WBGT 단에 항상 떠 있는 형태이다.

3.2 트랜잭션 프로세스 모델

예약 업무마다 가장 잘 부합하는 예약시의 기능별 업무 흐름이 있겠지만 가장 일반적이고 안정성 있는 기능별 예약 업무 흐름을 제안하여 이 예약 업무 흐름에 대하여 설계를 하고자 한다.

아래 그림 3은 본 논문에서 제안한 기능별 예약 업무의 흐름이다.

WBGT 모델은 기존 예약 시스템에서 각 기능별로 분리해 놓은 Read-Transaction과 Write-Transaction의 여러 단계를 그림 3과 같이 하나의 트랜잭션으로 처리하는 모델로, 트랜잭션의 시작(T_s)부터 트랜잭션의 종료(T_e) 시까지 발생되는 다중의 트랜잭션을 하나의 트랜잭션으로 묶어서 처리하는 방식이다.

예약 단계의 첫 번째 단계에서는 사용자가 예약을 하기 위하여 예약 시스템에 들어와 일정 등의 기초 데이

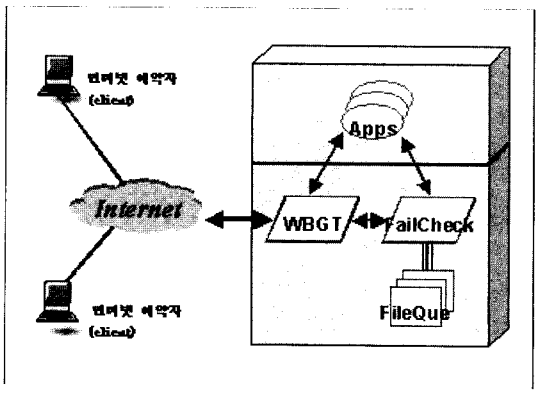

그림 2. 시스템 모델

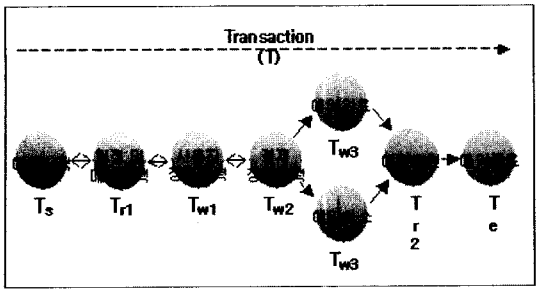

그림 3. 예약 업무 흐름

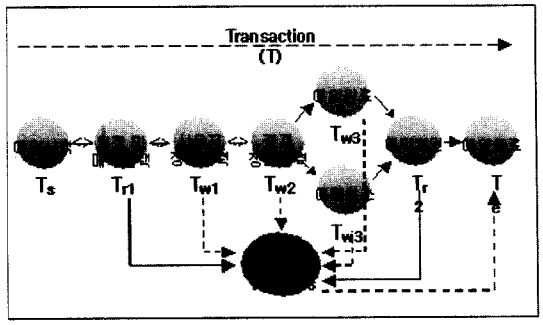

그림 4. 예약 업무 내의 FailCheck 프로세스 흐름

터를 조회하는 R-Transaction의 수행이 예약 업무 트랜잭션의 시작 시점이 된다. R-Transaction을 수행하는 함수는 Search() 함수이다. 이때 예약 업무에 대해 트랜잭션 관리를 위한 트랜잭션의 고유 순번을 발생 시키는 Make_Trans action_Serial_No() 함수와 Make_Transaction_Serial_No() 에서 발생시킨 트랜잭션의 고유 순번과 사용자의 예약 좌석 수를 기준으로 좌석을 잠금 시키는 SeatLock() 함수가 병행되어 진다. 그 후 사용자로부터 예약을 하기 위한 기초 데이터를 입력 받아 최종적인 예약을 완료 시키는 Insert_Reserve_Transaction() 함수가 수행이 되고, 처음 단계에서 잠금 시켰던 좌석을 해제 시켜주는 SeatUnLock() 함수가 수행 되어진다. 마지막으로 예약의 최종 결과를 사용자에게 확인 시켜주는 Select_Reserve_Info() 함수가 수행되면서 예약 업무의 트랜잭션이 종료된다[17-19].

이처럼 인터넷 예약자로부터 요청 받아 처리 되어지는 한 예약 업무의 전반적인 트랜잭션 T는 트랜잭션의 시작부터 종료 시까지 n개의 Read-Transaction과 m개의 Write-Transaction으로 이루어져 있으며, 이의 표현은 다음과 같다.

즉, 이 트랜잭션의 형태는 트랜잭션의 시작인 T_s가 시작되어 Read-Transaction인 T_{ri}에서 T_m, Write-Transaction 인 T_{wj}에서 T_{wm}, 트랜잭션의 마지막인 T_e까지 처리될 동안 T_{r1}에서 Read 되었던 좌석을 잠금 하여 T_{wm}의 처리가 끝나고 난 후에야 잠금을 해제 하도록 한다. 그로 인하여 사용자는 T_{r1}에서 확인 하였던 좌석에 대한 신뢰성을 보장 받을 수 있게 된다.

3.3 FailCheck 프로세스 모델

WBGT 모델 외에 기존에 각 기능별로 분리되어 있던 다중의 트랜잭션을 하나의 트랜잭션으로 묶음으로 인하여 발생되는 문제에 대한 처리 프로세스인 FailCheck 프로세스에 대한 모델을 설계하고자 한다.

FailCheck 프로세스는 예약 시점에 쌓이는 FileQueue(또는 MemoryQueue. 본 논문에서는 FileQueue를 사용한다.)의 데이터를 실시간 또는 일정 시간 간격으로 체크하여[9,10,18] FileQueue의 데이터를 토대로 기타 문제점 발생으로 인하여 잠겨 있는 좌석을 해제하도록 하는 모델이다.

FailCheck 프로세스는 그림 4와 같이 예약 트랜잭션의 T_s부터 T_e까지의 모든 트랜잭션 중에 항상 떠있는 형태로, WBGT의 시작과 끝을 체크하여 데이터의 잠금으로 인해 발생되는 문제점을 처리한다. 또한 일정 시간을 주기로 돌면서 FileQueue에 쌓여있는 처리되지 못한 데이터를 체크 하거나 기타 문제점 발생으로 인한 좌석 데이터의 잠금 문제를 해결하여 WBGT 모델의 시스템을 안정적으로 유지 및 관리하는 역할을 담당한다.

우선 FailCheck 프로세스는 WBGT의 트랜잭션 관리 및 좌석의 잠금 관리를 위하여 예약 업무의 시작 시점에서는 Insert_Lcoking_Transaction() 함수를, 예약의 종료 시점에서는 Delete_Lcoking_Transaction() 함수를 수행한다. 그리고 WBGT의 시작부터 종료까지 또는 그 외의 모든 상황에서 항상 일정한 간격으로 수행하면서 올바르게 종료되지 못하여 잠겨 있는 좌석 및 트랜잭션 등을 체크하여 처리하는 Timer_Check() 함수가 항상 대기 상태로 존재한다.

$$T = \{ T_s, T_{ri} .. T_m, T_{wj} .. T_{wn}, T_e \}$$

$0 <= i <= n, 1 <= j <= m$

T_s : Start -Transaction　　T_e :End -Transaction

T_r : Read -Transaction　　T_w : Write -Transaction

4. WBGT의 알고리즘

우선 첫 번째 검색 알고리즘은 프로그램별 일정 및 잔여 좌석을 조회하는 프로세스로 트랜잭션 구성 중 T_{r1}에 해당한다. 예매의 첫 번째 단계이기도 한 이 프로세스는 인터넷 예약자에게 예약 가능한 프로그램을 제공하는 역할을 담당하며, 예약 작업의 하나의 트랜잭션을 위한

```
Algorithm Search (P, C)
Inout  P : Profram Code ,
       C : Reaaaerve Ticket Count
Output Schedule in Program,
       Rn : Reserve Number
Begin
       1. Rn = Mate_Transaction_Serial_No( ) ;
       2. Select  Schedule in Profram ;
       3.return ;
End
```

고유 순번을 만들어 내는 프로세스로 예약이 완료된 후에 예약 번호를 부여하는 Make_Transaction_Serial_No() 프로세스가 내부적으로 더 추가되어 있다.

다음은 두 번째 좌석에 대해 잠금 및 해제를 하는 프로세스로 T_{w1}에 해당한다.

좌석 잠금 프로세스는 예매의 두 번째 단계로 이 프로세스부터 예매 종료 시까지 좌석을 잠금 함으로써 인터넷 예약자에게 좌석에 대한 신뢰성을 보장해 주는 역할을 담당한다. 그리고 예외 처리에 관련 되서 좌석을 해제하는 SeatUnLock 프로세스가 추가되어 있다. 좌석 해제 프로세스는 예약 종료 시나 전체적인 트랜잭션 내에서의 에러 발생시 잠겨 있던 좌석을 해제하는 역할을 하는 프로세스로 예약 시스템에 대하여 안정성을 유지하는 역할을 담당한다.

세 번째 좌석의 잠금 및 해제에 대한 트랜잭션을 체크하는 알고리즘(Insert_ Lcoking_Transaction(RN)/ Delete_Lcoking_Tra nsaction(RN))이 있다. 예약시 좌석을 잠금 하는 트랜잭션을 처리하는 프로세스로 본 논문에서 제시한 FileQueue의 사용을 대체 하기위한 프로세스로, 예약시 좌석을 잠금 후 해제하지 못할 경우의 처리를 위하여 잠금 트랜잭션에 대하여 LOG 를 남기고, 이의 처리 후나 예약의 완료 후 이를 삭제하는 프로세스이다.

네 번째 최종 예약을 하는 예약 알고리즘(Insert_ Reserve_Transaction(RN))이 있다. 예약의 가장 중요한 프로세스로 말 그대로 인터넷 예약자로부터 요청 받은 내용을 기준으로 예약을 하는 프로세스이다. 또한 예약 후 잠겨있던 좌석을 해제하는 기능을 담당하는 SeatUnLock() 프로세스와 트랜잭션 관리를 위하여 예약 첫 단계에서 처리 하였던 데이터를 삭제 시키는 Delete_Lcoking_Tr ansaction() 프로세스가 있다. 이 두 프로세스는 예약 프로세스의 수행 완료시 혹은 에러 시 처리하는 역할을 담당한다.

다섯 번째 예약 확인 프로세스 알고리즘(Select_ Reserve_Info(RN)) 이다. 예약 프로세스 다음에 수행되는 예약의 맨 마지막 단계의 프로세스로 예약완료 후

```
Algorithm Search (P, C)
Inout  P : Profram Code ,
       S : Schedule Info ,
       C : Reaaaerve Ticket Count
       Rn : Reserve Number
Output Seat Locking
Begin
       1. Update Seat Resserve Ticket Count
          Where ProgramCode, ScheduleInfo ;
       2. If (error)  {
              Delete_locking_Transaction(RN); }
       3. Return ;
End

Algorithm Search (RN)
Inout   Rn : Reserve Number
Output Seaat UnLocking
Begin
       1. Update Seat Resserve Ticket Count
          Where RN
       2. If (error)  {FailCheck Call; }
       3. Return ;
End
```

인터넷 예약자에게 예약내용을 확인시키게 하는 프로세스이다. Select _Reserve_Info() 프로세스는 예약 완료 후 예약 내용을 한 번 더 읽어 들이는 프로세스로 인터넷 예약자에게 예약을 확인시켜 주는 역할을 담당한다.

여섯 번째 예약을 취소하는 취소 알고리즘(Delete _Reserve_Transaction(RN)) 이다. 예약되어 있는 예약건에 대하여 인터넷 예약자로부터 요청 받은 내용을 기준으로 예약을 취소하는 프로세스이다. 예약 프로세스에서 트랜잭션 발생동안 좌석을 잠금 하여 트랜잭션이 끝날 때 해제 시키는 기능과는 약간 다르게, 취소 프로세스는 예약 건을 취소시키고 예약되어 있어서 예약을 하지 못했던 좌석을 예약 가능한 좌석으로 바꾸어주는 역할을 담당한다.

마지막으로 Timer_Check 프로세스는 예약 시스템에 항상 떠있는 프로세스로 관리자가 지정한 일정 간격마다 예약 트랜잭션 중에 발생되는 문제 등으로 인하여 잠겨있는 좌석을 체크하여 이를 처리해 시스템의 안정성을 유지 시켜주는 프로세스이다.

예약을 하기위해 좌석이 잠금 된 후 처리가 정상적이지 않아 잠금에 대한 해제 및 처리가 되지 못한 데이터를 체크하여, SeatUnLock() 프로세스로 좌석의 잠금이 풀리지않고 있는 데이터들을 해제 시키는 등의 작업을 하고, 처리가 완료 후 완료된 트랜잭션을 체크하여 삭제시키는 Delete_Lcoking_Transaction() 프로세스를 수행한다.

```
Algorithm Timer_Check ( )
Begin
  While (1) {
    1. Select Transaction Where Lock ,
    2. for (retch count ) {
    3. IF [Time > 10 minutes) {
        SeatUnLock (RN) ;
        Delete_Lcoking_Transaction (RN) ;
        Commit ;
    4.   } else {   Continue ;   } }
      }
    5. sleep (10) ;
End
```

5. WBGT의 구현

아래 그림 5는 사용자1이 예약 시스템에 들어간 처음 화면이다. 예약할 항목에 대하여 상세 정보를 보여준다.

상영일	상영시간	총좌석수	잠긴좌석수	예약가능좌석수	금 액		
2003/12/10	10:00	10	0	10	5000	2매	예매하기

그림 5. 사용자1의 예약시작 화면

이 화면에서 예약가능 좌석 수내에서 예매하고자 하는 장수를 선택한 후 예매하기를 누르면 그 다음부터가 예매하고자 하는 좌석의 수만큼 좌석을 잠금 함과 동시에 예약에 대해 트랜잭션을 시작하는 부분이 되겠다. 이 부분이 T_s, T_{r1}과 T_{w1}의 시작 부분이다. 그 후 예매를 완료하기 까지 거치는 중간 단계인 결재 정보 입력 중에도 예약자가 선택한 좌석은 계속 잠금 되어 있는 상태가 된다. 이 부분이 T_{w1}과 T_{w2}의 시작 부분이고, 예약이 완료되어 사용자에게 최종적으로 완료 상태를 보여줄 때에는 예약이 완료된 후 잠금 되어 있는 좌석에 대하여 잠금을 해제 시키게 된다. 이 부분이 T_{w2}가 끝난 후 T_e가 된 단계이다.

사용자 1이 예약시작 화면에서 예약 매수를 선택한 뒤 예매하기 버튼을 누른 후 다음 단계로 넘어가는 시점. 즉 Tw1 이후에 들어오는 사용자2의 초기 조회화면은 아래 그림 6과 같다.

첫 번째 사용자가 2장 예매를 선택하여 예매하기 버튼을 누른 후부터 계속 2장의 좌석이 잠금 상태이기 때문에 다른 사용자에게는 그 2좌석을 제외한 다른 좌석만 예약이 가능하게 나오게 되고, 사용자2도 사용자1과 마찬가지로 예매하기 버튼을 누르면 2좌석을 잠금 하고

상영일	상영시간	총좌석수	잠긴좌석수	예약가능좌석수	금 액		
2003/12/10	10:00	10	2	8	5000	2매	예매하기

그림 6. 사용자 2의 예약 시작 화면

예약 진행을 하게 되면, 사용자3이 들어와 조회를 할 때 사용자3 에게는 잠긴 좌석수가 4, 예약 가능 좌석수가 6이 된다. 그러나 만약 이 시점에서 사용자1이 예약을 완료 하였다면, 사용자3 에게는 잠긴 좌석수가 2, 예약 가능 좌석수가 6이 된다.

6. 분 석

현 예약 시스템과 WBGT 모델을 적용한 예약 시스템의 가장 비교되는 기능을 비교 한다면, 아래 표 2와 같다.

현 예약 시스템에서는 인터넷 예약자의 예약 요청한 순서와 상관없이 예약 종료의 순서대로 예약이 이루어지는 형태이다. 이러한 상황은 사용자의 컴퓨터 활용 능력과 컴퓨터 및 네트워크의 성능에 의존하여 예약이 이루어지는 상황을 발생 시킨다. 또한 사용자의 의미 없는 예약 작업이 이루어질 수 있다. 이에 반해 WBGT 모델을 적용한 예약 시스템은 사용자의 컴퓨터 활용 능력과 그 외의 환경에 의존 하지 않고, 예약 작업이 들어온 순서대로 처리하기 때문에 사용자에게 의미 없는 예약 작업을 하게하는 불편함을 감소시킬 뿐만 아니라 초기의 잔여 좌석에 대한 신뢰성을 보장할 수 있다.

즉, WBGT 모델을 적용한 예약 시스템은 사용자가 예약을 하기 위하여 들어와 예약 매수를 선택하고 조회를 하는 시점에서부터 최종 예약을 완료하는 시점까지

표 2. 예약 시스템 비교

예약내용	현 예약 시스템	BMT 모델을 적용한 예약시스템
예약초기	잔여 좌석수 2매	잔여 좌석수 2매
인터넷 예약자 1 2매 예약 시작	잔여 좌석수 2매	잔여 좌석수 0매
인터넷 예약자 2 2매 예약 시작	잔여 좌석수 2매	잔여 좌석수 0매 (예약하지 못함)
인터넷 예약자 2 2매 예약 종료	인터넷 예약자 2 2매 예약 완료	
인터넷 예약자 1 2매 예약 종료	잔여 좌석 부족으로 예약 실패	인터넷 예약자 1 2매 예약 완료

하나의 트랜잭션으로 구성하여 초기 조회 시점에서부터 선택한 좌석에 대하여 잠금을 하고, 최종 예약을 완료하는 시점에서 잠금을 해제 시킨다. 그로 인하여 예약자에게 초기 조회 시점에서 확인하였던 예약 가능한 좌석의 수를 예약이 종료하는 시점까지 보장해줄 수 있다. 또한 FailCheck 프로세스가 예약 실패 시나 WBGT의 예외 사항 발생시 트랜잭션을 시작할 때 잠금 하였던 좌석의 관리를 해주기 때문에 다음 예약자도 신뢰성을 보장 받을 수 있다.

7. 결론 및 향후과제

본 논문에서는 웹 기반의 전자상거래에서 기능별로 나누어져 있던 각각의 트랜잭션을 하나의 트랜잭션으로 묶어 구현하는 새로운 WBGT 모델을 제시하였다.

WBGT 모델은 초기 조회 시점에서부터 선택한 좌석에 대하여 잠금을 하고, 최종 예약을 완료하는 시점에서 잠금을 해제 시키는 하나의 트랜잭션으로 구성한다. 이러한 형태는 예약자에게 조회 시점의 데이터를 보장하여 사용자로 하여금 불필요한 예약 작업을 줄일 수 있게 해주며, 예약을 위해 들어온 사용자에게 순서대로 예약을 할 수 있게 하는 정당성을 부여하여 예약에 대한 신뢰성을 보장하는 장점이 있다. 특히 WBGT 모델은 기존 예약 분야에서도 단기간 내에 좌석을 선매 하려는 트랜잭션이 많을 경우에 그 효과를 극대화 할 수 있다. 그러나 WBGT 모델은 처음부터 좌석에 대한 잠금을 해야 하고 예약 완료 시 잠금을 해제하는 모델이기 때문에, 트랜잭션을 보장해야 하는 구간이 길어져 이에 따른 추가적인 시스템의 부하가 예상되고 예약 완료시까지 시스템적인 문제가 발생 하거나, 사용자의 요구에 의해서 트랜잭션이 취소가 되는 경우에는 트랜잭션을 초기화 시키려는 추가의 노력이 필요하다. 따라서 이 모델은 인터넷 실시간 예약에서의 신뢰성 보장 측면에서는 개선의 효과를 기대 할 수 있지만 성능에 대한 객관적인 근거가 제시되지 않았고 최종 예약을 의도하지 않은 트랜잭션에 의해 선의 예약자들에 대한 원천적인 기회 봉쇄를 야기 할 수 있는 모델이기 때문에 선택적인 분야에 적용 되어야 할 것이며 분야별로 성능이나 신뢰성 향상에 대한 객관적인 근거를 제공하기 위한 연구가 진행 되어야 할 것이다.

따라서 본 논문에서는 초기화 되어져야 하는 트랜잭션을 위해서 FailCheck 프로세스를 추가 하여 이를 보완 하였다. 그러나 향후 FailCheck 프로세스의 기능을 대신할 수 있는, 또는 미들웨어 레벨이나 데이터베이스 레벨에 추가 시키는 등의 추가적인 연구가 요구된다.

참고문헌

[1] http://www.ibm.com
[2] http://www.terms.co.kr
[3] http://www.tmax.co.kr
[4] http://www.koreanair.co.kr
[5] http://flyasiana.com
[6] http://www.barota.com
[7] http://www.easyticket.co.kr
[8] http://www.ticketlink.co.kr
[9] 윤용익, "실시간 완료 규약 프로토콜", 한국정보처리학회 학회지, Vol. 5, No.4. 1998.
[10] 윤용익, "실시간 분산 시스템을 위한 동적 트랜잭션 처리 방안에 관한 연구", 한국정보과학회 논문지(C), 제5권 6호. 1999.
[11] W. Derks, J. Dehnert, P. Grefen, W. Jonker. "Customized Atomicity Specification for Transactional Workflows", 2001.
[12] Yong-Ik Yoon, "Optimized Compensatio n Transaction For Mobile Telecommunic ation System", IEICE Transactions on Fundamentals of Electronics, Communic ation and Computer Sciences, VolE79-A, No.6, PP. 804-811, 1996.
[13] Reiter, Michael K. "Anonymous Web Transactions with Crowds Association for Computing Machinery", Vol. 42 No. 2, 1999.
[14] Philip A.Bernstein, "Principles of Trans action Processing", 1997.
[15] Jin Suk Baek and Yong-Ik Yoon. 「A Study of Application Modeling for Reliability in the Web Environment°ª. Proceeding of EALPIIT2000, China, pp.44-51. 2000.
[16] Yong-Ik Yoon, "Trinity Transaction Processing for Distributed Real-time Processing Systems", International Conference on Information technology and Application, SimYang, China, 2001.
[17] W.Derks, J.Dehnert, P.Grefen, W.Jonker. "Customized Atomicity Specification for Transactional Workflows", 2001.
[18] Younas, M., "Ensuring Recovery for SACReD Web Transactions in the E-commerce Applications". LECTURE NOTES IN COMPUTER SCIENCE. Vol.-No. 2453 2002.
[19] Alkhatib, G. "Transaction Management in Distributed Database Systems: the Case of Oracle's Two-Phase Commit", JOURNAL OF INFORMATION SYSTEMS EDUCATION. Vol. 13 No. 2, 2002.