Tạp chí Tin học và Điều khiển học, T. 19, S. 4 (2003), 391–398

GIAO THỨC HỢP THỨC CHO QUẢN LÝ GIAO TÁC PHÂN TÁN DI ĐỘNG

NGUYỄN KHẮC LỊCH, LÊ HỮU LẬP

Học viện Công nghệ Bưu chính - Viễn thông

Abstract. With the rapid growth and convenience of wireless technology, nowadays. It opens a new era of mobile computing and a new orientation in communication and data processing. Mobile databases are hosted on the lightweight devices like Laptop, palmtop, PDA, Pocket PC, Cellular phones ([6])... The most well-known 2PC ([2]) is used in distributed database, but it is far from being adapted to the mobile databases, because the typical mobile environments are off-line execution, moving and disconnected computing. To solve these problems, we proposed a new improved commitment protocol called the Commit Protocol for Mobile Distributed Transaction Management. It supports off-line execution, moving, disconnected computing, also it decreases the cost of wireless communication.

Tóm tắt. Sự phát triển nhanh chóng của công nghệ không dây và các thiết bị di động ngày nay đã mở ra một kỷ nguyên mới về tính toán, di động và xu hướng mới trong truyền thông và xử lý dữ liệu. Cơ sở dữ liệu di động được cài đặt trong các thiết bị nhỏ nhẹ như máy tính xách tay, PALM, máy trợ giúp cá nhân(PDA), máy tính bỏ túi (Poctket PC), máy di động cầm tay (Cellular phones) ([6])... Việc quản lý giao tác và điều khiển tương tranh trong cơ sở dữ liệu phân tán di động phức tạp và khó khăn hơn nhiều việc quản lý giao tác trong môi trường phân tán truyền thống. Giao thức sử dụng cho cơ sở dữ liệu phân tán là 2PC ([2]) nhưng giao thức này khó áp dụng cho môi trường phân tán di động và nếu áp dụng thì sẽ bộc lộ nhiều nhược điểm. Để giải quyết các nhược điểm đó chúng tôi đề xuất một giao thức hợp thức cải tiến mới gọi là CPM "Giao thức hợp thức cho quản lý giao tác phân tán di động". Giao thức này hỗ trợ việc thực hiện ngoại tuyến, di chuyển, tính toán không kết nối và giảm được chi phí truyền thông.

1. MỞ ĐẦU

Giao thức được sử dụng rộng rãi cho cơ sở dữ liệu phân tán (DDBMS) ([9, 10, 12]) là 2PC do Gray đề xuất ([2]). Nhưng giao thức này khó áp dụng cho môi trường phân tán di động và nếu áp dụng thì sẽ bộc lộ nhiều nhược điểm. Chẳng hạn, người sử dụng có thể ngắt kết nối thiết bị di động tạm thời để xử lý ngoại tuyến, điều này có thể dẫn đến hủy bỏ giao tác, do đó các ứng dụng có thể bị hủy bỏ thường xuyên. Để giải quyết vấn đề này chúng tôi đề xuất một giao thức hợp thức cải tiến gọi là CPM (The Commit Protocol for Mobile Distributed Transaction Management). Giao thức này hỗ trợ việc thực hiện ngoại tuyến, di chuyển, tính toán không kết nối và giảm được chi phí truyền thông vì bản chất của giao thức này không cần pha chuẩn bị như 2PC. Bài báo còn so sánh CPM với các giao thức hợp thức khác.

2. GIAO THỨC CPM

Trong CPM, các thành phần tương tác trong giai đoạn thực hiện và kết thúc gồm:

- APP (Application): yêu cầu sự thực hiện của một giao tác;
- P (Participant): thành viên thực hiện các thao tác của giao tác;

- PA (Participant Agent): đại diện cho các thành viên trong việc thực hiện, kết thúc và phục hồi của giao thức CPM;
- C(Coordinator): bộ điều phối điều khiển sự hợp thức của giao tác.

Chú ý rằng C luôn nằm trên mạng cố định hoặc mạng vô tuyến chất lượng cao (vì lý do an toàn) còn các thành phần khác thì tuỳ ý, có thể nằm trên mạng cố định hoặc di động. Đây là giao thức điều khiển việc kết thúc của giao tác nên trong phạm vi của giao thức CPM chúng tôi chỉ mô tả các tiến trình hợp thức, còn các hoạt động khác như lập lịch trình của giao tác ([1, 9, 11]) hay phân bố các giao tác con thực hiện tại các thành viên được giả định là đã có.

Nội dung của giao thức CPM được mô tả như sau:

Gọi $T = \{T_1, T_2, ..., T_n\}$ là tập các giao tác,

$P = \{P_1, P_2, ..., P_m\}$ là tập các thành viên sẽ thực hiện tập giao tác T,

$S = \{S_1, S_2, ..., S_m\}$ là tập các trạm tương ứng với các thành viên P.

Với giao tác T_i bất kỳ nào đó $T_i \in T$. Gọi $J = \{1, 2, ..., a\} \subset \{1, 2, ..., m\}$, J là tập chỉ số ứng với các thành viên cùng thực hiện giao tác T_i.

Bộ lập lịch trình của hệ thống sẽ chia giao tác T_i thành các giao tác nhánh sẽ được thực hiện tại các thành viên tương ứng là

$T_i = \{T_{ij}/j \in J\}$, với j bất kỳ thì $T = \{op\}$, $op \in \{Read, Write\}$.

Giả sử tại trạm S_k nào đó, $S_k \in S$ có ứng dụng APP^k yêu cầu hệ thống thực hiện giao tác T_i và tương ứng có P^k, PA^k. Sau đây là các bước thực hiện của giao thức:

Bước 1. APP^k gửi chuỗi các thao tác của T_i cần thực hiện đến PA^k.

Bước 2. PA^k gửi T_i đến bộ lập lịch trình.

Bước 3. PA^k lấy từ bộ lập lịch trình các giao tác nhánh và gửi đến các thành viên tương ứng P^j, $j \in J$ trước khi gửi đi PA^k ghi nhật ký từng thao tác của các giao tác nhánh thuộc T_i vào bộ nhớ (RAM).

Bước 4. PA^k khởi động timer và chờ các xác nhận từ các P^j.

Bước 5. Các P^j, $j \in J$ sẽ thực hiện cục bộ từng T_{ij} tương ứng. Sau khi thực hiện xong từng thao tác sẽ gửi xác nhận của thao tác tương ứng về cho PA^k.

Bước 6. Nếu hết timeout và PA^k không nhận được toàn bộ xác nhận ACK^{ij} của T_i từ các thành viên nó sẽ gửi lệnh abort (lệnh hủy bỏ giao tác) đến C và C chỉ việc quảng bá lệnh hủy bỏ đến các thành viên.

Bước 7. Ngược lại nếu PA^k nhận được toàn bộ xác nhận của T_i từ các thành viên nó sẽ gửi lệnh commit (lệnh hợp thức của giao tác đó) đến C (bộ điều phối). Đến thời điểm này thì các tính chất ACI của giao tác đã được đảm bảo cục bộ bởi các P^j cho tất cả các giao tác nhánh của giao tác T_i. Tuy nhiên các thành viên sẽ không thấy được sự kết thúc của giao tác nên chúng không đảm bảo tính bền vững D (tính chất A,C,I,D được trình bày trong Mục 4) của giao tác. Tính chất D (các tính chất A, C, I, D được trình bày trong Mục 4) sẽ được đảm bảo bởi bộ điều phối C bằng cách PA^k sẽ đẩy nhật ký của T_i đến C và C sẽ ghi ra bộ lưu trữ ổn định (stable storage).

Bước 8. Sau khi hoàn thành các bước trên thì các tính chất ACID của giao tác được bảo đảm cho tất cả các nhánh của giao tác T_i. Vì vậy bộ điều phối C quyết định hợp thức (commit) và ghi quyết định đó vào bộ lưu trữ ổn định, sau đó quảng bá (broadcasting) quyết định hợp thức đến tất cả các thành viên PA^j ($j \in J$) và đợi nhận các xác nhận từ chúng, PA^j nhận được quyết định hợp thức sẽ đưa ra thêm nhiệm vụ ghi lại và thực hiện lệnh hợp thức đó cho P^j. Sau khi thực hiện xong, P^j sẽ chuyển xác nhận đến PA^j và PA^j chuyển tiếp đến C. Khi C nhận được toàn bộ xác nhận từ các P^j, nó sẽ hoàn tất giao tác và thông báo OK đến PA^k, sau đó PA^k sẽ gửi OK đến APP^k.

3. CÁC THUẬT TOÁN TRONG CPM

Các khái niệm mô tả trong thuật toán này đã được trình bày trong phần mô tả giao thức CPM

Thuật toán tại APP

Begin

1. APP^k: Gửi chuỗi các thao tác của giao tác T_i đến PA^k
2. APP^k: Đợi đến khi nhận được lệnh Ok hoặc Not Ok

End

Thuật toán tại PA

Begin

IF(PA^k nhận được giao tác T_i) then

Begin

PA^k gửi T_i đến bộ lập lịch trình và lấy ra các T tương ứng với các thành viên P^j, $j \in J$

PA^k ghi nhật ký từng thao tác của các giao tác nhánh T vào bộ nhớ (RAM)

PA^k gửi các T đến các thành viên tương ứng P^j, $j \in J$

PA^k khởi động timer và chờ các xác nhận từ các P^j

IF (Nếu hết timeout và PA^k không nhận được toàn bộ xác nhận ACK^{ij} của T_i) then

Begin

PA^k gửi lệnh Abort (lệnh hủy bỏ giao tác) đến C

PA^k gửi lệnh Not OK (không thực hiện được) đến APP^k

End

Else

Begin

IF (PA^k nhận được toàn bộ xác nhận của Ti) then

Begin

PA^k gửi lệnh commit (lệnh hợp thức của giao tác đó) đến C

PA^k đẩy nhật ký của Ti đến C

PA^k đợi kết quả từ C

IF (PA^k nhận được kết quả từ C) then

PA^k đẩy kết quả về cho APP^k

End

End

End

Else

Begin

{Lúc này thuật toán dành cho các PA^j $(j \in J)$}

PA^j nhận được quyết định từ C

IF (quyết định là commit) Then

Begin

PA^j đưa lệnh ghi bản ghi "commit" và lệnh commit cho P^j
PA^j khởi động timer
PA^j đợi xác nhận từ P^j tương ứng
IF (PA^j nhận được xác nhận từ P^j) then
PA^j gửi xác nhận về cho C
IF (PA^j không nhận được xác nhận từ P^j và hết timeout) then
Begin
{Lúc này P^j tương ứng bị sự cố}
PA^j kiểm tra nhật ký cục bộ (LocalLog) tại P^j
IF (tồn tại bản ghi "commit") then
PA^j gửi xác nhận về cho C
Else
PA^j yêu cầu P^j thực hiện lại giao tác nhánh T nhờ nhật ký của C
End
End
Else
PA^j đưa lệnh abort cho P^j
End
End

Thuật toán tại P (thành viên)

Begin
P^j, $j \in J$ thực hiện các thao tác và gửi xác nhận về cho PA^k
P^j đợi nhận lệnh commit từ PA^j tương ứng
IF (P^j nhận được lệnh ghi bản ghi "commit" và lệnh commit) then
Begin
P^j tiến hành hợp thức giao tác nhánh T
P^j ghi bản ghi "commit" vào nhật ký cục bộ
End
Else
P^j hủy bỏ giao tác nhánh T
End

Thuật toán C (Bộ điều phối)

Begin
IF (C nhận được quyết định abort từ đại diện thành viên) then
Begin
C quảng bá lệnh abort đến các P^j, $j \in J$
End
Else
IF (C nhận được nhật ký (Log) và lệnh commit từ đại diện thành viên) then
Begin
C ghi nhật ký vào bộ nhớ ổn định stable_storage

C quảng bá lệnh commit đến các PA^j, $j \in J$
C đợi để nhận xác nhận từ các đại diện thành viên
IF (C nhận được hết các xác nhận của các giao tác nhánh T) then
Begin
C hoàn tất giao tác và gửi thông báo OK về cho ứng dụng tương ứng
End
End
End

4. SO SÁNH VỚI CÁC GIAO THỨC HỢP THỨC KHÁC

Giao tác là một tập các thao tác trên cơ sở dữ liệu và thỏa mãn các tính chất ACID (Atomicity, Consistency, Isolation, Durability) ([13, 15]).

Tính nhất quán (Consistency): Mỗi giao tác sau khi được thực hiện thành công sẽ duy trì tính nhất quán của cơ sở dữ liệu (thỏa tất cả các ràng buộc toàn vẹn), hệ quản trị cơ sở dữ liệu đảm bảo tính nhất quán được duy trì trên mỗi giao tác.

Tính cô lập (Isolation): Các tác dụng của các giao tác phải được cô lập với nhau khi mà chúng được thực hiện tương tranh (tính cô lập đã được hình thức hóa qua lý thuyết khả tuần tự).

Tính bền vững (Durability): Nếu một giao tác đã hoàn tất công việc thì kết quả của nó sẽ bền vững (không bị mất đi) ngay cả trong trường hợp hệ thống có sự cố), điều đó có nghĩa là kết quả của nó được ghi vào bộ nhớ ổn định.

Để đảm bảo tính nguyên tố của một giao tác truy nhập các đối tượng dữ liệu phân tán, tất cả các thành viên tham gia thực hiện giao tác phải hợp tác với nhau. Vì vậy chúng cùng nhất trí hủy bỏ hoặc hợp thức giao tác. Điều này đạt được qua giao thức hợp thức nguyên tố (ACP-Atomic Commit Protocol). Giao thức nổi tiếng nhất là 2PC. Nó đã được cài đặt ở vài hệ quản trị cơ sở dữ liệu thương mại như Oracle, DB2... Mặc dù được sử dụng rộng rãi nhưng 2PC bị xem như hoàn toàn không hiệu quả vì nó đưa ra một trễ lớn trong xử lý giao tác. Trễ này do chi phí thời gian vào việc xử lý các thông báo của bộ điều phối và các thao tác ghi nhật ký của nó.

Do các hạn chế đó dẫn đến việc nhiều nhà nghiên cứu đi tìm các phiên bản tối ưu của 2PC. Đó là các giao thức hợp thức đoán trừng PrC (presumed commit), hủy bỏ đoán trừng PrA (presumed Abort) và giao thức chuẩn bị sớm (Early Prepare) ([7, 13]). Các giao thức này giảm bớt trễ so với 2PC trong việc ghi nhật ký và giảm độ phức tạp thông báo.

Giao thức 1PC (one-phase commit) đề xuất bởi Gray ([3]) đã được xem xét lại và các biến thể của 1PC là CLP (Coordinator log protocol) và IYVP (Implicit Yes-Vote protocol) đã được đề xuất. 1PC giảm trễ bằng cách loại bỏ pha biểu quyết của 2PC. Mặc dù rất hiệu quả nhưng cho đến nay 1PC bị các nhà sản xuất các sản phẩm thương mại bỏ qua vì những giả thiết quá mạnh của nó.

Đối với giao thức 3PC thì độ trễ cũng lớn như 2PC nhưng cải thiện được tình trạng bị phong tỏa vào/ ra (blocking I/O) bằng cách thêm vào pha pre-commit.

Gần đây giao thức TCOT ([8]) (Timeout-based Mobile Transaction Commitment Protocol) được đề xuất bởi Kumar và cộng sự cải tiến bằng việc xét đến khía cạnh tối ưu timeout.

Nếu chúng ta giả sử n là số thành viên tham gia vào việc thực hiện giao tác thì chúng ta

sẽ có bảng so sánh về độ trễ như sau (chi phí trễ của các giao thức).

Giao thức	Số thông báo	Trễ ghi nhật ký	Trễ các bước truyền thông
2PC	$4(n-1)$	$1+2n$	4
PrA	$4(n-1)$	$1+2n$	4
PrC	$3(n-1)$	$2+n$	4
EP	$n-1$	$1+(n+n\_op)$	2
CL	n-1	1	2
IYV	$2(n-1)$	$1+n$	2
TCOT	$2(n-1)$	$1+n$	2
CPM	$2(n-1)$	$1+n$	2

Trong đó $n_$op là số thao tác được thực hiện bởi giao tác.

Tuy CPM khắc phục được xử lý ngoại tuyến và ngắt kết nối trong quá trình hợp thức nhưng CPM cũng có nhược điểm khi có quá nhiều giao tác bị trễ hợp thức, sẽ dẫn đến trường hợp bộ xử lý giao tác tại bộ điều phối phải dùng cơ chế vòng tròn để ngắt các giao tác chờ lâu nhất. Nếu không dùng cơ chế này sẽ dẫn đến tràn xử lý.

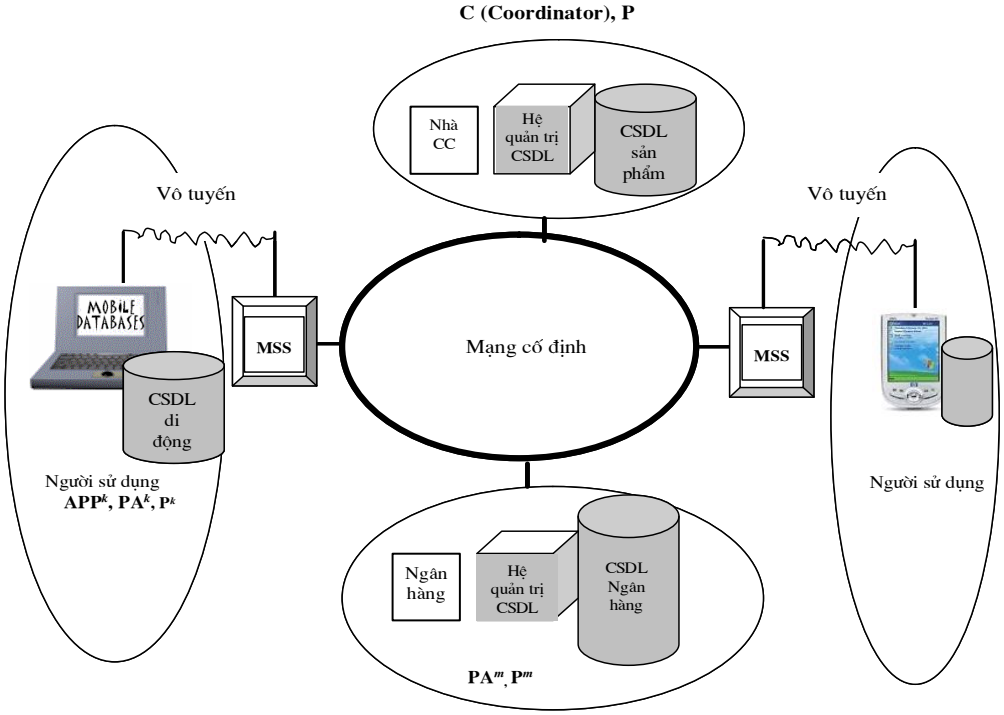
5. ÁP DỤNG CPM CHO THƯƠNG MẠI ĐIỆN TỬ DI ĐỘNG (M-COMMERCE)

Chúng ta đã chứng kiến sự phát triển cực kỳ nhanh chóng của các công nghệ vô tuyến, điều đó dẫn đến xu hướng phát triển tất yếu của thương mại điện tử di động (m-commerce). Chúng ta hãy cùng phân tích CPM khi áp dụng cho thương mại điện tử di động.

Hình 1 dưới đây sẽ mô tả kiến trúc tiêu biểu của thương mại điện tử di động bao gồm các khối sau:

+ Khối các nhà cung cấp sản phẩm gồm có CSDL về các sản phẩm được kết nối với mạng cố định và được thể hiện bằng các trang WEB bán hàng.
+ Khối ngân hàng gồm CSDL về tài khoản người sử dụng và thanh toán điện tử trên mạng.
+ Khối người sử dụng gồm những người sử dụng thiết bị di động như máy tính xách tay, PDA, Mobile phone... chứa các CSDL di động.
+ Khối mạng gồm mạng cố định và mạng vô tuyến.

Người sử dụng trong khi đang di chuyển trong mạng di động từ tế bào này sang tế bào khác sẽ thực hiện việc lựa chọn và mua sản phẩm thông qua các giao dịch di động. Đầu tiên NSD (người sử dụng) kết nối vô tuyến vào mạng và lựa chọn nhà cung cấp sản phẩm, tải xuống tài liệu liệt kê sản phẩm và số lượng có thể cho mỗi sản phẩm. Đến lúc này NSD có thể tạm ngắt kết nối và xử lý thông tin ngoại tuyến (lựa chọn sản phẩm off-line), giao tác sẽ bị trễ hợp thức (commit) cho đến khi NSD kết nối trở lại. Nhà cung cấp sản phẩm sẽ kết nối với ngân hàng để thông báo cho NSD số tiền phải trả. Sau đó NSD có thể quyết định mua sản phẩm hay không.



Hình 1. Kiến trúc CSDL di động phục vụ cho thương mại điện tử di động

Trong kiến trúc trên, giả sử các cấu thành của giao thức CPM được bố trí như sau: APP^k, PA^k được bố trí tại CSDL di động của NSD (thành viên Pk), C được bố trí tại CSDL cung cấp sản phẩm (thành viên P) còn PA^m thì được bố trí tại CSDL của ngân hàng (thành viên P^m).

APP^k chạy trên thiết bị di động của NSD và đưa ra chuỗi các thao tác đến PA^k, các thao tác của các giao tác này sẽ được PA^k ghi vào nhật ký và NSD có thể tạm ngắt kết nối, tiếp tục thực hiện xử lý thông tin cho đến khi hoàn thành giao tác cục bộ của mình. Sau khi đã lựa chọn xong sản phẩm NSD kết nối trở lại vào mạng, lập tức PA^k sẽ gửi toàn bộ nhật ký của nó lên C tại Server của nhà cung cấp và đưa ra lệnh hợp thức cho C, C sẽ ghi toàn bộ nhật ký này vào bộ lưu trữ ổn định. Sau khi ghi xong C quảng bá lệnh hợp thức (commit) đến các thành viên PA^k và PA^m và chờ xác nhận từ PA^k và PA^m. Sau khi nhận được các xác nhận từ các thành viên PA^k và PA^m thì giao tác hoàn thành và NSD có thể ngắt kết nối.

Nhận xét: Chúng ta thấy rất rõ với giao thức này, thứ nhất sẽ giảm thiểu việc hủy bỏ giao tác do kết nối tạm thời bị ngắt (do NSD tạm ngắt hoặc do mạng, thời gian ngắt không biết trước), thứ hai giao tác có thể hoàn thành cục bộ trong khi bị ngắt kết nối ra khỏi mạng (disconnect).

6. KẾT LUẬN

Bài báo đã đề xuất giao thức (CPM) và phát biểu dưới dạng các thuật toán, phân tích và chỉ ra mức độ phù hợp với nhiệm vụ quản lý giao tác phân tán di động, đồng thời cũng đã so sánh các lợi điểm với các giao thức khác (các biến thể của giao thức 1PC, 2PC, 3PC). CPM là một giao thức quản lý sự kết thúc các giao tác di động thỏa mãn các yếu tố sau:

- Hỗ trợ các giao tác ngoại tuyến và giảm thiểu việc hủy bỏ giao tác tại thời điểm kết nối lại;

- Có thể hợp thức một giao tác khi thành viên đang di chuyển;
- Hỗ trợ việc hợp thức các giao tác trong thời điểm bị tạm ngắt (disconnect);

Trong các bài báo tiếp theo, chúng tôi sẽ mô phỏng giao thức CPM trên máy tính để theo dõi hiệu quả của giao thức này.

TÀI LIỆU THAM KHẢO

[1] P.A. Bernstein, V. Hadzilacos, N. Goodman, *Concurrency control and recovery in database systems,* Addison Wesley Publishers, 1987.

[2] J. Gray, Notes on DataBase Operating Systems *Lecture Notes in Computer Science,* volume 60, Springer Verlag, 1978.

[3] J. Gray, A Comparison of the Byzantine Agreement Problem and the Transaction Commit Problem, *LNCS448,* Springer Verlag, 1987.

[4] ISO, Open System Interconnection Distributed Transaction Processing (OSITP) Model, *ISO IS 100261,* 1992.

[5] ISO, Interindustry Commands for Structured Card Query Language (SCQL), *ISO 78167,* 1997.

[6] IBM Corp, DB2 Everywhere for Windows CE and Palm OS Software, Administration and Application Programming Guide, *SC26967500,* 1999.

[7] Jstanmos, F. Cristian, A low-cost Atomic Commit Protocol, *Proceeding of the 9th IEEE Symposium on Reliable Distributed Systems,* 1990.

[8] V. Kumar, N. Prabhu, M. H. Dunham, and A. Y. Seydim, TCOT - A Timeout-Based Mobile Transaction Commitment Protocol, *IEEE Transactions on Computers* **51** (10) (2002).

[9] Nguyễn Khắc Lịch, "Quản lý giao tác và điều khiển tương tranh trong hệ cơ sở dữ liệu phân tán", Luận án thạc sĩ, Đại học quốc gia Hà nội 1999.

[10] Nguyễn Khắc Lịch, et al., Thiết kế và cài đặt kho thông tin Bưu điện, Các kết quả nghiên cứu khoa học và công nghệ của Viện KHKT BĐ (1999) 261–270.

[11] Nguyễn Khắc Lịch, et al., Mô hình cơ sở dữ liệu thời gian thực, Các kết quả N/C khoa học và công nghệ của Viện KHKT BĐ (2000) 382–390.

[12] Nguyen Khac Lich, et al., System Aided Management for Post and Telecom (SAMPT), *Proc of The 25th AIC Conference* (5/2001) 343–349.

[13] C. Mohan, B. Lindsay and R. Orbermarck, Transaction Management in the Distributed Database Management System, *ACM transactions on Database Systems,* 1986.

[14] *Object Management Group, Object Transaction Service, OMG Document 94.8.4,* OMG editor, August 1994.

[15] E. Pitoura, G. Samaras, *Data Management for Mobile Computing,* Kluwer Academic Publishers, 1998.

[16] G. Walborn, P.K. Chrysanthis, Using the Escrow Transactional Method to Manage Replicated Data in Disconnected Mobile Operations, *CS Technical Report 9432,* University of Pittsburgh, 1994.

[17] G. Walborn, P.K. Chrysanthis, PROMOTION: Support for Mobile Database Access, *Personal Technologies Journal,* September 1997.

Nhận bài ngày 12 - 9 - 2003
Nhận lại sau sửa ngày 21 - 10 - 2003