マルチメディア通信と分散処理　106－22
コンピュータセキュリティ　16－22
（２００２．２．１５）

モバイル環境に適した安全な転々流通機構の研究

森　謙作
（株）NTT ドコモ
〒239-8536 神奈川県横須賀市光の丘 3-5
e-mail: mori@mml.yrp.nttdocomo.co.jp

あらまし　一般的な電子実体の受け渡しは、２相コミットメントによる送受信者双方のデータベース（DB）の更新で実現される。しかし、この処理ではデータの授受時に通信に障害が発生した場合に、互いの DB の整合がとれなくなる可能性が存在する。本論文では、この DB に不整合が生じる可能性を回避した３つのトランザクション処理モデル、（ｉ）共通鍵を用いたモデル、（ii）公開鍵を用いたモデル、（iii）フラグを用いたモデルを提案した。また、シミュレーションにより本方式において、成功率、アボート率、データの不整合の発生率、一時的使用不可の発生率を求めた。その結果、提案モデルの構造を用いると、データの不整合が回避でき、一時的使用不可率も 2 相コミットメントのデータの不整合発生率と比較し、約 2 割～５割程度に抑えることが出来た。

キーワード　転々流通、トランザクション、公開鍵、共通鍵、フラグ

A study on the safe transferable distribution system on mobility environment

Kensaku Mori
NTT DoCoMo, INC.
3-5, Hirarinooka, Yokosuka, Kanagawa, 239-8536 Japan
e-mail: mori@mml.yrp.nttdocomo.co.jp

General transfer of the electronic entity is realized by updating the both (sender and receiver) databases by 2-phase commitment. In this commitment, DBs may be inconsistent when a communication error occurred in transaction processing.

To avoid this inconsistency caused by communication errors, we propose three types of transaction-processing models based on (ｉ) common-key , (ii) public-key , (iii) a flag and transaction.

In this paper, we show the rates of success, abortion, inconsistency, and temporal unavailability. Our models avoid Inconsistency and decrease 50-80 percents of temporal unavailability rate compared with two-phase commitment.

Keyword: transferable distribution, transaction, public-key cryptosystem, common-key cryptosystem, flag

１．はじめに

現在、本や音楽等のコンテンツをデジタル化し商品としてネットワーク上で販売・購入する機会が増加している［1］。今後、このような商取引をより発展させるには、現行の紙ベースのビジネスモデル(紙幣、本などの流通)をデジタル上で同様に実現すること、つまりデジタル化した現金やコンテンツ自体を安全に流通させる機構が必要となってくる。

このような機構の実現には、紙の性質（コピー・改竄不可能、複製困難性）をデジタル上でも満たすこと、つまり、コンテンツ自体

がコピーや改竄されることなく流通する性質（転々流通性）が求められる。

本研究は、通信の遮断が起き易いモバイル環境の上で、通信障害に影響されずに電子実体を安全に授受できる転々流通機構の確立を目的とする。

2．提案モデル

本研究では、転々流通機構を実現する最初のステップとして、2 者間のデータの授受に注目した。2者間のデータの授受は、2 相コミットメントと呼ばれるトランザクション処理を用いて、両者のデータベース（以下、DB と略す）内のデータを更新することで実現される。

2．1　2 相コミットメントの問題点

2 相コミットメントには、送信者・受信者・トランザクションマネージャ（トランザクション全体を管理するモジュール、以下 TM と略す）の３者が存在する。送受信者間のデータの授受内容は、まず各々の DB のテンポラリ領域に格納される。次に、TM が双方の DB にプリペア要求を送信し同期をとり、処理が正常に行えることを確認する。最後にコミット命令を送信する。コミット命令を受け取った送受信者は、受け取った時点でテンポラリ領域内の処理内容を自分の DB に反映させ、データの更新を行う。つまりコミット要求を受け取った時点で初めて処理が DB に反映される。

ここで、プリペア要求以前に TM からいずれかへの要求の送信に障害が発生すると、送受信者共にアボート処理が行われ、双方の DB は授受以前の状態（初期状態）に戻り整合性が保たれる。しかしコミット要求時に障害が発生すると、コミットに失敗した側はアボート処理され初期状態に戻り、コミットに成功した側は要求された処理を終了してしまい、通信時の障害によるデータの複製や紛失といった不整合が生じる可能性がある [2][3]。本研究では、このような不整合を回避できるトランザクションモデルの提案を行った。

2．2　提案モデル

上記の問題点を解決するモデルを提案する。提案するモデルでは、コミット失敗時の不整合の発生を回避できる。この回避とは、双方の DB に使用可能な同一のデータが同時に存在しないことを意味する。

2．3　共通鍵を用いたトランザクションモデル

共通鍵を用いたトランザクションモデルを図１に示す。このモデルの特徴は、授受の中で共通鍵を生成し、それを用いてデータを暗号化して送信する点、及びコミット処理を同時に行わず受信者側のコミット処理が正常に終了したことを確認して送信者がコミット処理を実行する点である。上記の 2 点を組み合わせることで、データの複製・紛失を回避している。このモデルを用いて、プリペア要求以降に通信に障害が発生した場合の、それぞれの DB の中身を表１に示す。このモデルでは、トランザクション処理を終了し、最後に鍵を送信する部分で障害が発生したに問題が生じる。共通鍵の送信は TCP 上で行うため、失敗すると再送要求などが発生するが、その時点で受信者側が（PC が落ちるなどの原因で）ネットワークから外れてしまった場合は、その時点で授受を終了させることは出来ない。この時点では、送信者側に共通鍵が、受信者側に暗号化されたデータとなり、どちらにも使用可能なデータが存在しない。この問題点

については、ここでは論ぜずに、実装依存の問題として取り扱う。解決案の例としては、ネットワーク上の紛失物預かり所などに送信者が鍵を預け、受信者側がそれを探しに来る形である。

２．４　公開鍵を用いたトランザクションモデル

公開鍵を用いたトランザクションモデルを図 2 に示す。このモデルの特徴は、受信者の持つ公開鍵を用いてデータを暗号化して送信する点、及びコミット処理を同時に行わず受信者側のコミット処理が正常に終了したことを確認して送信者がコミット処理を実行する点である。公開鍵を用いたモデルで、コミット時に障害が発生した場合の状態を表２に示す。

２．５　フラグと認証情報を用いたモデル

フラグを用いたモデルを図 3 に示す。構造的には上記の共通鍵モデルと同等である。
データは、自身が使用可能であるか否かのフラグを持つ。フラグによるチェックではデータが暗号化されていないため、使用不可のフラグがたったデータが渡った状態でも受信者が自身でフラグを倒してしまう可能性が有る。そのため、トランザクションマネージャは送受信が開始された段階でそのトランザクションの認証情報を生成し、フラグを倒す命令は送信者の ID がないと受け付けない形としておく。またもし通信が途中で切れた場合でも、認証情報を双方が持ち合わせていれば、そのデータは２者間でのみ授受を再開できる。このモデルで、コミット時に障害が発生した場合の状態を表３に示す。

３．実験

提案したモデルと既存のモデルについて、シミュレーションにより、成功率、アボート率、データの不整合及び一時的使用不可が発生する率を抽出し、比較検討を行った。

３．１　実験内容

概念的に、トランザクションには送信者、受信者、TM の３者が存在するが、実際には TM は送信側／受信側のいずれかに含まれる。今回の実験では TM の役割を送信者側に持たせ、ローカルでのソケット通信を用いて実験を行った。

それぞれのモデルについて、各イベント（データ送受信／要求送受信／ACK 送受信）にエラー発生率を設定した。エラー発生率の刻みは 0.01 である。エラー率を 0.1 とした場合は、データの送信時／受信時で 1%の確率でエラーが発生することを示す。１つのトランザクション内では、どの送受信でもエラーの発生率は同一とした。

それぞれのエラー率において、トランザクションを一万回ずつ実行し、計算結果を求め、提案モデルと２相コミットメントの性能を比較した。比較要素として、トランザクションの成功率、アボート率、不整合が発生する率、及び（データが暗号化され）一時的に使用不可となる確率を用いた。送信するデータ長は 2k バイトとした。

３．２　実験結果

まず図 4 に 2 相コミットメントを用いた場合のシミュレーション結果を示す。トランザクションには送信中と受信時にエラーが発生するため、合計で 12 ヶ所にエラー発生の可能性が有る。結果を見ると、2 相コミットメントでは、通信時のエラー率が 50%を超えるとほとんどがアボート処理されてしまう。また不正発生率はエラー率が 10%～20%の辺

りが一番高く、その後減少傾向にある。

本提案モデルの特徴は、データを利用不可にして送信する点と、コミットに時間差をつける点の 2 点であるが、まず２相コミットメントを派生させ、コミットに時間差をつけた場合の結果を図５に示す。コミットに時間差をつけることで、データの不整合の発生率を減らすことができたが、完全にはなくすことは出来ない。

図６に共通鍵を用いたトランザクションモデルのシミュレーション結果を示す。共通鍵を用いたモデルでは、データが送信側／受信側の双方に存在する可能性は回避しているが、最後に鍵を送信する時点で障害が発生すると、データが一時的に使用できなくなる。これを一時的使用不可率と呼ぶ。図に成功率、アボート率、及び一時的使用不可率を示す。図７に、公開鍵を用いたモデルのシミュレーション結果を示す。公開鍵鍵モデルでも、一時的に使用不可な状態が発生する。（表２参照）同様に成功率、アボート率、及び一時的使用不可率を示す。

３．３　考察

３．３．１　2 相コミットメント

エラー率が 10%～20%の辺りでアボート率がピークに達し、その後減少している。原因として、エラー率が 20%を超えるとトランザクション全体がアボートされる率が 80%と非常に高い点が上げられる。コミット処理まで行かずにプリペア要求以前でエラーが生じると、トランザクション全体がアボートされるためコミット時の不整合が起きにくい。

３．３．２　共通鍵モデル

このモデルでは、2 相コミットメントより通信に障害が発生した場合と比べ、アボート処理の回数が多い。
これは、２相コミットで同時にコミットした場合に受信側でエラーが発生する（不整合）割合が、共通鍵モデルで受信側へのコミットに障害が発生した場合のアボートへ置き換えられたためである。
そのため、エラー時のアボート率が 2 相コミットより高く、根本的に不整合が起きにくいモデルとなっている。
また、データが一時的に使用できなくなる状態が発生するが、これを 2 相コミットの不整合と等価と見ても、その割合は 2 相コミットの不整合発生率の 20％程度に抑えることが出来ている。
成功率は２相コミットメントに比べほぼ同一となっている。

３．３．３　公開鍵モデル

このモデルでは、受信側がコミット受信に失敗した場合に、公開鍵モデルでは、一時的不整合発生率が共通鍵モデルより高く２倍程度ある。更に、トランザクション全体の成功率が共通鍵モデルにくらべ、20%程度減少する。これは先に受信側が送信側へ鍵を渡すという処理があるため、共通鍵モデルより処理回数が増えることが原因であると予想できる。
一時的使用不可となる割合は 2 相コミットの不整合の発生率の 20％程度に抑えることが出来ている。

３．３．４　速度比較

２相コミットメント、フラグモデル、共通鍵モデルについて速度比較をおこなった。共通鍵は triple DES（鍵長 112bits）を用いた。結果を図８に示す。２相コミットとフラグモデルは速度的に差はないが、共通鍵モデルを用いると約３～１０倍の処理速度の差が出て

くる。共に処理速度は 1sec 以下であるが、公開鍵を用いたモデルでは、暗号／復号化の処理速度から約 1000 倍程度の時間がかかると想定される。これはトランザクションのみの処理速度であるため、今後ネゴシエーションなど手続き全体を含んだ処理時間で検討を進めていく必要があると考えられる。

4．まとめ

本実験では、シミュレーションから、２相コミットメント、共通鍵モデル、フラグモデルについて検討を行った。
提案したモデルでは、複製・紛失を回避すると共に、データの一時的使用不可率も 2 相コミットの不整合の発生率に比べ、20%～50%におさえることが出来た。

5．謝辞

本研究にご協力頂いた本郷室長、青野主任研究員、石井主任研究員、及び情報流通研究室の方々に感謝する。

[参考文献]

[1]日経 BP 社編, ”デジタルマネーのすべて” p21-29, 日経 BP 社, 1997

[2] P.Bernstein,E.newcomer 著、大磯他訳 “トランザクション処理システム入門”, 日経 BP 社,1998

[3] J.Lyon、et al. “Transaction Internet Protocol Ver 3.0”, RFC2371, June 1998.

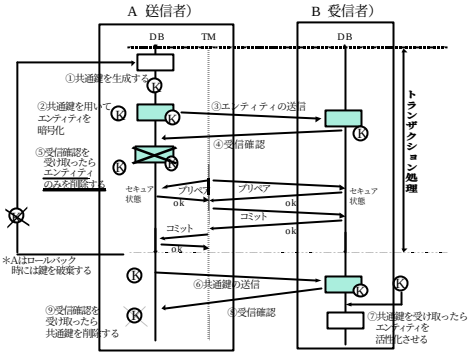

図１．提案モデル１（共通鍵の使用）

トランザクションのコミット／アボート後の状態（それぞれのDBに存在するもの）

	A 送信側)	B 受信側)	
コミット	共通鍵	暗号化されたエンティティ	
アボート	エンティティ	なし	
Bのコミットで失敗	エンティティ	なし	→ アボート処理
Bのokで失敗	エンティティ	暗号化されたエンティティ	→ アボート処理
Aのコミットで失敗	エンティティ	暗号化されたエンティティ	→ アボート処理

Bのコミットで失敗… Aのコミットは実行されず初期状態に戻る
Bのコミットも実行されずロールバックする
⇒アボートと同じ振舞いとなる

Bのokで失敗…Aのコミットは実行されない
Bのコミットは実行され、暗号化されたエンティティができるが
A側の共通鍵が破棄されるため、実体化はできない

表１．提案モデル１のトランザクション終了時処理途中（後に共通鍵の送信）の状態

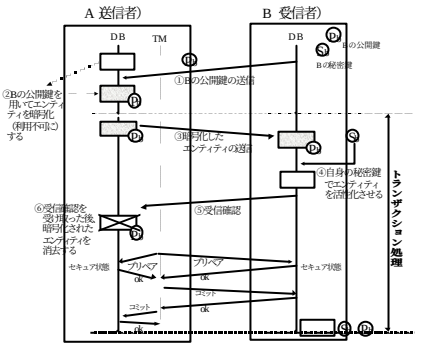

図２．提案モデル２（公開鍵の使用）

トランザクションのコミット／アボート後の状態（それぞれのDBに存在するもの）

	A 送信側)	B 受信側)	
コミット	なし	エンティティ	
アボート	Bの公開鍵をかけたエンティティ	なし	
Bのコミットで失敗	Bの公開鍵をかけたエンティティ	なし	→ アボート処理
Bのokで失敗	Bの公開鍵をかけたエンティティ	エンティティ	→ 成功
Aのコミットで失敗	Bの公開鍵をかけたエンティティ	エンティティ	→ Aに使用できないデータが残る)

Bのコミット失敗時… Aのコミットは実行されず初期状態に戻る
Bのコミットも実行されずロールバックする
⇒アボート処理

Bのokで失敗…Aのコミットは実行されない
Bのコミットは実行され、エンティティが活性化される。
A側のエンティティはTMにより破棄される。

Aのコミットで失敗…Bのokの失敗と同じ

表２．提案モデル２のトランザクション終了時の状態（すべての処理が終了した状態）

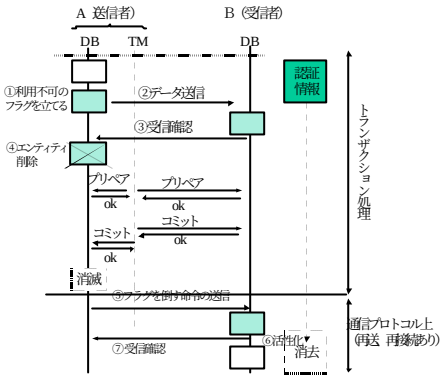

図３．提案モデル３（フラグの使用）

トランザクションのコミット／アボート後の状態（それぞれのDBに存在するもの）

	A 送信側）	B 受信側）	
コミット	なし	フラグの立ったエンティティ	
アボート	エンティティ	なし	
Bのコミットで失敗	エンティティ	なし	→ アボート処理
Bのokで失敗	エンティティ	フラグの立ったエンティティ	→ アボート処理
Aのコミットで失敗	エンティティ	フラグの立ったエンティティ	→ アボート処理

Bのコミットで失敗…Aのコミットは実行されず初期状態に戻る
Bのコミットも実行されずロールバックする
⇒アボートと同じ結果になる

Bのokで失敗…Aのコミットは実行されない
Bのコミットは実行され、暗号化されたエンティティができるが
A側の共通鍵が破棄されるため、実体化はできない

Aのコミットで失敗… Bのokの失敗と同じ

表２．提案モデル３のトランザクション終了時の状態（すべての処理が終了した状態）

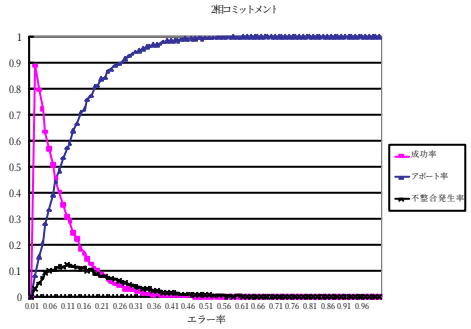

図４．２相コミットメント
シミュレーション結果

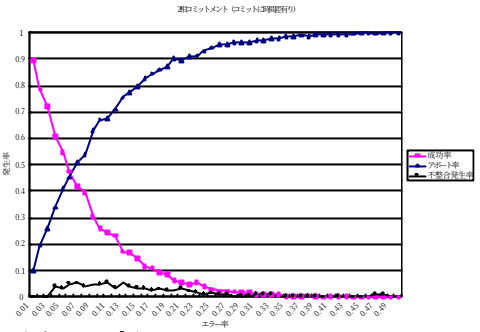

図５．２相コミットメントシミュレーション結果（時間差あり）

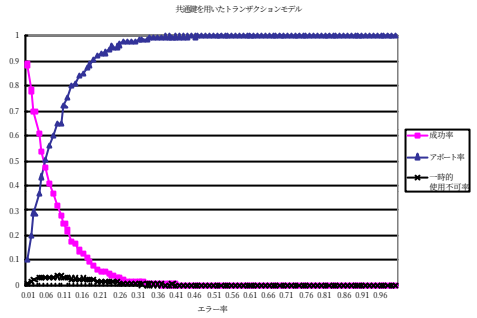

図６．共通鍵モデル
シミュレーション結果

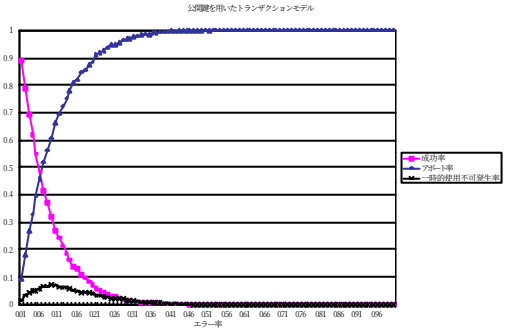

図７．公開鍵モデル
シミュレーション結果

モデル ＼ 通信速度	処理時間（μs） 64kbps	384kbps
2相コミットメント	30612.47 (0.3sec)	5812.47 (0.05sec)
フラグモデル	31211.47 (0.3sec)	6411.47 (0.06sec)
共通鍵モデル	89746.34 (0.9sec)	64946.34 (0.7sec)

図８．速度比較