LB-001

Model Driven Architecture における 実装非依存トランザクション計算モデルの提唱
A platform independent transaction computation model for Model Driven Architecture

滝本 雅之†　伊藤 和夫†　梅村 晃広†　岡 秀樹†　神谷 慎吾†
Masayuki Takimoto　Kazuo Ito　Akihiro Umemura　Hideki Oka　Shingo Kamiya

1. はじめに

OMG（Object Management Group）が提唱している MDA（Model Driven Architecture）では，厳密なモデル記述が要求される．しかし，UML（Unified Modeling Language）では，汎用性重視の観点から明確な計算モデルの規定をしていない[1]．一方，厳格な計算モデルとして広く知られている Shlaer-Mellor 法 のオブジェクト計算モデルでは，共通資源へのアクセス時に発生する最低限の順序制御しか規定がない．このためリカバリ処理などを実行するためには，業務ロジック設計者が状態遷移図の記述時にこの実現方式を考慮して設計する必要がある．

著者らは，Shlaer-Mellor 法によるオブジェクト計算モデルにおいて，イベント送受信時に永続キューを中継させ，さらにトランザクションの開始及び終了を明確化することで，業務系システムの設計者が容易にトランザクション処理を記述できる計算モデルを提唱する．

2. トランザクション計算モデル

2.1 Shlaer-Mellor 法におけるオブジェクト計算モデル[2] [3]

Shlaer-Mellor 法におけるオブジェクト計算モデルは，状態モデルと，メッセージ送受信で実際の処理を行う計算オブジェクト及びその制御を行う制御オブジェクトとで表現したモデル（図１）によって構成される．

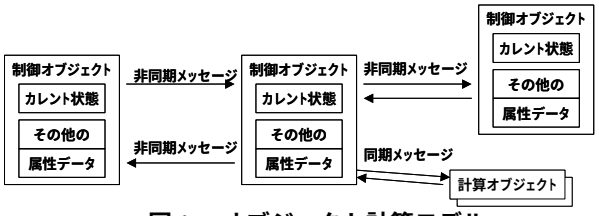

図１　オブジェクト計算モデル

オブジェクト計算モデルの主な特徴を，以下に示す．

（1）計算オブジェクト

計算オブジェクトは，インスタンス変数を保持し同期的なメソッド呼び出しを受け付ける．

（2）非同期メッセージ（イベント）

・　非同期メッセージの送達は，保証される．
・　同一の送信元から同一の送信先への非同期メッセージは，送信と同一の順序で受信される．他の受信順序及び通信遅延には，何の保証もない．

（3）制御オブジェクトと状態遷移

制御オブジェクトは，計算オブジェクトの特別な場合で，計算オブジェクトの性質に加えて，以下の機能を有する（図 2）．

・　各制御オブジェクトは，並列に動作する状態遷移機械である．
・　非同期メッセージを送受信する．
・　非同期メッセージの受信に伴い，状態遷移を行う．
・　状態遷移にアクション（遷移アクション）がある場合，そのアクションを実行する．また，アクションの実行が完了するまでは次のメッセージ受信は実行されない．
・　アクションは，通常の手続き型プログラム言語と同様の機能を有する他，計算オブジェクトのメソッド呼び出し，非同期メッセージの送信を実行できる．

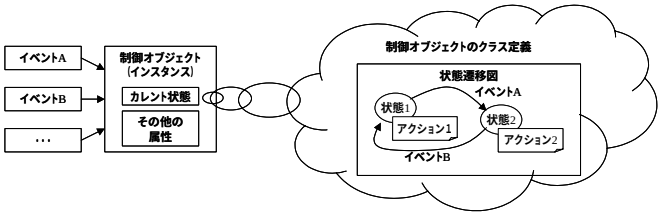

図２　制御オブジェクトと状態遷移

2.2 トランザクション計算モデルへの拡張

2.1 節で述べた Shlaer-Mellor 法では，排他制御のためにキューモデルに近い方式を採用している．しかし，共有リソースのリカバリなど，トランザクション処理に関する明確な規定がなされていない．このため，高度なロジックを業務ロジック設計者に強いることとなってしまう．Shlaer-Mellor 法を業務系システムの開発に適用するには，業務ロジック設計者の負担を軽減するための拡張が必要といえる．

そこで，従来の Shlaer-Mellor 法に対して以下の拡張を行うことにより，トランザクション処理をより容易に記述できるトランザクション計算モデル（Transaction Computation Model）を提唱する．

（1）トランザクションサブシステム

論理的に，単一のトランザクション管理下にある業務システムの集合をトランザクションサブシステムと呼ぶこととする．本稿では主に単一のトランザクションサブシステムについて議論する．

（2）メッセージ送受信

制御オブジェクト間のメッセージ送受信では，図３に示すように，各制御オブジェクトに暗黙の永続キューを持つキューモデルとして考える．

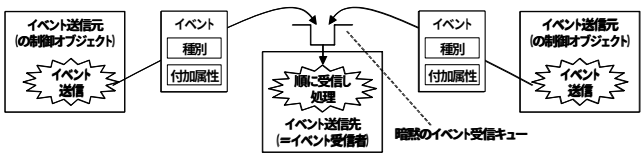

図３　永続キューを用いた非同期メッセージ

（3）トランザクションによる排他制御

トランザクションによる排他制御を実現するため以下の規定を追加した．

・　イベント受信の直前から，遷移アクションの完了及び永続キュー内部の操作までを 1 つのトランザクションとして扱う（図 4）．

†　株式会社エヌ・ティ・ティ・データ
技術開発本部　開発担当

ただし，トランザクション外部の動作は当該トランザクションに含まない．この規定により，BEGIN 及び COMMIT を明示する必要がなくなり，記述の簡略化が可能となる．

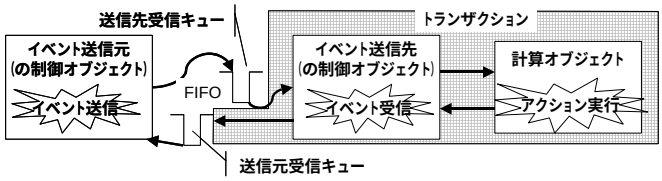

図４　トランザクション境界

3. 本モデルの検証[4] [5]

前節で述べたトランザクション計算モデルに関して，設計側が意図しないアボートと，プリンタや ATM などの取り消しが不可能な外部デバイスとの結合に関して本モデルがどのように振舞うかを検証する．

3.1 システムアボートの検証

設計者が意図しないアボートを大別するとシステム故障によるアボート（以下システムアボート）とスケジューラの共有オブジェクトへの並行アクセス制御に起因するアボートとに分類できる．本稿では，システムアボートに対する検証を行う．

システムアボートは，ハード的障害などにより発生する．この場合，障害の発生したシステムは強制終了する．本モデルは 2.2 節で述べたように永続キューが存在している．このため故障したシステムのエラー回復後，キューに対してリトライすること（図５）で，容易に ACID 属性を確保できる．

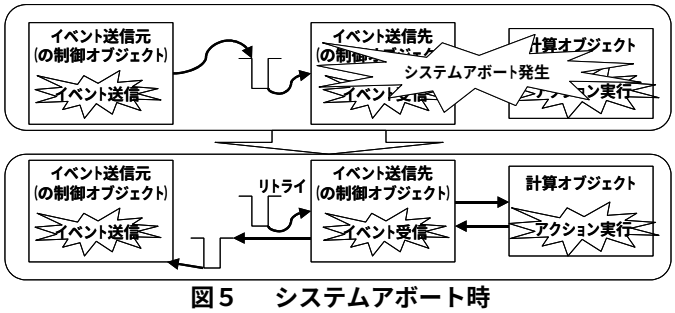

図５　システムアボート時

3.2 取り消せないデバイス への対応

発券機や ATM などの取り消すことのできない処理を含むシステムのモデル化を検証する．

このとき，対象デバイスが以下の条件を満足するようにすれば，本モデルを適用可能である．

（1）個々の外部デバイスは独立したトランザクションを有する．

（2）外部副作用が複数回発生しないことを保障する．

一般に外部副作用は，紙詰まりなどの物理的な障害に依存するため，業務ロジックから独立させるべきである．本モデルにおいて上記条件を満たすとき，業務ロジック設計者は，外部デバイスへメッセージを送信するところまでを意識すればよい．また，制御オブジェクト間のメッセージ交換と同様にみなすことができるため，モデルの可読性も向上する（図６）．

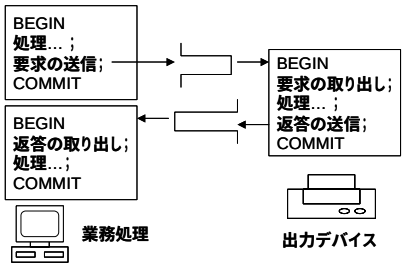

図６　外部副作用への対応

4. まとめ

本稿では Shlaer-Mellor の計算モデルに，

（1）非同期メッセージの送受信に永続キューを適用，

（2）非同期メッセージの受信からアクションの実行終了までを一つのトランザクションと規定，

の 2 規則を追加することにより，要求に非同期メッセージを用いるという制約のもと，トランザクション処理を記述可能な計算モデルが得られることを示した．

また，今後の課題として以下の点が挙げられる．

・　業務系システムで多用される同期的トランザクション要求への対応：　業務系システムでは制御オブジェクト間のメッセージ交換も同期型である場合が多い．同期メッセージ交換に対する検討も今後の課題である．

・　トランザクションサブシステムが複数ある場合の計算モデル検証：　複数のトランザクションサブシステム間の連携は，Web サービスやシステム統合などは一般的に行われており，このケースに関しては，更に詳細な検討を必要とする．

・　業務系システムへ実装するためのマッピングの検討：MDA の最重要課題である PIM-PSM 変換へ適用するためには，本モデルを実装環境にマッピングすることで，業務系システム開発に適用可能か，さらに変換後の性能についても検証する必要がある．

本モデルがシステム開発で利用できるレベルに達すれば，業務ロジック設計者と，システム方式設計者の分離が可能となり，開発効率の向上が期待できる．

5. 謝辞

本プロジェクトは情報処理振興事業協会（IPA）の次世代ソフトウェア開発事業の一環として行われました．本プロジェクトにおいて有益なコメントをお寄せくださった，東京大学の米澤 明憲教授，東京工業大学の柴山悦哉教授，IPA の林 浩史様に深く感謝いたします．

6. 参考文献

[1] Object Management Group: OMG Unified Modeling Language Specification (Action Semantics), (2002). www.omg.org

[2] Stephen J. Mellor, Marc J. Balcer: Executable UML *Addison Wesley* (2002).

[3] S.シュレイヤー， S.J.メラー: 続 オブジェクト指向 システム分析， 近代科学社（1995）．

[4] Jim Gray， 喜連川監訳: トランザクション処理-概念と技法-， 日経 BP 社 (2001).

[5]Philip Bernstein，Eric Newcomer: トランザクション処理システム入門， 日経 BP 社 (1998).