

オブジェクト指向実行検証系 ROAD/EE (2) ～統合方式と実行メカニズム～

1K-8

三菱電機(株)情報システム研究所
柳生 理子、中島 毅、田村 直樹、萩原 正敏

1.はじめに

オブジェクト指向実行検証系ROAD/EEは、OMT法のモデル記述を支援するCASEツール群であり、仕様実行する機能を備えている。

本論文では、ROAD/EEのシステム構成を示し、その中で特に特徴的な部分であるデータ統合方式と実行方式について述べる。

2.システム構成

図1にROAD/EEのシステム構成を示す。

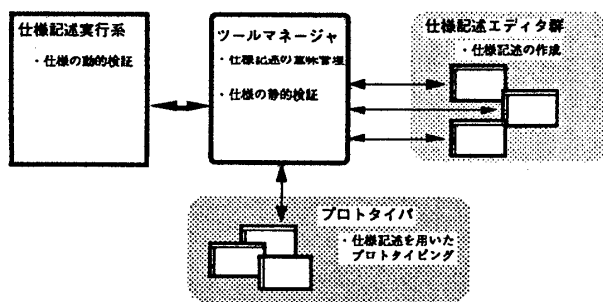


図1. ROAD/EEのシステム構成

以下で各部について述べる。

(1)仕様記述エディタ群

ROAD/EEでは、OMT法の仕様記述モデルに対応した仕様入力用エディタを提供している。現在、オブジェクト図エディタ、状態図エディタ、イベントフロー図エディタ等を用意している。

(2)ツールマネージャ

ツールマネージャは、複数の仕様記述エディタから入力される仕様記述から意味データを抽出し管理する。また、エディタから登録される意味データを基に、複数の仕様記述モデルにまたがる整合性のチェックを行う。さらに、エディタやプロトタイプからの様々な問い合わせを処理したり、仕様記述実行系の実行を制御し、実行結果をエディタ、プロトタイプに通知する。

(3)仕様記述実行系

仕様記述実行系は、ツールマネージャによって意味的に統合化された仕様を実行する部分である。実行系は、入力された仕様記述モデルを基に、インスタンスを作成し、各々のインスタンスを状態図に基づいて動作させる。この際、状態図中に現れるアク

ション、アクティビティ等の記述を基に各種演算も行う。更に、複数のインスタンスが並行に状態遷移を起こした時の非決定性の検出(複数の可能な状態遷移の発生、変数の同時更新)といった動的検証も行う。

(4)プロトタイプ

ツールマネージャが管理する仕様記述を基にプロトタイプリングを行うツールである。プロトタイプに関しては[1]で詳述する。

3.データ統合方式

ツールマネージャは、複数の仕様記述エディタを通して入力される仕様記述から意味データを抽出し管理する。ROAD/EEでは、中央のリポジトリを介して描画のためのデータを含む全データの一元管理を行う方式ではなく、次のようなデータ管理方式を採用している。

- ・ツールマネージャは、意味データのみを扱うリポジトリを保持する。リポジトリ内の全仕様要素が唯一のIDを持つことを保証する。
- ・各エディタはそれぞれの保存のためのデータとデータ形式を持つことができる。唯一の制限は、リポジトリへの登録時に返ってくるIDを保持することである。エディタはこのIDを用いてツールマネージャへ要素に対する再登録、名前付け、関係の検索を依頼する。
- ・エディタは、編集操作時にはなくある時点でとめて自分が持つ意味データを登録する。この方式の利点は次のようになる。
- ・エディタがリポジトリとは独立に動作可能である。
- ・リポジトリは、エディタにおける意味に関連しない編集操作やデータを関知しないのでよい。
- ・編集操作の度に意味データを登録しその意味的なチェックをする必要がないので編集操作へのオーバーヘッドが小さい。
- ・図的文法(エディタ内)と意味的なエラーのチェック(リポジトリ内)を明確に分けることができる。

3.実行方式

3.1 オブジェクトと通信のモデリング

OMT法は、システムの動的側面を表す仕様モデルとして、並列性と階層性を持たせた拡張状態遷移図 Statecharts[2]を採用している。Statechartsでは、

ROAD/EE (2) - Its Architecture for Data Integration and Execution -

Riko YAGIU, Tsuyoshi NAKAJIMA, Naoki TAMURA, Masatoshi HAGIWARA
Mitsubishi Electric Corporation

一つの並列状態は一つの計算プロセスと見なされ、プロセス間の通信はイベントのプロードキャストによって行う。

OMT法では、一つの並列状態を一つのオブジェクトと考える。クラスは所属するオブジェクトの動作を仕様化する単位である。オブジェクト間のメッセージ通信記法については、次の2種類の提案がある。

- ・ Statecharts内ではメッセージ送信の相手をクラス名で指定しておき、インスタンスの関連を記述する他のモデルによって実際の通信相手を決める [3]

- ・ 上のやり方でクラス名のかわりに通信ポート名を指定する [4]

ROAD/EEでは、OMT法で用いる関連の「役割」を送信相手を同定するために用いる。これは2つの提案の混合であり、通信相手の型と役割を指定する。

3.2 実時間のモデリング

Statechartsでは、刺激に対する応答を時系列的にとらえる状態遷移図を基本にしているため、実時間の経過とともに変化する自然量をモデル化できない。そのため、エンジニアリング系で扱うことが多い連続時間系の離散時間制御等の問題を細かにシミュレーションすることができない。

この問題に対して、Malerら [5]は、StatechartsのOR状態が連続時間量の時間変化を表す微分方程式を持てるように拡張し、その状態に入った際に、その微分方程式に従って時間経過とともに連続時間量を変化させる計算モデルを提案している。

ROAD/EEでは、基本的にこのMalerのモデルを採用している。これによって連続時間系をも含むシステムのモデリングと実行が可能になる。

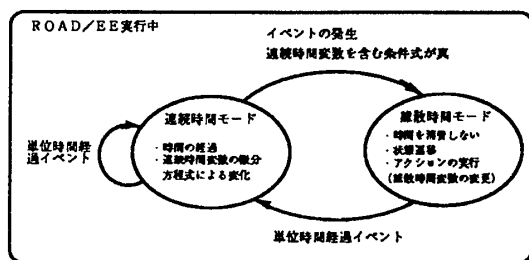


図2. ROAD/EEの実行モード

図2は、我々が提案する実行方式のモード遷移を示している。図2において、ROAD/EEの実行は、離散時間モードと連続時間モードの間を行き来する。離散時間モードでは、オブジェクトに対する直接のサービス要求、タイムアウト、条件式の評価値の真への変化をイベントとして受け、状態値や属性値の更新を行なう。連続時間モードでは、時刻をカウントし、オブジェクトの状態に定義された属性値の時

間変化式(微分方程式)に従って属性値を更新する。

3.3 実行環境

OMT法のモデリングでは、オブジェクト図に書かれたクラスを中心として、複数のモデルで記述された「仕様」が書かれる。このクラス対応の仕様に対して、複数の「実行環境」を定義することができる。実行環境には、次のものが定義されている。

- ・ 登場するオブジェクトインスタンス
- ・ インスタンスの初期状態と属性値
- ・ インスタンス間の関係

実行環境は、ツールマネージャで定義されるが、実行時には仕様記述実行系(図1)にロードされる。実行が開始されると、入ってくるイベントに対応して、初期値として与えられた登場インスタンスの状態値と属性値を順次変更して行く。

ROAD/EEでは、次のイベント生成の方法がある。

- ・ イベントトレース図よりスケジュールされた外部イベント列を与える
- ・ オブジェクトにイベントを送ったり属性値を変更したりするUIF部品から直接入力する [1]

仕様記述実行系の結果は、ツールマネージャを通じて、起動中のプロトタイプとエディタプロセスに伝えられる。

4. まとめ

ROAD/EEにおける、仕様記述の意味データのみを統合するデータ統合方式と、より細やかなプロトタイピングを行うための実行メカニズムについて述べた。

参考文献

- [1] 田村他, オブジェクト指向実行検証系ROAD/EE (3) ~プロトタイピング~, 第50回情報処理学会全国大会, 1K-9
- [2] D. Harel, "On Visual Formalisms," *Comm. of the ACM*, Vol. 31, No. 5, pp. 514-530, May 1988
- [3] D. Coleman et al., "Introducing Objectcharts or How to use Statecharts in Object-Oriented Design," *IEEE Trans. of SE*, Vol. 18, No. 1, pp. 9-18, Jan. 1992
- [4] N. G. Leveson et al., "Experiences Using Statecharts for a System Requirements Specification," *IEEE 6th Int'l Workshops on S/W Specification and Design*, pp. 31-41, 1991
- [5] O. Maler et al., "From timed to Hybrid Systems," *LNCS600, Real-Time Theory in Practice*, pp. 447-484, Springer-Verlag, 1991