4U－1

ROAD/EEにおける実行可能仕様と実行エンジン実現方法について

和泉 秀幸　中島 毅　田村 直樹

三菱電機（株）情報技術総合研究所

1 はじめに

ソフトウェアシステムへの要求を記述するモデルとして、Rumbaughらが提案するOMT法[1]がある。複数の図法を使って仕様を記述することで、仕様をもれなく記述できる反面、作成した仕様からシステムの動作イメージがつかみにくい、図法に不慣れな人には理解が難しいといった問題がある。

このような問題を解決するため、我々はOMT法で記述された仕様をもとにアニメーションなどを利用して検証する環境ROAD/EE(Realtime Object oriented Analysis and Design / Execution Environment)[2]の開発を進めている。

実行エンジンは、ROAD/EEの1機能部分であり、与えられた要求仕様とインスタンス情報を基に仕様の実行を行う。ここでは、OMT法で要求仕様を記述した場合における実行時の意味論と、それに基づく実行エンジンの実現について述べる。

2 実行エンジンの位置付け

2.1 実行エンジンとROAD/EE

ROAD/EEは次のような要素で構成される。

- 仕様作成用エディタ
  要求仕様に対するオブジェクト図、状態図などを作成するためのエディタ群。
- プロトタイパ
  各インスタンス対応で動作する図柄（アイコン）、操作入力I/Fの作成を支援する。また、実行結果からアニメーションを動作させる。
- 実行エンジン
  エディタ群で作成された要求仕様を、プロトタイパからの指示に従って実行する。
- ツールマネージャ
  各ツールを取りまとめる。ツール間でのデータの一貫性を維持し、データ通信I/Fを提供する。

以下にROAD/EEでのプロトタイピングの手順と実行エンジンの位置付けを示す。(図1参照)

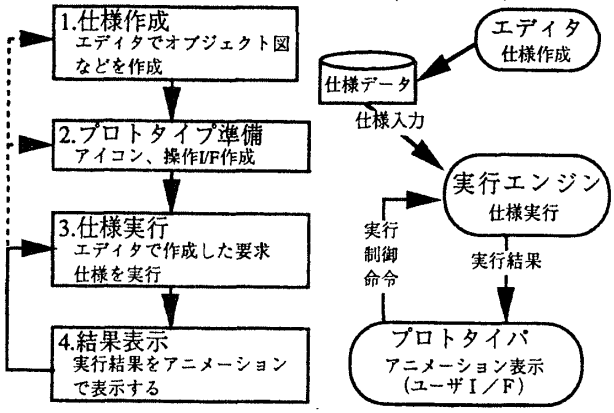

図1. プロタイピングの手順と実行エンジンの位置付け

2.2 実行エンジンの達成目標

実行エンジンは基本的に分析段階の仕様を対象とする。要求仕様を入力言語として実行し、結果報告、妥当性を検証する。その達成すべき要件を示す。

1. 明確な実行意味論の定義
   OMT法で記述された仕様は明確な実行意味論をもたないため、これを定義する。その際、(1)仕様記述の解釈の妥当性、(2)非決定性の回避とチェック可能性の追及に留意する必要がある。
2. 効率
   効率のよい仕様実行の実現。
3. 仕様としての柔軟性の追及
   あいまいな型宣言(宣言時には型を決めずに、値を代入した時点で型を決定する)、省略記法といった簡便な仕様作成への対応など。

3 実行可能な仕様への意味づけ

3.1 仕様の実行順序問題

OMT法で記述されたクラスの複数のインスタンスや、1インスタンス内の複数のAND状態は、

Executable Specification Language and Execution Engine for ROAD/EE,
Hideyuki IZUMI, Tsuyoshi NAKAJIMA, Naoki TAMURA,
Information Technology R & D Center,
Mitsubishi Electric Corp.
5-1-1 Ofuna, Kamakura, Kanagawa 247, Japan

状態図の元々の実行意味論 [3] から並列に動作可能な状態機械である。一般に並行プロセスの実行は半順序である。このため、「同じ入力でも実行順序により結果が異なる場合がある」という非決定性の問題がある。

例えば、図2は AND 関係の状態遷移図の例である。図2で、イベント E1 によりイベント E2、E3 が派生して発生する。状態 C では、E2 → E3 の順で発生した場合 状態 C3 へ遷移するが、E3 → E2 の順で発生した場合 状態 C2 までしか遷移しない。

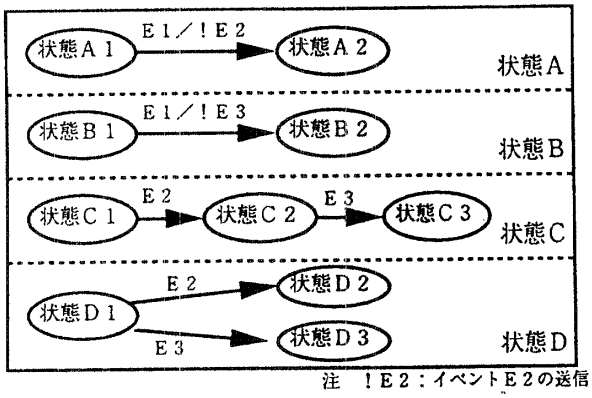

図2. 問題発生の例

3.2 実行可能な仕様の実行順序

要求仕様としてとらえた場合、同じ外部イベント列に対して、同じ結果が出力されることが正しい（ここで外部イベントとは、実行エンジンに対して外部すなわちプロトタイパから送られてくるイベントであり、実行エンジン内部で派生的に発生するものと区別する）。実行エンジンでは非決定性の問題に対して、仕様のバグとして報告することで対応するが、実現にあたっては非決定の判定が現実的な時間で計算できる必要があるため、実行時に次の解釈を行う。

「外部イベントに対する処理は同時である」

同時と解釈した時の処理を図3を用いて説明する。外部イベントが送られると、実行エンジン内部では、状態遷移、属性値の更新、オペレーション呼出し、イベント発生などが派生的に起きる。これらの一連の処理では、(1) 処理中に参照される値 (属性値、現在状態) は外部イベント発生直前の値、(2) 処理中に更新された値 (属性値の更新、遷移先の状態) が有効になるのは一連の処理終了時、にする。

図2の例にこの解釈を行うと、状態 C では 状態 C1 のみが遷移元状態となり、実行順序による非決定性はなくなる。しかし、状態 D では 状態 D1 が遷移元であり、状態 D2、状態 D3 への非決定性の遷移が残る。実行エンジンでは、このようなイベントの内部伝搬順序に非依存である非決定性を、仕様のバグとして報告する。

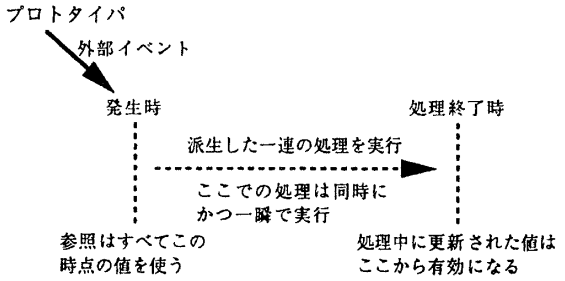

図3. 外部イベント処理の概要

尚、この解釈では記述力の低下を伴うが、要求仕様の解釈の仕方として適当な範囲と考える。なぜなら、1外部イベントで、図2の状態 C のように「C1 → C2 → C3」と連続遷移したり、同じ属性値の更新が何回も行われるのは、仕様として見づらく不適当なためである。

4 実行エンジン実現

実行エンジンに必要な要件を満たすため、次のような機能を実現する。

- 要求仕様の妥当性判定機能 (要件 1)
  仕様実行時のみ判定可能な仕様上の制約 (関連として接続できるインスタンスの個数制約など) をチェックする機能。外部イベント処理終了時など、情報更新が済んだ直後にチェックする。
- 最適化機能 (要件 2)
  静的な情報である状態遷移先などの情報をコンパイルすることで、仕様実行を効率的に行う。
- 仮想計算機 (要件 3)
  あいまいな型宣言や省略記法を解釈して計算を実行する仮想計算機。スタックを設け関数呼出しも可能にする。

5 おわりに

OMT 法で記述した要求仕様に対して実行時の意味づけを行った。これに基づいて、要求仕様の仕様記述としての妥当性、非決定性の回避とチェックが可能な実行エンジンの実現を検討した。

参考文献

[1] J.Rumbough 他, “オブジェクト指向方法論 OMT,” 凸版, 1992.7.

[2] 中島他, “新しい要求分析スタイルの構築 ～ オブジェクト指向仕様の実行検証系 ROAD/EE ～,” 本会 サマーワークショップ・イン・立山, pp.73-80 (1995-7).

[3] D. Harel et al., “On the Formal Semantics of Statecharts,” In Proceedings Symposium on Logic in Computer Science, pp.54-64 (1987)