ソフトウェア工学 127－6
（2000．5．26）

オブジェクト指向分析モデルの検証と公理系の提案

立石 孝彰，青木 利晃，片山 卓也
北陸先端科学技術大学院大学 情報科学研究科

近年，大規模かつ複雑なソフトウェアの開発において，オブジェクト指向方法論を用いた開発が主流になってきている．この方法論では，開発対象システムをオブジェクトという概念を用いて整理し，その結果，システムの複雑さを軽減させることが可能になった．しかし，分析モデルの検証手法は提案されていない．このため，分析工程での誤りが，設計または実装の工程になって発見されると，再びシステムの分析をやり直す必要が生じる．このことは，開発効率の低下をもたらす．また，分析における誤りを発見できないと，開発されたシステムに重大な欠陥をもたらすことがある．本論文では，開発対象システムに対する分析モデルにおいて，オブジェクトのもつ性質に関する検証を行なうための手法と，そのための公理系を提案する．この公理系は，構築した分析モデルから，計算機により自動生成できるような形式で定義する．

An Axiomatic System for Verifying Analysis Models

TAKAAKI TATEISHI, TOSHIAKI AOKI and TAKUYA KATAYAMA
School of Information Science,
Japan Advanced Institute of Science and Technology

The scale of software product is becoming larger as a result of the rapid progress and increasing use of computer systems. Object-oriented methodologies have been proposed for the development of such large systems. In object-oriented methodologies, a target system is organized into collective behavior of objects. Though these methodologies reduce the complexity, it is still difficult to verify the system and to provide it with high reliability. In this paper, we propose a verification method for checking properties which always hold in the analysis model, and an axiomatic system for this verification. The axiomatic system is defined so that we can automatically generate it from the constructed analysis model. This is because it is costly that we define an axiomatic system for every constructed analysis model.

1. はじめに

近年，様々なオブジェクト指向方法論が提案されている．オブジェクト指向方法論を用いることで，大規模なシステム開発の複雑さを軽減させることができる．これらの手法では，分析，設計，実装の3つの工程に分けることで，開発を行なうことができる．分析工程では，開発システムの論理的な振舞を捉えるために，分析モデルを構築する．分析以降の工程では，分析モデルを用いて開発を行なうため，分析工程でモデルの正当性を保証することは重要である．

そこでまず，分析モデルの形式化を行なう．開発システムに出現するオブジェクトを，クラス図を用いて表す．また，各々のオブジェクトの振舞を，属性とイベントを伴う状態遷移図を用いて表す．クラス図と状態遷移図を用いて分析を行なう手法は，OMT[3]，UML[4]等で広く導入されている．クラス図と状態遷移図には，オブジェクトに対する制約を記述することができる．制約をモデルに記述することによって，開発システムが持つ性質を明らかにすることができる．

状態遷移図に対する検証手法として，**モデル検査法**と呼ばれる手法が提案されている．この検証手法では，すべての到達可能な状態を探索して，状態遷移機械に関する性質の検証を行なう．このため，状態爆発という問題が生じるが，BDD[5]と呼ばれる効率的な表現を用いることで回避することができる．しかし，無限の値を持ち得る属性を伴う状態遷移図では，探索する状態数が無限となるため，この手法を適用することはできない．本論文で提案する検証手法では，状態遷移図に関する帰納法を用いて，数学的な証明を行なう．このことより，無限の値を持ち得る属性を，直接取り扱うことができる．

本論文では，分析モデルにおいて，オブジェクトに与えられた制約に関する検証手法と，検証に必要な公理系を提案する．2節では，対象とする分析モデルの形式的な定義と，検証の概要を示す．3節では，検証に必要な公理系を提案する．4節では，本論文で提案した公理系と検証手法を例題に適用する．5節では，本研究に対する今後の展望について述べる．最後の6節では，本論文のまとめを行なう．

2. 分析モデルと検証の方針

この節では，分析モデルの形式的な定義と，オブジェクトが持つ属性に対する検証の方針を示す．

2.1 分析モデルの定義

オブジェクト指向分析において，開発システムは，クラスと状態遷移図の概念を用いてモデル化される．オブジェクトは，属性を持つことができる．オブジェクトの振舞は，クラス毎に定義される．そして，オブジェクトの振舞を，状態遷移図で示す．

本論文で用いる分析モデルは，形式的には次の通りである．

$$(C, O, STD, M_i, M_b)$$

C : クラスの集合

O : オブジェクトの集合

STD : 状態遷移図の集合

M_i : $O \to C$

M_b : $C \to STD$

M_i は，オブジェクトがどのクラスから実体化したのかを表す．オブジェクト o が，クラス c から実体化することを，$c = M_i(o)$ と書く．クラスとオブジェクトの実体化関係を表すために，記号 $\lhd$ を次の通りに定義する．

$$o \lhd c \stackrel{\text{def}}{\Leftrightarrow} c = M_i(o)$$

M_b は，クラスから実体化したオブジェクトの振舞を表す．クラス c から実体化したオブジェクトの振舞が，状態遷移図 std で表現されるとき，$std = M_b(c)$ と書く．状態遷移図 std は，形式的には次の通りに定義する．

$$std = (S, E_i, E_o, B, A, T, s_0)$$

S : 状態の集合

E_i : 入力イベントの集合

E_o : 出力イベントの集合

B : 遷移条件の集合

A : アクションの集合

$T \subseteq S \times E_i \times B \times P(E_o) \times A \times S$
: 状態遷移の集合

$s_0 \in S$
: 初期状態

オブジェクトは，イベントを用いて通信を行なうことによって，協調動作を行なう．状態遷移図は，イベント出力と入力を伴うものとする．出力されるイベントの集合を E_o で表し，入力されるイベントの集合を E_i で表す．$P(E_o)$ は，E_o の巾集合である．

図 1 は，状態遷移 $(s_1, e_1, n == 0, \{e_2, e_3\}, n := n+1, s_2)$ を表している．この遷移は，次の意味を持つ．オブジェクトが，状態 s_1 であるときに，イベント e_1 を受けとり，遷移条件 $n == 0$ が成り立てば，オブジェクトの状態は，s_2 に変わる．ここで，n は，整数型の属性であり，$==$ は，2 つの整数値を引数にとり真偽値を返す中置演算子である．遷移が起こると，アクション $n := n+1$ が実行され，イベント e_2, e_3 が，出力される．アクション $n := n+1$ により，整数型の属性 n が持つ整数は，1 だけ増加する．

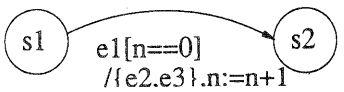


図 1 状態遷移

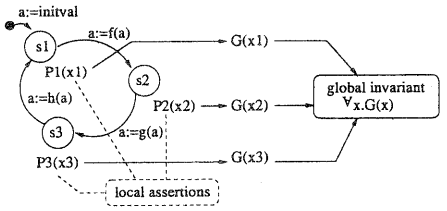


図 2 検証手法

2.2 検証手法

状態遷移図のすべての状態において，常に成り立って欲しい性質を**大域表明** (global assertion) と呼び，すべての状態において，常に成り立つ性質を**大域不変表明** (global invariant) と呼ぶ．

図 2 を用いて，大域表明 $\forall x.G(x)$ を検証する場合を以下で示す．ここで，a は属性を表し，その初期値は initval である．x_1, x_2, x_3 は，それぞれ，状態 s_1, s_2, s_3 における属性 a の値である．

(1) それぞれの状態に**局所表明** (local assertion) を割り当てる．局所表明とは，ある一つの状態で成り立って欲しい性質のことである．図 2 では，状態 s_1,s_2,s_3 にそれぞれ局所表明 $P_1(x_1)$,$P_2(x_2)$,$P_3(x_3)$ を割り当てている．

(2) 割り当てた局所表明が，それぞれの状態で成り立つことを証明する．この証明には，状態遷移図に関する帰納法を用いる．具体的には，図 2 の場合には，次のことを証明する．

(a) $\forall x_1.P_1(x_1)$

(b-1) $P_1(x_1)$ が成り立つならば，
$P_2(x_2)[f(x_1)/x_2]$ が成り立つ

(b-2) $P_2(x_2)$ が成り立つならば，
$P_3(x_3)[g(x_2)/x_3]$ が成り立つ

(b-3) $P_3(x_3)$ が成り立つならば，
$P_1(x_1)[h(x_3)/x_1]$ が成り立つ

ここで，f, g, h は，関数である．$t[a/b]$ は，t に出現する b を a で置き換えて得られた式である．この置き換えは，$b := a$ というアクショ

ンを表現したものである．以上を証明すると，$P_1(x_1)$,$P_2(x_2)$,$P_3(x_3)$ が常にそれぞれの状態 s_1, s_2, s_3 で成り立つことが言える．常にそれぞれの状態で，局所表明が成り立つとき，それらの局所表明は，妥当であるという．妥当性が示された局所表明を局所不変表明 (local invariant) という．

(3) 大域表明が，すべての状態でも常に成り立つことを証明するには，局所不変表明から大域表明が導けることを示せば十分である．図2の場合，まず局所不変表明 $P_1(x_1)$,$P_2(x_2)$,$P_3(x_3)$ から，$G(x_1)$,$G(x_2)$,$G(x_3)$ を導く．そして，x_1,x_2,x_3 は，すべての状態における属性の値を表すため，$G(x)$ を導くことができる．

次節では，前述の証明を行なうための公理系を提案する．

3. 公 理 系

分析モデルにおいて誤りが見つかると，分析モデルを再び構築し直して，そして，検証を行なう．このような分析モデルの構築プロセスでは，分析モデルの構築と検証が何度も行なわれる．このため，分析モデルに対する検証のための公理系を，何度も構築する必要がある．そこで，本論文では，分析モデルから自動的に公理を生成することを考える．そのために，分析モデルの雛形から，公理系の雛形を構成できるようにする．この公理系の雛形を，以下に示す．

3.1 局所不変性

ある一つの状態に割り当てられた局所表明が妥当であるとは，その局所表明が，その状態で常に成り立つことである．オブジェクト obj において，局所表明 $P_{s_1}, P_{s_2}, \cdots, P_{s_n}$ が妥当であるためには，Class_VALID obj P_{s_1} P_{s_2} $\cdots$ P_{s_n} が成り立たなければならない．ここで，Class_VALID は以下の通りに定義する．

定義 1 (LVALID-DEF)

$$\mathsf{Class\_VALID}\ obj\ P_{s_1}\ P_{s_2} \cdots\ P_{s_n} = \bigwedge_t \mathcal{V}_t$$

where

$$\mathcal{V}_{t_0} = P_{s_1}(\mathsf{initval}_1, \mathsf{initval}_2, \cdots, \mathsf{initval}_m)$$

$$\mathcal{V}_{t_i} = P_s(obj.\mathsf{attr}_1(s), \cdots, obj.\mathsf{attr}_m(s)) \wedge (\mathsf{TransCond}_{t_i}) \Rightarrow P_{s'}(f_{i1}(obj.\mathsf{attr}_1(s), \cdots, obj.\mathsf{attr}_m(s)), \cdots, f_{im}(obj.\mathsf{attr}_1(s), \cdots, obj.\mathsf{attr}_m(s)))$$

そして，それぞれの識別子は以下の通りである．

- m は，属性の個数を表す．
- $obj.attr(s)$ は，オブジェクト obj の状態 s における属性 $attr$ の値を表す．

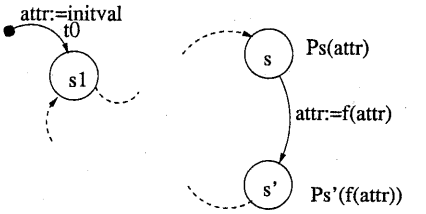


図 3 局所表明の妥当性

- t_i は，遷移を表す．
- f_{ij} は，遷移 t_i に伴うアクションを表す関数である．
- s, s' は，遷移 t_i に対する前状態と後状態を表す．
- $\mathsf{TransCond}_{t_i}$ は，遷移 t_i に伴う遷移条件を表す．

定義1において，P_{s_1},P_{s_2},$\cdots$,P_{s_n} は，それぞれ状態 $s_1, s_2, \cdots, s_n$ に割り当てた局所表明を表す．s_1 は，状態遷移図の初期状態である．そして，$\mathsf{initval}_i$ は，P_{s_1} の i 番目の属性の初期値を表す．

局所不変表明は，それぞれの状態に割り当てた局所表明が妥当であることから導かれる．このことを表す公理は次の通りである．

公理 1 (LINV-AX)

$$\forall obj\ P_{s_1}\ P_{s_2}\ \cdots\ P_{s_n}\ .\ obj \triangleleft \mathsf{Class} \Rightarrow (\mathsf{Class\_VALID}\ obj\ P_{s_1}\ P_{s_2}\ \cdots\ P_{s_n} \Rightarrow (\bigwedge_{i=1,2,\cdots,n} P_{s_i}(obj.\mathsf{attr}_1(s_i), \cdots, obj.\mathsf{attr}_m(s_i))))$$

3.2 大域不変性

大域表明 G を導くには，G が，それぞれの状態で常に成り立つことを証明すれば十分である．大域不変表明は，それぞれの状態において大域表明が成り立つことから導くことができる．このことを表す公理が，次の公理2である．

公理 2 (GINV-AX)

$$\forall obj\ .\ obj \triangleleft \mathsf{Class} \Rightarrow (\forall I.((\bigwedge_{i=1,2,\cdots,n} I(obj.\mathsf{attr}_1(s_i), \cdots, obj.\mathsf{attr}_m(s_i))) \Rightarrow (\forall s.I(obj.\mathsf{attr}_1(s), \cdots, obj.\mathsf{attr}_m(s)))))$$

例として，属性 $attr$ を持つオブジェクト obj を考える．オブジェクト obj は，状態 $s_1, s_2, \cdots, s_i$ を含む状態遷移図を，振舞として持つものとする．このとき，大域不変表明 G は，以下に示す手順で証明できる．まず最初に，各々の状態に局所表明 $P_{s_1}(obj.attr(s_1))$，$P_{s_2}(obj.attr(s_2))$，$\cdots$，$P_{s_i}(obj.attr(s_i))$ を割り当てる．これらは，公理1を用いることで，$P_{s_1}(obj.attr(s_1)) \wedge P_{s_2}(obj.attr(s_2)) \wedge \cdots \wedge P_{s_i}(obj.attr(s_i))$ を導くことができる．これは，$P_{s_1}(obj.attr(s_1))$, $P_{s_2}(obj.attr(s_2))$, $\cdots$, $P_{s_i}(obj.attr(s_i))$ が，局所不変表明であることを表す．次に，

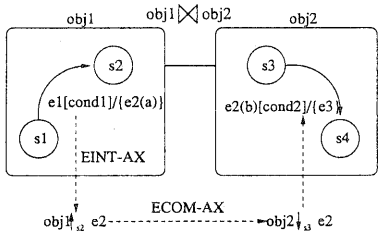


図 4　イベント通信

$$P_{s_1}(obj.attr(s_1)) \Rightarrow G(obj.attr(s_1))$$
$$P_{s_2}(obj.attr(s_2)) \Rightarrow G(obj.attr(s_2))$$
$$\vdots$$
$$P_{s_i}(obj.attr(s_i)) \Rightarrow G(obj.attr(s_i))$$

を証明して，大域表明を導く．このとき，$G(obj.attr(s_1)) \wedge G(obj.attr(s_2)) \wedge \cdots \wedge G(obj.attr(s_i))$ が成り立つことを導くことができる．そして，公理 2 を用いて，$\forall s.G(obj.attr(s))$ を導くことができる．

3.3　イベント通信

オブジェクトは，イベントを用いて他のオブジェクトと通信を行なう．遷移が起きると，オブジェクトはイベントを出力する．そして，イベントを受けとることで遷移が起きる．図 4 は，オブジェクト obj_1 と obj_2 の通信を表したものである．オブジェクト obj_1 が，状態 s_1 で，イベント e_1 を受けとり，かつ，遷移条件 $cond_1$ が成り立つと遷移が起こる．そして，属性 a を伴うイベント e_2 が出力される．

このようなイベントを用いたオブジェクト間の通信は，**イベント出力公理**と**イベント通信公理**の 2 つの公理で表す．イベント出力公理は，遷移によってイベントが出力されることを表す公理である．この公理を次の公理 3 に示す．

公理 3 (EOUT-AX)

$\forall obj\ .\ obj \lhd \mathsf{Class}$
$\Rightarrow ((obj \downarrow_s e) \wedge \mathsf{TransCond}$
$\quad \Rightarrow (obj \uparrow_{s'} e'_1 \wedge \cdots \wedge obj \uparrow_{s'} e'_n) \wedge \mathsf{P(attr)})$

ただし，次の通りに定義する．

- $obj \downarrow_s e$ は，オブジェクト obj が，状態 s において，イベント e を受けとったことを表す．
- $obj \uparrow_s e$ は，オブジェクト obj が，状態 s において，イベント e を出力したことを表す．
- TransCond は遷移条件である．
- P(attr) は，属性 attr が持つ性質を表す．

イベント通信公理は，あるオブジェクトから出力されたイベントが，別のオブジェクトによって受けとられることを表した公理である．本論文で提案している分析モデルでは，リンクのあるオブジェクト間でのみ，イベント通信が行なわれる．リンクは，クラス間の関連が実体化したものである．オブジェクト o_1 と，オブジェクト o_2 にリンクがあることを，$o_1 \bowtie o_2$ と表す．図 4 を見ると，オブジェクト obj_1 と，オブジェクト obj_2 にはリンクがあるので，$obj_1 \bowtie obj_2$ が成り立つ．イベント通信公理は，次の通りである．

公理 4 (ECOM-AX)

$\vdash \forall obj_1\ obj_2\ .\ obj_1 \bowtie obj_2$
$\quad \Rightarrow (\forall P\ .\ (obj_1 \uparrow_s e) \wedge P(a_1, a_2, \cdots, a_n))$
$\qquad \Rightarrow (obj_2 \downarrow_{s'} e \wedge P(b_1, b_2, \cdots, b_n)))$

オブジェクトは，イベントを用いて属性値の受渡しを行なうことができる．公理 4 では，$a_1, a_2, \cdots, a_n$ は，出力イベントの属性を表す．$b_1, b_2, \cdots, b_n$ は，入力イベントの属性を表す．

図 4 の例において，オブジェクト obj_1 が状態 s_1 でイベント e_1 を受けとり，遷移条件 $cond_1$ が成り立っていると仮定する．このことは，$obj_1 \downarrow_{e_1} s_1 \wedge cond_1$ と表すことができる．公理 3 を用いることで，$obj_1 \uparrow_{e_2} s_2 \wedge P(a)$ を導くことができる．ここで，P は，出力イベント e_2 が持つ属性 a の性質を表現している表明である．次に，$obj_1 \bowtie obj_2$ を仮定すると，公理 4 を用いることで，$obj_2 \downarrow_{e_2} s_3 \wedge P(b)$ を導くことができる．$obj_2 \downarrow_{e_2} s_3$ は，オブジェクト obj_2 が状態 s_3 においてイベント e_2 を受けとったことを表し，$P(b)$ は，属性 b のもつ性質を表している．このことから，イベント e_2 は，オブジェクト obj_1 から obj_2 に渡され，同時に，属性 a の性質 P が，属性 b に渡されたことを示すことができた．

4. 例　　題

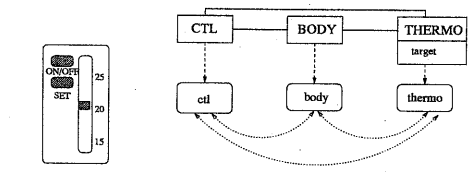
4.1　分析モデル

ここで，例として，小規模なエアコンの分析モデルを考えることにする．エアコンは，コントローラーと温度調節器と本体から構成されていると考える．コントローラーを用いて，温度の設定や，エアコンの電源を ON にすることができる．

このエアコンの分析モデルのクラス図を，図 5 に示す．

エアコンと温度調節器は，コントローラーによって電源の ON/OFF が，行なわれるものとする．また，温度調節器では本体を制御するための基準温度の設定を行なうことができる．設定温度は，コントローラーで調節される．エアコンは，温度計によってアイドル/アクティブの状態に変更されるものとする．アイドル/アクティブの状態が，すぐに切り替わることを避けるために，アイドル状態において，室内温度が基準温度よりも 2 度高ければアクティブ状態になるようにしなければならない．同様に，アクティブ状態において，室内温度が基準温度よりも 2 度低ければアイドル状態になるようにしなければならない．本体/コ

ントローラー/温度調節器の関係を表すために，図5で示すクラス図を用いる．このクラス図には3つのクラス BODY,CTL,THERMO と，3つのオブジェクト body,ctl,thermo がある．これら3つのクラスは，それぞれ本体，コントローラー，温度調節器を表すクラスである．3つのオブジェクトは，それぞれのクラスから実体化したものである．



コントローラー　　クラス図

図5 コントローラとクラス図

図6は，3つのオブジェクトの振舞を，状態遷移図 STDBody,STDCtl,STDThermo を用いてそれぞれ表したものである．コントローラーの電源ボタンを押すと，イベント ctl_pow が，オブジェクト ctl から出力される．オブジェクト body は，イベント ctl_pow を受けとり，エアコンの電源が ON になる．そして，温度設定ボタンを押すと，イベント set_target が属性 n を持って出力される．属性 n の値は，設定温度を表す．出力されたイベント set_target は，オブジェクト thermo によって受けとられる．そして，基準温度 target が変化する．基準温度が室内温度より低いとき，エアコンは冷却を始める．そして，同時にしばらく冷却し続けるために，基準温度の値を2度だけ下げる．

4.2 検　　証

ここで，エアコンの分析モデルに対して，基準温度は，常に30度以下であるということを検証する．この制約は，次の通りに形式的に記述できる．

$\forall s.\mathrm{thermo.target}(s) \leq 30$

そして，証明のために，以下の仮定を導入する．

$$
\begin{aligned}
\mathrm{ASSUMS} &= \{a_1, a_2, a_3, a_4, a_5, a_6\} \\
\mathsf{a}_1 &= \forall s\ .\ \mathrm{thermo} \downarrow_s \mathrm{temp} \\
\mathsf{a}_2 &= \forall s\ .\ \mathrm{ctl} \downarrow_s \mathrm{push\_pow} \\
\mathsf{a}_3 &= \forall s\ .\ \mathrm{ctl} \downarrow_s \mathrm{ctl\_set} \\
\mathsf{a}_4 &= \mathrm{ctl} \bowtie \mathrm{body} \\
\mathsf{a}_5 &= \mathrm{body} \bowtie \mathrm{thermo} \\
\mathsf{a}_6 &= \mathrm{thermo} \bowtie \mathrm{ctl}
\end{aligned}
$$

仮定 a_1 は，温度調節器 thermo が，室内温度の変化を検出することを表す．仮定 a_2 と a_3 は，コントローラー ctl のボタンが押されて，イベント push_pow と，ctl_set が，それぞれ押されたことを表す．仮定 a_4 から

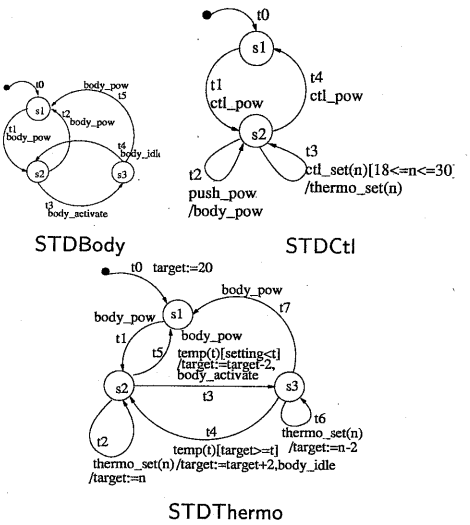


図6 状態遷移図

a_6 は，それぞれのオブジェクト間に，リンクが存在することを表す．

まず，状態 s_1,s_2,s_3 のそれぞれに，局所表明 $P_{\mathsf{s}_1}, P_{\mathsf{s}_2}, P_{\mathsf{s}_3}$ を割り当てる．

$$
\begin{aligned}
P_{\mathsf{s}_1}(x) &= x \leq 30 \\
P_{\mathsf{s}_2}(x) &= x \leq 30 \\
P_{\mathsf{s}_3}(x) &= x \leq 28
\end{aligned}
$$

公理1-4の公理に，エアコンの分析モデルを当てはめることによって，この検証のための公理を生成することができる．こうして生成された公理系の一部を，付録Aの公理 **AC1**, **AC2**, **AC3**, **AC4** と，定義 **AC1** に示す．

次に，すべての状態において，与えられた局所表明が成り立つことを証明する．この証明の一部を付録Aに示す．P_{s_1}, P_{s_2}, P_{s_3} が局所不変表明であることを示すには，$\mathcal{CV}_{t_0}, \mathcal{CV}_{t_1}, \cdots, \mathcal{CV}_{t_7}$ を示せば十分である．$\mathcal{CV}_{t_3}$, $\mathcal{CV}_{t_6}$ についての証明は，証明 **P1** と **P2** にそれぞれ示した．最後に，P(thermo.target) が大域不変表明であることの証明は，証明 **P3** に示した．以上から，P(thermo.target) は，常に成り立つことを示すことができる．

5. 今後の展望

5.1 計算機支援

エアコンの例で示したように，この検証では，一般に複雑で長い証明ステップを伴う．一方，計算機上で証明を行なうための定理証明器が提案されており，証明を効率的にかつ，正確に行なうことができる．さらに，モデル構築支援や，プロトタイプ実行を行なうツールと連携させることによって，効率的なシステム

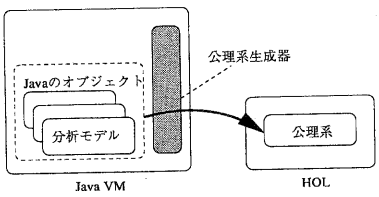


図 7 検証支援環境

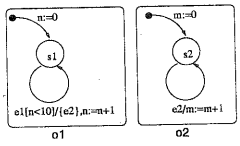


図 8 検証不可能な例

開発を行なうことが期待できる．

3 節で述べたように，本稿で提案した公理系の定義を用いると，構築された分析モデルに基づいて，公理系を自動生成することができる．現在，定理証明器 HOL と，Java を用いて，検証支援のための環境を開発している．この検証支援環境のアーキテクチャを図 7 に示す．分析モデルと，公理系生成器は，Java を用いて実装する．公理系生成器は，HOL が読み込むコードを生成する．

5.2 複数のオブジェクトにまたがる性質

図 8 において，オブジェクト o_1 で出力されたイベント e_1 は，オブジェクト o_2 で受けとることができる．オブジェクト o_1 では，遷移条件 $n \leq 10$ によって，イベント e_1 が出力される回数は 10 回以下である．一方，オブジェクト o_2 は，イベント e_2 を受け取ると m に 1 を加える．ここで，$m \leq 10$ を証明したいとする．これを証明するためには，イベント e_2 がオブジェクト o_1 から 1 回出力されると，必ず，オブジェクト o_2 が 1 回受けとるという事実が必要である．しかしながら，図 8 のモデルは，この事実を表現しておらず，我々が提案した公理系においても証明できない．このような性質を扱うためには，オブジェクト間の通信の特性を整理し，それを表現する公理を導入する必要がある．通信の特性としては，出力されたイベントは必ず 1 回だけ受け取られる，イベントが失なわれる，イベントが重複する等が考えられる．これらの特性を我々の公理系で取扱可能にすることが，今後の課題の 1 つである．

5.3 制約記述言語

現在，広く利用されているモデル表記法として UML がある．UML には，クラス図，オブジェクト図，ステートチャート図等の多くのモデル表記法が定義されている．そして，これらのモデル中に制約を記述するための制約記述言語として，*OCL*(Object Constraint Language) が定義されている．しかし，モデルに与えられた制約を検証するための手法は定められていない．そこで，本研究で提案した検証手法と公理系を拡張して，OCL を扱うことができるようにする．

6. ま と め

本論文では，分析モデルにおいて与えられた制約を検証するための手法と，そのための公理系を提案した．この検証手法は，分析モデルに対する公理系を生成し，その公理系を用いて証明を行うものである．公理系は，構築した分析モデルから，自動生成可能な形式で定義されている．現在，公理系を自動生成するためのシステムを，HOL と Java を用いて構築中である．

参 考 文 献

1) T.Aoki, T.Katayama: How to support verification of object-oriented analysis models using HOL, Sci/ISAS' 99 at Orlando USA, pp.525-532, 1999
2) T.Aoki, T.Katayama: Unification and Consistency Verification of Object-Oriented Analysis Models, APSEC '98 at Taipei Taiwan, pp.296-303, 1998
3) J.Rumbaugh, M.Blaha, M.Premerlani, F.Eddy and W.Lorensen: Object-Oriented Modeling and Design, Prentice Hall, 1991
4) Object Management Group, Inc: OMT Unified Modeling Language Specification (version 1.3), 1999
5) E.Clarke, O.Grumberg and D.Long: Verification Tools for Finite-State Concurrent Systems, LNCS 803, 1993

付　　録

A.1 公　　理

公理 AC1 (LINV-AX)

$\forall obj\ P_{s_1}\ P_{s_2}\ P_{s_3}\ .\ obj \lhd \mathsf{Thermo}$
$\Rightarrow (\mathsf{Thermo\_VALID}\ obj\ P_{s_1}\ P_{s_2}\ P_{s_3}$
$\quad \Rightarrow (\bigwedge_{i=1,2,3} P_{s_i}(obj.\mathsf{target}(\mathsf{STDThermo.s}_i))))$

定義 AC1 (LVALID-DEF)

$\mathsf{Thermo\_VALID\ thermo}\ P_{s_1}\ P_{s_2}\ P_{s_3} = \bigwedge_{i=0,\cdots,7} \mathcal{CV}_{t_i}$

$\mathcal{CV}_{t_3} =$
$P_{s_3}(\mathsf{thermo.target(STDThermo.s_3)})$
$\wedge(\mathsf{thermo} \downarrow_{\mathsf{STDThermo.s_3}} \mathsf{temp})$
$\wedge((\mathsf{thermo.target(STDThermo.s_3)}) \geq (\mathsf{temp.t}))$
$\Rightarrow P_{s_2}(\mathsf{thermo.target(STDThermo.s_3)} + 2)$
$\mathcal{CV}_{t_6} =$
$P_{s_3}(\mathsf{thermo.target(STDThermo.s_3)})$
$\wedge(\mathsf{thermo} \downarrow_{\mathsf{STDThermo.s_3}} \mathsf{thermo\_set})$
$\Rightarrow P_{s_3}(\mathsf{thermo\_set.n} - 2)$

公理 AC2 (GINV-AX)

$\forall obj\ .\ obj \lhd \mathsf{Thermo}$
$\Rightarrow ((\bigwedge_{i=1,2,3} G(obj.\mathsf{target}(\mathsf{STDThermo.s}_i)))$
$\quad \Rightarrow (\forall s.G(obj.\mathsf{target}(\mathsf{STDThermo.s}))))$

公理 AC3 (EOUT-AX)

$\forall obj\ .\ obj \lhd \mathsf{Ctl}$
$\quad \Rightarrow ((obj \downarrow_{\mathsf{STDCtl.s_2}} \mathsf{push\_pow})$
$\quad\quad \Rightarrow (obj \uparrow_{\mathsf{STDCtl.s_2}} \mathsf{body\_pow}))$
$\forall obj\ .\ obj \lhd \mathsf{Ctl}$
$\quad \Rightarrow ((obj \downarrow_{\mathsf{STDCtl.s_2}} \mathsf{ctl\_set})$
$\quad\quad \Rightarrow (obj \uparrow_{\mathsf{STDCtl.s_2}} \mathsf{thermo\_set})$
$\quad\quad\quad \wedge(18 \leq \mathsf{thermo\_set.n} \leq 30))$

公理 AC4 (ECOM-AX)

$\forall obj_1\ obj_2\ .\ obj_1 \bowtie obj_2$
$\Rightarrow ((\forall P\ .\ (obj_1 \uparrow_{\mathsf{STDCtl.s_2}} \mathsf{thermo\_set}) \wedge P(\mathsf{thermo\_set.n}))$
$\quad \Rightarrow (obj_2 \downarrow_{\mathsf{STDThermo.s_3}} \mathsf{thermo\_set}) \wedge P(\mathsf{thermo\_set.n}))$
$\forall obj_1\ obj_2\ .\ obj_1 \bowtie obj_2$
$\Rightarrow ((\forall P\ .\ (obj_1 \uparrow_{\mathsf{STDThermo.s_2}} \mathsf{body\_idle}))$
$\quad \Rightarrow (obj_2 \downarrow_{\mathsf{STDBody.s_3}} \mathsf{body\_idle}))$

A.2 証　　明

証明 **P1**

$$\dfrac{\dfrac{\mathsf{ASSUMS} \vdash \mathsf{T}}{\mathsf{ASSUMS} \vdash ((\mathsf{thermo.target(STDThermo.s_1)}) \leq 30) \Rightarrow ((\mathsf{thermo.target(STDThermo.s_2)}) - 2 \leq 28)}}{\mathsf{ASSUMS} \vdash P_{s_2}(\mathsf{thermo.target(STDThermo.s_2)}) \Rightarrow P_{s_3}(\mathsf{thermo.target(STDThermo.s_2)} - 2)}$$
$$\overline{\mathsf{ASSUMS} \vdash \mathcal{CV}_{t_3}}$$

証明 **P2**

$$\dfrac{\dfrac{\mathsf{ASSUMS} \vdash \mathsf{ASSUMS}}{\mathsf{ASSUMS} \vdash (\mathsf{thermo} \uparrow_{\mathsf{STDCtl.s_2}} \mathsf{thermo\_set}) \wedge (18 \leq \mathsf{thermo\_set.n} \leq 30)}}{\mathsf{ASSUME} \vdash (\mathsf{thermo} \downarrow_{\mathsf{STDThermo.s_3}} \mathsf{thermo\_set}) \wedge (18 \leq \mathsf{thermo\_set.n} \leq 30)}$$
$$\overline{\mathsf{ASSUMS} \vdash \mathcal{CV}_{t_6}}$$

証明 **P3**

$$\dfrac{\dfrac{\mathsf{ASSUMS} \vdash \mathsf{T}}{\mathsf{ASSUMS} \vdash \bigwedge_{i=0,\cdots,7} \mathcal{CV}_{t_i}}}{\mathsf{ASSUMS} \vdash (\mathsf{thermo.target(STDThermo.s_1)} \leq 30)\ (\mathsf{thermo.target(STDThermo.s_2)} \leq 30)\ (\mathsf{thermo.target(STDThermo.s_3)} \leq 28)}$$
$$\dfrac{}{\mathsf{ASSUMS} \vdash (\mathsf{thermo.target(STDThermo.s_1)} \leq 30) \wedge (\mathsf{thermo.target(STDThermo.s_2)} \leq 30) \wedge (\mathsf{thermo.target(STDThermo.s_3)} \leq 30)}$$
$$\overline{\mathsf{ASSUMS} \vdash \forall s.\mathsf{thermo.target}(s) \leq 30}$$