時間制約を含むシステムソフトウェアのテスト手法に関する研究
～Timed I/O Statechart を用いた機能仕様に対するテスト手法とその実装～

北海道大学大学院工学研究科　○漆原映彦、金井理、岸浪建史

要　旨

本研究は、状態遷移型の挙動を持つソフトウェアに対する機能テストの自動化手法を提案することを目的とする。本研究では、ソフトウェアの機能仕様記述として、UML/Statechartに時間制約の概念を加え拡張したTimed I/O Statechartを新しく提案し、次にこの仕様記述に基づいて、テスト系列生成、時間制約を含むテスト入出力タイミングの導出、及びテスト実施の自動化手法を開発したのでこれを報告する。

1．はじめに

ソフトウェアの効率的な開発と品質向上のため、体系的、網羅的なソフトウェアテスト手法が必要とされている。すでに著者らは、オブジェクト指向開発における状態遷移の標準的仕様記述である UML / Statechart[1]に基づいた仕様ベースの自動テスト手法を提案してきた[2]。しかし、Statechartは、多くのソフトウェアの挙動仕様記述に必要な時間制約を表すことが出来ない。そこで本研究では、時間制約を含む状態遷移の仕様記述法であるTimed I/O Statechartを新たに提案し、さらにこれに基づく体系的、網羅的テスト自動化手法を開発することを目的とする。

2．Timed I/O Statechart 仕様に基づくテスト手法の概要

図１に提案するテスト手法の概要を示す。手順としては①ソフトウェアに要求される時間制約を含む挙動を、Timed I/O Statechartにより仕様記述する。②この仕様から、テスト系列集合（入力動作系列、期待出力動作系列、入出力動作の時間制約系列のおのおのの集合）を自動生成する。③生成されたテスト系列の時間制約系列を満たし、かつ最後までテスト系列が入出力可能なテスト入出力タイミング条件を導出する。④別途開発されたテスト対象ソフトウェアに対し、②で求めた入力動作系列を入力し、実出力と②で求めた期待出力動作系列と比較する。その際、③で求めた入力タイミング条件を満たすように入力し、出力の時刻が出力タイミング条件を満たしているかをチェックする。これらの条件が全て満たされている場合このソフトウェアを合格とする。

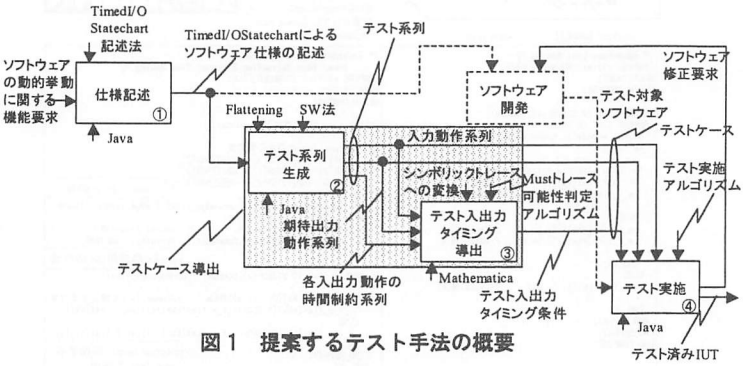


図１　提案するテスト手法の概要

3．Timed I/O Statechart による仕様記述

本研究では、提案するテスト手法に対する要求条件として下記を考慮した。

①仕様に基づいたテスト対象ソフトウェアのブラックボックステストが可能であること。

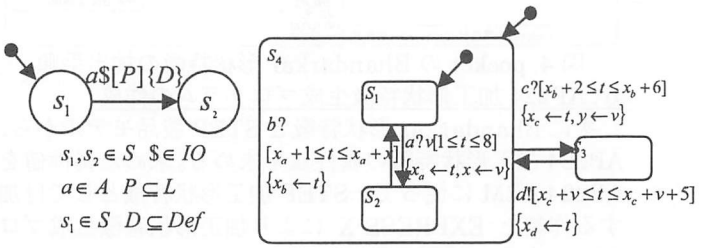
②階層性を持つ状態遷移表現が仕様で記述可能であること。

③挙動に関する時間的な制約条件が仕様上で表現可能であり、また実挙動がその制約条件を満たしているか否か検証できるテストが実施できること。

①②については、すでにStatechart仕様に基づくブラックボックステストが提案されている[2]。一方、③についてはTimed I/O Automatonを用いた階層性を持たない状態遷移仕様に基づいた時間制約を含むテスト系列生成法のみが提案されている[3]。したがって、本研究ではTimed I/O Automaton に状態の階層性を導入し Statechart 化したTimed I/O Statechartを新たに仕様記述方法として提案する。Timed I/O Statechart(TSC)の定義を以下に示す。

$$TSC =< S_0, S, A, IO, V, t, L, Def, \sigma, \delta, X >$$

ただし、S_0：初期状態集合、S：状態集合、σ：$(S \to 2^S)$



(a)状態遷移記述要素　　(b)記述例

図２　Timed I/O Statechart

下位状態を与える関数、A：入出力動作名集合、δ：状態遷移関数、IO：入出力区別記号の集合$(\{!,?\} \cup \{?v\}(v \in V))$、$V$：遷移時刻を表す変数集合、$t$：グローバルクロック変数、$L$：時間制約を記述する線形不等式$P_i$の論理式集合、$Def$：代入文の集合、$X$：各変数の初期値集合、である。Timed I/O Statechartの各遷移は図2(a)のように表現される。また、図2(b)はTimed I/O Statechartの簡単な記述例を示す。

4．テストケースの導出

テストケースとは、テスト系列とテスト入出力タイミング条件の集合から成る。またテスト系列は、入力動作系列の集合、期待出力動作系列の集合、入出力動作の時間制約系列の集合からなっている。ここでは、このテストケースをTimed I/O Statechartから自動生成する手順を述べる。

4.1．テスト系列の生成

まず、テスト系列生成の前にTimed I/O Statechartを、これと等価な階層性を持たないTimed I/O Automatonに、[2]に記述されている分解アルゴリズムを用いて分解する。

次に、テスト系列の生成にはSW法[4]を用いる。SW法の特徴としてはテスト系列が若干長くなるが、オートマトンの構造によらずテスト系列が必ず存在すると言った汎用性を持つことである。SW 法では、状態存在の確認、及び遷移存在の確認を行う入出力動作系列を網羅的に作り出すことができ、ここで作成された各テスト系列は

$\alpha=(a_1\$_1,P_1,D_1)..(a_n\$_n,P_n,D_n)$ のように表され、ここで $a_i\in A$ 、 $\$_i\in IO$ 、 $P_i\subseteq L$ 、 $D_i\subseteq Def$ である。

4.2. テスト入出力タイミングの導出

まず、導出されたテスト系列αに対して入出力動作 $a_1,...,a_n$ の実行時刻を表す変数 $t_1,...t_n$ を導入し、加えてαの時間制約 P_i に含まれる変数 $x_1,x_2,...,x_k\in V$ の値を、代入される変数 $t_1,...t_n$ に置き換える。置き換えたものをシンボリックトレースと呼び、 $w=(a_1\$_1,t_1,\overline{P_1})..(a_n\$_n,t_n,\overline{P_n})$ （ただし、t_i:各遷移の実行時刻、$\overline{P_i}:P_i$ の変数に対し各入出力の実行時刻を代入したもの）で表す。

次に、シンボリックトレースに対しテスト入出力タイミング条件の導出を行うために、Mustトレース可能性判定アルゴリズムを適用する。ここで「Mustトレース可能」とは、あるテスト系列中のある遷移における出力動作が適当な範囲内で実行されれば、その遷移の後の各入出力動作には必ず可能な実行タイミング（有限な幅を持つ動作時刻範囲）が存在する性質を持つことを意味する。前述のシンボリックトレースのままでは、テスト系列が最後まで入力できない時刻範囲を持つことがあるためテスト系列が最後まで入出力できる（Mustトレース可能な）入出力タイミング条件を以下のアルゴリズムで導出する。

① w 内の最終遷移の実行時刻 t_n に対する時間制約を次の3つに分類する。

i) $\{f(t_1,t_2,...,t_{n-1})\le t_n\}$ （時刻 t_n を左辺に含まない不等式）

ii) $\{t_n\le g_i(t_1,t_2,...,t_{n-1})\}$ （時刻 t_n を右辺に含まない不等式）

iii) $R(t_1,t_2,...,t_{n-1})$ （時間変数 t_n を含まない論理式）

②①の遷移の入出力タイミングの上下限値を導出する。まず、入出力タイミングの上限値は $t_n^{sup}=\min[g(t_1,...,t_{n-1})]$ さらに入出力タイミングの下限値は $t_n^{inf}=\max[f(t_1,...,t_{n-1})]$ より計算できる。これらより、この遷移のタイミング条件は $t_n^{inf}\le t_n^{sup}$ かつ $R(t_1,t_2,...,t_{n-1})$ が真でとなる。ここで、この条件を $TrCondMust_n$ とする。

③①②と同様に、この系列内においてに対する $TrCondMust_i$ を $i=n-1$ から $i=0$ まで順次計算していき、各遷移 $a_i(0\le i\le n-1)$ の実行可能なタイミング条件の上下限値 t_i^{inf},t_i^{sup} を求めていく。

このアルゴリズムにより 4.1 で生成されたテスト系列の各遷移に対し入出力タイミング条件を表す時間の上下限値が式の形で導出できる。これらの式を各遷移のテスト入出力タイミング条件としてこれらの式は次節のテスト実施アルゴリズム内で用いられる。

5. テスト実施アルゴリズム

4 で生成、導出されたテストケースをテスト対象のソフトウェアに実際に入力する。テストの実施アルゴリズムの概要は次のようになる（図 3）。まず、Mustトレース可能性判定によって生成されたテスト系列の入力タイミング条件（入力時刻の上下限値の間）に合致する時刻を一つ選び入力系列をテスト対象ソフトウェアに入力してゆく。次にその入力の時刻から、その後の入出力タイミング条件を表す上下限値を数値として求めていく。また、実出力が期待出力と同じであり、かつ出力を受け取った時刻が先ほど求めた出力タイミング内であれば、その遷移が正しく実装されていると判断する。これが全てのテスト系列内の全ての遷移について成り立つ時、テスト対象ソフトウェアが正しいと判定する。

6. ケーススタディ

前述のテスト手法を、テスト系列生成、テスト実施には

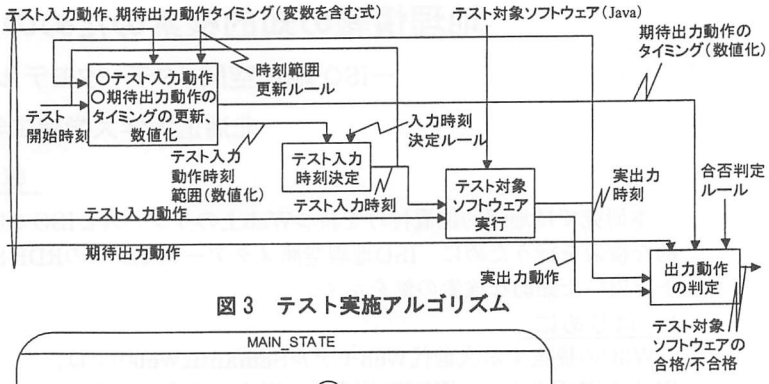


図 3　テスト実施アルゴリズム

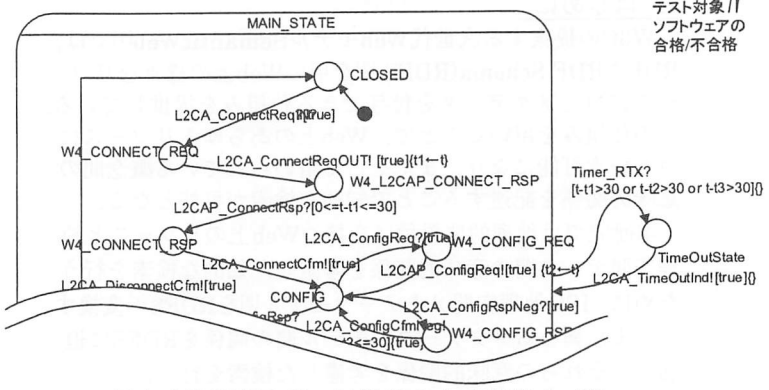


図 4　L2CAP 層の Timed I/O Statechart 仕様（一部）

Java を、またテスト入出力タイミング導出は Mathematica で実装を行った。ケーススタディとして Bluetooth の L2CAP 層のプロトコル[5]を実行する通信制御用の Java ソフトウェアをテスト対象とした。このプロトコルの Timed I/O Statechart（一部）を図 4 に示す。この L2CAP 層は上位層と下位層との通信を行いながら相手のデバイスの L2CAP 層と通信を行う。まず、相手のデバイスに対して接続要求を出し、これに対し反応を待つ。接続されるとデータを送受信し、切断の要求を出して切断する。いずれの状態でも、要求を出してから反応が 30 秒以内にない場合には初期状態に戻る、というタイムアウトの時間制約がこの Timed I/O Statechart の仕様に含まれている。

この Timed I/O Statechart から 4 で述べた手順によりテストケースを生成し、別途この仕様に基づき開発した Java ソフトウェアをテスト対象とし、自動的なテスト実施を行った。その結果、SW 法によって検証できる構造の実装誤りに加え、時間的な制約の実装誤り（例えば 30 秒経過しても初期状態に戻らないような実装等）を事前に検出できた。

7. まとめ

本研究では、Timed I/O Statechart を提案し、それによって記述された仕様に対して、テスト系列生成、テスト入出力タイミングの導出、テスト実施に至る一連のテスト手法の開発、実装を行った。また、ケーススタディによって本手法の有効性を確認することが出来た。

参考文献

[1]OMG Unified ModelingLanguage Ver1.3

[2]魚住他:Guard 条件を網羅した UML Statechart 仕様からのテストケース生成法と制御プログラム検証への応用,情報処理学会第 62 回大会講演論文集

[3]深田他:あるクラスの時間オートマトンに対する適合性試験系列生成の一手法,情報処理学会論文誌,v.40,n.1, (1999) pp85-94

[4]佐藤他：“通信システムの段階的な試験のための試験系列自動生成手法とその実現” 情報処理学会論文誌（1990）Oct Vol31 No.10 pp1486-1496

[5]Bluetooth Specification http://www.bluetooth.com/