Vol.2020-EMB-55 No.2
2020/12/11

自然言語仕様書からのテストケースの自動生成 ー機能試験への品質試験ケースの追加試験ツール

吉井 亮介[1]　青山 裕介[1]　久代 紀之[1]

概要：システム開発において，システムテストは重要な工程である．システムテストにおけるテストケース設計は，テスト設計者が，システム仕様書に記載された各文を精読し，ここから “条件 → 期待動作” のタプルを取出し機能試験ケースを抽出すること．さらに，テスト設計者のドメイン知識を用い，各機能が満足すべき品質要求を充足しているかどうかを確認する品質試験ケースを追記するタスクとみなすことができる．現状，上記タスクは，テスト設計者が保有するノウハウと製品知識により暗黙的になされているため，設計されたテストケースの品質と作業効率が，テスト設計者個々のスキルに大きく依存してしまうという課題があった．さらに，テストケースの設計根拠が明示されないため，レビューでの欠陥・漏れの抽出が困難で，テスト設計のノウハウを組織として蓄積しにくいという課題もあった．
本研究では，これら課題を解決するために，自然言語で記載されたシステム仕様書各文から，“条件 → 期待動作“のタプルを形式記述として自動抽出するアルゴリズムを開発するとともに，システムドメインで必要とされる品質試験ケースをパターンとして抽出し，これらパターンを形式記述した．さらには，システム仕様書から抽出された機能試験ケース（形式記述）と品質試験ケース（形式記述）を論理的に結合し，機能試験に品質試験項目を追記した試験ケースを自動生成するツールを開発した．

Test Case Generation Tools from natural language specifications A Tool for Supplimentating Quality Test Cases to Functional Test Cases

Abstract: System testing is an important process in system development. Test designers read each sentence in a system specification document very carefully and extract a tuple of “condition → expected behavior”. Then the test designer ellanorate functional test cases with by utilizeing their own domain knowledge, and compensate quality test cases to satisfy quality requirements for each function. The above tasks are performed implicitely and holistically, quality and efficiency of the tasks heavily depend on each test engineer’s skill. There are difficulties to detect errors and flows in test cases design throuh review, because design rationale is usually not to be explained explicitelly.
In this paper, we have developed algorithms that extracting tuples of “condition → expected behavior” and discribing as semi-formal descriptions. We have also supporting tools, which supplements quality test cases on quality test patterns to the functinal test cases, and converting the test cases including quality test cases into decision tables.

1. はじめに

システム開発において，システムがどのように動作すべきかどうかという要素は，システムにおける機能要求に該当し，開発において重要な要素であり，正しく認識されていなければならない．システム仕様書はこの機能要求を定義，記述したものであり，開発者はこの文書の各文を精読し，システムの設計，開発を行うことになる．また，開発されたシステムをテストすることも重要な工程である．実際の製品が機能要求を満たすかどうかをシステムテストによって試験していく．テスト設計者は，仕様書から試験項目を抽出し，それらを “条件 期待動作” のタプルとして，記述することで，機能要求のテストケースを作成可能である．

一方で，機能要求に対し品質要求というものもある．これは，機能要求を十分に満たしたうえで，ISO25000 シリーズ [1] で定められている規格のような，セキュリティ，使用性といったシステムの品質について定めた要素であり，適切な品質要求が十分に満たされていることが重要である．

[1] 九州工業大学
Kyushu Institute of Technology

現状，仕様書を精読し，テストケースの生成を行うという作業が行われるために，生成されるテストケースはテスト設計者の知識に依存してしまう．また，仕様書の表記によって解釈が変化する可能性もある．

さらに，品質テストについても，テスト設計者が製品ドメインにおける品質要求の知識を保有している場合があり，品質要求のテストケースの設計もテスト設計者の知識に依存してしまうという課題もある．

本研究では，これらの課題を解決するために，自然言語で記載された仕様書から表記に依存しない形式記述に変換し，テストケースを自動生成するアルゴリズムを開発する．また，テスト設計者の知識として蓄積されている品質テストケースをパターンとして抽出し，知識の蓄積やシステムテストへの再利用が可能になるよう形式化する．さらには，これらの処理をツールとして実装し，テスト設計の効率化を図る．

2. パターンの記述・蓄積

2.1 テスト設計ノウハウのパターン化

自然言語で記載された仕様書から生成されるテストケースは，仕様書に明示的に記載されるものは機能要求に関するものであることから，品質試験項目についての情報は不十分である．品質要求についての知識は，テスト設計者の知識に依存するものであり，共有する必要がある．そこで，品質試験項目についての知識をパターンとして共有する．

パターンとは，ソフトウェア設計において頻出する問題に対して適用される解決策のことである．本研究では，テスト設計がテスト設計者の知識に依存しないよう，製品ドメインの品質テストのパターンを言語化し，蓄積する．製品ドメインごとに蓄積されたパターンは，複数の製品の品質テストのために再利用することができ，テスト設計の作業効率化を可能とする．

2.2 パターンの記述方法

パターンの記述方法を統一することは，他人の記述したパターンとの形式の一致を意味し，情報の蓄積に有効な手段である．表 1 にパターンの記述方法を示す [2]．各項目は，パターンに含まれるべき要素ごとに分けられている．開発に関する知識から，パターンとして蓄積，再利用すべきと判断されたものを抽出し，各項目について記述する．表 1 の 1 項目である「解決（セミ形式記述）」には，「解決」の項目で記述した事柄をセミ形式化し，記述する（セミ形式記述については 3.1 節を参照）．この項目により，そのパターンを適用する際にどのようなテストを行うべきかを論理的に理解することができる．表 2 はパターンの記述例となる．表中の内容は，[3] でテストの具体例として示されているものを表 1 に要約したものである．

表 1　パターンの記述方法
Table 1　Pattern description method

項目	内容
名前	パターンの内容を指す造語
状況	このパターンが適用されるべき状況
問題	起こりうる問題
解決	問題に対する解決策
解決（セミ形式記述）	解決の欄をセミ形式記述でまとめたもの
結果	解決策が適用されたあとの状況
フォース	問題解決にあたり，考慮すべき事柄

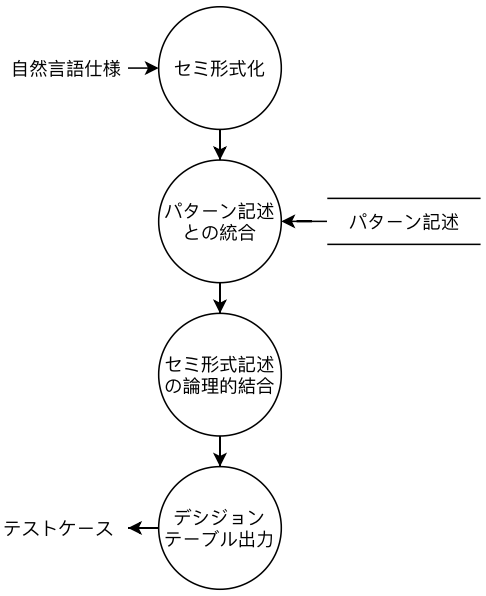

図 1　パターンを適用した自然言語仕様からのテストケース生成
Fig. 1　Test case generation from natural language specifications with patterns applied

3. テストケースの生成

本研究では，品質試験項目を含むテストケースを自動生成するための手法を提案する．自然言語仕様からテストケースが生成されるまでの流れを図 1 に示す．各処理について，以降に説明する．

3.1 自然言語仕様の形式化

自然言語で記述された仕様書は，論理関係に曖昧さがあるために，テスト設計者の解釈に誤りが生じる場合がある．[4] では，自然言語の文をセミ形式記述と呼ぶ表現で論理記述することにより，論理関係の曖昧さを排除するアルゴリズムを提案している．

セミ形式記述では，日本語文を単文化し，各単文を「関係語（主体, 対象, 制約）」という形式（以降命題プリミティブと呼ぶ）で表現することで，テストの各操作・確認項目のパラメータを明確化する．また，命題プリミティブ同士を命題論理の記号 “!”（否定），“&”（論理積），“|”（論理和），“–>”（論理包含）を用いて接続することで，厳密な

表 2　パターンの記述例
Table 2　Pattern description example

項目	内容
名前	誤入力に対するエラー表示
状況	ユーザによる入力を正しく処理する
問題	入力内容によって，ソフトウェアが誤作動することがある
解決	以下のような入力に対し，誤った入力であれば適切なエラーメッセージが 1 回表示されることを確認する（ここでいう適切とは，ユーザが何を誤ったかが理解できるということ）．型の異なる入力（整数が有効なボックスに実数・文字を入力するなど），長い入力（許容を超える長さの文字列を入力する），数値の境界（各データ型で定義されている数値の範囲を超えたり，0 を入力（正負の境界値）したりする）
解決（セミ形式記述）	入力する（ユーザ, 文字・数値, 型の異なる入力, 長すぎる入力, 数値の境界）–> 表示する（ソフトウェア, エラーメッセージ, 適切な）
結果	エラーメッセージによって，入力の誤りをユーザに伝えることができる
フォース	想定外の入力に対しても，正常に動作する必要がある

論理関係の記述を行う．

セミ形式記述への変換はルールベースのアルゴリズムとして実現されており，形態素解析器 Juman++ [6]（v2.0.0-rc2[*1]）と日本語構文解析器 KNP [7]（v4.1.9[*2]）によって日本語文を解析した結果得られる feature [8] のデータ（格や品詞などの文法データ）を用いて文節ごとに変換する．このルールは，例えば，用言の形態素を含む文節を命題プリミティブにおける関係語にする，体言でありかつガ格の助詞を含む文節を命題プリミティブにおける主体にする，用言の活用形が条件形の場合には係り先の命題プリミティブと論理包含で結合する，といったものが Python スクリプトとして記述されている．

[4] の変換アルゴリズムを用いることで，例えば「ユーザがボタンを押すと，ポットはランプを 3 秒間点灯させる．」という自然言語の文は，[4] のアルゴリズムによって「押す（ユーザ, ボタン） –> 点灯する（ポット, ランプ, 3 秒間）」と変換される．

3.2　パターン記述との統合

自然言語からセミ形式記述への変換により，論理関係の曖昧さを排除した．パターン記述との統合は 3.3 節で論理的な結合をするために必要な機能である．セミ形式の仕様文と，その仕様に適用するセミ形式のパターンのそれぞれで使用されている単語を統一することが目標である．
統合の流れを以下に示す．ここでの単語とは，セミ形式記述における関係語，主体，対象，制約に該当する文字列を示している．

(1) 仕様書（セミ形式）とパターン（セミ形式）から 1 文ずつ選出する．
(2) 各セミ形式記述を単語ごとに分解する．
(3) 各単語に対し，変換後の単語をユーザが指定する．表は単語の変換表としてイメージしたものである．表頭にパターン（セミ形式）の各単語，表側に仕様書（セミ形式）の各単語，表体の各セルに変換後の単語が格納される．各セルに入力する単語は，表頭，表側それぞれの単語の変換後を入力する．どちらも変換の必要がない場合は，空白とする．
(4) 各単語を指定された単語に変換し，新たなセミ形式記述として出力する．

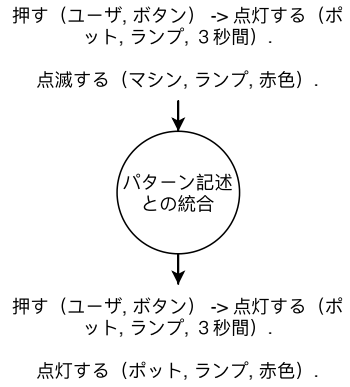

図 2　パターン記述との統合例
Fig. 2　Example of integration with pattern description

図 2 は統合の 1 例である．仕様文（セミ形式）として「押す（ユーザ, ボタン） –> 点灯する（ポット, ランプ, 3 秒間）.」，パターン（セミ形式）として「点滅する（マシン, ランプ, 赤色）.」を選出したと仮定している．パターン（セミ形式）の関係語「点滅する」を「点灯する」に，主体「マシン」を「ポット」に変換する場合の変換表は表 3 となる．各列に注目すると，「点滅する」の列に「点灯する」，「マシン」の列に「ポット」と入力されたセルが存在するため，それぞれの単語は変換され，新たなセミ形式記述が出力される．各行に注目すると，「点灯する」，「ポット」の行に空でないセルが存在するが，表側の単語と同じものであるため，単語の変換は行われない．よって，仕様文（セミ形式）は変換前のまま出力される．

[*1] https://github.com/ku-nlp/jumanpp/releases
[*2] http://nlp.ist.i.kyoto-u.ac.jp/index.php?KNP

表 3　パターンとの統合の例
Table 3　Example of integration of pattern and specification statement

		パターン記述			
		点滅する	マシン	ランプ	赤色
機能仕様	押す ユーザ ボタン 点灯する ポット ランプ 3 秒間	点灯する		ポット	

3.3　セミ形式記述の論理的結合

3.1 節，3.2 節で，自然言語からセミ形式記述の変換を行い，仕様文（セミ形式）とパターン（セミ形式）の単語を統一することができた．ここでは，2 つのセミ形式記述を論理的に結合し，1 つのセミ形式記述へと変換する処理を行う．以下に，セミ形式記述の論理的結合の順序について示す．

(1) 2 つのセミ形式記述を選出する．
(2) それぞれのセミ形式記述に含まれる命題プリミティブから 1 つずつを選出し，比較する．命題プリミティブを構成する要素のうち，関係語，主体，対象がすべて一致した場合，それらは結合可能と判断し，両命題プリミティブに含まれる制約の全要素を制約に持つ命題プリミティブを構成する．
(3) 2 つのセミ形式記述の論理構造に対して，表 4 を参照しながら適切な結合を行い，1 つのセミ形式記述を出力する．表 4 は，2 つのセミ形式記述の論理構造によって，結合結果が変化することを示している．また，各セミ形式記述を構成する命題プリミティブ（A1，A2，A3，B，C）については，以下に詳細を示す．

- A1，A2 は，関係語，主体，対象がすべて一致している命題プリミティブである．
- A3 は，A1，A2 と関係語，主体，対象がすべて一致しており，A1，A2 それぞれが持つ制約をすべて制約として持っている．
- B，C は，A1，A2，A3 と関係語，主体，対象の少なくとも 1 つは不一致（B，C 間も同様）である．

結合可能な命題プリミティブが選出できなかった場合，2 つのセミ形式記述は結合不可能とし，2 つのセミ形式記述をそのまま出力し，本処理を終了する．

図 3，4 は，論理的結合の例である．図 3 は，どちらも命題プリミティブ単体である場合の結合例である．それぞれに含まれる制約が 1 つのセミ形式記述の制約として並べられていることを確認できる．図 4 は，どちらも “–>”（論理包含）を含む場合の結合例である．2 つのセミ形式記述はそれぞれ “–>” の後件部分が一致しており，前件部分は論理和で結合され，後件部分は制約が並べられた状態で 1 つのセミ形式記述に結合されていることを確認できる．

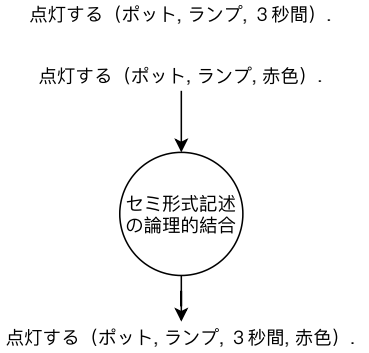

図 3　セミ形式記述の結合例 1
Fig. 3　Combination example of semi-formal description 1

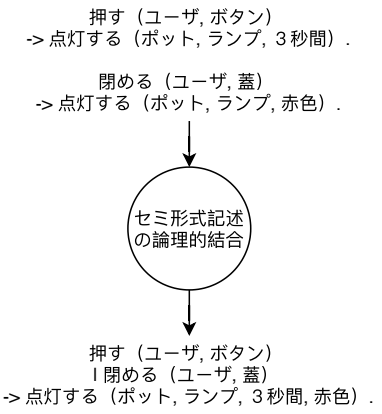

図 4　セミ形式記述の結合例 2
Fig. 4　Combination example of semi-formal description 2

3.4　デシジョンテーブル出力

前節の 3.3 によって仕様文（セミ形式）とパターン（セミ形式）を結合することでパターンを適用した仕様文（セミ形式）が生成される．このセミ形式記述をデシジョンテーブルへと変換することで，網羅的な論理関係の組み合わせでテストケースを生成する．

セミ形式記述は，命題論理の記号を用いて仕様が記述されているため，各命題プリミティブが取りうる真理値を求めることでテストケースが生成できる．この事実に基づき，命題プリミティブの充足可能な真理値を網羅的に求め，デシジョンテーブルを生成する [4] の手法により，パターンを適用した仕様分からデシジョンテーブルを生成する．

4.　ツールへの実装

3 章における一連の処理を，アプリケーションとして実装する．話題沸騰ポット第 7 版 [5] から 3 文を抜粋し，テストケースの生成までの具体的な実行例を示す．これに加えて，テスト設計者がセミ形式記述を理解できるよう，論

表 4　セミ形式記述の論理的結合
Table 4　Logical combination of semi-formal descriptions

	A1	A1 & B	A1 |B	A1 –> B	B –> A1
A2	A3	A3 & B	A3 |B	A3 –> B	B –> A3
A2 & C	-	A3 & B & C	A3 & C |B	A3 & C –> B	B –> A3 & C
A2 |C	-	-	A3 |C |B	A3 |C –> B	B –> A3 |C
A2 –> C	-	-	-	A3 –> B |C	B –> A3 –> C
C –> A2	-	-	-	-	B |C –> A3

```
# 蓋センサが3sec以上onとなったら、蓋が閉じられたと判断する。
# ============================================
成る(蓋センサ, ??, 3sec以上onと) -> 閉じる(蓋, ??) & 判断する(??, ??).

# 蓋が閉じられ、水量が適正な場合、沸騰行為をする。
# ============================================
閉じる(蓋, ??) & is(??, _, 適正) -> する(水量, 沸騰行為).

# 蓋が閉じられても、水量が異常な場合、状態はアイドルのままである。
# ============================================
閉じる(蓋, ??) -> is(水量, _, 異常) -> is(状態, _, アイドルのまま).
```

図 5　自然言語仕様の形式化
Fig. 5　Semiformalize

理的結合によって出力されたセミ形式記述の論理構造を可視化する処理も行う．抜粋した自然言語仕様文は以下のとおりである．

- 蓋センサが 3sec 以上 on となったら，蓋が閉じられたと判断する．
- 蓋が閉じられ，水量が適正な場合，沸騰行為をする．
- 蓋が閉じられても，水量が異常な場合，状態はアイドルのままである．

4.1　自然言語仕様の形式化

図 5 は，[4] にて開発された変換ツールによって 3 文をそれぞれセミ形式化したものである．“#” で始まる行はコメントとして扱われる．

4.2　パターン記述

表 1 に従い作成したパターン記述を表 1 に示す．本来パターンの記述はそのシステムドメインの知識を有しているテスト設計者が自身の知識から抽出するものであるが，使用している仕様書が話題沸騰ポットであるため，架空のパターンとして記述した．実際の仕様書，パターンを用いた実験の前に，2 つのセミ形式記述を統合し，論理的結合，デシジョンテーブルの生成が可能かどうかを確認することが重要であると考えられるため，架空のパターンを用いることは問題ないと考えられる．以降はこのパターンを適用するものとして処理を行う．

4.3　パターン統合

表 5 の「解決（セミ形式記述）」の内容と各仕様文（セミ形式）を統合する様子を図 6 に示す．この画面は表形式になっており，表体にはユーザ入力が可能である．入力されているセルには「水量」と書かれており，これにより，パターンのセミ形式記述の単語が「ポット」から「水量」へと変換され，関係語，主体，対象が同一である命題プリミティブが仕様文とパターンの両方に含まれることとなり，論理的結合へと進めることが可能になる．これらの単語が変換される様子は，図 7，8 のようにプレビュー画面を用意することで，逐一確認しながら表への入力作業を行うことが可能である．

表 5　パターン記述
Table 5　Pattern description

項目	内容
名前	異常を検出したときの伝達
状況	ポットが異常を検出する
問題	ポットが異常な用法で使用される
解決	ポットが異常事態を検出とき，ブザーを 1sec ごとに鳴らし，ユーザに異常を知らせる
解決（セミ形式記述）	is（ポット, _, 異常） –> 鳴らす（ポット, ブザー, 1sec ごとに）.
結果	ランプ表示によって，異常が起きていることをユーザに伝えることができる
フォース	想定外の使用が行われたときにユーザに伝える必要がある

4.4　セミ形式記述の論理的結合

図 9 は，統合後のセミ形式記述を結合したあとのセミ形式記述をプレビューできる画面である．ここでは，単語の変換が行われたセミ形式記述について統合後を示している．

4.5　デシジョンテーブルの自動生成

論理的結合によって 1 つに結合されたセミ形式記述をデシジョンテーブルに変換する．[4] で開発されたツールに入力することで図 10 のような結果を得られる．

4.6　セミ形式記述の論理構造の可視化

論理的結合が行われた後のセミ形式記述の論理構造を [4] に従い命題ネットワークとして可視化する．図 11 はその出力である．

5.　まとめ

本研究では，システム仕様書から抽出された試験ケース

図 6　パターン統合
Fig. 6　Pattern integration

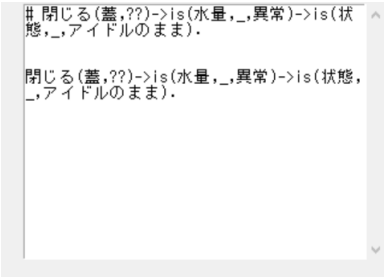

図 7　パターン統合のプレビュー（仕様書）
Fig. 7　Preview of pattern integration

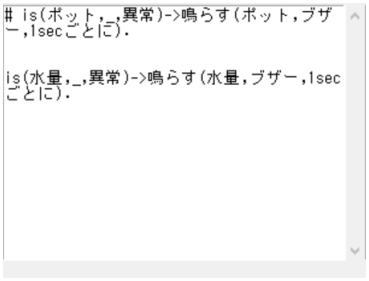

図 8　パターン統合のプレビュー（パターン）
Fig. 8　Preview of pattern integration

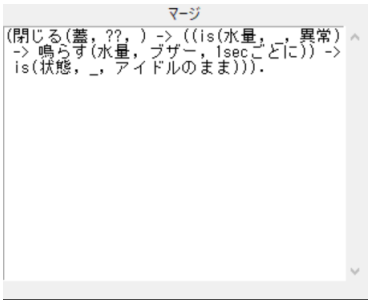

図 9　セミ形式記述の論理的結合
Fig. 9　Logical combination of semi-formal descriptions

（形式記述）と品質試験ケース（形式記述）を論理的に結合し，試験ケースを自動生成するアルゴリズムを開発した．

今後の予定として，ツールへの実装を行い，実際のシステム開発現場の中で行われる試験に本ツールを適用し，設計プロセスの明示化と作業効率化，および品質試験のパターン化によるテスト設計のノウハウの蓄積と再利用に貢献できることについての評価を行う予定である．また，評価結果をもとに，ツールの使いやすさやわかりやすさの向上など，システム開発現場での有用性を高めるためのツールの精緻化を行っていく．

参考文献

[1] JIS X 25010:2013 システム及びソフトウェア製品の品質要求及び評価（SQuaRE）－システム及びソフトウェア品質モデル, http://kikakurui.com/x25/X25010-2013-01.html
[2] 久代紀之, テスト設計ノウハウのパターン記述による蓄積とその活用方法, 2018.
[3] J.A.Whittaker：“How to Break Software: A Practical Guide to Testing”, Addison-Wesley, 2007
[4] 青山裕介，黒岩丈瑠，久代紀之：テストケース生成のためのシステム仕様書の論理記述変換アルゴリズム, 情報処理学会論文誌， Vol. 61, No. 3, pp. 521–534 (2020).
[5] 話題沸騰ポット要求仕様書（GOMA-1015 型）第 7 版， http://www.sessame.jp/workinggroup/WorkingGroup2/POT_Specification_v7.PDF
[6] Tolmachev, A., Kawahara, D. and Kurohashi, S.: Juman++: A Morphological Analysis Toolkit for Scriptio Continua, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 54–59 (2018).
[7] Kawahara, D. and Kurohashi, S.: A fully-lexicalized probabilistic model for Japanese syntactic and case structure analysis, *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Association for Computational Linguistics, pp. 176–183 (2006).
[8] 河原大輔：KNP で付与される feature 一覧, http://nlp.ist.i.kyoto-u.ac.jp/index.php?KNP (2013). Last accessed: May 31, 2019.

		antecedent=T 0	antecedent=T 1	antecedent=T 2	antecedent=F 3
condition	閉じる(蓋, ??)	T	T	T	F
action	is(水量, _, 異常)	F	T	T	-
action	鳴らす(水量, ブザー, 1secごとに)	-	F	T	-
action	is(状態, _, アイドルのまま)	T	-	T	-

図 10　デシジョンテーブルの生成
Fig. 10　Generation of decision table

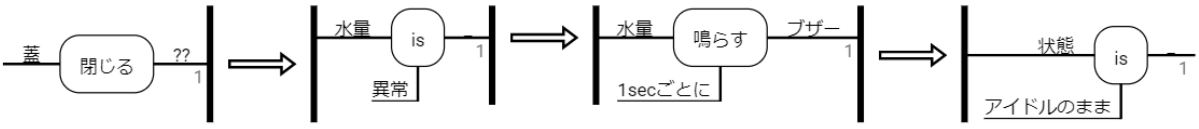

図 11　セミ形式記述の論理構造の可視化
Fig. 11　Visualization of the logical structure of semi-formal specifications