コンテキスト指向ソフトウェア設計における振る舞い検証に関する研究
―エアコンシステムを題材にして―

2020SE031 真砂樹
指導教員：張漢明

1　はじめに

近年,FIWARE[1] などのコンテキスト指向を用いたフレームワークが普及している．コンテキスト指向とは, プログラムの実行時における外部環境やシステムの内部状態などの状況をコンテキストとし, 様々な要件にまたがって存在する横断的関心事をモジュール化することで, 状況に応じてプログラムの振る舞いを変化させる手法である．

コンテキスト指向開発では, プログラミングの段階で横断的関心事を合成したときに, 意図しない振る舞いを検出する場合がある．設計の誤りをプログラミングの段階で特定することは困難であり, 一般的に手戻りの発生による修正のコストは高い．設計段階において振る舞い検証を行うことは, コンテキスト指向開発において重要である．

本研究の目的は, コンテキスト指向ソフトウェア設計における振る舞い検証を支援することである．振る舞いとは操作の順序関係を定義したものであり, 振る舞い検証では, 設計が要求仕様を満たしていることを検証する．上記の目的を達成するための技術課題は以下の 2 点である．

1. 振る舞い記述の形式化
2. 形式仕様記述の効率化

本研究の基本的なアイデアは, コンテキスト指向ソフトウェア設計における振る舞い検証に形式仕様言語を導入し, 形式仕様と UML の間の関係を分析して相互利用する方法を提示することである．形式仕様言語は, 厳密な文法や意味論を持つ言語を用いて, システムの仕様を記述することができる．本研究では, 形式仕様として VDM++, プロセス代数 CSP を,UML としてクラス図, シーケンス図を用いる．本稿では, コンテキスト指向ソフトウェアの具体例としてエアコンシステムを取り上げ, 振る舞い検証を行う．そこから得た知見を基に VDM++ および CSP と UML の間の関係を分析し, 検証を支援するための技術を考察する．

2　関連技術と関連研究

2.1　関連技術

コンテキスト指向ソフトウェアの概要

コンテキスト指向とは, プログラムの実行時における外部環境やシステムの内部状態などの状況をコンテキストとし, 様々な要件に跨って存在する横断的関心事をモジュール化することで, 状況に応じてプログラムの振る舞いを変化させる手法である．コンテキスト指向プログラミング (Context-Oriented Programming: COP)[2] では, コンテキストに依存した振る舞いをモジュール化するために層（layer）やコンテキストに応じてそれらを切り替える層活性（layer activation）といった言語要素を持つ．

VDM++ の概要

VDM++[3] は, オブジェクト指向の概念を取り入れた形式仕様記述言語であり, 設計段階におけるシステムの構造や機能の抽象的な記述に適している．

CSP の概要

CSP(Communicating Sequential Processes)[4] は, 並行システムのモデルを形式的に記述し, 検証するための理論である．本研究では, コンテキスト指向ソフトウェアを並行動作しているシステムと捉えて振る舞いを記述する．

2.2　関連研究

鵜林らの研究 [5] では, 電気ポットを例題にコンテキストを考慮したプロダクトライン開発手法を提案し, 形式手法の一つである VDM++ を用いて仕様を記述することで, 設計段階における妥当性確認の有用性を示した．

3　具体例: エアコンシステム

エアコンシステムとは, 室内温度によって冷房と暖房の運転を切り替えるシステムである．室内温度をコンテキストとし, コンテキストによって運転を切り替えることでコンテキスト指向に基づいて設計する．

3.1　UML 図

エアコンシステムのクラス図とシーケンス図の例を**図 1**, **図 2** に示す．

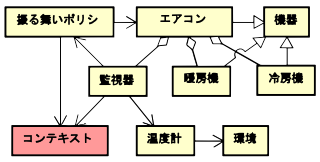

図 1　エアコンシステムのクラス図

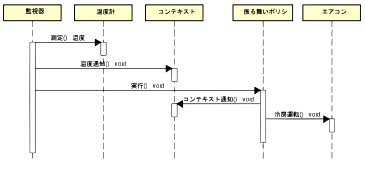

図 2　エアコンシステムのシーケンス図

図 1 のクラス図では, エアコンシステムの静的な構造を表している. 本記述では, 監視器が周期的に温度計の測定や振る舞いポリシを実行する. **図 2** のシーケンス図では, エアコンシステムの動的な振る舞いの一部を表している.

3.2 VDM++ による記述

テスト仕様の VDM++ 記述例を以下に示す.

```
operations
テスト 1 : () ==> seq of ログ ‘振る舞い
テスト 1() == (
    初期設定 (); 環境. 室内温度設定 (28);
    エアコン. 起動 (); 環境. 室内温度設定 (28);
    監視器. 実行 (); 環境. 室内温度設定 (19);
    監視器. 実行 (); ログ. 表示 ();
);
```

テスト仕様では外部環境と外部からの操作の系列をテストケースとし, システムの振る舞いを記録したログを確認することで意図した通りに振る舞うか確認することができた.

3.3 CSP を用いた振る舞い検証

振る舞い検証では, 仕様と実現が同じ振る舞いであることを検証する. 仕様とは着目した操作の順番を規定したものであり, 実現は仕様の操作以外を隠蔽した記述である.

CSP では 2 つのプロセスが等しいことを詳細化関係で表現する. 詳細化関係は 2 つの集合の包含関係からなり, 互いが包含関係にあるとき等価であるといえる.

各クラスのプロセスを並行合成したものを以下に示す.

```
E_T = ENV [|{|env_temp|}|] THERMOMETER
E_T_C = E_T [|{||}|] CONTEXT
E_T_C_P = E_T_C [|{|cnt_level|}|] POLICY
E_T_C_P_A = E_T_C_P [|{|air|}|] AIRCONDITIONER
E_T_C_P_A_M = E_T_C_P_A
[{|env_temp,notify_t,policy_go,monitor.stop|}|] MONITOR
SYSTEM = E_T_C_P_A_M
```

プロセス SYSTEM は環境, 温度計, コンテキスト, 振る舞いポリシ, エアコン, 監視器の逐次プロセスをそれぞれ並行合成したプロセスである. SYSTEM では, 逐次プロセスを 1 つずつ合成していき, システム全体の振る舞いを記述している.

仕様の CSP 記述を以下に示す.

```
SPEC = [] t:TEMP @ env_temp!t ->
        if high_temp(t)
        then cooler.on -> SPEC_00(t)
        else heater.on -> SPEC_00(t)
SPEC_00(pm) = env_temp?t ->
            if low_temp(pm) and high_temp(t)
            then heater.off -> cooler.on -> SPEC_00(t)
            else if high_temp(pm) and low_temp(t)
            then cooler.off -> heater.on -> SPEC_00(t)
            else SPEC_00(t)
high_temp(t) = t >= BOUNDARY
low_temp(t) = not high_temp(t)
```

SPEC では, 室内温度が基準温度をまたいだ時に冷房と暖房の運転が切り替わることを記述している.

FDR の assert を用いた検証を以下に示す.

```
S = SYSTEM \ diff(Events,{|env_temp,cooler,heater|})
assert S [T= SPEC; SPEC [T= S
```

プロセス S は SYSTEM の env_temp,cooler,heater に着目したプロセスである. 検証の結果,TURE が返され, 着目した事象に対して, 実現 S と仕様 SPEC の振る舞いが同じであることが確認できた.

4 考察

4.1 VDM++ と CSP

本具体例において,VDM++ と CSP の対応関係を分析し,VDM++ の記述と CSP 間の対応付けが可能であると考えられる. 具体的には,CSP の型定義（datatype）や channel によるイベントの定義, 各プロセスのイベントの順序は VDM++ と対応付けできる. 各プロセスの並行合成に関しても VDM++ のメソッドの実行関係から同期するイベントを得ることができる. また,CSP の記述は VDM++ によるテストケースのすべての振る舞いを規定している. CSP を用いて要求仕様を記述することで VDM++ のテストケースを効率よく生成することが期待できる.

4.2 CSP 記述の支援技術

VDM++ と対応付けできない部分を分析して支援技術を考察する. VDM++ から得ることのできない部分の一つとして, 周期的に実行される操作が終わるタイミングが挙げられる. ここを支援する技術として, シーケンス図を用いて動的な振る舞いを記述することが考えられる. さらに,CSP の振る舞いは状態マシン図で表現できる可能性があり,CSP 記述なしで振る舞い検証を行うことが期待できる. また, エアコンシステムに機能を追加し, 複雑化した場合についても VDM++ との対応付けは可能であると見込まれ, 振る舞い検証を行うことができると考えられる.

5 おわりに

本研究では, コンテキスト指向ソフトウェア設計における振る舞い検証の支援を目的とし, エアコンシステムの具体例を用いて検証を行い, そこから得た知見を基に形式仕様と UML の対応関係の分析, 検証を支援するための技術を考察した. 今後の課題として, 様々な事例に適用させ, 支援技術の有用性を議論することが挙げられる.

参考文献

[1] FIWARE, https://www.fiware.org/.

[2] 紙名哲生： 文脈指向プログラミングの要素技術と展望 , コンピュータソフトウェア , Vol.31 , No.1 , 2014.

[3] 石川冬樹： VDM++ による形式仕様記述 , 近代科学社 , 2011.

[4] C. A. R. Hoare : Communicating Sequential Processes , Prentice-Hall , 1985.

[5] 鵜林尚靖, 金川太俊, 瀬戸敏喜, 中島震, 平山雅之：コンテキストベース・プロダクトライン開発と VDM++ の適用 , 情報処理学会論文誌 , Vol.48 , No.8 , 2007.