B-007

自動コード生成を目的としたテンプレートベースによる UML 上位設計の作成と整合性検査

Template-based UML Upper Level Design Creation and Consistency Check for Automatic Code Generation

畑瀬尚之　　和崎克己
Takayuki Hatase　　Katsumi Wasaki

信州大学総合理工学研究科
Graduate School of Science and Technology, Shinshu University
信州大学工学部
Faculty of Engineering, Shinshu University

1　はじめに

ソフトウェア開発の速い段階でのプロトタイピングによる要求確認は欠陥発覚の遅れに対して有用な手段である．本研究の最終的な目的は上位設計群を利用したプロトタイピングの自動コード生成であり，その準備研究として VDMJWeb サービス [1] が挙げられる．これは上位設計群を用いることで，ユーザによる早期の妥当性確認を可能にするものであるが，用いる上位設計群の間での整合性確認は行われていない．用いる上位設計で整合性が取れていなければ，プロトタイピングに不整合の混入や，コード生成が困難となる事が考えられるため，上位設計群に対し整合性検査を行うことが必須である．検査対象は，プロトタイピングの自動コード生成に必要となる，テンプレートベースで記述された UML クラス図，UML アクティビティ図とする．従来研究 [2] では，クラス図とアクティビティ図の要素の一致検査を行った．本研究では，テンプレートをベースとするアクティビティ図に対してのロジック層の構造検査を行ったものである．

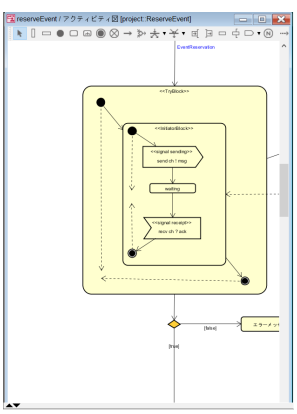
2　テンプレートを用いた UML 記述

2.1　テンプレートによる UML 記述支援の必要性

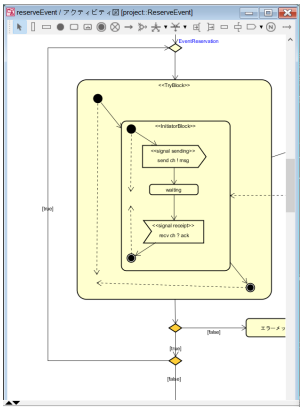
提案手法による整合性検査は，検査を行う上での曖昧さの排除といった目的から，複数の UML 図記述規則をユーザに強いている．これにより，UML 図記述のコストの増大や不整合混入といった問題が考えられるため，テンプレートベースでの UML 図記述を行い，問題の緩和を図る．UML 図記述支援としては，テンプレート構造の挿入支援やデザインパターンの作成を行う．

2.2　テンプレート構造の挿入支援

システムの実行系列を記述する際，頻出構造と考えられる制御構文・例外処理をテンプレート構造としてアクティビティ図上に作成し，それらをアクティビティ図内の任意のフローへの挿入を支援する機能を astah*professional にプラグインとして実装した．挿入時には，ノードが重ならないよう，元のノードの位置を自動調整し挿入を行う．挿入するテンプレート構造はワークフロー型となっており，挿入部であるフローの選択も必ず 1 つのみとしているため，テンプレート構造の挿入によって元のアクティビティ図のフローとしての整合性が崩れることはないと考える．



(A-0) ベースとなるアクティビティ図



(A-1) 仕様変更に合わせたアクティビティ図

図 1　仕様変更に伴うアクティビティ図の変更例

2.3　デザインパターンを用いた UML 図記述

対象ドメインを定めたデザインパターンを作成することにより，ユーザがこれをベースとし，仕様に合わせた要素の追加や変更を行うことで，UML 図記述のコスト削減，不整合混入の抑制となると考える．また，デザインパターンが記述された UML プロジェクト内に，サポートする制御構文・例外処理のテンプレート構造の記述や，ログインシステムといった汎用性があり広く用いられるシステムモデルをライブラリとして提供する．

3　テンプレートを用いた仕様の追加・変更への追従

制御構文・例外処理のテンプレート構造や挿入支援機能の使用性の評価を行うため，各種テンプレート構造が盛り込まれ，簡素な仕様の定められたデザインパターンを作成する．これをベースに仕様の追加・変更を行い，それに伴う UML 図への要素の変更・追加をテンプレート挿入支援機能を用いて行う．

ベースとなる UML 図は，イベントに対しユーザが必要項目を入力し，予約を行うというものである．これを A-0 とし，その仕様の一部である「予約可能な日数は 1 日分のみとする」に対し，「複数日同時に予約可能である」という仕様の変更を行ったものを A-1 とする．仕様変更により変化のあったアクティビティ図部分を図 1(A-0) と図 1(A-1) に示す．A-1 を作成するにあたり，A-0 のクラス図・アクティビティ図に対して以下のように変更を行った．

表 1　ロジック層検査における不整合と対応するユーザフィードバック例

評価値		不整合内容	ノート内容
1.Critical	31	終了ノードまで到達せず，フローが途切れている	Flow is broken at [node]
	32	接続されているフロー数が記述規則を満たさない	Worong the number of the [node] 's Flow
	41	分岐と合流のノードが合致しない	Unable to match any node with [node]
	42	分岐，ループ構造下のまま終了ノードへ到達した	Finished before node appeared that matched [node]
2.Warning	31	開始・終了ノード間に出現しないノード	[node] does not appear until FinalNode

1. クラス図
   (a) 複数日の指定を行うため，予約日の start と end を要素として追加
2. アクティビティ図
   (a) 予約内容入力部分に予約日の start と end を追加
   (b) 入力された予約日が予約可能であるかの照合部分をループによる処理に変更

1(a) によるクラス図への要素の追加，2(a) によるアクティビティ図への要素の追加は，所定箇所に手書きでの追加となる．2(b) によるアクティビティ図へ変更は挿入支援機能を用いて行った．図 1(A-0) が入力された予約日が予約可能であるかの照合部分であり，仕様変更に対応するため，該当部分を do-while 構造によって括ったものが図 1(A-1) である．現在の挿入支援機能は 1 つのフローを選択し，その間に指定構造を挿入するものであるため，図 1 のような，既存ノードをループ構造によって括る形を作るには，テンプレート構造を挿入後，フローを接続し直す必要がある．

4　アクティビティ図に対する構造検査

アクティビティ図に記述された構造に対し，静的検査を行う．コード生成対象言語により，記述可能となる構造も変わるが，提案手法では Java 言語を対象としたコード生成を目標しており，対応する構造も Java 言語でサポートされたものとなる．記述可能とする制御構文・例外処理は，if/switch-case 文や for/while 文，break/continue 文，try-catch 文とする．検査内容としては，(1) 開始ノードから終了ノードへと到達できない処理手順や到達不可能なノードといった，アクティビティ図としてフローの接続が正常であるかの検査と，(2) サポートする構造に対して設けた記述規則の逸脱をしていないかといった，コード生成を行う上で満たすべきとした構造となっているかの検査を行う．表 1 にアクティビティ図に対するロジック層検査における不整合例を示す．表 1 中の評価値 1.32 や 1.41，1.42 はアクティビティ図の表現としては問題ないが，与えた記述規則を逸脱する構造のため不整合である．

各構造に対する記述規則については，サポートする制御構文・例外処理がそれぞれ，その構造を表現する上で満たすべき規則として設ける．コード生成・解析を行う場合の制限であり，主に接続フローや用いるノード間の関係に対して設けている．各記述規則を満たし，ノードやフロー内にラベリングが施されたテンプレート構造を用意する．挿入支援を用いてこれらが使用されることで，検査時に，全探査を行わずとも構造解析，不整合検知を行うことが可能である．

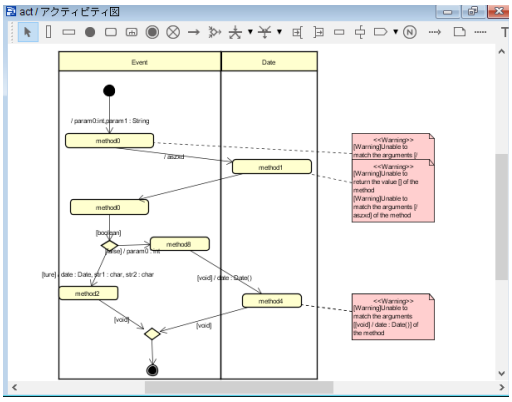


図 2　検査結果のユーザフィードバック UI の例

5　astah*プラグインとしての実装試行

提案手法による整合性検査と，そのフィードバックを astah*professional[4] のプラグインとして実装を行った．フィードバックとして，表 1 に示す不整合内容に対応したエラー文を不整合要素と紐づけた形で図上にノートとして生成し，フィードバックを行う．図 2 に，検査結果のユーザフィードバック UI の例を示す．このとき，示された不整合内容に対し修正が行われ，不整合が解消されていれば，次の検査時にはノートが削除される．

6　まとめと今後の課題

自動コード生成を目的とした上で利用される上位設計群の整合性検査の提案と検査部の一部実装と，不整合混入抑制のための，テンプレート構造挿入支援機能の実装と使用性テストを行った．今後の課題として，生成対象言語別にサポートされる制御構文・例外処理のテンプレート構造の作成や，用途によるバリエーションを持たせた各種テンプレート構造の表現を行うことで，生成対象の拡大を図る．また，複数フローの選択による挿入形の変更といった，挿入支援機能の拡張を行い，より不整合混入抑制の働きとなるものを目指す．

参考文献

[1] 村林, 多田, 和崎：VDMJ と Apache Axis2 を用いた上流工程におけるモデル実行環境の構築, FIT2014 講演論文集, (B-108), pp.155–158, 2014

[2] 畑瀬, 和崎： 自動コード生成を目的としたテンプレートベースでの UML 上位設計に対する整合性検査, 情報処理学会 全国大会 講演論文集, (5K-06), pp.195–196, 2020

[3] G.Pintér, I.Majzik： Modeling and Analysis of Exception Handling by Using UML Statecharts, LNCS 3409, pp.58–67, 2004

[4] 株式会社チェンジビジョン：astah*professional. http://astah.change-vision.com/ja/product