3L-7

実現性の高い振舞いモデルにおける データの依存関係分析によるクラス設計

岡田 康治† 松浦 佐江子‡

芝浦工業大学大学院 理工学研究科†‡

1. はじめに

組込みソフトウェアはハードウェアからの入力を基に処理を行い，ハードウェアに結果を出力する．入力値や出力の応答に誤差があるため，システムの要求を実現するには実装して動作させた結果を設計にフィードバックする必要がある．しかし，近年導入が進む UML クラス図やシーケンス図を用いて段階的にソフトウェアを分析・設計してから実装する開発手法では，フィードバック箇所が多い為，モデルと実装が乖離しやすい．

我々はこれまで，フィードバックするモデルが減れば整合性を取りやすいと考え，規定の走行ロボットと走行ラインが引かれたコースの基で走行技術を競う ET ロボコン大会[1]を題材に，システムの要求から想定される振舞いとデータをアクティビティ図でトップダウンにモデル化し，モデルから生成したソースコードにより振舞いの問題点を特定・修正し，モデルへとフィードバックすることで実装と整合性がとれた実現性の高いモデルを作成する振舞い・データモデリングを提案してきた[2].

しかし，振舞い・データモデリングの設計はデータが構造化されていない，共通部分が共通化されていないといった，保守上の問題点が存在する．開発したソフトウェアの保守性を向上するリファクタリング手法が知られている．そこで，本稿では，リファクタリングを適用するための導入として振舞い・データモデリングで定義したデータの依存関係を基に，クラスを設計する方法を提案する．また，ET ロボコン大会用ソフトウェア開発への適用結果から作成したクラスに対して効果的なリファクタリングが可能か考察する．

2. 振舞い・データモデル

振舞いデータモデリングの成果物は振舞い・データモデル(図 1)とそのモデルを実装したソースコードである．振舞い・データモデルはシステムの振舞いをユースケースの順序，ユースケース記述，具体的な制御ロジックに分け，それぞれナビゲーション・シナリオ・ロジックと呼びアクティビティ図で記述し，振舞い呼び出しアクションで構造化したモデルである．ロジックモデルには各ハードウェアが責任を持つ振舞いとデータを記述したハードウェアパーティションを記述している．ロジックモデルの作成時は，可能であればモデルカタログ[3]を利用する．モデルカタログはよく用いる制御のモデルが定義されたもので，その中に定義されたクラス図(図 2)から，クラス，属性，メソッド，引数の名前を用語として，ロジックモデルの振舞いやデータに対応付けることで，モデルの妥当性を向上している．データはその役割から修正方針が決まるため，表 1 のようにデータを分類し，そ

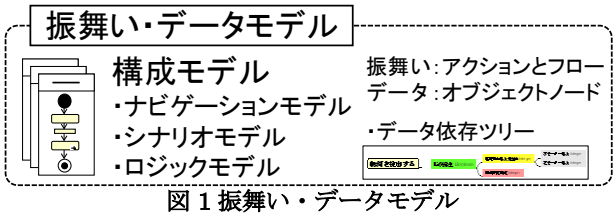

図 1 振舞い・データモデル

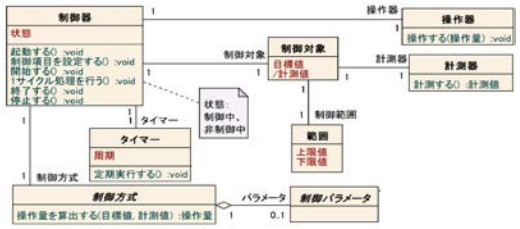

図 2 目標制御のモデルカタログのクラス図

表 1 データの分類と設計情報

分類	役割	設計情報
生成データ	ソフトウェア内で生成・保持・更新され状態を表す	依存するデータ名
計算データ	計算によって値が決定される	依存するデータ名
定数データ	運用環境の計測に基づくドメイン固有の定数	計測対象の名前
実験データ	実験によって値を調整する定数	変更時の変更理由
HWデータ	ハードウェアからの入力値	利用するAPI名
参照データ	他アクティビティ図で定義されるデータへの参照	参照先モデル・データ名

の特徴を補足する設計情報を付記する．この設計情報に書いたデータの依存や参照関係から，データの入力から出力への変換系列を表すデータ依存ツリーを生成した．

また，モデル要素から C++のスケルトンコードへの変換規則を定義して，モデルに対応したコードの実装と，コードの改善結果のフィードバックを支援し，両者の整合性を取りやすくしている．

3. データ依存関係分析によるクラス設計

クラス設計は，データの依存関係を利用して，データをクラスやその属性や操作として定義する．クラス設計はシナリオとハードウェアのクラス定義，クラスの定義，クラスの操作と属性の定義，クラス関係の定義の順に行う．図 3 は設計したクラス図の例である．このクラスは「倒立ライントレースする」ロジックモデルをクラス化したものである．本事例での倒立ライントレースは左右二輪の走行ロボットに対して，倒立振子制御をしつつ，光センサーから読み取った輝度からラインのエッジに沿って前進するよう左右のモーターを制御するものである．このラインのエッジに沿うように制御する部分に目標制御のモデルカタログを利用した．また，このモデルが参照するデータを生成する「エッジ輝度を決定する」ロジックモデルと，このロジックを利用する「ベーシックステージを走行する」シナリオモデルのクラスも含む．

3.1. シナリオとハードウェアのクラス定義

シナリオとハードウェアはロジックから明確に分離できるため別パッケージに定義する．シナリオパッケージはナビゲーションモデルとシナリオモデルに対応する．シナリオクラスは自分自身に関連を持つことでシナリオモデルの順序を定義するナビゲーションモデルを実現する．

Designing Classes by the Data-dependence Analysis in the Realizable Behavioral Model
†Koji OKADA ‡Saeko MATSUURA
†‡Graduate School of Engineering and Science, Shibaura Institute of Technology

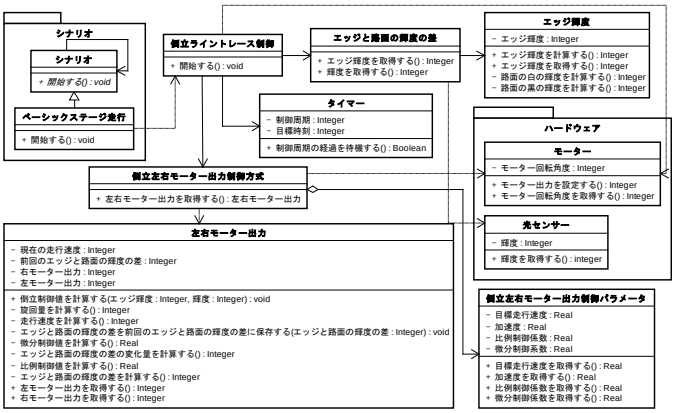

図 3 作成したクラス図の例

「ベーシックステージを走行する」シナリオモデルを実現するクラスを，シナリオクラスを継承した「ベーシックステージ走行」クラスとして定義した．

ハードウェアパッケージはロジックモデルのハードウェアパーティションに対応する．「倒立ライントレースする」モデルには「光センサー」というハードウェアパーティションがあり「輝度を取得する」というアクションと「輝度」というオブジェクトノードが記述されている．パーティション名をクラス名，オブジェクトノードを属性，アクションをメソッドしてクラスを定義した．また，ロボットの左右のモーターもそれぞれ別パーティションとして定義していたが，これらは共通する語句を抜き出すとアクションとオブジェクトノードが一致するため，纏めて「モーター」クラスとして定義した．

3.2. クラスの決定

モデルカタログに定義されたクラス構造や，データの参照関係からクラスを定義する．

「倒立ライントレースする」モデルでは目標制御を利用しているので，そのクラス構造(図 2)を踏襲して「倒立ライントレース」，「タイマー」，「エッジと路面の輝度差」，「倒立左右モーター出力制御方式」，「倒立左右モーター出力制御パラメータ」，「左右モーター」クラスを定義した．クラス名は，モデリング時に用語と対応付けたデータ名を元に適切な名称を付ける．例えば，「エッジと路面の輝度差」クラスは同名のデータを図 2 の「制御対象」と対応付けていたのでこの名前にした．一方，「制御方式」クラスは制御量を算出する操作が「倒立制御値を計算する」アクションに対応し，その戻り値を「左モーター出力」「右モーター出力」から「左右モーター出力」クラスとしたため，「倒立左右モーター出力制御方式」というクラス名を付けた．

「エッジ輝度を決定する」モデルは規範となるクラス構造が存在しないため，「倒立ライントレースする」モデルから参照されるエッジ輝度をクラスとして定義した．

3.3. クラスの属性と操作の定義

データの依存関係を用いて全てのデータをクラスへ割り当て，属性や操作を定義する．クラス名の決定時に参考にしたデータ名を起点に，依存するデータを同じクラスに割り当てる．例えば「左右モーター」クラスは左モーター出力，右モーター出力という計算データが起点となる．計算データは処理の結果を表すため，このデータを計算する「倒立制御値を計算する」アクションを操作に定義した．データ依存ツリーのこれらのデータの下には「前回のエッジと路面の輝度の差」という生成データがある．生成データは状態を保持更新する必要があるため，属性に加えこのデータを更新する「エッジと路面の輝度の差を前回のエッジと路面の輝度の差に保存する」アクションを操作に定義した．データ依存ツリーの更に下には「エッジと路面の輝度の差」クラスへ割り当てた同名データが定義されている．そのデータまでの全データを「左右モーター出力」クラスへ割り当てた．

「エッジと路面の輝度の差」データは「エッジ輝度」と「輝度」というデータに依存する．「エッジ輝度」データは「エッジ輝度」クラスに，「輝度」データは「光センサー」クラスへ割り当てているため，「エッジと路面の輝度差」クラスにはこれらのデータは割り当てられないが，図 2 では「制御対象」クラスが目標値と計測値を持つため，それに合わせるために getter を定義した．

3.4. クラス関係の定義

最後にクラス間の関係を定義する．「エッジと路面輝度の差」データと「エッジ輝度」データのように依存関係のあるデータ間でクラスが分かれているものに対して，依存関係を結んだ．また，「倒立ライントレース制御器」と「エッジと路面の輝度の差」のようなモデルカタログのクラス図で関連が引かれているクラスにはそれを踏襲して関連を結んだ．

4. 考察

データの依存関係からリファクタリングの糸口が掴めるか考察する．図 3 を見ると左右モーター出力クラスが他のクラスに比べて大きく，「巨大なクラス」が生じていると判断できる．このクラスに関するデータ依存ツリーの一部を抜粋すると図 4 のように走行速度というデータが一つのデータの塊を作っていることが読み取れる．また，走行速度が依存する一部のデータは倒立左右モーター出力制御パラメータクラスにも定義されており，走行速度に関するデータが複数のクラスへ分散している状態であることが分かる．そこで，3 章で述べたクラス抽出を用いることで走行速度を 2 つのクラスから切り離した走行速度クラスが作成できる．これによって巨大なクラスとデータの分散が改善できた．

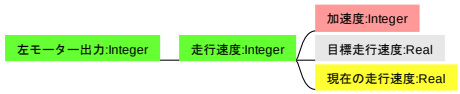

図 4 倒立ライントレース制御の依存ツリー(抜粋)

5. おわりに

本稿では，振舞い・データモデルからリファクタリングを適用するためのデータの依存関係によるクラス設計方法を提案した．また，設計したクラスに対して，データの依存関係からリファクタリング方法を検討できることを確認した．今後の課題としては，4 章で述べたようにデータ依存ツリーや振舞い・データモデルからリファクタリング方法を判断出来る事例が確認できたため，これらを分析してクラス設計方法へ組み込めるか検討する．

参考文献

[1] ET ロボコン，http://www.etrobo.jp/ (2014/01/12 時点)
[2] 岡田康治，松浦佐江子，“組込みソフトウェア開発における振る舞いとデータに基づくトップダウンなモデリング手法”，ソフトウェアエンジニアリングシンポジウム 2013 論文集,2013,p.1-2
[3] モデルカタログ，http://www.umtp-japan.org/modules/introduction1/index.php?id=47 (2014/01/12 時点)