3L-8

# 目標制御モデルに対するモデル検査の適用

本間洋光† 結城海斗† 鈴木智之† 力武克彰†
仙台高等専門学校†

## １．背景

形式手法の一つである形式仕様記述は長い歴史を持ち，産業界への適用が進められてきた．しかし，その適用は基幹系など欠陥が重大な障害を引き起こすシステムの開発への導入が大半であり，適用範囲は限られてきた．そこで，本研究では組込みソフトウェア開発における，モデル駆動開発の手本として UMTP JAPAN により提供されるモデルカタログ[1]から目標制御モデルを例に取り，目標制御モデルを用いた開発過程に形式仕様記述を適用した過程をまとめ，より一般的なシステムへの適用事例として報告する．

## ２． 研究概要

本研究では，２輪ロボットの走行制御ソフトウェアの開発を念頭に目標制御モデルを元にし PSM を作成し，PSM に対して形式仕様記述の一つである VDM++の適用を目指す．PSM とはプラットフォームに依存しないモデル（PIM）を元にし，開発対象の問題領域やプラットフォームの仕様を反映させたモデルを指す．

## ３．目標制御モデルとその PIM について

目標制御とは計測値が目標値となるように制御を行う仕組みである．温度調節や物体の走行速度制御などに組込まれ利用される．カタログではこの目標制御ついて，要求分析から PIM の導出までが説明されている．ここで VDM++による検証のベースとなるクラス図を図１に示す．

このクラス構造においては，制御器クラスが中心的な役割を果たす．制御器クラスが制御対象クラスを経由し，現在の計測値を取得する．実際の操作量の演算は制御方式クラスに委譲し，演算された操作量を操作器クラスに渡すことで計測値を目標値に近づける．

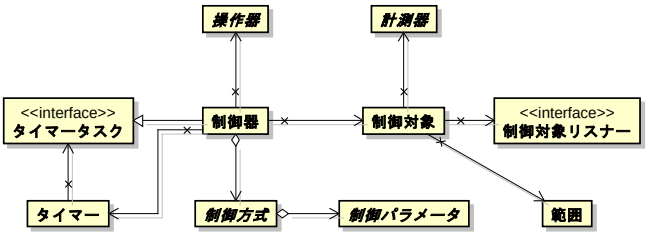


図 1 PIM のクラス図

## 4. 開発対象とその PSM の導出について

走行制御ソフトウェアを搭載する走行体の外観を図２に示す．走行体は LEGOMINDSTORMS NXT を用いて組み立てられる．中央液晶画面を備えたブロックにはマイクロプロセッサが搭載され作成するプログラムを書き込むことが可能である．また，各種センサを取り付けることが可能であり，走行制御ソフトウェアでは光センサ，エンコーダ内蔵サーボモータを２つ利用する．側面両側に取り付けられた車輪は，モータにより駆動する．

図 2 走行体外観

走行制御ソフトウェアではカタログに掲載された目標制御モデルを元に，ライントレースを実現するため，輝度値制御，並びに曲率を制御する．ライントレースでは，走行体を白地に黒線で描かれたレーンに沿って走行させる．ライントレースを実現するために，車体底部に搭載された光センサを用いて路面の明るさを計測し，黒線と白線の間における輝度値に近づく様制御を行う．また，一定の曲がり具合での走行を実現するため，曲率についても制御を行う．

---

**Applying model checking to Target Control Model**
Hiromitsu HOMMA† Yuki KAITO†
Tomoyuki SUZUKI †Yoshiaki RIKITAKE†
†Sendai National College of Technology

実際の開発では，PIM にもとづき，プラットフォームに依存した PSM を作成する．そこで，本研究では輝度値制御，曲率制御の２種類の PSM を作成した．ここでは，輝度値制御についてのみ，導出したクラス図を示す．

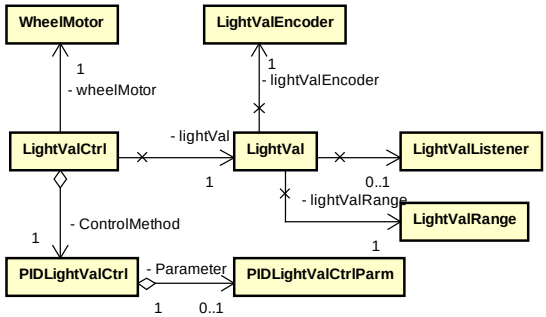


図 3　輝度値制御の PSM

図 3 では，PIM で示されたクラス間の関係は維持したまま，クラスの名称を輝度値制御に則したものに変更している．また，制御方式は PID 制御のみの使用を前提とした．また，操作器はモータとなるため WheelMotor クラスを操作器に該当するクラスとした．

## 5. VDM++による仕様の形式化

次に VDM++による仕様の形式化を行う．目標制御の実行状態を制御器クラスは保持する．その状態は制御器クラスの制御開始，終了を意味する操作の不変条件として記述される．

また，目標制御を実際に実行する操作 doCtrl()をスレッドとして定義しこの中で目標制御の演算及びモータの回転が実現される．その際 PIM で与えられたシーケンス図によって，操作器へのアクセスは排他制御であると記載されているため，mutex によって排他制御をモデル化した．以上の３点について VDM++によって記述すると図４が得られる．

## 6. VDM++によるモデル化による課題の発見

本システムにおいては，WheelMotor クラスのオブジェクトは共有オブジェクトであり，輝度値制御と曲率制御の双方からアクセスされ得る．しかし，両クラスにおいて独立して演算した結果が WheelMotor クラス渡される設計になっている．これは，PIM 作成時点で，複数スレッドへの対応の考慮漏れであると考えられる．そのため，PIM まで立ち戻りクラス構造の変更を行った．その結果が図 5 である．

```
  stopCtrl()
      ext wr state:CtrlState
      pre state = <On>
      post state~ = <Off>;
    startCtrl()
      ext wr state:CtrlState
      pre state = <Off>
      post    state~    =    <On>;
thread
  doLightValCtrl();
  sync
    mutex(setTurn);
```

図 4　VDM++による記述

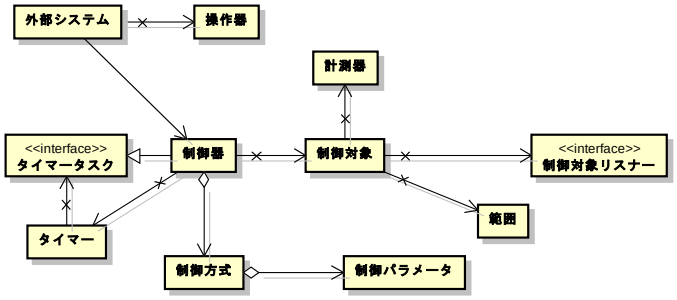


図 5　修正後の PIM

図３から図５への変更点は，操作器へのアクセスを一元化するため，操作器クラスを外部システムクラスからアクセスするように変更した点である．外部システムクラスは PSM においては，輝度値制御と曲率制御それぞれで演算された操作量をどの程度採用するかという情報を保持しており，それに基づき計算した，最終的な操作量を操作器クラスを渡す役割を担う

## 7. まとめ

本研究では，基幹系ではなく，より一般的なシステムへの VDM++の適用の事例の報告を目標に，２輪ロボット走行ソフトウェアの PSM に対し仕様の形式化を進めた．その結果，自然言語や UML で与えられた仕様の VDM++によるモデル化の例を示すことができた．また，PIM 時点で考慮から外れていた複数スレッドへの対応という要求の漏れを指摘し，その解決案を PIM モデルの修正という形で示すことができた．

【参考文献】

[1] UMTP 組込みモデリング部会：組込み分野のための UML モデルカタログ (2012.04.15)
http://www.umtpapan.org/themes/original2/pdf/built/ uilt_uml_catalog.pdf