社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
KBSE2008-31 (2008-11)

組込みソフトウェアにおける段階的モデル駆動開発実験

伊藤　邦彦† 　天川　美那† 　松浦　佐江子‡

† 芝浦工業大学大学院 〒337-8570 埼玉県さいたま市見沼区深作 307
‡ 芝浦工業大学 〒337-8570 埼玉県さいたま市見沼区深作 307
E-mail: † {m108015, m107007}@shibaura-it.ac.jp, ‡ matsuura@se.shibaura-it.jp

あらまし 2007 年および 2008 年の ET ロボコンに参加し，自律走行する車両型ライントレースロボットのモデル駆動開発を行った。2007 年はシステムをハードウェア，外部環境，制御の 3 つの観点から ET ロボコンに特化しない「自律走行車でのレース」モデル構築し，ハードウェア，外部環境，制御を ET ロボコンに特化させることで実装を行っていった．しかし，「自律走行車でのレース」モデルから実装までを行うにはハードウェア・外部環境が制御にどのように影響を及ぼすかという分析が不十分であり繋がりが不明確となった．2008 年はモデルを詳細化する際に付与するプラットフォームの情報を分析し，ハードウェアの能力・外部環境の制約を段階的に取り入れることでモデルの実現を図った．本稿では，モデル駆動開発における組込みソフトウェアの要求から実現までの段階的開発について考察し，これを報告する．

キーワード モデル駆動開発，組込みシステム，ET ロボコン

Model Driven Development Experiment of Embedded System

Kunihiko ITO† 　Mina AMAKAWA† 　and 　Saeko MATSUURA‡

† Shibaura Institute of Technology Graduate School 307 Fukasaku, Minuma-ku, Saitama, 337-8570 Japan
‡ Shibaura Institute of Technology 307 Fukasaku, Minuma-ku, Saitama, 337-8570 Japan
E-mail: † {m108015,m107007}@shibaura-it.ac.jp, ‡ matsuura@se.shibaura-it.jp

Abstract

We have participated in the Embedded Technology Robot Contest in 2007 and 2008. The aim of the contest is to design a good model of a line tracking robot that has satisfactory high speed using LEGO MINDSTORMS. Since 2007, a concept of Model Driven Architecture has been introduced in our model of the robot, so that three parts of the system consists of hardware components, the system environment, and the control software. We assumed the hardware components and the system environments are independent platform for the control software. However, a way of transforming a Platform Independent Model to a Platform Specified Model resulted in not being defined clearly, because of insufficient analysis of the platform. It means that the properties of hardware components are not sufficiently defined in the model, even though they are estimated in the actual system environments. In 2008, we improve a way of platform analysis so that properties of hardware components and the conditions of the system environments are introduced into a model stepwise. This paper describes such an experiment of Model Driven Development of an embedded system.

Keyword Model Driven Development, Embedded System, Embedded Technology Robot Contest

1. はじめに

近年ソフトウェア開発は，大規模・複雑化しただけではなく，開発期間の短縮化が求められ，ユーザニーズの多様化がすすんでいる．特に，組込みソフトウェア開発においては，ハードウェアや動作を行う外部環境の変更によりソフトウェアに大きな影響を与えるという問題がある．そこで，ハードウェア・外部環境の変化に対応できる再利用性の高いソフトウェアを構築することが求められている．組込みソフトウェアでは，ハードウェアと外部環境の役割をモデルに採り入れる

ことによりハードウェア・外部環境の変化に適用可能な再利用性の高いソフトウェアを作成することができる．仕様レベルのモデルをシミュレーション実行しテストを行うことで実装プラットフォームに依存しないモデルを構築する方法が試みられている[1]．しかし，ハードウェア・外部環境といったプラットフォームに依存しないモデル(PIM)からプラットフォームに依存したモデル(PSM)に変換することは困難である．変換が困難となる理由は，ハードウェアの性能は実環境における実験によって計測することができるが、その結果を厳密にモデルに反映することが難しく、各プラットフォームの分析が十分に行えないということが挙げられる．

本稿では，ET ソフトウェアロボットコンテスト[2]の事例を使い PIM から PSM に変換するためのプラットフォーム分析を段階的に行うことで，プラットフォームの情報がモデルに与える影響について考察する．

2 章は開発事例である ET ロボコンの概要と設計コンセプトを説明する．3 章は段階的なプラットフォームの分析方法について具体的に述べる．4 章ではプラットフォーム分析後の結果と実装について述べる．

2. ET ロボコン

ET ロボコンは規定されたコースを同一に規定された LEGO Mindstorms[3]で分析，設計したソフトウェアを搭載して自律走行を競うライントラッキングレースである．ET ロボコンでは，モデルの評価と走行性能の評価を行う．競技内容から得られる走行性能には，以下が挙げられる．「黒線で描かれたレーンをリアルタイムで検出しながら走行する」という条件より，ラインの上を走行する性能が必要である．「制限時間(2 分)以内に走行しなくてはならいない」という条件より，2 分以内に走行できる速度の設定が必要である．また，合計タイムは「走行タイム」から「ボーナスタイム」を減算するため「ボーナスタイム」についても考慮しなくてはならない．「ボーナスタイム」を得られる条件は表 1 である．表 1 の新ショートカットとは，間隔 30mm の湾曲した点線コースである．新ショートカットでは，通常のライントレース走行とは全く異なった走法を行わなくてはいけない難所である．また，ツインループについてもループ内に入るためにループの位置を把握しどのような方向に進むかを判断し，走行することが求められるため，通常の走行方法とは異なった方法を採る必要があると考えられる．

表 1　ボーナスタイムの設定

ボーナス対象	ボーナスタイム
中間ゲート通過	1回ごとに5秒
ゴールゲート通過	1回ごとに5秒
新ショートカット走破	1周目：10秒 2周目：20秒
ツインループゲート通過	1回目、2回目： 5秒 3回目、4回目：10秒
ゴール後停止(オーバーランなし)	10秒
ゴール後停止(オーバーランあり)	5秒

「ボーナスタイム」である「ゴール後停止」ではゴール地点を認識する必要がある．ゴール地点を認識する指標としてゴール直前にある坂道を識別する灰色線がある．これを認識することで場所を認識することが可能である．今回採り入れる走行性能は

1. ラインの上を走行する
2. 2 分以内に走行する
3. 灰色線を認識する

である．

また，速く走行することだけではなく，よいモデルを作成するという点から事例として ET ロボコンを採用した．よいモデルとは再利用性の高いモデルとし，ET ロボコンに特化しないモデルを作成する．

図 1 は ET ロボコンで規定された走行体である．走行体には，外部情報を取り入れる入力部分として前方にタッチセンサーとライトセンサーを持っており，出力部分として方向を決定するステアリングモーターと推進力を出力するトラクションモーターがある．コースには走行路となる黒色の走行ラインがひかれている．

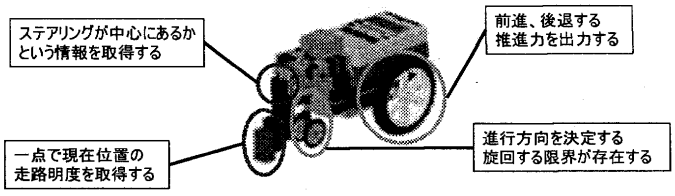

図 1　ET ロボコンで規定された走行体

2.1. モデルの作成

再利用性の高いソフトウェアを作成するため，ET ロボコンに特化しないモデルとしてハードウェアと外部環境について独立したモデルを作成する．そこで，「自律走行車でのレース」モデルを PIM として作成する．PIM はモーター，センサーの数，ステアリング機構，コースの形状について独立したモデルを作成する．「自律走行車でのレース」モデルが図 2 である．自律走行するとは地図を見て，運転方法を決定して，車体を動作させるということを行っていると考え，図 2 では「地図を見る」，「運転方法を決定する」，「車体」の

3 つの領域に分割し，それぞれの領域に変更が生じても変更箇所が特定できるように作成した．「地図を見る」領域では事前に把握している路の形状を判断し，現在の位置を特定する．「運転方法を決定する」領域では路の形状に合わせて運転方法を決定し車体に動作を伝える．「車体」領域では車体が行える動作をまとめ実際に動作する命令を伝える．このように作成した PIM から「ET ロボコンでのレース」モデルを PSM として作成していく．この手順を踏まえることで再利用性の向上を図った．

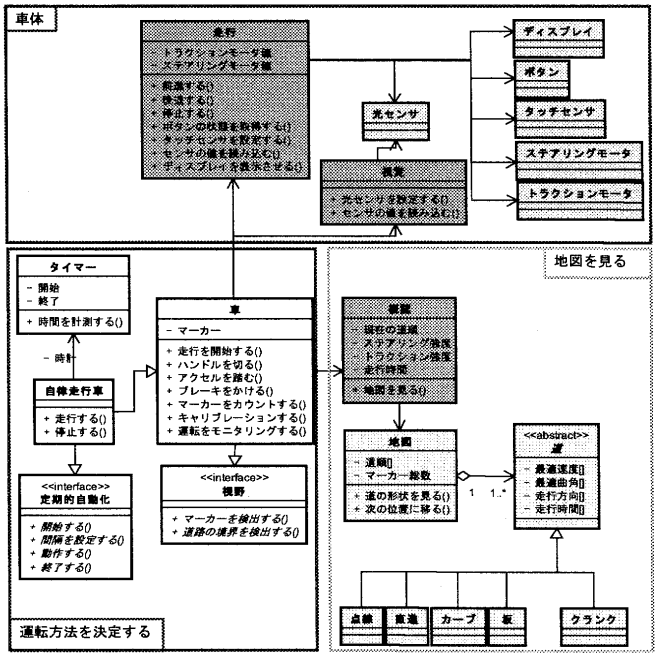

図 2　自律走行車でのレースモデル

2.2. PIM から PSM 変換の問題点と改善案

PIM から PSM に変換し，実装を行っていったが，システムの内部に走路を読む入力の情報と動作を行う出力が入り混じりどのような要素が関わり持つか分析されないまま実装を行ったために，プラットフォームの情報が散在するソースコードとなってしまった．そのため，PIM の独立性を保つことができなかった．この原因はプラットフォームを外部環境とハードウェアとしたが，二つのプラットフォームに関連があるため，プラットフォームが複雑になったためである．そのため，図 2 の運転方法を決定することが複雑となった．

PIM から PSM へ変換するためには，二つのプラットフォームの関連を十分に分析することが必要となり，どのように運転方法を決定するかということを分析しなくてはならない．そこで，各プラットフォームの複雑度を段階的に採り入れ，具体化していくことによりプラットフォームの分析を行っていく方法を採る．

3. プラットフォーム分析

プラットフォームを分析する PIM として ET ロボコンに特化しない「自律走行車でレースを行うモデル」を扱う．PIM から走行性能から段階的にプラットフォームの分析を行い PSM に変換する．段階的にプラットフォームを採り入れる順番は

1. ライントレースロボット
2. 2 分に以内に走行する 1 つの光センサーによるライントレース
3. 灰色線を認識する

である．これらを段階的に分析し，運転方法を決定するクラスに段階的に採り入れる．

3.1. ライントレースロボット

PIM からライントラッキングレースを行うモデルへ具体化を行う．ライントレースロボットとは，ラインからのずれを検出し，方向を決定，前進するロボットである．よって，ライントラッキングレースモデルにおいて取り入れる制約は表 2 である．表 2 の制約を満たしライントレースを行うステートマシン図を図 3 とする．ここでは，ステートマシン図は外部環境を状態とし，入力を遷移条件としている．出力を実行アクションとして記述する．図 3 でのライントレースは常にライン上にいることを想定しラインから外れたときのみ復帰動作を行うということを行っている．

表 2　ライントラッキングレースの制約

プラットフォーム	制約
外部環境	一定幅のラインがある
ハードウェア	路面状況を取得できるセンサーを持つ
	前進することができる
	進行方向を決定することができる

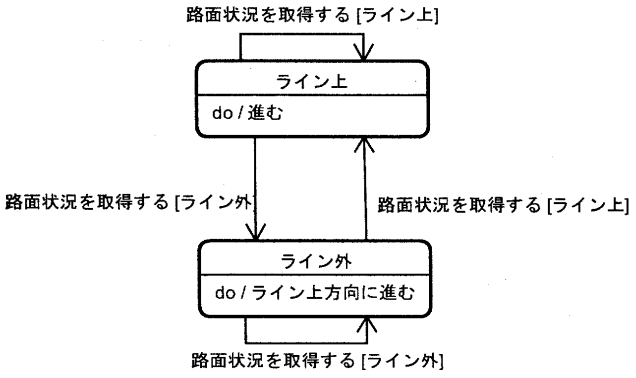

図 3　ライントレースロボットのステートマシン図

3.2. 光センサーによるライントレース

ここでは，2 分以内走行でき，直線・カーブを同様

に走行できる速度に設定する．速度はコース上に存在するカーブ半径最小のものを走行可能な速度を実験により算出した．速度によるモータパワーの表は表 3 である．

表 3　カーブ半径によるモータパワー

	カーブ半径					
	300		350		400	
	内側	外側	内側	外側	内側	外側
モータパワー	205	205	230	230	255	255

次に ET ロボコンで規定されているハードウェアの能力について具体化を行う．規定のハードウェアで路面状況を取得するために使用できるセンサーは光センサーだけであるため取り入れる制約は表 4 となる．ここで，光センサーは 1 つしか持たないため，走行方法をエッジ走行とする．エッジ走行とはライン上であればライン外方向へ進みライン外であればライン上方向へ進むという走行方法である．エッジ走行ではライン上とライン外を頻繁に行き来することになる．

また，路面状況を取得する光センサーの能力について実験を行うと人の目では，はっきりと黒色と白色が区別されているが，光センサーでは区別がされていないことがわかった(図 4)．これは，今回の使用される光センサーが直径約 1 センチの円内の明度を平均値として取得するためである．そのため，ライン上とライン外の間に約 1 センチの灰色の境界部分を持つことがわかった．光センサーの能力を考慮したステートマシン図は図 5 となる．境界という状態を新たに追加し，遷移条件を色による条件へと変更を加えた．ステートマシン図から 3 つの色を識別することができればよいことがわかる．

表 4　光センサーを考慮した制約

プラットフォーム	制約
外部環境	一定幅のラインがある
	ラインの色が黒色． それ以外を白色とする
ハードウェア	前方に明度を取得する光センサーを
	前進することができる
	進行方向を決定することができる
	進行方向操作により 光センサーの位置が変わる

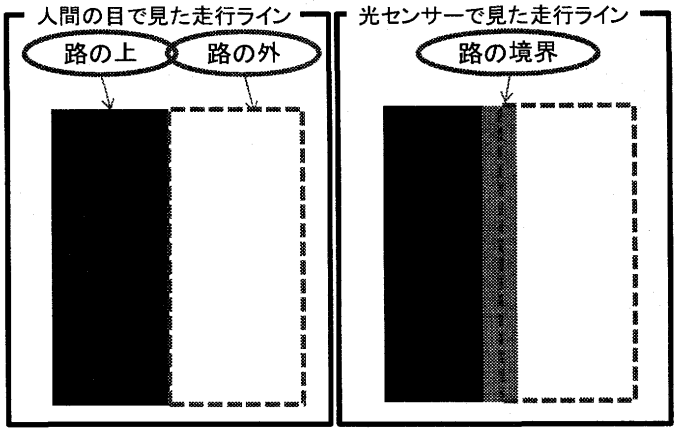

図 4　人間の目と光センサーで見た走行ラインの違い

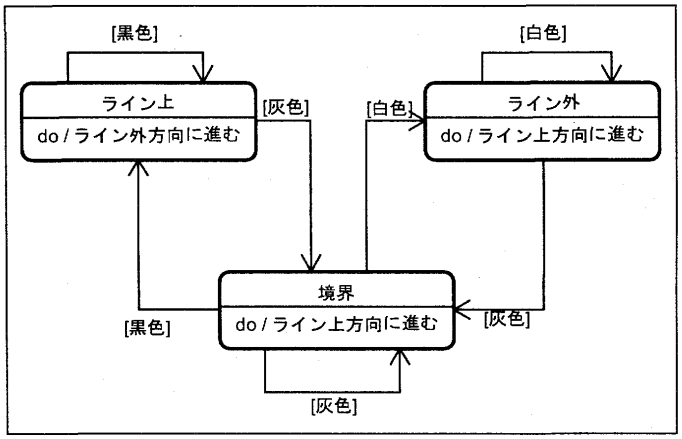

図 5　光センサーを考慮したステートマシン図

3.3. 灰色線の認識

2007 年度，2008 年度の ET ロボコンのコースでは，坂道の入口と出口に灰色線がある．この灰色線を認識することで坂道を認識できることができる．また，坂道出口はゴール付近であるため，ゴール時に止まる指標ともなる．灰色線を採り入れた時の制約は表 5 である．灰色線を採り入れたことにより一点で境界であるかライン上の灰色線であるかを判別することができなくなる．そのため，一点ではなく履歴を取ることで複数点の情報を基に判別する方法が考えられる．灰色線の時と灰色線のないラインでの明度の読み取りの違いを実環境での実験により得た明度の値が図 6 である．図 6 の前半は通常のラインであり，後半は灰色線上を走行しているときの読み取った明度の値を表している．図 6 より境界と灰色線の大きな違いは境界と灰色線の幅と長さであり，通常のラインと灰色線の違いは黒色が存在しないことが分かる．このことから灰色の読み取り回数を履歴とすることで灰色線を判別する．灰色線と通常ラインの軌跡の違いを図で表すと図 7 となる．黒色を読み取ってから次の黒色を読み取るまでの灰色

の回数を判別する指標とする．図 8 が灰色線を考慮したステートマシン図である．灰色線での出力動作はライン上と同様の動作を行う．また，灰色線以外の出力動作は図 5 と同様である．

表 5　灰色線による制約

プラットフォー	制約
外部環境	一定幅のラインがある
	ラインの色が黒色． それ以外を白色とする
	直線ライン上に灰色線がある
	灰色線と境界の明度が一致する
ハードウェア	前方に明度を取得する光センサーを
	前進することができる
	進行方向を決定することができる
	進行方向操作により 光センサーの位置が変わる

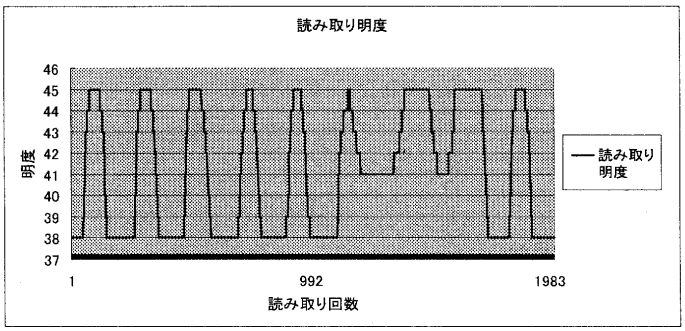

図 6　灰色線の明度の読み取り

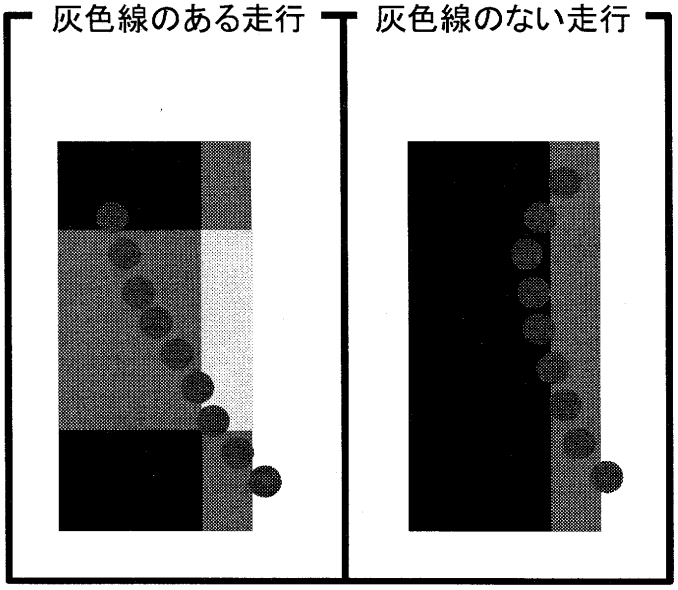

図 7　灰色線と通常ラインの軌跡の違い

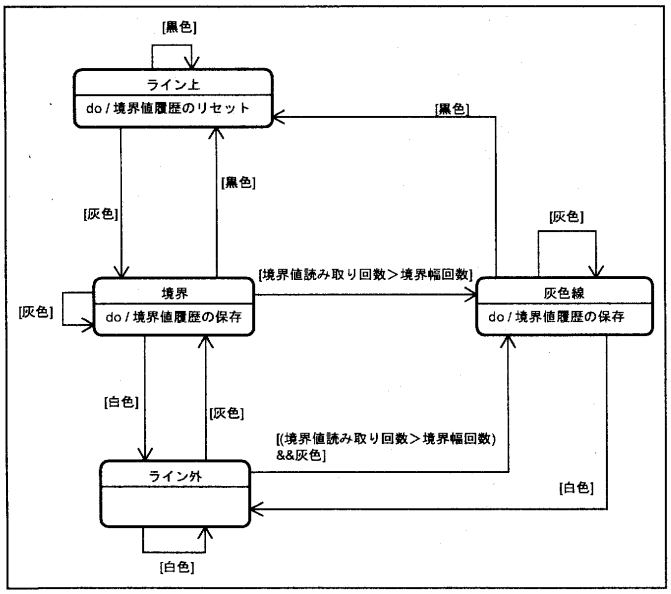

図 8　灰色線を考慮したステートマシン図

4. 実装と結果

3 章でのプラットフォーム分析を段階的に行い，プラットフォームの複雑度をモデルに採り入れ，実装を行った．プラットフォーム分析のクラス図が図 9 である．プラットフォーム分析では灰色線を読み取るまでしか行っていいないため，図 2 のような道の形状を読み取る性能を持っていない．そのため，実際のコースでは直線やカーブ・坂道などがあるが，それらを走行可能な一定速度で走行するよう設定した．図 2 では，ハードウェアの領域として光センサーを「車体」領域としていたが，実際には外部環境と係るため，外部環境と係る部分として切り分ける必要があった．

実装では，光センサーから得た明度を色と対応付けは，光センサーの個体差・外部環境の照度・走行路の材質などが影響するためその環境に適した設定をその場で行うことのできる機構を付け加えた．灰色線の認識では，境界幅回数を具体的な数値として設定しなくてはならないが，車体の進入角度により回数が異なるため，境界幅回数を厳密に定義することができなかった．そのため，カーブなどで灰色線と誤認識するということが起こった．しかしながら，コース全体を完走することができた．光センサーによるライントレースまでは十分に分析できたといえる．

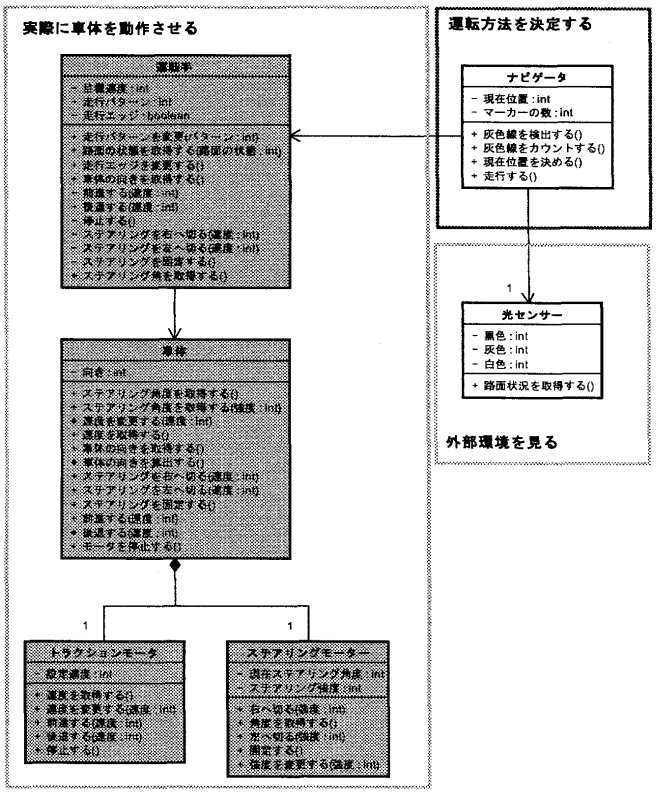

図 9　プラットフォーム分析後のクラス図

5. おわりに

ハードウェア・外部環境に依存しないモデルから実装可能なモデルまでを段階的にプラットフォームを分析することにより，どういった制約があり，どのような要素がモデルに影響を与えるかということを明確にすることができた.「自律走行車でのレース」モデルから「ET ロボコンでのレース」モデルに変換するには外部環境に係る入力部分と出力部分を切り分ける必要があることがわかった．事前に入力として与えるコース全体の構成の入力と動的に取得する入力がありそれらについても段階的に取り入れていかなくてはならい．段階的に外部環境とハードウェアの係わりを分析し，採り込んでいくことにより，必要な要素を確定させる．次の段階での分析では何の要素を明確にしなくてはならないかということがわかる．また，灰色線を認識する方法として灰色の数をカウントしてきた．この方法を採ったときカーブでの誤認識が見られた．このことから，カーブにおいても灰色の値を多く読み取るということがわかる．しかし，灰色の読み取り回数だけでは路の形状を判断するには限界があるため，次の段階ではタッチセンサーの能力について採り入れることにより，ステアリング位置が中心にあるという情報が取得可能となる．このことにより「自律走行するレース」モデルにあった道の形状を採り入れることができると予想できる．

また，プラットフォーム分析により得た制約と要素が実行可能モデルのシミュレーション時においても，このような制約と要素としても使用可能であると考える．

文　献

[1] 伊藤 恵, 久保秋 真,“組み込みソフトウェア開発のための仕様シミュレーション”, 情報処理学会研究報告, Vol.2002, No.64 (2002)
[2] ET ソフトウェアロボットコンテスト; http://www.etrobo.jp/
[3] LEGO Mindstorms; http://Mindstorms.lego.com/