モデルベースによる超小型モビリティ用組込みシステム開発

茂木 和弘*

A Model-Based Approach for Micro Mobility Systems

Kazuhiro MOTEGI*

*群馬大学大学院理工学府知能機械創製部門 (〒373-0057 群馬県太田市本町 29-1)

*Division of Mechanical Science and Technology, Faculty of Science and Technology, Gunma University (29-1 Hon-cho, Ota, Gunma 373-0057)

1. 超小型モビリティ

世界の石油などの化石燃料の生産量のピーク（ピークオイル）は、21 世紀の中ごろと予想されていた。しかし、新規油田の開発や推定されていた埋蔵量の修正などにより生産量のピークは修正されている。しかし、工場や自動車などによる二酸化炭素排出量の増加は地球温暖化などの環境問題の原因となっているため、自動車は内燃機関から電気自動車への移行が進められている。国や企業の戦略が大きく影響しているが、中国、インド、フランス、ノルウェーのように内燃機関から電気自動車へ切り替え（予定も含む）や優遇が進んでいる¹⁾。

また、日本の都市における既存車の利用実態は、交通手段として自動車に依存する傾向にあり、その移動距離は 10 km 圏内が約 6 割を占め、自動車への乗車人数は 2 人以下が多い²⁾。このような利用に対し、乗車定員が 4 から 5 名、航続距離が 500 km 以上を標準とする既存車は、過剰性能・過剰品質となっている。このため、日常生活の足となる輸送手段として、環境性能に優れた 1 から 2 人乗り程度の超小型モビリティの開発が進められている。しかし、超小型モビリティは既存車と比較して航続距離が短いために、バッテリー切れに対する不安や充電頻度の増加と充電時間の長さに対する不満が生じ、普及を妨げる要因となっている。この課題を解決するため、回生ブレーキによる電力の回収、ハイブリッド電源の利用、さらにより高度な電源制御などを行う必要がある。これらを電子コントロールユニットで制御するために組込みシステムのモデルベース開発が行われているが、従来の開発環境ではモデルベース開発の利点を享受できていない。

2. 電気自動車の組込みシステム開発

2.1 組込みシステム開発

組込みシステムには共通する 4 つの特性があり、この特性を考慮して組込みシステム開発は行われている³⁾。

- ・専用化されたシステムである

- ・リソース制約が厳しい
- ・高い信頼性が必要である
- ・リアルタイム性が必要である

組込みシステム開発はハードウェア設計、ハードウェア実装、ソフトウェア設計、ソフトウェア実装というプロセスで開発を行うことが一般的である。すなわち、組込みシステム開発はハードウェア開発が優先される。なぜなら、ハードウェアが生産コストと性能に直結するためである。ハードウェア仕様を先に確定させた後に、ソフトウェアの柔軟性を活用してシステムの開発を行う。組込みソフトウェア開発手法にはウォーターフォールモデル、V 字モデル、プロトタイプングモデル、そしてアジャイルプログラミングモデル等がある⁴⁾。従来手法を適用した組込みシステム開発は、紙ベースの仕様書で行われるため、書き手の意図した情報を読み手に誤解なく伝達することが困難であり、開発プロセスにおける工程間やメンバ間でミス・コミュニケーションを引き起こすリスクがある。この従来手法の各工程の課題を以下で述べる。

A) 設計工程

設計工程ではプロトタイプを構築し、その出力結果と仕様書に定義された所望の出力結果を比較することで妥当性の評価を行う。しかし、開発に着手した段階では紙ベースの仕様書の内容は不完全であり、必要となる詳細な設計情報は設計中に明らかになることが多く、仕様書に逐次追加していかなければならない。

B) 実装工程

実装工程では紙ベースの仕様書に基づき、プログラミング言語やハードウェア記述言語を使用してハンドコーディングを行う。ハンドコーディングはそれなりに時間がかかり、また高いスキルを持ったメンバの確保は困難であるので、コードの品質はメンバのスキルに依存し、かつバラツキが生じる可能性がある。また、タイプミスや仕様書に記述された内容がコーディングメンバに正しく伝わらないことを原因とするエラーが混入する可能性も否定できない。また、過去に

開発されたソフトウェア資産の再利用を行うために、他人の作成したプログラムコードを参照しようとしたが解読することができず、結局再利用をあきらめてしまったという話はよくある。すなわち、従来手法によるソフトウェア資産は属人性が高く再利用性が低いということを意味する。

C) テスト工程

テスト工程では、仕様書に基づきソフトウェアのテストを行う。テストによって検出されたエラーについて、ハンドコーディングで発生したタイプミスのような単純なエラーであれば容易に修正できるが、単体テストでは検出されず統合テストのような最終段階で検出されたエラーの場合、エラーの修正は容易ではない。このようなエラーは、システム内部の機能単位（サブシステム）間の相互作用といった要因が複雑に絡み合って発生することが多く、原因を特定し修正するには紙ベースの仕様書を拠りどころとして、工程間をまたがってコミュニケーションを行う必要があるもので、莫大な費用と時間がかかる。

ここで、組込みソフトウェア産業における開発の実態について述べる。2010年3月、経済産業省は組込みソフトウェアに関連する工業会・協会などの団体に加盟する企業のプロジェクト責任者を対象に実態調査を行った⁵⁾。これによると、開発ソフトウェアの行数は全体で平均100万行程度、新規部分で平均45万行程度に上った。また、プロジェクトの投入人数は5人以上20人未満が全体の約5割を占め、職種構成はソフトウェアエンジニアが全体の約4割を占め、ソフトウェア開発費は費用全体の約6割を占めた。プログラムの作成手法はハンドコーディングが約9割を占めた。また、ソフトウェアの不具合発生時、不具合を修正するためにかかる工数はソフトウェア実装工程が最も大きかった。2013年4月26日、独立行政法人情報処理推進機構が2012年度ソフトウェア産業の実態把握を目的として組込みシステム開発企業などを対象に実施した調査に関する報告書を公開した⁶⁾。報告書では、開発ソフトウェアの中で過去のソフトウェア資産をそのまま再利用している比率は約2割にとどまっていること、組込み機器製品の不具合の約5割がソフトウェアであること、ソフトウェア不具合の約8割は下流（実装・テスト）工程で発見され、不具合の発生原因は上流工程が約6割を実装・デバッグ工程が約3割を占めていることが示されている。

これらの実態調査結果より、組込みシステム開発企業は従来手法による課題に直面しているといえる。チームを組む、紙ベースの仕様書とハンドコーディングを用いての大規模組込みソフトウェア開発では、ミス・コミュニケーションが発生し、下流工程まで不具合は発見できず、手戻りと修正の手間は大きくなり、ソフトウェア資産の再利用性は低くなるのである。このため、開発プロセスの見直し

や革新的な手法への取り組みは急務である。開発プロセスを改善する手法の1つとしてモデルベース開発がある。モデルベース開発手法による組込みシステム開発プロセスは、対象となるシステムを記述したモデルを仕様として定義し、このモデルを拠りどころとして開発プロセスを再構築する。特徴は以下の4つである。

A) モデルによる仕様の表現・定義「実行可能な仕様書」
状態遷移図やフローチャートなどの基本的なレベルでのモデルは、情報共有やドキュメント化、設計品質の向上といった点において改善効果が認められている。しかし、これは従来の紙ベースの仕様書と大差ない。モデルベース開発手法におけるモデルは、モデルベース開発ツールによって作成される出力結果の確認機能が備わったモデルである。このモデルは、開発しようとしているシステムの挙動をコンピュータ上（仮想空間）でシミュレーション（数値実験）できるので、「実行可能な仕様書」と定義される。

B) シミュレーションによる設計の詳細化、妥当性検証
実行可能な仕様書であるモデルは、紙ベースの仕様書で記述しきれなかった設計情報、すなわち、シミュレーション条件や実行結果といった情報が付加され、システム設計の段階で設計情報として共有することが可能となる。そして、シミュレーション結果から所望の出力結果も明示的になり、これを拠りどころとして設計の詳細化を進めることができる。また、モデル化されたシステム全体のシミュレーションを実行し、その結果をシステム内部の機能単位ごとに分析することで、コンポーネント間の相互作用を把握できる。また、モデルの共有を進めることにより、エラーの早期検出や、工程間にわたるエラー要因の特定の際のコミュニケーション改善においても効果が期待できる。

C) モデルからの自動コード生成による実装
使用するモデルベース開発ツールによっては、モデルから自動コード生成（オートコーディング）を行うことができる。自動コード生成は、従来手法におけるハンドコーディングによるエラーを排除し、コードの品質のバラツキを抑え、属人性が低くなり、ソフトウェアの再利用性の向上が期待できる。また、実装工程における工数の削減も期待できる。

D) テスト・検証におけるモデルの再利用
モデルベース開発ツールによるが、モデルから全テストケースの自動生成を行うことができる。この生成されたテストケースを使用して実機とモデルの両方をテストし、出力結果の比較・判定を行うことで検証時間の短縮や自動化が期待できる。また、検証作業の質・結果の定量化によってチームでの情報共有が可能となる。

これらの特徴よりモデルベース開発手法による組込みソ

ソフトウェア開発は、従来手法において困難であった「仕様の明確化」,「開発プロセス全体のコミュニケーションの改善」,「開発の上流工程における詳細な検討」,「資産の再利用性の向上」に対し有効である。既に自動車の新機能の90%が電子制御部品によって実現され、その開発費用の半数以上がソフトウェア開発であるといわれている。ソフトウェアの大規模化と複雑化でソフトウェアに潜む誤り(バグ)の存在性は飛躍的に高まり、ソフトウェアの不具合を原因とする欠陥や故障が増加している。このため、ソフトウェアに潜む誤りを事前に発見することが大きな課題となっている。しかし、自動車に実装される電子コントロールユニット数は増加の傾向にあり、これらに対するテスト項目は飛躍的に増大している。これに伴い、実機による試行テストを繰り返して品質を熟成させるという方法は費用の観点から限界がある。また、テストの網羅性の観点からも問題がある。こうした問題を是正し、開発効率や品質の向上、コスト低減を実現する手法として、モデルベース開発が自動車業界に普及している。

2.2 超小型モビリティの組込みシステム開発

超小型モビリティは「自動車よりコンパクトで小回りが利き、環境性能に優れ、地域の手軽な移動の足となる1から2人乗り程度の車両」として定義される。超小型モビリティは低炭素社会の実現に資するとともに、都市や地域の新たな交通手段、観光・地域振興、高齢者や子育て世代の移動支援など、生活・移動の質の向上をもたらす新たなカテゴリの乗り物として期待されている。超小型モビリティとは以下に示す条件を満たす車両である。

- ・長さ、幅、高さが軽自動車の規格内の三・四輪自動車
- ・乗車定員2人以下(2個の年少者用補助乗車装置を取り付けたものにあつては、3人以下)

・定格出力8kW以下(または排気量125cc以下)

超小型モビリティは低炭素社会の実現に資するとともに、都市や地域の新たな交通手段、観光・地域振興、高齢者や子育て世代の移動支援など、生活・移動の質の向上をもたらす新たなカテゴリの乗り物として期待されている。しかし、超小型モビリティには、A)航続距離が短い、B)バッテリーへの充電時間が長い、C)バッテリーの充電頻度が高いという課題がある。これらの課題は以下に示す方法で解決できると考える。

- ブレーキ命令時、モータの運動エネルギーを電気エネルギーに変換することでブレーキをかけ、変換できた電気エネルギーをバッテリーに回収する。
- 従来のバッテリーにリチウムイオンキャパシタを加える(ハイブリッド電源)ことで、従来のバッテリーでは困難であった急速充放電を可能にする。
- ワイレス給電技術を利用することで駐車場停止中の自動充電が可能となり、利用者は充電操作の手間を感じなくなる。さらに、信号待ち充電や移動中充電を実現できれば、航続距離の飛躍的な増加につながる。

これらの解決方法に対するシステムは、図1に示すように(a)モータ駆動・回生制御システム、(b)ハイブリッド電源管理システム、そして(c)ワイヤレス充電システムが考えられる。図1中の超小型モビリティはトヨタ車体が開発・販売している超小型電気自動車のCOMS(コムス)である。各システムは少なくとも1つの他のシステムと結合され、また3つのシステムはどれもプラント(制御対象)の仕様が一意的ではない。このため、各システム間のスムーズな結合、プラントの仕様変更に対応可能な成果物を獲得、コントローラのプロトタイプ(マイコン)の設計・実装の柔軟な対応、再利用性の高い成果物の蓄積による生産

図1. 超小型モビリティの組込みシステム

性の向上が期待できるモデルベース開発を適用した並行開発は重要になっている。

3. 超小型モビリティ統合開発環境

大規模なモデルベース開発を適用した並行開発には、開発の引き継ぎ、技術者間のスキルレベルと開発速度の違い、モデルの再利用性、工程間のトレーサビリティなどの解決しなければならない問題が多い。モデルベース開発の利点を享受しながらこれらの課題を解決するには、プロジェクト管理手法、技術者スキル標準、標準化された成果物、これらを一元管理する専用ライブラリ、そして専用ライブラリへのアクセスを支援する GUI (Graphical User Interface) で構成された統合開発環境が必要になる。統合開発環境の運用概要を図2に示す。モデルベース開発の検証手法には、Model In the Loop Simulation (MILS), Rapid Control Prototyping (RCP), Software In the Loop Simulation (SILS), Processor In the Loop Simulation (PILS), そして Hardware In the Loop Simulation (HILS) などがある。MILS は、制御装置（コントローラ）と制御対象（プラント）といった組込み機器全体をモデルで表現し、コンピュータ上でモデルを使用したシミュレーションを行うことにより設計工程において制御仕様の詳細な検証を行う手法である。正確なシミュレーションを行うには、プラントモデルのパラメータを詳細に設定することやコントローラにおけるハードウェア（マイコン）の動作を仕様に合わせて設定することが必要である。RCP は、MILS で作成したコントローラモデルを実機に接続して検証を行う手法である。コントローラモデルは専用のエミュレータ（プロトタイプ）に

ロードすることによりリアルタイムで動作することができる。SILS は、コントローラにモデルではなく自動生成された C 言語などのプログラムを使用し、これをプラントモデルと接続して検証を行う手法である。PILS は、コントローラにマイコンを使用し、これをプラントモデルと接続して検証を行う手法である。HILS は、コントローラに開発したハードウェアを使用し、これをプラントモデルと接続して検証を行う手法である。プラントモデルは専用のエミュレータにロードすることによりリアルタイムで動作することができる。図2に示した開発ツールは一例であり、開発に応じて変更される。

統合開発環境を運用するユーザにはプロジェクトマネージャとメンバ（開発技術者）がいる。プロジェクトマネージャは専用ライブラリにアクセスし、プロジェクト管理手法を再利用することでメンバが行うプロジェクトをマネジメントする。プロジェクトマネジメントではプロジェクトマネージャとメンバが工程表、進捗状況報告書、課題管理表、成果物管理表という4つのドキュメントをやりとりする。これら4つのドキュメントは専用ライブラリで一元管理する。プロジェクトにはシステム開発とスキル獲得がある。メンバはこの2つのプロジェクトに取り組む。スキル獲得プロジェクトでは、メンバは自分のスキルマップと開発分野のキャリアマップを確認し、不足しているスキルを分析し、対象となる教育プログラムの履修によって不足スキルの獲得を行う。スキルマップ、キャリアマップ、教育プログラムは専用ライブラリで一元管理する。システム開発プロジェクトでは、メンバはプラントモデリング工程やモデルベース開発手法によるコントローラ開発の各工程で

図2. 統合開発環境の運用概要

統合開発環境から必要な計測・検証環境を運用し、専用ライブラリから過去の資産（成果物）を再利用しながらシステム開発を行う。開発によって生成された成果物は専用ライブラリで一元管理される。また、専用ライブラリへのアクセスは必ず GUI を介して行う。

3.1 プロジェクトマネジメント

モデルベース開発の利点を享受した開発を行うには、システム開発やスキル獲得といったプロジェクトをマネジメントする必要がある。プロジェクトマネジメントはプロジェクト管理手法をベースに行われる。さまざまなプロジェクトマネジメント手法が提案されているが、ここでは以下のドキュメントをプロジェクトで使用することを考える。これらは、ET ロボコン（Embedded Technology ソフトウェアデザインロボットコンテスト）^{†7)} 活動におけるプロジェクトマネジメントに修正を加えたものである。

A) 工程表

各プロジェクトにおいて、どの工程の、どの作業を、どのメンバが、いつ行うのか、の計画と実施状況をリアルタイムかつ視覚的にチームで共有するドキュメントである。また、作成・更新を行う権限はプロジェクトマネージャのみに与えられている。

B) 進捗状況報告書

プロジェクトマネージャが工程表・課題管理表・成果物管理表を更新するための資料としての役割とプロジェクトマネージャが各メンバの作業手順・作業能力を把握し、作業のアドバイスや課題の割り振り、リスク回避を行う役割がある。また、メンバは定期的に進捗状況報告書を作成し、プロジェクトマネージャに提出する義務がある。

C) 課題管理表

課題をチームで共有する役割がある。また、メンバは課題が抽出された場合、必ず課題管理表に登録する義務があり、プロジェクトマネージャは登録された課題の優先度付けとメンバへの割り振りを行い（課題管理表を更新し）、必要があれば工程表を（作業を追加するなど）更新する。また、プロジェクトマネージャとメンバは定期的に課題管理表を確認する義務がある。

D) 成果物管理表

成果物を標準化し、それをチームで共有する役割がある。登録された成果物はチーム全体で承認を行い（この過程で成果物は標準化される）、その後プロジェクトマネージャ（あるいはプロジェクトマネージャに任命されたメンバ）が専用ライブラリに登録する。すなわち、メンバは成果物を専用ライブラリに勝手に登録することはできない。成果物管理表への登録とチームの承認を経ることで標準化された成果物を専用ライブラリに登録できる（成果物をチームで共有できる）。

以上の4つのドキュメントの役割を明確化することで、

開発プロセスに関する課題の解決が期待できる。

3.2 スキル獲得

メンバがスキルを獲得する際に重要なことは、開発に必要なスキルには何があるのか、メンバは何のスキルをどのレベルで持っているのか、スキルを獲得するにはどうすればよいのか、どのスキルから獲得すべきか、であり、これらは明確にされていなければならない。そこで、スキル獲得プロジェクトに取り組む前に、ETSS-JMAAB (Embedded Technology Skill Standards-Japan MATLAB Automotive Advisory Board)⁸⁾ をカスタマイズしたスキル標準を作成する。スキル標準の構成は ETSS-JMAAB と同様にスキル基準、キャリア基準、教育研修基準とし、スキル基準ではプロジェクトに必要な全てのスキルとそのレベルを定義し、各メンバのスキルマップを作成する。キャリア基準では本プロジェクトの作業内容を分類し、それを職種として定義し、各職種での必須スキル群とその必須レベルを定義したキャリアマップを作成する。教育研修基準では各メンバのスキルマップと各職種のキャリアマップの差異を埋めるための教育プログラム群を作成し、さらにその履修手順を視覚化する。これらを専用ライブラリに登録し、プロジェクトマネージャによる管理のもとで、システム開発プロジェクトと並行してスキル獲得プロジェクトに取り組む。具体的には、メンバは自分のスキルマップと開発分野のキャリアマップの差異を確認し、不足しているスキルを教育プログラムの履修によって獲得し、スキルマップを更新する。これがスキル獲得プロジェクトである。

4. まとめ

生活・移動の質の向上をもたらすと期待される超小型モビリティは、航続距離が短い、バッテリーへの充電時間が長い、バッテリーの充電頻度が高いという課題を克服しなければ普及が難しい。このためには、モータ駆動・回生制御システム、ハイブリッド電源管理システム、そしてワイヤレス充電システムが結合したシステムの開発が考えられる。このような開発では、プラントの仕様が一次的でなくともモデルベース開発手法でシステムを並行して開発できる統合開発環境が必要になっている。統合開発環境はシステム開発の効率化だけでなく、プロジェクトに携わるメンバのスキル獲得の機能も必要と考える。

(2018.6.17-受理)

文 献

- 1) “自動車：超小型モビリティについて—国土交通省 都市局 自動車局,” [Online]. Available at: http://www.mlit.go.jp/jidosha/jidosha_fr1_000043.html. [参照：15-6 月-2018]
- 2) “自動車産業を巡る構造変化とその対応について（経済産業省 製造産業局 自動車課）,” [Online]. Available at: <http://www3.keizaireport.com/report.php/RID/329514/?fe>. [参照：

15-6 月-2018]

- 3) 高田広章, 宿口雅弘, 南角茂樹, 岸田昌巳: “リアルタイム OS と組み込み技術の基礎—実践 μ ITRON プログラミング,” CQ 出版, 2004
- 4) “II-24-2. 組み込みシステム開発手順と関連文書 | 日本 OSS 推進フォーラム,” [Online]. Available at: <http://ossforum.jp/en/node/1052>. [参照: 15-6 月-2018]
- 5) “2010 年版組み込みソフトウェア産業実態調査報告書の公表について (METI/経済産業省),” [Online]. Available at: http://www.meti.go.jp/policy/mono_info_service/joho/downloadfiles/2010software_research/index.html. [参照: 15-6 月-2018]
- 6) “2012 年度『ソフトウェア産業の実態把握に関する調査』の報告書を公開: IPA 独立行政法人 情報処理推進機構,” [Online]. Available at: <https://www.ipa.go.jp/sec/softwareengineering/reports/20130426.html>. [参照: 15-6 月-2018]
- 7) “ET ロボコン 2018 公式サイト—組み込みソフトウェア技術教育をテーマとした『ET ロボコン』開催のご案内,” [Online]. Available at: <http://www.etrobo.jp/2018/>. [参照: 17-6 月-2018]
- 8) “JMAAB (Japan MATLAB Automotive Advisory Board),” [Online]. Available at: <http://jmaab.mathworks.jp/>. [参照: 17-6 月-2018]
- 9) “About the Unified Modeling Language Specification Version 2.5.1,” [Online]. Available at: <https://www.omg.org/spec/UML/About-UML/>. [参照: 27-6 月-2018]

† 用語解説

ET ロボコン: 統一モデリング言語 (UML, Unified Modeling Language)⁹⁾ などのモデリング手法で設計したソフトウェアを LEGO MINDSTORMS NXT を使用した自律型のロボットに実装して決められたコースを走行させ, そのスピードを競うコンテストである。若手組み込み技術者教育をテーマにしているため, コンテストでは走行タイムを競うだけではなく, 開発ソフトウェアの機能やその妥当性を説明したドキュメント (モデリングシート) の出来栄も評価される。UML は, 1990 年代初頭に開発された数々のモデリング手法のアプローチを統合し 1997 年に標準化された。UML を使う利点の 1 つとして, 上流工程から下流工程まで UML モデルによる対応が可能である点が挙げられる。UML はソフトウェアのオブジェクト指向の分析・設計の過程で用いられる図形言語であり, Version2.x では 13 種類の図 (ダイアグラム) が定義されている。

著者紹介

茂木和弘 (もてぎ かずひろ)

1993 年 3 月 群馬大学大学院工学研究科博士後期課程電子情報工学専攻修了

1993 年 4 月 群馬大学工学部情報工学科 助手

2007 年 4 月 群馬大学大学院工学研究科生産システム工学専攻 助教

2014 年 4 月 群馬大学大学院理工学府知能機械創製部門 助教