

初学者の MATLAB/Simulink の知識習得を促進する パターンランゲージの提案

The Pattern Language to Acquire Knowledge of MATLAB/Simulink for Beginners

三菱電機株式会社 設計システム技術センター

Mitsubishi Electric Corporation

○伊藤 弘毅¹⁾

○Hiroki Itoh¹⁾

Abstract "The pattern language for beginner of MATLAB/Simulink" aims at encouraging entry-level developers to acquire knowledge of Model-Based Development, MBD. The pattern language, which consists of 7 patterns, could share knowledge about general concept and operation of the tool and effective ways to enhance modeling in a team development. The paper indicates the description of each pattern and the points to express the knowledge considering to beginners.

1 はじめに

モデルベース開発(以下 MBD)は、汎用マシン上でモデルを構築して振る舞いをシミュレーションすることで、開発対象の制御仕様等を実機なしで机上検証することを可能にする開発技術である。MBDは、特に自動車業界で広く導入されており、MathWorksが開発している MATLAB/Simulink が事実上の標準ツールとして使用されている。

MBD 適用による効果を最大限に享受するには、ツールの使い方やモデルによるロジックの構築方法について十分に知識を有していることが条件となり、各開発者に必要な知識を効率的に習得させることが課題となる。組織で MBD を運用していくにあたり、モデリングする際に遵守すべき内容を整理したガイドラインが複数公開されている。MBD のモデリングに関する指針は、JMAAB が発行しているガイドラインに記載がある[1]。また、MathWorks は ISO 26262 や AUTOSAR に対応するためのベストプラクティスを整理したホワイトペーパーを公表している[2][3]。これらの文書に記載されている内容は、ブロックの使い方や推奨される設定など、具体的な実装方法に関する内容が主となっている。

MBD を実践するには、各開発者はモデルの実装方針だけでなく、モデリングの効率的な進め方や質の高いモデルの構築に役立つ概念を認識しておく必要がある。ツールの使い方に関しては、MathWorks のリファレンスを参照したり、トレーニングを受講したりすることにより学習することができる。しかし、モデリングを実践する際の工夫や、MBD を実際の開発現場で運用していくための考え方といったノウハウを、先述の方法で習得することは難しい。ノウハウは、モデリングを実践して経験を積み重ねていく中で得られるものであるため、特に MBD プロジェクトに参加

三菱電機株式会社 設計システム技術センター

Design Systems Engineering Center, Mitsubishi Electric Corporation

兵庫県尼崎市塚口本町 8-1-1 Tel: 06-6497-7355 e-mail: Ito.Hiroki@dr.MitsubishiElectric.co.jp

8-1-1, Tsukaguchi-Honmachi, Amagasaki, Hyogo, Japan

1) 三菱電機株式会社 設計システム技術センター ソフトウェア技術推進部 システムソフトウェア開発技術グループ

Design Systems Engineering Center, Mitsubishi Electric Corporation

【キーワード:】パターンランゲージ, モデルベース開発, MATLAB, Simulink, 人材育成, 教育

したばかりの初学者が即座に習得することは難しい。また、現場の経験者が持つ知見は共有されているとは限らないため、初学者がその知識を認識し習得できるとは限らない。そのため、MBDに関する実践的なノウハウを初学者に効率よく伝達していくことが、開発プロジェクトを安定して運営するための課題となる。

そこで、筆者は初学者がMBDを実施する際に共通的に必要となる工夫点や、効率的なモデリングを実践するための考え方を言語化し、体系的に整理することを試みた。整理にあたって、知識を記述する枠組みとしてパターンランゲージを採用した。パターンランゲージは、元々は建築分野において町や建物を設計する上での知識を共有するために考案され[4]、90年代からはソフトウェア開発の分野に応用されるようになった記述方法である。ソフトウェア開発におけるパターンランゲージは、PLoPの会議等を通じて多く発表されてきたが、MBDの知見を共有するものは見当たらない。

筆者は、主に経験の浅いMBDの開発者がプロジェクトに参画してMATLAB/Simulinkを使用する際に、効率的にモデリングを進めるために必要となる知見や考え方を伝達することを目的とした「初学者のMATLAB/SimulinkによるMBDの適用を促進するパターンランゲージ」を作成した。本パターンランゲージは、複数のMBDプロジェクトに参画する中で得られた知見をパターンとして言語化し、集約することにより作成した。本取組みにより、MBDプロジェクトに参画する初学者に対し、経験者が持つノウハウを効率的に伝達できるようになることを目指す。

本論文は以下の章から構成される。2章では、本論文で提起するパターンランゲージの概要を記述する。3章では、パターンランゲージを構成する各パターンの記述を示す。4章は、各パターンの効果を評価した結果を示す。5章において、今後の展望を述べる。

2. パターンランゲージの概要

表 1 パターン一覧

カテゴリ	ID	名前	概要
モデルをつくる	P1	簡単な実験	簡単なシミュレーションをして未知のブロックの動きを確認する
	P2	ダイアログを開かない	ダブルクリックでダイアログを開かずにペインを使って編集作業をする
	P3	専門家の助言	不明点があったら自分で執拗に調べずに早めに有識者に尋ねる
	P4	サンプルの力を借りる	ブロックの使い方が分からなかったら、サンプルモデルを動かしてみる
チームでモデルをつくる	P5	見た目をまねる	作成済みのモデルの形を真似して、似た機能を作成する
	P6	見た目に訴える	モデリングのポイントがあれば、文章だけでなく視覚的に表現する
	P7	過去の失敗から学ぶ	エラーの対処法を記録しておいて、他のメンバに共有する

2.1 構成パターン

本パターンランゲージは、7つのパターンから構成され、適用場面に対応した2つのカテゴリを持つ。一つ目のカテゴリは「モデルをつくる」で、開発者個人で一つのモデルを作成するときには有用なパターンをまとめたものである。二つ目のカテゴリは「チームでモデルをつくる」で、複数人のチームでモデルを作成するときには有用なパターンをまとめたものである。これらのパターンは、筆者がMBDプロジェクトにおいてモデリングの実務を実践した中で、開発の中で繰り返し発生した課題とその対応策について、汎用的に他のプロジェクトでも知見を活用できるものを抽出して得られたものである。

本パターンランゲージが持つパターンの概要と所属するカテゴリを表 1 に示す。各パターンの詳細な内容については、3 章で記述する。

2.2 パターン記述の工夫

パターンには様々な記述方法[5]があるが、初学者に対して知識を浸透させていく目的を鑑みると、可能な限り平易で分かりやすくパターンを表現する必要があると考える。パターンの典型的な記述方法である Canonical Form[6]は、「状況」、「問題」、「フォース」、「解決方法」、「結果状況」と項目立てて表現する。この記述方式は体系的に知識を整理するには有用だが、パターンに見識のない開発者は必ずしもこの記述方式に慣れているとは限らない。特に、新しく MBD の知識習得を進めている開発者にとって、パターンの記述は可能な限り平易かつ馴染みやすい表現にすることで、内容を理解するための負担を生じないようにすることが重要と考えた。

そこで筆者は、コラボレーション・パターン[7]の「その状況において」、「そこで」、「その結果」という記述項目を採用し、記述に流れを持たせてパターンの知識が伝わるような構成とした。また、本文を口語的な文章にしたり、途中にイラストや説明の図を挿入したりすることで、開発者がパターンを読む敷居を下げるように意識した。特に、Fearless Change[8]のパターン記述にて頻繁に確認できる、新しい技術に挑戦する開発者を勇気づける「~しよう」といった言葉を多く使用するようにした。

上記のとおりパターン記述を工夫することにより、初学者がパターンを最後まで読み切り、モデリングの知見を円滑に習得できるように意識した。

3. パターン記述

本章では、「初学者の MATLAB/Simulink による MBD の適用を促進するパターンランゲージ」を構成する 7 つのパターン記述を示す。なお、紙面の都合上、パターン記述のイラストは省略して載せている。

3.1 簡単な実験

使い方が分からないブロックがあったら、簡単なモデルを作って試しに動かしてみよう

あなたはモデルに新しいロジックを組み込もうとしているときに、役立ちそうなブロックを見つけた。そのブロックを使えば、ロジックをエレガントに実装できそうな予感がする。ただ、そのブロックは今まで使ったことがなく、使い方がよく分かっていない。

その状況において

使おうとしているブロックの使い方や設定が分からず、ドキュメントをいたずらに検索して時間を浪費してしまう。

そこで

使い方が分からないブロックがあったら、モデルにいれて試しに動かしてみよう。

新しく空のモデルを用意して、そのブロックを入れてシミュレーションしてみて結果を確認してみよう。Simulink では、シミュレーションボタンを押すだけで簡単に動きを確認することができる。もし、想定に対して惜しい動きをしているのであれば、その時はドキュメントを参照して設定内容を見直してみよう。ドキュメントだけでなくサンプルの力を借りることも考えよう。また、実験の結果わからないことがあれば、専門家の助言を得て疑問点を解決しよう。簡単な実験で作ったモデルは、専門家に問題点を伝えるのに大いに役に立つ。

その結果

手元でブロックの動きを確認することができるようになるため、振る舞いを理解する時間が少なくて済むようになる。

3.2 ダイアログを開かない

モデルやブロックのプロパティは、ダイアログではなくペインで変更してみよう

モデルに追加したブロックのパラメータを変更しようとしている。モデルに追加したブロックのプロパティを変更するには、それをダブルクリックしてブロックパラメーターダイアログを開いて変更する。

その状況において

一つ一つダイアログを開いて設定を変更するのは非常に手間である。変更するブロックの数が少ないときは大きな問題ではないが、複数のブロックに対して設定をしなければならぬときはとても面倒である。

そこで

ダブルクリックしてダイアログを開くのではなく、ペインでプロパティを変更してみよう。ブロックのパラメータを変更するときは、モデルデータエディタ (Ctrl+Shift+E) を積極的に利用しよう。モデルデータエディタを利用すれば、ダイアログを開くことなく、複数ブロックのパラメータを一つの画面のなかで変更することができる。また、ブロックを右クリックして開くブロックプロパティについても、ダイアログなしで変更できるときがある。ブロックプロパティは、プロパティインスペクターを利用することで、ダイアログを開くことなくモデルの画面上で編集することができる。また同様に、ブロックの追加もライブラリブラウザーによりダイアログを開くことなく行える。ブロックを追加するときはモデル上でダブルクリックし、ブロック名を入力することで追加できる。

その結果

モデルに複数のブロックを追加し、ブロックのパラメータやプロパティを変更するとき、一回一回ダイアログを開く必要性がなくなるため、モデリングの時間を短縮させることができる。これは特に、モデルをつなげるときに最低限の IF 定義をするときやロジックの再利用をするときに便利である。

3.3 専門家の助言

分からないことやうまくいかないことがあったら、早めに専門家のアドバイスを求めよう

ブロックの使い方が分からない。あるいは、モデルをシミュレーションしようとしても、エラーが起きたり意図通りに動作しなくて困っている。

その状況において

なかなか問題を解決する方法が見つからず、新たな価値を生み出すはずの貴重な多くの時間を失ってしまう。

MATLAB/Simulink は商用ということもあり、他のプログラミング言語と比べるとインターネット上にある情報量は少ない。途中で解決を諦め、実現したいロジックの実装を断念することになるかもしれない。

そこで

専門家の意見を積極的に聞いてみよう。MATLAB の保守サービスを契約しているのであれば、サービスリクエストを出して、自分がしてみたいことや困っていることを、直接 MathWorks に質問してみよう。また、近くに経験者がいるときは、早めに声をかけて相談してみよう。質問するときには、いま自分が何に困っているのかを明確にする作業が必要になってくる。その時は、簡単な実験をすると効率的だ。小さなモデルのサンプルを作ることで、いま自分がやりたいことが何であり、何に困っているということが整理できてくる。

その結果

モデリングの課題解決に要する時間を減らすことができる。そして、もし問題を解決できたときには、過去の失敗から学ぶため得られた知見をチームに共有しよう。

3.4 サンプルの力を借りる

ブロックやサブシステムの使い方が分からなかったら、サンプルモデルを動かしてみよう

あるブロックがロジックをつくるのに使えそうな感じだが、その使い方がよく分かっていない。

その状況において

ブロックの使い方がよく分からず、多くの時間を失ってしまい、予期せぬ動作をしてしまうかもしれない。

ブロックの使い方はリファレンスとして書かれているが、字面のみだとなかなか動きを掴むのが難しい。どの設定がキモでどう設定したらいいのか、前例がなければ使うのに躊躇してしまう。

そこで

使えそうなモデルのサンプルを見つけて、試しに動かしてみよう。

最初に、使い方が分からないブロックがあったら、MathWorks のリファレンスにあるサンプルを動かしてみよう。動くモデルでブロックの働きを確認すると、使い方のヒントになるだろう。

また、モデルの作り方や動きをチームの他の人に伝えたいときもある。その時は、自分でサンプルモデルをつくらう。言葉で伝えるよりも、動くモデルで伝えたほうが分かりやすい。

サンプルモデルには、簡単な実験で作ったモデルが使えるかも知れない。もし、単純なブロックの使い方だけでなく、モデルの構造に伝えたいメッセージがあるのであれば、それを見た目に訴えるように表現しよう。

その結果

サンプルを確認することで、ブロックの振る舞いがよく理解できるようになり、ロジックの実装を効率よく進めることができるようになる。

また、他の人にサンプルを通じてメッセージを伝えることができ、チーム全体のモデリングの質の向上が見込める。

3.5 見た目に訴える

モデリングで意図するメッセージを、視覚的にメンバに伝えよう

モデリングした内容について、メンバと共有したい情報があるとき、それを全て文章で表現するのは難しいときがある。

その状況において

全ての情報を文字で伝えようとするのは面倒だし、うまく伝わらないかもしれない。

情報を伝える一番単純な方法は、文章で伝えることである。そのため、意識的でなければすべて文字で表現してしまい、結果として情報が伝わらないことが起こるかもしれない。

そこで

ブロックの配置や背景色などを工夫することで、情報を伝えられないか考えてみよう。

最もわかりやすいのは、処理の時系列な流れを視覚的に訴えるということである。人間は、基本的に左側を過去、右側を未来に捉える傾向がある。その他、縦に処理を並べて書くことで、その処理が並列的に実行されることを伝えることができる。

背景色を変更すると、モデリングするときに目に飛び込んで来やすいので、効果的にメンバに注意喚起ができるようになる。例えば、変更を意図しないモデルの背景をグレーに変えるのも良い。同様に、暫定的な実装があったときにブロックの背景を赤く変更することでそれを伝えるのも効果的だろう。

その結果

メンバのモデリングの意図を効果的に伝えることができる。文字情報だけで伝えることが難しいことも、配置や色を変えたりすることで、他のメンバに訴えかけることができる。

3.6 見た目をまねる

モデルの全体の構成を考えると、ほかの似たモデルの見た目をマネしてみよう

複雑なロジックのモデルを作ろうとしているとき、あるべき構成が思いつかないでいる。

その状況において

アドホックなモデリングをしてしまい、品質の良くないモデルが出来上がってしまう

また、チームでモデリングをしているときは、他の人が書いたモデルと違った構成になってしまい、全体として一貫性のないモデルとなってしまう。

そこで

追加しようとしているロジックと似たもののモデルがあったら、その見た目をマネしてみよう。

見た目をマネするというのは、使っているブロックや信号線の接続の仕方を見て、同じような構成になりそうなところがあれば採用するという意味である。Simulink のモデルはグラフィカルに表現されるため、一般のプログラミング言語よりもマネがしやすい。

また、似たような性質を持つロジックをいくつかモデリングすると分かっているときは、それらを実現しやすいモデルの構成をあらかじめ検討してテンプレートを作っておこう。

その結果

モデルの構成に迷いがなくなり、スムーズにロジックの実装を進めることができる。また、ほかの人がモデルのロジックを確認するときに、見慣れた構成であることから、モデルが実現しようとしているロジックに集中して見るようになる。

3.7 過去の失敗から学ぶ

発生したエラーの情報は、他のメンバーに共有できるように記録に残しておこう

モデルをコンパイルしたが、これまで初めて見るエラーが発生した。エラーメッセージを見てみたが、どう解決したらよいか検討がつかない。

その状況において

エラーを解決するのに時間がかかってしまう。モデリングが進まない。

エラーメッセージの内容だけでは、何が起きているのか想像できないときもある。他のメンバーが以前にエラーに遭遇したことを覚えていても、解決方法を覚えていない場合もある。

そこで

発生したエラーが解決したら、その情報をメンバが確認できる場所に記録しておこう。

記録として残しておけば、あとで他のメンバが同じエラーに遭遇したときに、すぐに解決できるようになる。同じチームでモデリングをしていると、同じようなエラーに遭遇するものである。

また、自身においても一度遭遇したエラーに、時間を置いてまた出くわすことはよくある。

その結果

発生したエラーを迅速に解決でき、モデリングの作業にすぐに戻ることができるようになる。

4. 評価

筆者は、提案パターンランゲージの効果を確認するため、2つの評価実験をした。一つは、初学者に対して実施したモデリングのケーススタディである。もう一つは、経験者に対して実施したパターンの効果に関するアンケートである。

4.1 ケーススタディによる評価

本評価において、被験者に簡単なモデリングの課題に取り組んでもらい、パターンに記述された知識を理解する前後の作業内容を比較することで、モデリングの作業効率が向上するかどうかを検証した。被験者は、MBDの初学者4名である。

ケーススタディでは、まず被験者はパターンの知識がない状態で一つのモデリングの課題に取り組んだ。その後、本パターンランゲージにおける「モデルをつくる」カテゴリの4つのパターンに対する説明を受けた後に、異なるモデリング課題に取り組んだ。なお、モデリング課題の難

表 2 パターン説明前後の適用件数

被験者	説明前	説明後
A	1	4
B	0	1
C	0	3
D	0	2

表 3 アンケート評価者の属性

所属組織	人数	経験年数	人数
A	2	2年未満	1
B	2	2年以上5年未満	2
C	2	5年以上10年未満	2
D	1	10年以上	2

表 4 アンケートによる質問項目と評価点数

質問項目	P1	P2	P3	P4	P5	P6	P7
[Q1] パターンのコンテキストに過去悩まされた経験を持つか 4:よくある, 3:たまにある, 2:まれにある, 1:ない	▲7 ▼0	▲3 ▼4	▲7 ▼0	▲6 ▼1	▲6 ▼1	▲6 ▼1	▲7 ▼0
[Q2] 今パターンを見て, 昔聞いておけばよかったと思う内容か 4:思う, 3:やや思う 2:あまり思わない, 1:思わない	▲4 ▼3	▲7 ▼0	▲7 ▼0	▲6 ▼1	▲5 ▼2	▲5 ▼2	▲7 ▼0
[Q3] 初学者に伝わりやすい・伝えやすい内容となっているか 4:思う, 3:やや思う 2:あまり思わない, 1:思わない	▲6 ▼1	▲5 ▼2	▲5 ▼2	▲5 ▼2	▲6 ▼1	▲6 ▼1	▲6 ▼1

易度が結果に影響しないように、課題の出題順番を入れ替えて実験するように考慮した。上記 2 回のモデリング課題の結果を比較することにより、当該パターンに対する初学者の理解度と適用によるモデリング作業の効率改善の効果を評価した。

パターンの説明前後における、課題に対するパターンの適用件数の変化を表 2 に示す。本表に示すとおり、パターンの説明前と比較して説明後のパターンの適用件数が多く、初学者においてもパターンの記述を理解でき、即座にモデリング作業に反映できることを確認した。

パターンの適用によって、以下に示すモデリング効率が向上する効果を確認できた。

- 簡単な実験：

ブロックの仕様をモデルで動作確認することで実装効率が向上、制限時間内での課題完了を実現

- ダイアログを開かない：

複数入出力端子のプロパティの設定に要する時間が、平均 44%削減 (67 秒→38 秒)

- 専門家の助言：

適宜有識者に質問してモデリングを進めることで、制限時間内での課題完了を実現

- サンプルの力を借りる：

サンプルモデルを使って未知のブロックを動作確認することで、制限時間内での課題完了を実現

併せて、ケーススタディ後に被験者に感想を尋ねたところ、「専門家の助言は役に立つ」など好意的な意見があった一方、「説明を補足する図があったほうがよい」といったパターンの改善に繋がる意見を得ることもできた。

4.2 アンケートによる評価

4.1 に示したケーススタディによる評価に加え、有識者に対するアンケートにより、パターンの有効性を評価する実験をした。全 7 パターンに対して、4 段階評価の 3 つの質問を設定した。パターンを評価した有識者は 7 人である。有識者の情報を表 3 に付記する。所属組織や経験年数に幅を持たせるように評価者を設定した。

表 4 に、質問項目と有識者が評価した点数の内訳を記載した。ここで、表中の▲印は 4 点または 3 点の評価件数、▼印は 2 点または 1 点の評価件数を示している。列は、表 1 で示した ID でパターンを並べている。

Q1 と Q2 の評価結果を確認する限り、一部を除き 3 点以上の点数となった件数が過半数となっ

ている。このことから、各有識者はパターンが対象とするコンテキストに問題を感じ、パターンの内容を事前に知識として習得することに価値を感じていることが分かる。一方で、比較的低い点数の質問項目もいくつか確認できた。例えば、「簡単な実験」パターンについては、モデリングを始めて早い段階でパターンを無意識的に実践している場合が多く、パターンを明示的に伝える必要性を低く感じる有識者が多かった。「ダイアログを開かない」パターンに関しては、当該機能をサポートしていない MATLAB のバージョンを利用している場合はパターンを適用できないほか、小さいモニタを利用している場合は作業領域が狭くなり逆に非効率となるという意見が得られ、開発者の置かれたコンテキストに応じてパターンの有効性に関して評価が分かれることが分かった。また、有識者においてもすべてのパターンの知識を有しているとは限らず、幾つかのパターンの内容について初めて知って参考になったという意見も得られた。

Q3 の評価結果に関しても、概ね高評価の件数が多かったことから、初学者においても理解しやすくパターンを記述できていると思われる。一方で、具体例を記述した方が良いなど、より伝わりやすいパターンにするための改善案についても意見を得られた。

5. おわりに

4 章に記載のとおり、本パターンランゲージを利用することにより、MBD 導入にあたって必要となる MATLAB/Simulink の概念や操作方法に関する知識や、チームで MBD を実践することに役立つ知識を開発者に効率的に伝達できることを確認した。

今後の展開として、以下の二点を挙げる。

- ・ MBD の応用知識に関するパターンの拡充

本論文で提案したパターンランゲージは、初学者の MBD に関する知識習得を促進することを目的に作成したものである。初学者に対する支援の一方、パターンの拡充により一定の経験者が持つべき知識を補完することも、開発プロジェクトに対して有効であると考えられる。具体的には、作成した個々のモデルを統合し、一つの大規模なモデルを構築する際に有効となる知識をパターン化すると有効であると推察している。また、MBD のプロセスが成熟してくると、開発を効率化するためにタスクを自動化する試みが行われる。この自動化に対する考慮点もパターン化できる余地があると考えている。

- ・ アンケート結果に基づくパターンの洗練

パターンは一度作成して完了するものではなく、パターンコミュニティにおけるシェファードイングというレビューが象徴する通り、内容を改善するプロセスが重要視されている [9]。本パターンランゲージも、アンケート結果をフィードバックとして、記述を改善させていきたい。

参考文献

- [1] “制御モデリングガイドライン Ver6”. <https://jmaab.jp/>, (参照 2022-07-11)
- [2] Jason Moore and John Lee, 11 Best Practices for Developing ISO 26262 Applications with Simulink, 2019.
- [3] David Jaffry and Holly Keener, 10 Best Practices for Deploying AUTOSAR Using Simulink, 2020.
- [4] Christopher Alexander et al., “A Pattern Language: Towns, Buildings, Construction”, Oxford University Press, 1977.
- [5] Eric Gamma et al., “Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.
- [6] C2 Wiki, “Canonical Form”, <http://wiki.c2.com/?CanonicalForm>, 2011.
- [7] コラボレーション・パターンプロジェクト, “Collaboration Patterns”, 2012.
- [8] Mary Lynn Manns and Linda Rising, 川口恭伸監訳, “Fearless Change アジャイルに効く アイデアを組織に広めるための 48 のパターン”, 丸善出版, 2014.
- [9] 江渡浩一郎, “パターン, Wiki, XP - 時を超えた創造の原則”, 技術評論社, 2009.