

MBAC
MBD／ACG 普及ガイドライン



2022/10/XX MBAC MBD 普及 WG

目次

1. Introduction
 - 1.1. 目的
 - 1.2. 適用範囲
 - 1.3. 文書の構成
 - 1.4. 用語・略語
2. MBD／ACG 適用の意義
3. MBD／ACG 適用の障壁
4. MBD／ACG 普及ガイド
5. Information
 - A. MBD／ACG 適用事例
 - B. Toolbox モデルケース
 - C. ACG 使用ガイドライン
 - D. 教育コンテンツ情報
 - E. その他の参考資料

1. Introduction

1.1. 目的

このガイドラインは、MBD を航空機システム開発及びソフトウェア開発へ導入する際の、様々な課題に対し、解決の手がかりとなる情報をまとめた文書である。

また、そもそも、MBD を開発へ導入することで、どういうメリットがあるのか、それを必要なツールのスポンサーに承認してもらうために活用できる情報を収集し収録した。

本書は、システム開発に MBD を活用し、効率化を図りたいと考えているが、導入の方法、適用先、ツール導入、必要なスキルの面で悩みを抱えている担当者を読者として想定している。また、記載の情報を参考に社内関係先への説明に使用することも想定している。

1.2. 適用範囲

本書では、Simulink® で実施する、システムシミュレーションモデリング、シミュレーション、検証、コード生成を対象として検討する。

1.3. 文書の構成

本書は、図 1.3-1 に示す構成となっている。

1.4. 用語・略語

a) MBD : Model Based Development / Model Based Design

モデルを使用して行う開発／設計。

b) Model

対象とするシステムの仕様、設計、分析結果、検証等に関する情報を単純化、抽象化し、体系的に整理した情報の構造体。システムやソフトウェアの構造、ふるまい、インタフェース等を、ブロックダイアグラム等で表現したもの。特にシミュレーション可能なものを指してモデルと呼ぶことがある。本書で使用を想定する Simulink® 等のツールを使用することで、Model 上での計算、シミュレーションを行うことができ、設計仕様の Validation に使用することができる。

c) ACG : Auto Code Generation

自動コード生成。シミュレーション可能なモデルから、ツールの機能を用いて、実装ソースコードを自動生成すること。

2. MBD／ACG 適用の意義

現在、MBD（モデルベース開発）は、自動車や航空・宇宙開発等の分野で、デファクトスタンダードな開発手法となっている。一般的に、以下のようなメリットがあるとされている。

- (1) システム設計上流でシミュレーションを行うことにより、仕様妥当性やサブシステム間インタフェースの整合を、早期に検証することができる。
- (2) 検証、ドキュメント作成の自動化、実装コードの自動生成等の機能を使用することで、ソフトウェア品質の向上、製造コストの削減といった効果が得られる。

本書では、MBD のメリットを十分に引き出すうえで欠かせない、作業の自動化、とりわけ ACG（Auto Code Generation）についても、適用・普及のガイドラインを示す。

上記のメリットに加え、WG 内で議論した、MBD／ACG 適用の意義について、本項で示す。

2.1 MBD 導入のメリット

WG 内の議論で、MBD 導入のメリットとして、以下の様なものを抽出した。

No	MBD 導入のメリット	詳細
1	システム 不適合の早期発見	モデルによるシステム統合シミュレーションを活用することにより、システム統合時に発覚する不適合を、開発の早い段階で発見、解消することができる。 開発の上流で品質を高めることにより、試験フェーズ等でのリワークコストを削減することができる。
2	ソフトウェア品質の向上	設計にモデルを用いることにより、仕様の可視化、曖昧さの排除が可能となる。これにより、ソフトウェアの品質を向上させることができる。また、モデルは動的検証が可能であり、設計仕様の段階での不適合発見が可能となる。
3	業務の自動化ができる	モデリングツールの機能を使用し、テストやレポート作成を自動化することができる。
4	データの有効活用	電子データであるモデルは、ツールの機能を使用し、トレーサビリティを持たせることや、検証カバレッジを測定する等が可能であり、製品品質の向上につながる。
5	再利用性	一連のデータ（要求、モデル、検証結果、検証手順（スクリプト）等）をモデル内で統合的に管理することにより、各モデルは再利用できる。

2.2 ACG 導入のメリット

WG 内の議論で、ACG 導入のメリットとして、以下の様なものを抽出した。

No	ACG 導入のメリット	詳細
1	ソフトウェア開発の効率化	従来のように人の手でコーディングするのに比べ、ソースコードの自動生成を利用すると作業時間が大幅に短縮される。
2	ソフトウェア品質の向上	モデルの段階で検証され、人の手を介さずソースコードが生成されるため、ヒューマンエラーが入り込まず、安定的に高品質なソフトウェアが作成できる。 モデルの段階でも検証されているため、コードに対しては、モデルとコードの動作が一致していることを、モデルの段階で用いた検証ケースを利用して確認することができる。
3	業務の自動化ができる	モデルの検証に使用したケースを用い、コードのテストを自動化することができる。設計(モデル)変更に伴う再帰テストの際にも効果的である。

3. MBD／ACG 適用の障壁

MBD／ACG の適用には、前項で示したようなメリットがある一方で、社内への導入が進まないと感じている会社が多くあるという情報がある。なぜ導入が進まないのかについて分析し、以下に抽出した。4 項以降では、それぞれの導入の障壁に対する打ち手となる情報を掲載する。

3.1 MBD 導入の障壁

No	MBD 導入の障壁	詳細
1	ツール導入コスト	ツール価格が高いことに加え、費用に対する効果を定量的に示すことが難しく、投資に踏み切れない。
2	教育コスト	MBD 及びツールを活用するまでに、トレーニングが必要であるが、習熟には一定のコストがかかる。自社内に有識者、エキスパートが育っておらず、教えられる人が足りない。
3	過去資産の活用が難しくなる	(プログラム言語で書かれた)過去資産を活用することが難しくなり、移行のための手間が必要になることから敬遠される。
4	システムシミュレーションに必要な、環境、メカ、電気などの物理現象のモデル化が難しい	ソフトウェアで実現されるコントローラのモデリング以上に、制御対象となる物理現象をモデル化することが難しいという印象を持たれている。

No	MBD 導入の障壁	詳細
5	ツールの選定が難しい	どのツールを選択すれば目的の課題解決ができるのか、選定が難しい。MATLAB はツールボックスという形で必要な機能を拡充していく仕組みであるが、どのツールボックスの組み合わせが自社業務に必要なのかが判断しづらい。
6	MBD 適用による作業プロセスが明確になっていない	従来のシステム／ソフトウェア開発と対比して、プロセスのどこが変わるのが明確になっていない。これを明確にすることにより、MBD で追加になる作業と、削減できる作業が明確になり、効果や負担の軽減がどの程度かが見えてくる。

3.2 ACG 導入の障壁

No	ACG 導入の障壁	詳細
1	ACG の作業要件、標準がない	高品質ソフトウェアとするために、どのような手順で実施する必要があるのか、定められた手順書、要件、標準が整備されていない。
2	ACG で生成されるコードとハンドコード部の統合方法が明確でない。	(1 とも関連) オートコードとハンドコードを統合する方法について明確な手順化がされておらず、実施方法に迷う。
3	システムモデルが ACG できるレベルで設計されない	ACG を行うにはソフトウェア実装まで考慮したモデリングが必要となるが、システム設計の段階でそこまで精緻なモデリングがされないことが多い。ACG を行うために、モデルを再作成したり、修正したりする手間が必要になり、ACG を使わない選択に至ることがある。
4	ACG の品質／可読性に対する不安感	自動で生成されるコードの妥当性について不安を感じる人がいる。コードの可読性が高くない場合は特に不安感を持たれる。

3.3 各障壁に対する評価

3.1 及び 3.2 で MBD/ACG 導入に対する抽出した各項目に対し、WG 内で議論した結果を以下に示す。【】は、後に示す参考文献の記号)

No	MBD 導入の障壁	対策
1	ツール導入コスト【A】	ツールの価格以上に効果が得られることを説明できる必要がある。他社の有効活用事例や調査研究資料等を活用する。
2	教育コスト【D】	教育コスト以上に効果が得られることを説明できる必要がある。また、効率的に教育、学習が行えるように、教育資料、有償・無償のセミナー情報、学習方法について整理しておく必要がある。
3	過去資産の活用が難しくなる【C】	過去資産の活用、過去資産とモデルの統合の方法をマニュアルの形でまとめられていると導入しやすい。
4	システムシミュレーションに必要な、環境、メカ、電気などの物理現象のモデル化が難しい	モデリングに限定された課題ではないと言える。制御対象の現象をどのように理解、表現するかは、MBD でない設計においても必要なこと。
5	ツールの選定が難しい【B】	コントローラ開発を行う業種の中では、基本的な構成があると考えられるので、使用目的と対応付けたツール構成案を用意すると考えやすい。
6	MBD 適用による作業プロセスが明確になっていない【C】	従来のプロセスと対比できる、標準プロセスの例を共有できると適用がしやすい。

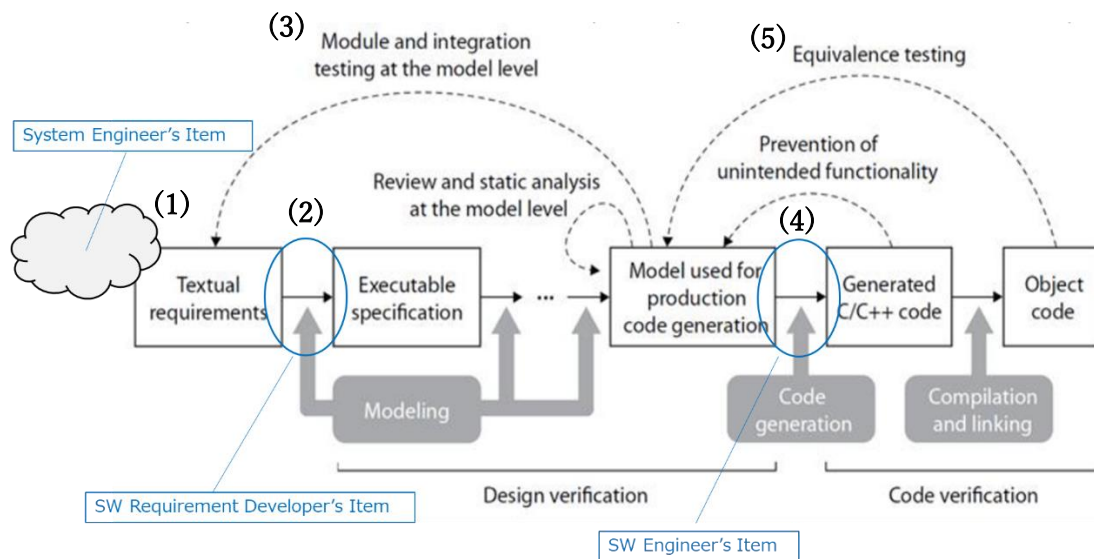
No	ACG 導入の障壁	対策
1	ACG の作業要件、標準がない【C】	ACG の作業要件、標準が必要。プロセスを規定しガイドライン等にまとめることが必要。テスト環境、手順(自動化スクリプト)等、確立し、共有できると有効。
2	ACG で生成されるコードとハンドコード部の統合方法が明確でない。【C】	自動生成コードの実装要領をガイドライン等にまとめることが必要。
3	システムモデルが ACG できるレベルで設計されない【C】	システムエンジニアとソフトウェアエンジニアの間で、どのくらいの精度のモデルを授受するべきかを取り決め、マニュアル等で明確にする必要がある。モデルはソフトウェアコードまで利用することが望ましい。
4	ACG の品質/可読性に対する不安感【C】	自動生成されたコードの検証方法を定め、標準化する必要がある。また、モデリング規約を定め、これに従うことで、ある程度可読性のあるコードを生成させることができる。

4. MBD/ACG 普及ガイド

適用先の工程及び適用のステップ

MBD/ACG 導入し、自社組織に普及させて行くに際し、**従来の開発プロセス成果物を段階的にモデルに置換**していくことが良いと考えられる。

図 4-1 に一般的なシステム/ソフトウェア開発プロセスを示す。図中の各工程について、以下の要領で置き換えていくことが一案として考えられる。各工程での導入は順序を示すものではないため、同時並行、効果的なプロセスを選択して導入してもよい。



(出典：MathWorks 資料に WG 加筆)

図 4-1 一般的な MBD システム/ソフトウェア開発プロセス

- (1) SW Requirements 作成工程において、従来の**要求文書のうち、モデルによる表現に適した部分**を、Executable specification に置き換える。
- (2) 作成した Executable Specification Model 上で要求仕様の**レビュー及び動的検証**を行う。
- (3) Specification Model を System Engineering ヘフィードバックし、System Model を構築、**System の機能性、I/F の検証**を行う。
- (4) 検証済みとなった Specification Model より、**Auto Code Generation** を行う。
- (5) **Auto Generated Code と Specification Model の一貫性**を検証する。

Model の作成（１）は、MBD の前提であり、モデル表現が適した自然言語要求を Model に置換する。

Model の検証（２）により、SW Requirement のフェーズで意図通りに動作することを確認することができ、後の工程での不適合抑制に効果がある。またこの工程で作成するテストケースは後のコードテストのケースとして再使用することが効果的である。

System Model 統合（３）を実施することで、システム統合後に発覚する I/F 不整合や不適合を、開発の早い段階で発見することができる。

モデル化された箇所のコーディングの工程を、人の手によるコーディングから、Auto Code Generation に置き換える（４）ことは MBD の中でも最もコスト効果が得られる部分である。ACG を行うため、一定の制約のもと、Modeling をすることが必要となるが、導入当初から、この工程でコスト及び品質向上の効果を上げることを目指すことを推奨する。

Auto Generated Code と、Model の動きが同一であることをテストにより検証（５）する。その際、（２）で作成したモデルテストケースを再使用し、テスト自動化環境等を構築することにより、コーディング、テストのコストダウンに効果が期待できる。

適用対象システム、ソフトウェア

MBD は各種システム、ソフトウェアへ活用することが可能である。ただし、適用に際し、コスト効果を見極める必要がある。一般には、単純、小規模なプログラムの場合に MBD の適用効果は小さく、複雑、大規模で不適合の発生が相当数発生するプログラムになるほど MBD の適用効果は大きいと想定される。

一方、大規模システムに適用するには適切なプロセスが必要である。プロセスを検証するために、小規模なシステム（パイロットプログラムでも良い）でプロセスの構築、検証を行った上で大規模システムに適用することも有効な手段である。

適用対象としては、システムの制御を行うコントローラ開発に最も多く適用されているが、他分野での活用方法や Toolbox の提供も広がっているため、ツールデベロッパーへ問い合わせるとソリューションを提供してもらえる可能性がある。

5. Information

以下に示す、MBD／ACG を導入する際の参考情報を、別紙として掲載する。

- A. MBD／ACG 適用事例
- B. Tool Box モデルケース
- C. ACG 使用ガイドライン
- D. 教育コンテンツ情報
- E. その他の資料

A. MBD／ACG 適用事例

1. MBD 適用事例

(1) MBD を導入した各社コメント

MathWorks 社資料「モデルベースデザイン（M B D）導入効果」

<https://content.mathworks.com/viewer/623294d0f99d7d2a2f276db2>

から MBD を導入した効果に対するコメントを紹介する。MBD を導入した航空宇宙・防衛メーカーのコメントによれば、MBD が開発効率や生産効率の向上に寄与していることがうかがえた。また、日本メーカーも MBD 導入効果をコメントしており、その中には製品開発における新技術開発にも貢献したとの指摘もあった。

航空宇宙・防衛機器メーカー	各社コメント
エアバス	ソフトウェアのテスト工数を 2 / 3 に短縮した
B A E	バンドコーディングより 1. 5 ～ 2 倍効率化した
インブラエル	開発期間を 6 ヶ月以上短縮できた
ハネウェル	生産性が 5 倍以上向上した
ロッキード・マーチン	開発効率が 2 倍になり、設計更新が 1 日でできるようになった
K A R I （韓国航空宇宙研究院）	開発期間が半減し、設計の繰り返しを最小化した

メーカー	効果	内容
村田製作所	コスト削減	エコマネシステム制御ソフトの開発期間を 5 0 % 短縮し、欠陥のないコード生成を実現
住友重機械工業	製品性能向上 コスト削減	油圧ショベル制御ソフトウェア開発で製品の燃費効率を 1 5 % 向上し、エンジニアの作業工数を 5 0 % 削減した
日産自動車	コスト削減 製品性能向上	エンジンの空燃比コントローラのゲイン調整時間を 9 0 % 短縮し、CO ガス排出を半分以下にした
グローリー	コスト削減	ソフトウェア開発におけるバグの早期発見が可能になった。開発工数を 3 6 % 削減した

日立 Astemo	コスト削減	車間距離制御装置（ACC）のコントローラ開発の時間を半減した。手作業によるコード生成を排除し、テストの速度と効率が向上した
トヨタ自動車	新技術開発	モデルで車両の動きを「先読み」しながら「最適」な操作を決定するといった熟練ドライバーのような自動運転制御を実現した
日産自動車	製品の信頼性向上	MBD ツール（Polyspace）によりサプライヤーのバグを検出しソフトウェアの信頼性が向上した

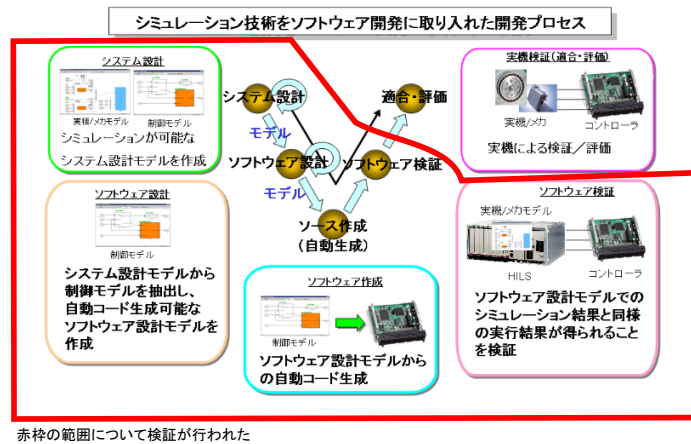
(2) MBD 開発の定量的効果

次の資料に同じ装置制御コントローラの開発に対し、C 言語を用いたレガシー開発と MATLAB/Simulink を用いた MBD 開発を行って、使用工数等について比較し MBD 開発の定量的効果が示されている。

「モデルベース開発ツールを活用した際のコストの効果検証 実施報告書」IPA 独立行政法人 情報処理推進機構 2013.2、<https://www.ipa.go.jp/files/000026867.pdf>

項目	内容
対象作業	ボンディング装置用制御コントローラ開発
開発の範囲	システム設計～ソフトウェア検証
開発手段	レガシー開発：C 言語を使用 モデルベース開発：MATLAB/Simulink、自動コード生成機能利用
比較方法	設計、実装の各プロセスにて混入した不具合を、検証プロセスにて検出した際に修正が必要となるプロセスを明確にするとともに、その修正に掛かった工数をプロセスごとに算出し、レガシー開発とモデルベース開発それぞれでの最終的なプロセス別の生産性を算出し比較する
評価	レガシー開発に対する工数削減率 <ul style="list-style-type: none"> 全体から見たプロセス毎の工数削減率（％） 全体の工数削減率（％）

検証の範囲



検証の範囲

* 出典:「モデルベース開発ツールを活用した際のコストの効果検証 実施報告書」IPA 独立行政法人 情報処理推進機構 2013.2

MBD 開発の利点として、①工数の削減、②コーディングの短縮、③レビュー時間の短縮、④手戻りの軽減がうたわれており、それぞれの項目に対しレガシー開発と MBD 開発の工数が比較され、MBD 開発の工数削減効果が示されている。いずれの項目も MBD が工数削減や作業効率アップに寄与していることがわかる。

MBD開発の効果

No.	MBD開発の利点	
1	工数の削減	コーディングの短縮、レビュー時間の短縮、手戻りの軽減により全体的な工数が削減できる

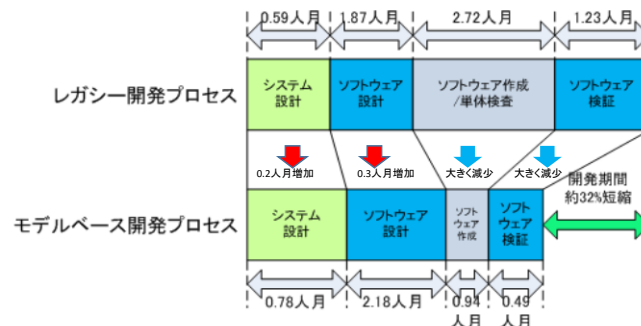


図 32 モデルベース開発とレガシー開発の工数(再掲)

表 19 全体工数削減率

レガシー開発期間	6.41 人月
モデルベース開発期間	4.39 人月
削減率	約 32%

* 出典:「モデルベース開発ツールを活用した際のコストの効果検証 実施報告書」IPA 独立行政法人 情報処理推進機構 2013.2

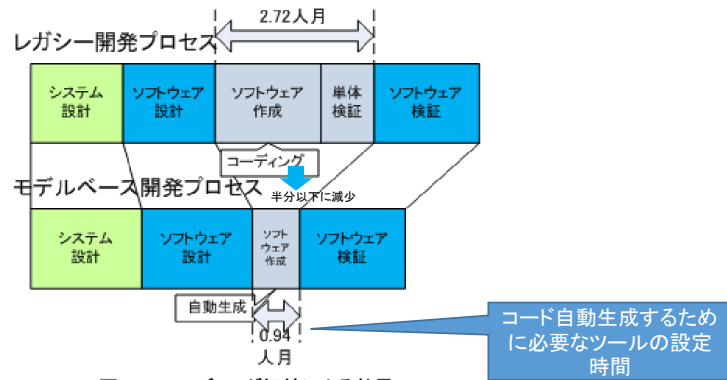
表 22 適用レベル 3 の工数削減率

※ ○は適用、×は非適用 適合・評価プロセスは実験で実施していないため - とする

適用レベル	適用プロセス		評価(削減率)	
L3	システム設計	○	-3%	全体 32%
	ソフトウェア設計	○	-5%	
	ソフトウェア作成	○	28%	
	ソフトウェア検証	○	12%	
	適合・評価	-	-	

* 出典:「モデルベース開発ツールを活用した際のコストの効果検証 実施報告書」IPA 独立行政法人 情報処理推進機構 2013.2

No.	MBD開発の利点	
2	コーディングの短縮	<ul style="list-style-type: none"> 自動コード生成によりコードを作成する工数がない ヒューマンエラーが入り込まない



プロセス	コーディングの工数
レガシー開発	2.72人月
モデルベース開発	0.94人月 (ツール設定)

* 出典:「モデルベース開発ツールを活用した際のコストの効果検証 実施報告書」IPA 独立行政法人 情報処理推進機構 2013.2

No.	MBD開発の利点	
3	レビュー時間の短縮	作成するモデルは抽象的な図で表現されコードよりも見やすい。レビュー者にも理解しやすい

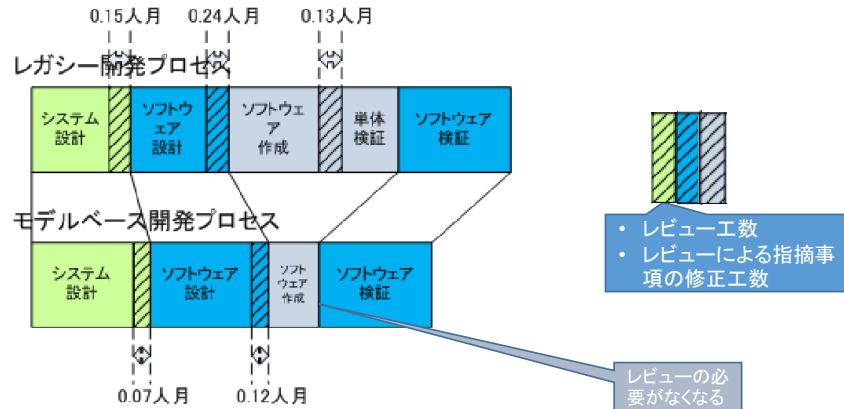
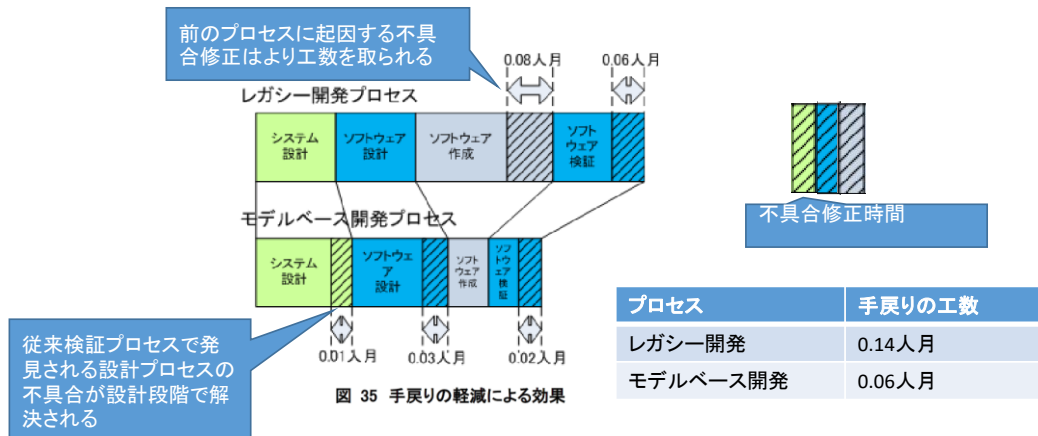


表 20 レビュー総工数

プロセス	レビューに掛かる全体工数
レガシー開発	0.52 人月
モデルベース開発	0.19 人月

* 出典:「モデルベース開発ツールを活用した際のコストの効果検証 実施報告書」IPA 独立行政法人 情報処理推進機構 2013.2

No.	MBD開発の利点	
4	手戻りの軽減	PC上でモデルを使ってシミュレートし検証を行いながら設計するため、システム設計段階でプラントモデルを作成する必要があり、ソフトウェア設計ではプラントモデルを使ったシミュレーションが必要になる。しかしながら、アルゴリズムの不具合をコーディング前に発見することができ単体検証時の手戻りを軽減することができる



* 出典:「モデルベース開発ツールを活用した際のコストの効果検証 実施報告書」IPA 独立行政法人 情報処理推進機構 2013.2

以下に示す別紙に、様々な企業、分野における MBD／ACG 適用の事例について情報をまとめる。

番号	資料名
[A1]	MBD 導入効果例.pdf

B. Toolbox モデルケース

以下に示す別紙に、用途別ツール構成例、及びツール別の適用シーンの例をリスト化した。用途別ツール構成例では、適用を検討している業務に対し、必要となる Toolbox の組み合わせ例を示し、導入するツール構成の目安として用いることを想定している。また、ツール別適用シーン例では、各 Toolbox の持つ機能概要と、その Toolbox がどういった業務に活用できるかを表に整理した。

番号	資料名
[B1]	用途別 ツール構成.xlsx
[B2]	ツール別 適用シーン.xlsx
[B3]	ARP4754A_DO178C_DO331 用 MathWorks ツールチェーン.pdf

C. ACG 使用ガイドライン

本項では、ACG を使用するにあたり、遵守する必要がある項目をまとめている。ガイドラインを遵守することによって、コード生成時のエラーの低減や生成コードの効率向上などの効果が見込める。

ガイドライン遵守のための作業概要は以下の通りである。

1. 規約に準拠したモデルの作成
コード生成及び検証時にエラーとなることやコードが意図しない挙動となることを防止するために、規約に準拠したモデルを作成する。
2. モデルの設定
モデルの設定を変更し、Embedded Coder でコードを生成できるようにする。
3. コードの生成
Embedded Coder を使ってモデルからコードを生成する。
4. コードの検証
生成したコードがモデルと等価であるか、MISRA C 等のコーディングスタンダードに準拠しているかどうか等を検証する。

以下、各作業の内容を説明する。

1. 規約に準拠したモデルの作成
高信頼性のソフトウェアをモデルによって作成する場合、以下のモデリング規約を参考にモデルを作成する必要がある。

- (1) JMAAB モデリングガイドライン (CONTROL ALGORITHM MODELING GUIDELINES USING MATLAB®, Simulink®, and Stateflow®)

自動車用制御装置のコントローラモデルを運用する上で、作成者と使用者の間で容易に共通の理解が得られるように、Simulink / Stateflow モデルの記述について重要な基本的なルールを規定したものである。

JMAAB モデリングガイドラインに準拠しているかどうかは、Simulink Check にてチェックが可能である。

- (2) MBAC モデリング規約[C1]

MBAC モデリング規約 WG にて作成したモデリング規約である。JMAAB モデリングガイドライン パージョン 5.1 を基に、テラリングを実施した規約である。

また、JMAAB モデリングガイドラインと MBAC モデリング規約の差異を別資料[C3]にてまとめているので、必要に応じて参照されたい。

- (3) Embedded Coder コード生成制約

Embedded Coder を使用してモデルを生成する場合に必要な、モデルに対する制約である。以下のページにてルールがまとめられている。

<https://jp.mathworks.com/help/rtw/ug/supported-products-and-block-usage.html>

(4) DO-178/DO-331 準拠のためのガイドライン

DO-178 及び DO-331 で要求される、完全性、明確性、決定性を満たしたモデルであるかどうかをチェックするためのガイドラインである。以下のページにてルールがまとめられているとともに、Simulink Check にてチェックが可能である。

https://jp.mathworks.com/help/slcheck/ug/model-checks-for-do-178cdo-331-standard-compliance_bup4bx5.html

(5) Simulink Code Inspector Compatibility Check

ソースコードとモデルの等価性を確認できるツールである「Simulink Code Inspector」が使用できるモデルであるかをチェックするためのガイドラインである。Simulink Check にてチェックが可能である。

これら 5 つのガイドラインは包含関係となっている。大まかな関係性を図 A-1 に示す。なお、図内の円内に入るモデルが、それぞれの規約に準拠したモデルである。

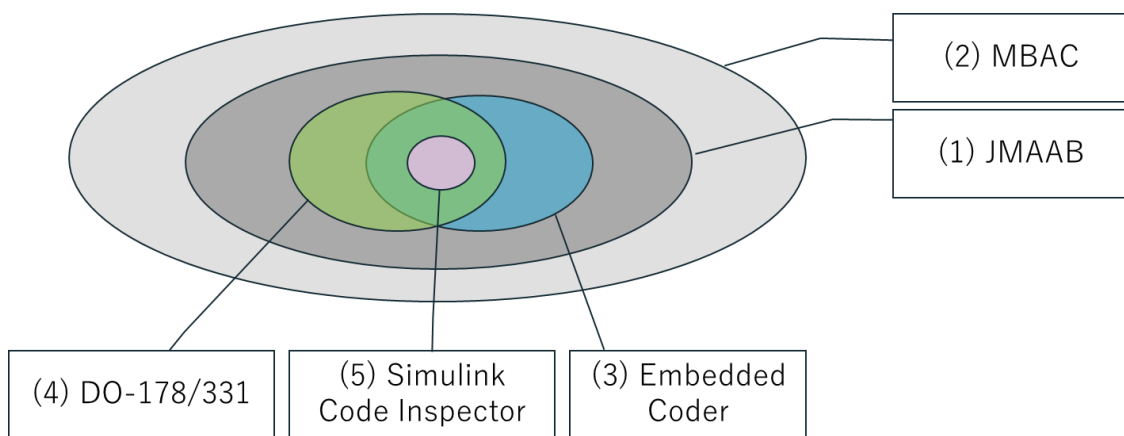


図 C-1 モデリング規約の包含関係

図より、Simulink Code Inspector Compatibility Check が一番厳しい規約となっている。最初から Simulink Code Inspector Compatibility Check に準拠するモデルを作成することは慣れが必要であるため、一番規約の緩い MBAC のモデリング規約から順番に規約を満たしていくことが重要であると考え。

2. モデルの設定

Embedded Coder にて意図したとおりにコードが生成できるように、モデルの設定を変更する。モデルの設定方法は、別資料[C2]を参照のこと。

3. コードの生成

Embedded Coder を使用して、モデルからコードを生成する。

4. コードの検証

3.にて生成したコードを、以下のツールを使用して検証する。詳細は別資料[C4]、[C5]、[C6]を参照のこと。

(1) Simulink Code Inspector

モデルとコードが等価であることを検証する。

(2) Polyspace Bug Finder

MISRA C に準拠していることを検証する。

(3) Polyspace Code Prover

オーバーフロー、NULL 参照等の実行時エラーがないことを検証する。

(4) Simulink Test

Back-to-Back Test により、ソースコードの振る舞いがモデルと同等であることを検証する。

なお、上記に挙げたツールは、ツールが正しく動作することを保証するための仕組みである Tool Qualification 認証を受けている。検証ツールとしての認証を持つツールを利用して、モデルとコードの等価性を立証する。

番号	資料名
[C1]	MBAC モデル作成規約
[C2]	MBD-P-003GeneratingSourceCode.pdf
[C3]	JMAAB-MBAC 対応表(サブ ID 対応版).xlsx
[C4]	Simulink Code Inspector によるモデルと自動生成コードの等価性検査手順
[C5]	ソフトウェアバグを根絶する C/C++ 静的コード解析
[C6]	Simulink Test による PIL Back2Back テスト手順

D. 教育コンテンツ情報

以下に示す別紙に、MBD に関する学習、スキルアップに有効な、教育、トレーニング、動画等の情報を整理した。

番号	資料名
[D1]	教育コンテンツ.xlsx

E. その他の参考資料

番号	資料名
[E1]	<u>MBD 導入お役立ち情報</u>
[E2]	<u>IPA MBD コスト効果検証</u>
[E3]	<u>IPA 組み込み動向 2018</u>
[E4]	DO-331