ホワイトペーパー

大規模組織向け MATLAB および Simulink のバージョン アップグレード

著者: Judy Wohletz、Vinod Reddy、および Jim Ross、MathWorks

MathWorks®

目次

1 はじめに

これまで開発チームは、新しい開発プロジェクトに着手する前に各ツールを新規バージョンにアップグレードし、複数年にわたる製品開発のライフサイクルが完了するまで同じバージョンを使用してきました。しかし、企業がこれまでにないスピードで新しい技術を導入していることから、製品の開発中にアップグレードや新しい技術の調査を行うことが困難になっています。プロアクティブなアップグレード戦略を採用することで、組織はバージョンを変更するリスクと新しい技術や手法を取り入れることができないリスクのバランスを取ることができます。

MATLAB および Simulink を新規リリースにアップグレードすることで、モデルベースデザイン (MBD、モデルベース開発) の最新技術を組織で利用できるようになり、生産性と成果の向上につながります。アップグレードの純利益は、投資収益率 (ROI) として測定することができます。収益は、生産性の向上とコストやリスクの削減を含めて計算されます。投資は、アップグレードの評価と展開のためのコストに加え、アップグレードの新機能を有効活用するために必要なプロセスやツールの変更のためのコストとして測定されます。

アップグレードが成功するかどうかは、それを実装するために用いる戦略に左右されます。このホワイトペーパーでは、最新の MATLAB および Simulink 製品を最大限に活用し、利益達成に関連するコスト、リスク、および混乱を最小限に抑えることを目的とした企業全体のアップグレードのための体系的なアプローチについて、MathWorks からの推奨事項をご紹介します。これらの推奨事項は、MATLAB および Simulink の新規バージョンへのアップグレードをお客様にご案内した際の実際の経験に基づいています。

アップグレードのプロセスは、主に 6 つのフェーズで構成されています。

1. 評価
2. 計画
3. 移行
4. テスト
5. リリース
6. サポート

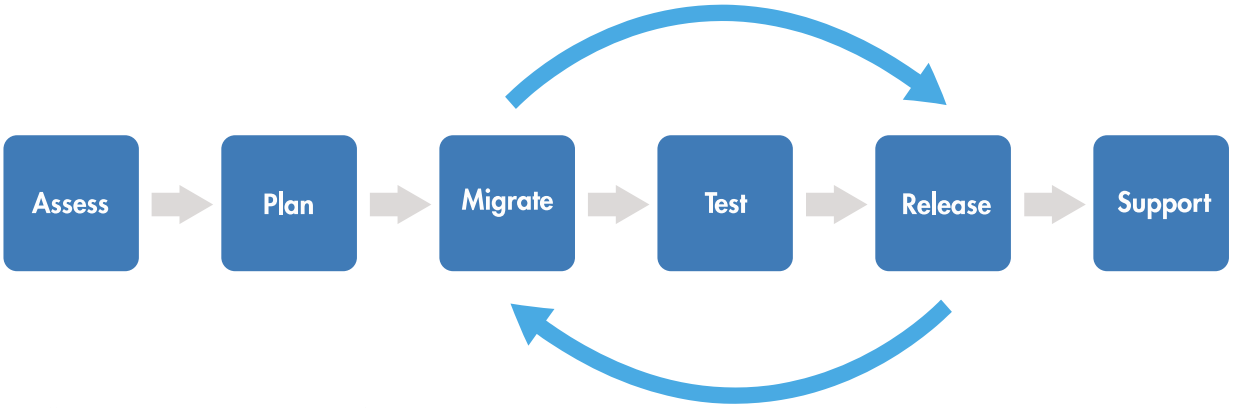
プロセスのどのフェーズにおいても、最も重要な手順はコミュニケーションです。バージョンアップグレードの間、すべての利害関係者に十分に情報を行きわたらせることが重要です。評価、結果、およびアップグレード決定の主な理由を早い時期に通知することで、ユーザーはあらかじめ準備をしておくことができます。アップグレードの意思決定がなされたら、組織的な計画を迅速に伝達して、各所で計画を策定できるようにする必要があります。チームがアップグレードに取り掛かる際に、タイムライン、成功例、および問題点が伝達されていると、チームの成功が促進されます。

効率的で堅牢なアップグレード プロセスのための最終的な考慮事項は、継続的インテグレーション (CI) の使用です。開発ワークフローに CI 環境を使用すると、再現性が向上し、テストの実行に必要な作業を減らすことができます。また、CI 環境の使用により、日々の業務にも効果をもたらし、アップグレードの評価に必要な作業を大幅に削減することができます。CI が既に開発ワークフローに組み込まれている場合は、少しの変更を加えるだけで、特定の MATLAB バージョンのテストを行う柔軟性が得られる可能性があります。CI が現在の開発ワークフローに組み込まれていない場合は、CI に取り組むことをお勧めします。

こうした取り組みを組み合わせて、リスクを最小限に抑え、アップグレードの成功の可能性を最大限に高めることができます。

2 概要

ROI を最大化するには、アップグレードを行う前に、アップグレードのプロセス、考えられるアップグレードパス、さらに最も重要なこととして、見込まれる利益と付随するコストを把握することが必要です。新規バージョンへのアップグレードに必要なタスクやアクティビティは、複数の論理的なフェーズにグループ化することができます。具体的には、評価、計画、移行、テスト、リリース、およびサポートのフェーズです。



一般的なアップグレード プロジェクトにおける各フェーズのワークフロー。

プロセスの各フェーズには、トリガー、インプット、アクティビティ、アウトプット、および終了基準があります。このプロセス自体を支えるのが、持続戦略です。対象を指定したコミュニケーション計画を使用し、アップグレードの意図、状況、決定、タイムライン、およびアップグレード中に確認された重要な問題について、すべての利害関係者にもれなく情報を提供することにより、このワークフロー全体をサポートする必要があります。

2.1 評価

アップグレード プロセスの最初のフェーズの目的は、アップグレードの全体的な効果を理解し、利益がコスト、リスク、および労力を上回るものかどうかを評価することです。ここでの目的は、中断を招く問題が存在するかどうかを早期に判断し、その特定のバージョンを評価する際に追加の労力を費やさずに済むようにすることです。アップグレード プロジェクトの次のフェーズに進む前に、十分な ROI が必要です。

評価 目標: アップグレードの影響を理解する	
トリガー	1. オペレーティング システムのサポート期限 2. MATLAB および Simulink の新機能 (全く新しい製品や機能を含む) 3. 新しいハードウェア開発プラットフォーム 4. OEM/ベンダー/パートナーのコラボレーション 5. 新しい開発プロジェクト
インプット	1. アップグレード予定の現在の開発プロジェクトの状況 2. 現在のバージョンとアップグレード バージョンの間で MATLAB および Simulink が何回リリースされているか 3. MATLAB および Simulink の現在のバージョン、サードパーティのソフトウェアおよびハードウェア、開発またはアプリケーションのソフトウェアおよびハードウェア、コンピューターおよびオペレーティング システムなどの開発プラットフォーム 4. アップグレードのタイムライン

アクティビティ	1. 初期の対象バージョンを選択 2. 現在の問題と、新規バージョンでの対応が期待される要件の一覧を作成 3. アップグレード アドバイザーを使用して、モデル、コード、およびスクリプトの少量の代表的なサンプルを移行するチームを特定 4. モデル、コード、およびスクリプトの少量の代表的なサンプルを移行し、バージョン移行の効果をより深く理解 5. ユーザーが一般的に実行するタスクに対して回帰テストを実行し、新規バージョンでのタスクにかかる時間に関するメトリクスを収集 6. このチームにアップグレード プロジェクトを確実に実行する能力があるか、追加の技術的なトレーニングが必要かどうかを評価 7. チームがアップグレード プロジェクトに対処するのに十分なリソースを有しているかどうかを評価 8. 以下が記載されたアップグレード評価レポートを準備 • アップグレードの手順 • 利益や、シミュレーション速度、コードのサイズなどのメトリクスを含む結果 • 課題 • エラーメッセージ • 現時点で把握されている見込み利益、推定コスト、およびリスクに基づく ROI 予測
アウトプット	1. 実行/中止の判断 2. 初期の対象バージョン 3. アップグレード評価レポート

2.2 計画

計画フェーズの目的は、アップグレード プロジェクト全体の範囲と計画を定義することです。

計画 目標: アップグレード プロジェクト全体の範囲を定義する	
トリガー	実行の決定
インプット	アップグレード評価レポート
アクティビティ	1. ビジネスケースの作成 2. 影響を受けるモデル、開発プロジェクト、組織、および主要なステークホルダーを特定し、範囲を確定
アウトプット	1. アップグレードのためのビジネスケース 2. アップグレードのタイムライン、影響を受ける開発プロジェクト、、モデル、組織、およびステークホルダーの一覧、必要なリソース、トレーニング、労力、コスト、およびリスクの見積もり、依存関係、およびハードウェアとソフトウェアの対象バージョンを含む、アップグレード計画

2.3 移行、テスト、およびリリース

移行、テスト、およびリリースの各フェーズは密接に関連し、モデルを新規リリースにアップグレードするために反復的に行われます。

2.3.1 移行

移行フェーズでは、Simulink モデル、MATLAB コード、MATLAB アプリ、カスタムツールを反復的にアップグレードして、新規リリースに対応させます。ここで言うカスタムツールとは、MATLAB および Simulink を使用した環境で作成されたアプリケーションや設定を指します。カスタムツールには、次のものが含まれます。

- カスタム Simulink ブロックライブラリ
- S-Function ブロック
- MATLAB 起動スクリプト
- Embedded Coder ディクショナリのカスタム ストレージ クラス
- カスタムのシステム ターゲット ファイル
- MATLAB Report Generator または Simulink Report Generator セットアップファイル
- カスタムのモデル アドバイザー チェック
- モデリング スタイル ガイドライン
- 既定のコンフィギュレーション パラメーター設定
- MATLAB および Simulink 環境のカスタマイズに使用される MATLAB スクリプト

移行フェーズは、3 つのサブフェーズを複数回繰り返して実行されます。ここでのサブフェーズとは、初期移行、カスタムツール移行、および移行の自動化です。このような反復作業では、通常は自動と手動を組み合わせた、モデル、スクリプト、ユーザー インターフェイス、テンプレート、サードパーティツールとのインターフェイス、およびその他のアプリケーションをアップグレードするためのアプローチが必要になります。

このフェーズでのアクティビティでは、新規バージョンへのモデルの移行のみを行います。以下のセクションで説明するプロセスのテストフェーズでも、引き続きモデルのテストが必要です。

2.3.1.1 初期移行

初期移行のサブフェーズでは、手順やツールを開発し、新機能の理解を深めることを目的として、組織内で開発された代表的なモデルを表した、少数の Simulink モデルと MATLAB プロジェクトファイルのセットを移行します。この作業は、現在のバージョンでのアップグレードとモデルの改善を支援するために設計された*アップグレード アドバイザー*というツールを使用して行います。このサブフェーズにより、次のサブフェーズがサポートされ、リスクと労力が軽減されます。

初期移行 目標: モデルのサブセットを移行する	
トリガー	承認されたビジネスケース
インプット	1. アップグレード評価レポート 2. アップグレード計画 3. モデルおよびその他の関連するアーティファクト

アクティビティ	1. モデルに対してアップグレード アドバイザーを実行 2. モデルのブロック線図更新、シミュレーション、およびコード生成を正常に実行 3. Simulink で診断ビューアーを使用して、エラーメッセージおよび警告メッセージを文書化 4. エラーがある場合は解決 5. 警告メッセージを比較して、新規バージョンに特有のものか、現行バージョンに存在するものかを判断 6. 警告メッセージの重大度を評価 7. 警告メッセージを解決する必要があるかどうかを判断 8. その他のすべての問題を「アップグレード問題リスト」に文書化
アウトプット	1. 更新済みアップグレード評価レポート 2. 新規アップグレード問題リスト 3. 更新済みアップグレード計画 4. 新規バージョンに移行したモデルのサブセット

2.3.1.2 カスタムツールの移行

カスタムツールの移行サブフェーズでは、カスタムツールの移行に加え、組み込み機能を採用してカスタムツール機能の使用を終了することも重要です。ここでの長期的な目標は、組み込み機能を採用することで、カスタムツールをなくすことです。これにより、将来のアップグレードがより簡単になり、自動化を促進することができます。将来のアップグレードにかかる労力やコストも実質的に軽減されます。

カスタムツールの移行 目標: モデルとカスタムツールをアップグレードして、自動化ツールを開発する	
トリガー	1. 承認済みアップグレード計画 2. 移行済みモデルのサブセット
インプット	1. アップグレード評価レポート 2. アップグレード問題リスト 3. モデルおよびその他の関連するアーティファクト 4. カスタムツール
アクティビティ	1. カスタムツールを新規バージョンにアップグレード 2. 可能な場合、カスタムツールを MATLAB および Simulink の組み込み機能に置き換え 3. 新規バージョンにおけるカスタムツールの妥当性を確認 4. アップグレード プロセスの手動による手順を自動化することで、労力を削減し整合性を確保する自動化ツールを開発
アウトプット	1. 更新済みアップグレード評価レポート 2. 更新済みアップグレード問題リスト 3. 更新済みアップグレード計画 4. アップグレード済みカスタムツール 5. 自動化ツール

2.3.1.3 移行の自動化

移行の自動化サブフェーズでは、前のサブフェーズで策定した手順と自動化ツールを使用して、残りのモデルをアップグレードします。このフェーズの主な 2 つの目的は、移行に関する問題を解決し、自動化ツールを改良して他のチームでも使用できるようにすることです。

移行の自動化 目標: モデル、カスタムツール、自動化ツールに関連する既知の問題をし、モデル、カスタムツール、自動化ツールをアップグレードする	
トリガー	1. モデルとカスタムツールがアップグレード済み 2. 自動化ツールが開発済み
インプット	1. 自動化ツール 2. アップグレード済みカスタムツール 3. 残りのモデルおよびその他の関連するアーティファクト 4. アップグレード問題リスト
アクティビティ	1. 残りのすべてのモデルで、自動化ツールをアップグレード アドバイザーとともに実行(自動化ツールがアップグレード アドバイザーを呼び出します) 2. モデルのブロック線図更新、シミュレーション、およびコード生成を正常に実行 3. Simulink で診断ビューアーを使用して、エラーメッセージおよび警告メッセージを文書化 4. エラーがある場合は解決 5. 移行の妨げとなる警告を解決 6. すべての問題を「アップグレード問題リスト」に文書化 7. カスタムツールおよび自動化ツール用ヘルプドキュメントを更新 8. カスタムツールおよび自動化ツール用リリースノートを作成 9. カスタムツールおよび自動化ツールの改善により問題を解決 10. 新旧バージョンの相違点、カスタムツール、および自動化ツールに関連するトレーニング教材を作成または更新
アウトプット	1. 改善済み自動化ツール 2. 更新済みアップグレード評価レポート 3. 更新済みアップグレード問題リスト 4. 更新済みアップグレード計画 5. 新規バージョンにアップグレードされたモデル 6. カスタムツールおよび自動化ツール用リリースノート 7. カスタムツールおよび自動化ツール用更新済みヘルプドキュメント 8. 新規または更新済みトレーニング教材

2.3.2 テスト

テストフェーズの目的は、新規バージョンを使用してモデルとコードで生成された結果が、旧バージョンを使用して生成された結果と、許容範囲内で機能的および数値的に等価であるかを確認することです。

テスト 目標: アップグレードモデルの妥当性を確認する	
トリガー	モデルがアップグレード済み
インプット	1. アップグレード済みカスタムツール 2. モデルおよびその他の関連するアーティファクト 3. 入力テストデータおよび期待される出力
アクティビティ	1. ユーザーがよく実行するタスクに対する回帰テストを実行し、結果を旧バージョンと比較 2. オープンループ、モデルインザループ (MIL)、ソフトウェアインザループ (SIL)、およびプロセッサインザループ (PIL) の各シミュレーションをデスクトップで実行 3. ラボでのハードウェアインザループ (HIL) シミュレーション、ラピッド プロトタイピング、およびオンターゲット ラピッド プロトタイピングを実行 4. 通常実行されるシステムテストやハードウェアテストを実行 5. 結果を確認して妥当性確認済みモデルを承認 6. 必要に応じて追加のテストデータおよび期待される出力を定義
アウトプット	1. 新規バージョンで妥当性を確認したモデル 2. 追加のテストデータおよび期待される出力 (必要な場合) 3. テストレポートおよび関連するアーティファクト

2.3.3 リリース

リリースフェーズでは、アップグレード済みモデル、カスタムツール、MATLAB および Simulink の新規バージョンをリリースします。

リリース 目標: MATLAB の新規バージョンとカスタムツールをリリースする	
トリガー	テストフェーズ完了
インプット	1. 妥当性確認済みモデル 2. 入力テストデータおよび期待される出力 3. カスタムツール用ヘルプドキュメント 4. カスタムツール用リリースノート 5. アップグレード計画
アクティビティ	1. アップグレード計画を更新 2. トレーニングクラスをスケジュールおよび実施 3. すべての資料のリポジトリを作成 4. ユーザーに通知
アウトプット	1. 更新済みアップグレード計画 2. リリース済みバージョンのドキュメント、ツール、およびトレーニング

2.4 サポート

サポートフェーズで組織は、モデル、カスタムツール、および MATLAB と Simulink 製品のユーザー向けに、問題発生時の継続的なサポートを行います。

サポート 目標: 報告された問題を解決する	
トリガー	ユーザーが問題を特定
インプット	1. カスタムツール 2. 問題のあるモデル 3. ヘルプ ドキュメンテーション 4. トレーニング教材
アクティビティ	1. 問題を再現 2. 根本原因を特定 3. 問題を修正 (モデル、カスタムツール、またはその他の場所) 4. 新規バージョンをユーザーに通知 5. リポジトリを更新 6. 問題追跡リストを更新 7. カスタムツール用リリースノートを更新 (必要な場合) 8. 修正を加えた更新済みバージョンをリリース (必要な場合)
アウトプット	1. 更新済みリリースノート 2. モデルやカスタムツールの修正済みバージョン 3. 更新済み問題追跡リスト

3 アップグレードの詳細なワークフロー

本章では、第 2 章で定義した 6 つのプロセスフェーズを復習しながら、各手順の詳細を説明します。

3.1 評価

評価フェーズの目的は、アップグレードの全体的な効果を理解し、アップグレードの利益がコスト、リスク、および労力を上回るかどうかを判断することです。ここでの目的は、中断を招く問題が存在するかどうかを早期に判断し、その特定のバージョンを評価する際に追加の労力を費やさずに済むようにすることです。アップグレードの ROI が十分でない場合、組織は次のバージョンへのアップグレードの延期を決定することがあります。アップグレードが延期されるたびに、将来のバージョンにアップグレードするための障壁が増えます。アップグレード プロセスは、新しい開発プロジェクト、新しい企業との共同作業、新しいハードウェア、新しいオペレーティング システムなど、数多くのイベントがトリガーになる可能性があります。

3.1.1 アップグレード プロジェクト開始時の考慮事項

アップグレード プロセスの最初の手順は、現状の評価です。この評価には、以下の手順の大半またはすべてが含まれます。

- 現在、会社で使用しているソフトウェアとハードウェアの一覧表を作成する
- 現在、会社および共同作業先の会社で使用している MATLAB および Simulink の現在のバージョンを文書で記録する
- MATLAB および Simulink 製品で使用しているサードパーティのソフトウェアおよびハードウェアのツールを文書で記録する

- サードパーティ ベンダーがサポート予定の MATLAB および Simulink 製品のバージョンを確認する
- 社内でサポートしている、また将来サポートする予定のコンピューターとオペレーティング システムを特定する
- 今回のアップグレードで影響を受ける社内の主要なステークホルダーと協力し、アップグレードの大まかなタイムラインへの合意を得る
- 以前のアップグレードでの問題を確認する

3.1.2 対象となる MATLAB および Simulink のバージョンの選択

アップグレードする MATLAB および Simulink のバージョンを決定する前に、特定のバージョンの新機能に関する *リリースノート*、既知の問題に関する *バグレポート*、*システム要件*、および *サポートされているコンパイラ* を絞り込んで検索します。また、組織の状況に該当する場合は、*製品別のプラットフォームの可用性* と *MATLAB および Simulink 製品を実行するためのコンピューターの選択* も確認します。対象とする MATLAB および Simulink のバージョンを選択する際にもう一つ考慮が必要なことはアップデートです。MATLAB ウィンドウの右上にあるベルのアイコンをクリックすると、アップデートにアクセスできます。MathWorks では、最新のアップデートが利用できる場合は、組織や開発プロジェクトで実行可能になり次第すぐに使用することを推奨しています。サードパーティのソフトウェアを使用している場合は、ベンダーに連絡し、現在サポート対象および今後サポート予定の MATLAB と Simulink のバージョンとアップデートを確認します。また、特にリリース前の MATLAB および Simulink バージョンを対象にする場合は、MATLAB および Simulink の新規バージョンやアップデートがリリースされた後のサポートのタイミングを把握することも重要です。

アップグレードプロセスの途中で、対象とする MATLAB および Simulink のバージョンを変更する可能性があることに留意してください。予期せぬ問題が発生する可能性や、MathWorks がお客様の組織にとって利益のある新機能をリリースする可能性があります。プロセスの早い段階で特定のバージョンに固執するのではなく、**自社の状況に応じて ROI が最も高くなるバージョンを選択する柔軟性を維持します。** アップグレードの頻度が 1 年を超える場合は、新機能を利用できるまで何年も待つのではなく、対象バージョンを変更することで、その利益を得られるように準備しておく必要があります。

3.1.3 技術環境の整備

次の手順では、アップグレードの影響を受ける開発プロジェクトに対して、社内で使用しているコンピューター、オペレーティング システム、MATLAB および Simulink、開発用ソフトウェアおよびハードウェア、サードパーティ製ソフトウェアおよびハードウェアを利用できるかどうかを確認します。新規バージョンでモデルを保存すると、元のバージョンで開くことが難しくなるため、評価および移行作業のために専用のサンドボックスを確保します。最後に、新規バージョンでは、適切な結果を得るために追加の設定を検討する必要があります。

サンドボックスとは、リポジトリ内の 1 つ以上の新しい専用ブランチのことで、次のことを可能にします。

- 複数のユーザーまたはチーム間での共同作業
- ツールまたはライブラリへのアップデートを共有
- 既存の CI ワークフローを簡単に使用
- アップグレードの一環として開発された、新規または拡張された CI ワークフローの再利用を簡易化

評価フェーズでの初期テストに使用するサンドボックスでは、その後の移行、テスト、およびリリースフェーズで同一のアプローチをより大規模に適用できるように拡張可能であることが重要です。評価フェーズでのテストは最小限にすることを意図しているため、簡単に解決できない問題のチェック、警告の理解、パフォーマンスの問題の特定に重点を置く必要があります。移行フェーズやテストフェーズなどの後期のフェーズでは、より完全なテストを行うことをお勧めします。

3.1.4 初期テスト

技術的な環境が整い、アップグレード基準が確立されたら、旧バージョンから対象バージョンにアップグレードするモデルを 1 つ選択します。社内で使用しているモデルの大半を代表するような大規模モデルで、一般的なモデリングパターンを含み、会社のモデリングスタイルに合ったものを選ぶことを推奨します。

モデルの選択後、以下の手順で初期テストを行います。このプロセスでは通常、いくつかの手順を手動で行う必要がありますが、対象となる MATLAB および Simulink バージョンにアップグレードする場合の問題を特定するためにはこれが不可欠です。モデル、カスタムライブラリ、およびカスタムツールは、その後のテストの妨げとなる可能性のあるエラーメッセージの排除に必要な最小限の修正のみを適用して、「現状のままで」テストします。この評価フェーズの目的が、より大規模な移行を成功させるために事前に解決しておかなければならない問題を特定すると同時に、アップグレードの実現可能性をすばやく解析することである点に留意してください。

評価フェーズで推奨されるアクティビティは、以下のとおりです。

1. 現在のバージョンのテスト
    a. 現在のバージョンの MATLAB および Simulink を開く
    b. MATLAB パスにカスタムライブラリやカスタムツールを追加する
    c. 現在のバージョンの MATLAB および Simulink で Simulink モデルまたは MATLAB プロジェクトファイルを開くSimulink モデル コンフィギュレーションの推奨設定を使用する
    d. ブロック線図を更新する
    e. モデルをシミュレートする
    f. エラーメッセージをすべて解決し、警告に対する適切なアクション (ある場合) を判断する
2. 対象となるバージョンのテスト
    a. 対象となるバージョンの MATLAB および Simulink を開く
    b. MATLAB パスにカスタムライブラリやカスタムツールを追加する
    c. 対象となるバージョンの MATLAB および Simulink で Simulink モデルまたは MATLAB プロジェクトファイルを開く
    d. モデルに対してアップグレード アドバイザーを実行する
    e. 警告メッセージやエラーメッセージを記録する
        i. エラーメッセージを修正する
        ii. 追加のテストの妨げになる場合のみ、警告メッセージを修正する
        iii. 発生した問題をすべて文書化し、追跡する (他のモデルでも同じ問題が発生する可能性あり)
    f. 対象となるバージョンの MATLAB および Simulink をリリースする前に問題を解決する

選択したモデルを MATLAB および Simulink の対象バージョンにアップグレードする際に解決できない問題が発生した場合は、中間のバージョンでモデルをテストします。古いバージョンからアップグレードする場合は、MATLAB および Simulink の中間のバージョンにアップグレードしなければならない場合もあります。非互換性に関する警告やエラーメッセージを確実に確認するには、この追加手順が必要になる場合があります。アップグレードの間隔が、警告やエラーメッセージが段階的に廃止される時間枠よりも長い場合、問題の根本的な原因のデバッグに役立つ重要なメッセージを見逃す可能性があります。この方法は、デバッグや問題の分離には有効ですが、通常、中間のバージョンは、新しい対象バージョンに向かう途中の一時的な停留所のようなものにすぎません。推奨される方法は、現在の MATLAB および Simulink バージョンと対象となる MATLAB および Simulink バージョンの中間を選択することです。

3.1.5 回帰テスト

ユーザーがよく行う作業に対し回帰テストを行い、MATLAB および Simulink の新規バージョンでこれらの作業にかかる時間を大まかに把握します。可能な限り、以前のバージョンで同じタスクを実行し、テスト結果を比較します。このテストは、現在使用中のいくつかの大規模モデルで実行することをお勧めします。このアクティビティは、新規バージョンをリリースする前に潜在的な問題を検出するのに役立ちます。

評価フェーズでは、MATLAB および Simulink の特定のバージョンがアップグレード先の現実的な候補であるかどうかを回帰テストですばやく確認します。回帰テストは、予定されているアップグレード先のバージョンにかかわりなく、プレリリースごとにすばやく実行することが理想的です。次の表は、評価フェーズで MATLAB および Simulink の特定の

バージョンを評価する際にチェックする必要のある主な項目の一覧です。これは網羅的なリストではなく、特徴的な重要項目をまとめた代表的なリストです。評価フェーズで評価すべき最も重要な側面は、パフォーマンスと、解決できない致命的な問題の有無です。回帰テストに関するその他の情報については、テストのフェーズで説明します。

結果	理想	メモ
シミュレーション	シミュレーション エラーなし	モデルのシミュレーションでエラーが発生しても、解決可能であること。
コード生成	コード生成エラーなし	モデルのコード生成でエラーが発生しても、解決可能であること。
シミュレーション時間	以前のバージョンよりも高速	若干のシミュレーション時間の増加は許容できるが、標準として時間短縮を維持すること。
コード生成時間	以前のバージョンよりも高速	若干のコード生成時間の増加は許容できるが、標準として時間短縮を維持すること。

3.1.6 アップグレードの決定

回帰テストを完了し、テスト結果をアップグレード基準に照らして評価した後、アップグレード プロセスの次のフェーズに進むかアップグレードを後日に延期するかを決定できます。

3.2 計画

計画フェーズの目的は、アップグレード プロジェクトの全体的な範囲と計画を定義することです。これは、移行、テスト、リリース、およびサポートの各フェーズなど、プロセスの残りのフェーズを網羅するのに必要なものです。

3.2.1 アップグレード プロジェクトの管理

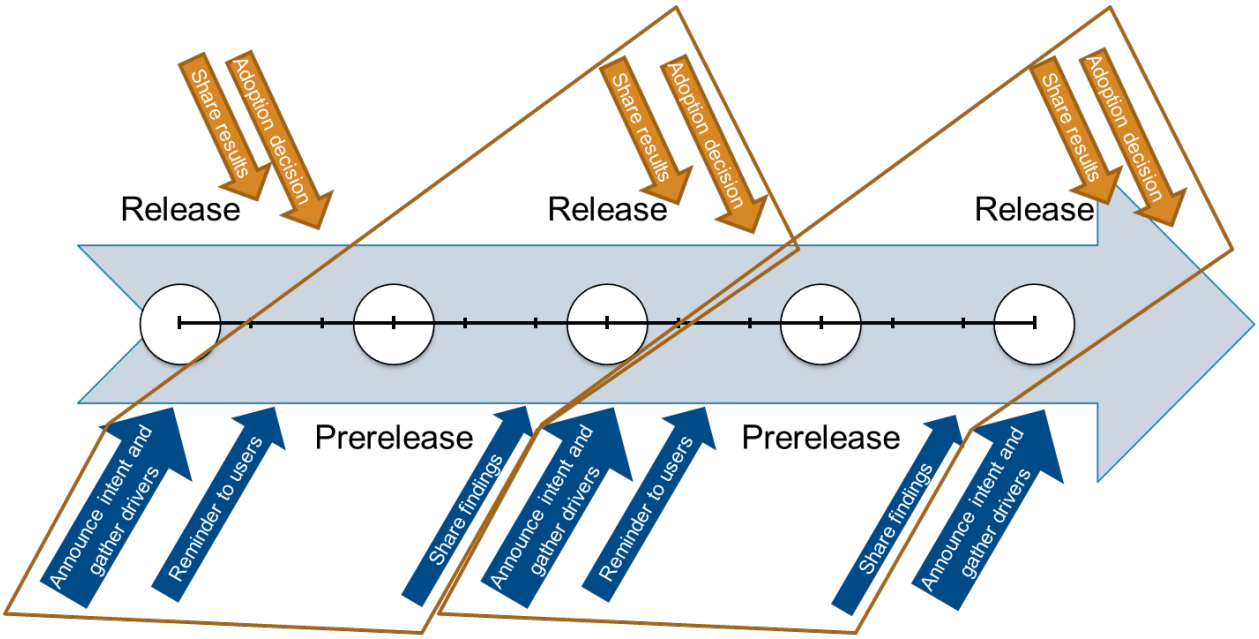
アップグレードの目標期日や、対象となる MATLAB および Simulink のバージョンなどのアップグレードの全体的な計画を作成します。この計画では、各フェーズおよび各フェーズの終了基準を満たすために必要なアクティビティを明確に定義します。また、各チームメンバーの役割とその役割に期待することや、アップグレード プロセスに関与するさまざまなチームを定義します。代表的な役割には、アップグレード リーダー、エンジニアリング チーム、経営陣スポンサー、IT 担当、開発プロジェクト マネージャー、ツールチーム、マネージャー、サードパーティ ベンダー、MathWorks コンサルタントなどがあります。詳細については、以下のスイムレーン図を参照してください。

フェーズ	アップグレード リーダー	エンジニアリング チーム	経営陣スポンサー	IT	開発プロジェクト マネージャー	ツールチーム
評価	主導	情報共有	情報共有	サポート	情報共有	貢献
計画	主導	貢献	貢献	貢献	貢献	貢献
移行	主導	貢献	情報共有	サポート	情報共有	貢献
テスト	主導	貢献	情報共有	情報共有	情報共有	貢献
リリース	貢献	主導	情報共有	情報共有	情報共有	主導
サポート	主導	貢献	情報共有	情報共有	情報共有	貢献

コミュニケーションはアップグレード プロセス全体を通して非常に重要です。計画、状況、決定事項、次の手順をステークホルダーと定期的に共有する必要があります。アップグレードの可能性については、連絡が早すぎるということはありません。アップグレードの目的となる新機能についてユーザーコミュニティに通知し、評価フェーズと移行フェーズの時間枠についても知らせます。各フェーズの結果は、うまくいったことや、ツールチームまたはユーザーが特定した問題点も含めて共有します。状況と決定事項に加え、時間枠の変更についても継続的に情報提供します。アップグレード全体を通してコミュニケーションに継続的に力を入れることが重要です。そして、最大限の成果をあげるには、各ステークホルダー グループのニーズを考慮し、適切にコミュニケーションを調整します。

マイルストーン ミーティングやステータス ミーティングをスケジュールし、アップグレードに関する定期的な更新情報やステータス レポートを主要なステークホルダーに提供します。このミーティングでは、各チームメンバーの状況を確認します。アップグレード プロセスで発生した問題は、これらのミーティングでチームと共有する必要があります。各フェーズで完了したアクティビティの技術的な見直しを定期的に行います。次の表と図は、MATLAB および Simulink のプレリリースとリリースのスケジュールに関連するコミュニケーションのタイムラインの例です。

コミュニケーションのタイムライン		
日付	対象者	トピックス
MATLAB および Simulink のプレリリースの 3 か月前	ユーザー基盤および経営陣	次回プレリリーステストのリマインダー アップグレードの目的 (以下を含める): • アップグレードの既知の推進要因のリスト • アップグレードに対する既知の障壁のリスト プレリリースをテストするボランティアの依頼
MATLAB および Simulink のプレリリースの 1 か月前	ユーザー基盤	次回プレリリーステストのリマインダー 次回プレリリーステストのスケジュール 既知の推進要因/障壁の変更点
MATLAB および Simulink のプレリリース	プレリリーステストの積極的な参加者	プレリリーステストの進行状況およびスケジュールの共有
MATLAB および Simulink のプレリリースの 1 か月後	ユーザー基盤	成功および問題点を含む、プレリリーステストの結果の共有 残りのプレリリーステスト計画がある場合はその共有
MATLAB および Simulink のリリースの 1 か月前	ユーザー基盤および経営陣	プレリリーステストの結果の共有 アップグレード評価計画の共有
MATLAB および Simulink のリリース	アップグレード評価の積極的な参加者	アップグレード評価の進行状況およびスケジュールの共有
MATLAB および Simulink のリリースの 1 か月後	ユーザー基盤	成功および問題点を含む、アップグレードの評価結果の共有 残りのアップグレード評価計画の共有 (ある場合)
MATLAB および Simulink のリリースの 2 か月後	ユーザー基盤および経営陣	準備ができ次第、計画およびスケジュールなど、アップグレードに関する決定事項の共有



MathWorks リリースに関連するコミュニケーションのタイムライン

3.2.2 ビジネスケースの作成

新規バージョンにアップグレードする前にビジネスケースを作成することで、アップグレードによる利益と関連コストが明確になります。MATLAB および Simulink 製品の新機能により、ワークフローが改善されるなどの利点もあります。また、現在の MATLAB および Simulink のバージョンが新しいオペレーティング システムやサードパーティのソフトウェアをサポートしていない場合、アップグレード用のビジネスケースが、オペレーティング システムやサードパーティ ソフトウェアのアップグレードをきっかけに実行に移される場合もあります。

3.2.3 事前の目標の定義

アップグレード プロセスの範囲を制限することは重要です。一度に多くのことをやろうとすると、アップグレード プロジェクトに遅延が発生したり、失敗したりするリスクが高まります。

可能な限り、新機能を導入することなく、「現状のまま」の状態でモデルをアップグレードします。新しいカスタムツールや MATLAB および Simulink の新規バージョンを導入するのと同時に新機能を導入すると、アップグレード プロセスが複雑になり、モデルの妥当性確認が困難になります。新機能は、モデルが完全にアップグレードされ、MATLAB および Simulink の新規バージョンで妥当性が確認されてから導入します。

アップグレード プロセス中に、モデルの更新、シミュレーション、コードの生成など、組織の一般的なワークフローに焦点を当てたアップグレードのテストを行います。モデルやカスタムツールのアップグレード後は、開発を担当したエンジニアが新規バージョンでモデルの妥当性を確認する必要があります。

可能な場合は、カスタムツールの機能を Simulink の組み込み機能に置き換えることを目標に含めるのが良い考えです。たとえば、カスタム ライブラリ ブロックを、同じ機能を備えた新しい Simulink ブロックに置き換えることを計画することができます。新規バージョンで適用されなくなったモデリング スタイル ガイドラインを削除し、使用予定の Simulink の新機能に合わせて新しいガイドラインを追加することを目標として設定します。カスタムツールのドキュメンテーションやトレーニング教材の評価を目標に加えることを検討します。

3.3 移行

移行フェーズでは、モデルおよびカスタムツールを反復的に新規バージョンにアップグレードします。それぞれの反復作業では、モデル、スクリプト、ユーザー インターフェイス、テンプレート、サードパーティツールとのインターフェイス、およびその他のアプリケーションをアップグレードするために、通常、自動による手順と手動による手順が必要となります。反復的なアプローチにより、アップグレード プロセスの各フェーズで強固な基盤を構築し、組織全体およびエンドユーザー側のリスクと労力を軽減すると同時に新規バージョンの信頼性を高めます。

移行フェーズでは、モデルは新規バージョンにアップグレードされるだけで、まだテストされていません。テストフェーズで、モデルのテストを行います。

3.3.1 環境設定

モデルを新規バージョンに移行する前に、モデリング環境を設定する必要があります。お使いの環境によっては、この手順に、カスタムツールへのアクセス、MATLAB パスの設定、コンパイラの設定、MATLAB プロジェクトファイルの開始などが含まれる場合があります。MATLAB および Simulink の新規バージョンで環境設定を行うためには、できるだけ変更を少なくすることが重要です。カスタムツールや MATLAB スクリプトの変更は、アップグレード プロセスのカスタムツール移行フェーズの開始まで延期します。評価フェーズで使用したサンドボックス手法をスケールアップし、複数のアップグレード プロジェクトで使用できるようにします。

多くの場合、移行フェーズの初期段階で標準のコンフィギュレーション設定を作成しておくと有利です。これによって、作業の重複を最小限に抑え、後で発生する統合に関する問題を減らすことができます。初期テストで得られた推奨設定は開始点として適していますが、新しい設定や変更された設定があれば各コードターゲットに対してそれをレビューし、設定が適切であることを確認します。移行プロセスの開始時には、これらのレビュー済みの設定をモデルに適用します。

3.3.2 移行するモデルの特定

プロセスの評価フェーズでは、新規バージョンのモデルを 1 つ選択してアップグレードし、テストを行うことにより、プロセスを評価して潜在的な問題を特定しました。移行フェーズでは、テスト対象をより多くのモデルに拡大します。利用可能なすべてのモデルをテストすることができない場合は、サイズやモデリングスタイルが異なるモデルをさまざまなチームから選択することをお勧めします。プロセスのこのフェーズでは、エッジケースや一般的ではないモデリングパターンを選択します。典型的なものとは異なるモデルを作成する特定のグループや個人を把握している場合は、それらのモデルをこのフェーズのテストに含めます。エッジケースをテストすることで、予期せぬ問題を発見し、MATLAB および Simulink の新規バージョンをリリースした後ではなく、アップグレード プロセス中に問題を解決することができます。そのため、生産納期に影響を与えることなく問題を解決する時間をより多く確保することができます。よくある高度な事例として、以下を含むモデルが挙げられます。

- 入れ子のライブラリ
- モデル参照ブロック
- Configurable subsystem やバリアント
- モデル参照ブロックと入れ子のライブラリの両方

3.3.3 初期移行

初期移行フェーズの目的は、ブロック線図の更新、シミュレーションの実行、コードの生成などの一般的なワークフローをテストすることです。このフェーズを完了するためのガイドとして、以下のアクティビティリストを使用します。

旧バージョン:

- MATLAB プロジェクトファイルを開く
- モデルのブロック線図更新、シミュレーション、およびコード生成を正常に実行
- Simulink で診断ビューアーを使用して、すべての警告メッセージを文書化

新規バージョン:

- MATLAB プロジェクトファイルを使用し、MATLAB でパスおよび環境を設定
- 選択したモデルに対してアップグレード アドバイザーを実行:
    - レポートを確認
    - 必要最小限の推奨される修正を適用 (選択したモデルの移行に支障のない修正はプロセスの後半で適用可能)
    - モデルのブロック線図更新、シミュレーション、およびコード生成を正常に実行
    - 診断ビューアーを使用してエラーメッセージおよび警告メッセージを文書化
    - エラーを解決
    - 警告メッセージを比較し、新規バージョンに固有のものか旧バージョンにも存在していたものかを判断
    - 警告メッセージの重大度を評価
    - モデルのアップデートまたは新機能の採用により警告メッセージを解決できるかどうかを評価
    - その他のすべての問題をアップグレード問題リストに文書化 (アップグレード アドバイザーで検出されなかった問題のあるカスタムツールの文書化など)

初期移行フェーズでは、モデルを更新できることを確認するために MATLAB ファイル、データディクショナリ、カスタム ブロック ライブラリ、およびカスタムツールに必要な最小限の変更のみを行います。モデルが依存するファイルは、MATLAB パスに追加する必要があります。新機能、最適化、コンフィギュレーション セット、および S-Function API の変更は、後続のカスタムツール移行のサブフェーズで完了する必要があります。

データディクショナリについては、モデルやデータディクショナリに関連するデータをロードし、データのロードやデータのロード後のモデルの正常な更新に関する問題を解決します。

カスタム ブロック ライブラリについては、アップグレード アドバイザーがモデル内に存在する Simulink カスタム ライブラリ ブロックを自動的に更新します。アップグレード アドバイザーは、組み込みの Simulink ブロックから作成されたカスタムブロックがカスタム ブロック ライブラリに存在し、ライブラリリンクがアクティブである場合はそれを更新し、モデルやカスタムライブラリに存在するカスタム S-Function ブロックを更新し、アップグレード アドバイザーのチェックごとに結果を生成します。結果の確認後、アップグレード アドバイザーによってブロックへの変更を自動的に適用できます。アップグレード アドバイザーはマスクを更新しないため、手動で更新する必要があります。

対象リリースよりもはるかに古いリリースからアップグレードした場合、まれにアップグレード アドバイザーがモデルの更新に失敗し、問題の根本的な原因を特定できない場合があります。このようなケースのほとんどは、MATLAB と Simulink のカスタマイズや、Simulink では意図されていない、あるいは予想されていないエッジケースのモデリングパターンが原因です。このような場合、問題の分離とデバッグのために、まずは MATLAB および Simulink の中間のリリースにアップグレードし、問題解決後に対象リリースにアップグレードすることが必要な場合があります。中間のバージョンで問題のデバッグを行う場合の詳細については、本ドキュメントの「初期テスト」を参照してください。

3.3.4 カスタムツールの移行

カスタムツールの移行だけでなく、組み込み機能の導入によりカスタムツール機能を廃止する機会を探ることも重要です。

3.3.4.1 カスタムツールのアップグレード

MATLAB および Simulink の新規バージョンにアップグレードする場合は、カスタム ストレージ クラス、カスタム システム ターゲット ファイル、MATLAB Report Generator および Simulink Report Generator 設定ファイル、カスタム モデル アドバイザー チェック、モデリング スタイル ガイドライン、およびカスタム MATLAB スクリプトの更新が必要になる場合があります。テンプレートモデルやデモモデルがある場合は、使用可能な Simulink の新機能のうち、使用すると有利であると考えられる機能を導入して、それらを修正することを検討してください。これは、コンフィギュレーション パラメー

ターの設定を MathWorks の推奨設定に移行する良い機会になります。また、これは、組織全体で共通のコンフィギュレーション パラメーター設定に合意する機会でもあります。

モデリング スタイル ガイドラインは、新規バージョンに合わせて更新する必要があります。適用されなくなったモデリング スタイル ガイドラインを削除し、使用予定の Simulink の新機能に応じたガイドラインを追加することを検討してください。

カスタムツールの開発と移行を行う場合は、それらのツールの要件と目的の機能を詳述した設計ドキュメントを作成することを推奨します。カスタムツールの開発については、運用環境のソフトウェアと同様の開発プロセスに従います。テスト計画とテストケースを作成し、MATLAB および Simulink の新規バージョンへのアップグレード時にそれらを使用してカスタムツールのテストを行います。可能であれば、テスト計画を自動化し、各プレリリースと対象となる MATLAB および Simulink のバージョンで自動テストを実行します。MATLAB および Simulink の新規バージョンにアップグレードしたら、必要に応じて設計ドキュメント、テスト計画、およびテストケースを更新します。

以下のセクションを確認し、アップグレードが必要なファイルを特定します。次に、Simulink の新機能、最適化、および文書化された API を確認し、カスタムツール機能を Simulink の組み込み機能に置き換えられる機会を探ります。

MATLAB スクリプトおよびファイル

モデルの更新、シミュレーション、およびコード生成をエラーなしで実行するために必要な MATLAB スクリプトおよびファイルをアップグレードします。以下はその例です。

MATLAB と関わる MATLAB スクリプト	Simulink と関わる MATLAB スクリプト
• MATLAB プロジェクト • 環境設定 MATLAB スクリプト • コンパイラ設定スクリプト • データ解析スクリプト • ユーザーが作成した GUI	• パラメーターを読み込むスクリプト • コンフィギュレーション セットを読み込むスクリプト • コンフィギュレーション オプションを設定するスクリプト • カスタムメニュー用スクリプト • ユーザー インターフェイス スクリプト • ブロックコールバックのスクリプト • マスク内のスクリプト

GUIDE を使用して GUI を開発した場合は、それらを App Designer に変換します。MathWorks では、モデルのコールバックではなく MATLAB プロジェクトファイルを使用することを推奨しています。モデルのコールバック (スクリプトのプリロードおよびポストロードなど) の使用を回避できない場合は、必ずコールバックの追跡を有効にして、コールバックに起因する問題をデバッグしやすくしてください。

Simulink のカスタマイズおよびコンフィギュレーション

モデルの更新、シミュレーション、およびコード生成をエラーなしで実行するために必要な Simulink ファイルをアップグレードします。以下はその例です。

シミュレーション	コード生成	検証と妥当性確認
• シミュレーション データ • ログのカスタマイズ • 可視化のカスタマイズ • レポート生成設定ファイル • データ比較ツール	• システム ターゲット ファイル (STF) • テンプレート makefile (TMF) • Target Language Compiler (TLC) ファイル • コード生成テンプレート (CGT) ファイル • コード置換ライブラリ (CRL) ファイル • SIL/PIL 環境	• ユニットテストのカスタマイズ • テストデータ • レポート • 信号およびテスト オーサリング ツール • カスタムのモデル アドバイザー チェック • Simulink Design Verifier のカスタマイズ • Polyspace 静的解析ツールの構成 • SIL/PIL 環境

コンフィギュレーション設定

アップグレードには常に、コンフィギュレーション設定の変更が伴います。つまり、新しい設定、変更された設定、非推奨となった設定です。これらの変更の特定、把握、文書化、適用が必要です。新規バージョンでは、モデルのコンフィギュレーション設定をモデル エクスプローラーからエクスポートできます。詳細については、Simulinkドキュメンテーションの「コンフィギュレーション セットを保存する」をご覧ください。この機能がない旧バージョンでは、カスタムスクリプトを作成してこの設定をエクスポートできます。

コンフィギュレーション設定の更新を移行フェーズの初期段階で行い、設定を十分にテストして、この作業がすべての人に利益をもたらし、モデルが統合に対応することを確認する必要があります。コンフィギュレーション設定の変更を採用するプロセスを用意しておくことをお勧めします。

データディクショナリ

データディクショナリを読み込み、モデルの読み込みまたは正常な更新に関する問題を解決します。MATLAB スクリプトと ASCII ベースのデータディクショナリを Simulink データディクショナリおよび Embedded Coder ディクショナリにアップグレードします。Embedded Coder ディクショナリの詳細については、*Simulink データディクショナリ*および*Embedded Coder ディクショナリ*のドキュメンテーションを参照してください。

カスタム ブロック ライブラリ

カスタム ブロック ライブラリを新規バージョンにアップグレードする場合は、その機会を利用してカスタムブロックを Simulink の組み込みブロックや機能に移行します。リリースノートを使用すると、新規のブロックや機能を絞り込んで検索することができます。また、新しい機能や強化された機能が利用できる場合は、カスタムブロックを改善し最適化する機会にもなります。

新しく利用できるようになった同じ機能をもつ組み込みブロックに、現在のカスタムブロックとは異なるダイアログ パラメーターの値が指定されている場合があります。このような場合、ブロックダイアログのパラメーターを古いパラメーター フィールドから新しいパラメーターフィールドにコピーすることができます。この手順は、手作業で、または MATLAB スクリプトを介して自動的に行うことができます。チームで使用しなくなったブロックを段階的に廃止する場合、MATLAB パス上に、ユーザーが今後アクセスしないように明確にマークを付けたレガシーのライブラリを作成する必要があります。MATLAB および Simulink の新規バージョンでモデルを更新し、アップグレード用の MATLAB スクリプトを実行した後に、このライブラリを削除することができます。お使いのライブラリでアップグレード アドバイザーを実行します。S-Function に変更が加えられた場合や、その他のプラットフォームをサポートする予定がある場合は、S-Function を再コンパイルすることが必要な場合があります。

カスタムのモデルアドバイザー設定

モデルアドバイザーのチェックと設定はリリースごとに変更するのが一般的です。アップグレードは、新しいチェックについて確認すると同時に、現在のモデルアドバイザーの設定をレビューする機会となります。カスタムチェックもテストし、必要に応じて更新します。組織のニーズおよびガイドラインに合ったバージョン固有の一連のチェックを事前に開発することで、各チームが費やす時間を短縮します。

3.3.4.2 カスタムツールのテスト

テスト計画とテストケースを使用してカスタムツールをテストし、MATLAB および Simulink の新規バージョンでカスタムツールが意図したとおりに動作することを確認します。次に、モデルのアップグレード プロセスのテストを開始します。複数の PC で、組織がサポートするすべてのオペレーティング システムにおいて、カスタムツールおよびアップグレード プロセスをテストすることをお勧めします。カスタムツールの自動回帰テストを開発していない場合は、この時点で作成し、カスタムツールのアップデートやアップグレードの際にいつでも再利用できるようにします。

3.3.4.3 サードパーティツールのテスト

MATLAB および Simulink の新規バージョンのリリース前に、すべてのサードパーティツールを新しいリリースの MATLAB および Simulink を使用してテストすることを推奨します。場合によっては、MATLAB および Simulink の新規バージョンと互換性のある、新規バージョンのサードパーティ ソフトウェアがすぐにリリースされないこともあります。サプライヤーにリリース時期を確認し、それに応じて計画を立てます。以下はその例です。

アップグレード時に検討すべきサードパーティ製品

- 要件管理ツール (例: IBM Rational DOORS)
- キャリブレーション サポート ツール
- MATLAB および Simulink 製品と統合されている構成管理ツール
- MATLAB および Simulink 製品と統合されている変更管理ツール
- オーサリング/システム エンジニアリング ツール
- コシミュレーション ツール
- IDE
- コンパイラ
- ハードウェア

3.3.5 移行の自動化

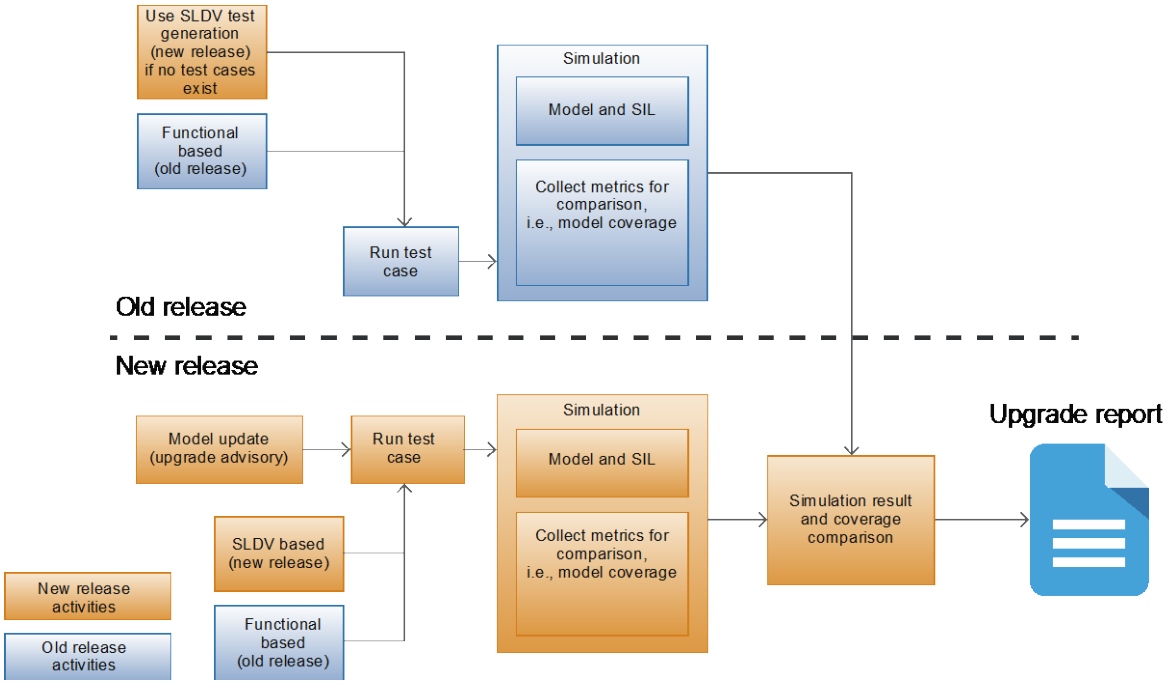
初期移行とカスタムツールの移行を実行したら、次の手順ではアップグレード プロセスを可能な限り自動化します。

3.3.5.1 アップグレード自動化ツール

モデルやカスタムツールのアップグレードプロセスは、自動化することを推奨します。他のすべてのアップグレードスクリプトを呼び出す単一のマスターのアップグレード用 MATLAB スクリプトを作成し、どのモデルをアップグレードする場合も 1 つの MATLAB スクリプトを実行するだけで済むようにすると便利です。このプロセスは可能な限り自動化します。この MATLAB スクリプトはさまざまな関数を呼び出して、以下を自動化します。

- アップグレード アドバイザー
- カスタムライブラリのコンテンツ
- コンフィギュレーション パラメーター
- ワークスペース オブジェクト
- アップグレード後の回帰テスト
- カスタムツールのアップグレード プロセス

これらの自動化ツールでは、各フェーズで発生したエラーや警告の詳細を提示するレポートが生成されます。Simulink Design Verifier (SLDV) および SIL シミュレーションを使用してこれを実行するワークフローの例を以下にご紹介します。このワークフローは、スクリプトのみに基づく手法、またはワークフローを自動化してアップグレード プロセスを誘導する UI を使用して実装できます。以下の図は、Simulink モデルをあるリリースから別のリリースに自動的にアップグレードする際に使用できるアップグレードプロセスの例を示しています。



アップグレード プロセスの例

3.3.5.2 テスト自動化ツール

MATLAB および Simulink の新規バージョンのリリース前に、カスタムツールを含むアップグレード スクリプトを、組織内ですべてのモデルに対して実行します。一部のモデルしかアップグレードできない場合は、モデリングスタイルやサイズが異なるモデルをさまざまなグループから選択します。モデルをアップグレードしたら、ブロック線図を更新し、シミュレートして、コードを生成します。警告メッセージを確認し、発生したエラーを文書化します。

このテスト手順は自動化することを推奨します。自動化により、複数のモデルに対してバッチモードでこれらのテストを実行し、問題を自動的に文書化できます。これらの問題の一部を解決するために、MATLAB および Simulink の新規バージョンのリリース前に、回避策や MATLAB スクリプトを開発することが必要な場合があります。このバッチモードは継続的インテグレーション サーバーに実装することもできます。CI サーバーを使用することで、演算リソースをプールできるだけでなく、組織でアップグレード プロセスを開始する準備ができ次第、アクティビティを自動的に発生させることができます。

3.3.5.3 残りのモデルの移行

ここまでで、カスタムツールとカスタム ブロック ライブラリをアップグレードし、アップグレード自動化ツールを開発しました。次の手順では、残りのモデルをアップグレードします。このアップグレード アクティビティは、ローカルマシンで実行する方法と、CI サーバーにスケールアップしてアップグレード プロセスを自動的に完了させる方法があります。

モデルをアップグレードする際に、発生した問題を文書化します。モデル、自動化ツール、カスタム ブロック ライブラリ、カスタムツールなどに問題が発生する可能性があります。移行フェーズで推奨される手順に従って問題を解決してから、自動化ツールを再度実行します。既知の問題がすべて解決されるまで、この手順を繰り返します。この反復的なループは CI サーバーでも実行できます。その場合、CI サーバーがアップグレード アクティビティを実行します。

3.4 テスト

テストフェーズの目的は、MATLAB および Simulink の新規バージョンにおけるモデルおよびコードが、旧バージョンのモデルおよびコードと許容範囲内で機能的および数値的に等価であるかを確認することです。

3.4.1 回帰テスト

ユーザーがよく行う作業に対し回帰テストを行い、MATLAB および Simulink の新規バージョンでこれらの作業にかかる時間を大まかに把握します。可能な限り、以前のバージョンで同じタスクを実行し、テスト結果を比較します。このテストは、現在使用中のいくつかの大規模モデルで実行することをお勧めします。

このアクティビティは、新規バージョンをリリースする前に潜在的な問題を検出するのに役立ちます。MATLAB および Simulink の新規バージョンへのアップグレードを延期することは、新規バージョンやカスタムツールのリリースを撤回することよりも簡単です。アップグレードの実装前に問題を特定し、解決策を見つけたり、必要に応じて回避策を開発したりすることができるため、テストは運用環境の開発プロジェクトに対するアップグレードの影響を最小限に抑えるために必要不可欠なものです。

回帰テストは、現在の結果のベースラインに対するさまざまなチェックを含め、徹底したものにする必要があります。整合性チェックは、把握する必要のある変更の特定に役立ち、最終的にアップグレードの成功 (または失敗) を左右します。次の表は、アップグレードの評価のためにチェックする必要のある主な項目の一覧を示したものです。これは網羅的なリストではなく、特徴的な重要項目をまとめた代表的なリストです。機能とパフォーマンスの両方を考慮する必要があることに留意します。当然のことながら、これらの項目に対応した (できれば自動化された) テストがあれば、このプロセスをさらに利用しやすくなります。アップグレード プロセスの一環として実行されるテストに対して改善を行った場合は、それらを開発ワークフローに統合できるだけでなく、統合する必要もあります。最終的に、テスト計画は各フェーズに合わせて適宜調整する必要があります。

結果	理想	許容範囲	メモ
シミュレーション結果	同一の結果	許容範囲内	記録された信号にモデルの出力がすべて含まれている必要があります。重要な中間結果とみなされる信号を追加で記録できます。
カバレッジメトリクス	同一の結果	カバレッジツールの変更によって説明される	100% に近いカバレッジを達成しているテストによるシミュレーション結果であれば信頼性が高くなります。 注: テストのカバレッジツールの変更によってバージョン間でカバレッジメトリクスが変わる可能性があります。このことを理解し、信頼できるテストを実施することが重要です。
コード生成	同等	エラーなし	SIL テストにより、ここでの「許容範囲」の結果を問題のないものにすることができます。
SIL テスト	同一の結果	許容範囲内	記録された信号にモデルの出力がすべて含まれている必要があります。重要な中間結果とみなされる信号を追加で記録できます。
シミュレーション時間	以前のリリースよりも高速	許容範囲内	以前のリリースと新規リリースの間の若干のシミュレーション時間の増加は許容できる場合があります。
コード生成時間	以前のリリースよりも高速	許容範囲内	以前のリリースと新規リリースの間の若干のコード生成時間の増加は許容できる場合があります。

サードパーティの互換性	同等	同等	サードパーティのソフトウェア、ツール、場合によりハードウェアは、新規バージョンの MATLAB と連携して動作する必要があります。
エラーと警告	類似の結果		新しいエラーが発生した場合は解決する必要があります。新しい警告が発生した場合は調査して理解する必要があります。

3.4.2 等価性テスト

等価性テストを実行し、以前のバージョンでの各モデルとコードのシミュレーション結果と新規バージョンのモデルとコードのシミュレーション結果を比較することをお勧めします。セクション 3.3.5.1 に示されたプロセスの例には、等価性テストのフェーズが含まれています。この等価性テストの完了後にモデルに追加するのは MATLAB および Simulink の新機能のみにすることが重要です。そうしない場合、モデルおよび生成されたコードがバージョン間で同等に機能しない可能性があります。Simulink Test は、MATLAB の複数リリースでのテストの実行、等価性テスト、およびリリース間でのコードテストをサポートしています。Simulink Design Verifier を使用すると、等価性テストのテストカバレッジが 100% になるテストケースを生成できます。

3.4.3 継続的インテグレーションのテスト

アップグレード、統合、シミュレーション、コード生成、等価性テスト、テストレポート生成を自動的に行うには継続的インテグレーション環境が推奨されます。大規模組織にとってこれは長期的な目標です。これを推進することで、アップグレード プロセスは各モデルの変換およびテストを手作業で行うよりもシームレスで費用効果が高いものになり、エンドユーザーへの影響も最小限に抑えることができます。CI サーバーを使用すると、アップグレード時に必要な手作業によるエンジニアリングの労力が大幅に削減されます。セクション 3.3.5.2 に記載されたワークフローは、すべてのモデルおよびツールのアップグレードに使用できます。CI サーバーは、モデルおよび MATLAB ファイルのアップグレードのために設定されたカスタムプロセスを実装する必要があります。各モデルおよび MATLAB ファイルのアップグレードが試行されるまで、それらのタスクが CI サーバーの各コンポーネントに対して繰り返されます。

3.4.4 ベータテスト

テストの最終フェーズでは、選ばれたユーザーによる小グループを用意し、カスタムツールとモデルのアップグレードプロセスのベータテストを行います。ベータテスターには、関連する Simulink モデルに対する深い見識があり、プロセスおよびツールを改善するためのフィードバックを提供できる、MATLAB および Simulink の経験豊富なユーザーを含めます。テスター グループは、必要なサードパーティのソフトウェアツールを使用して、日常業務で MATLAB および Simulink の新規バージョンとカスタムツールの使用を開始します。

3.4.5 アップデートのテスト

MATLAB および Simulink のアップデートをテストし、組織のテスト計画に組み入れます。MathWorks では、最新のアップデートが利用できる場合は、組織や開発プロジェクトで実行可能になり次第すぐに使用することを推奨しています。対象となる MATLAB および Simulink のバージョンを組織内でリリースする前に、そのバージョンで使用できるアップデートを確認し、最新のアップデートで以前に実行した回帰テストを繰り返します。組織でこのタイプのテストが CI サーバーを使用して自動化され、アップデートをすばやく評価できるテスト方法となっていれば理想的です。そのようになっていない場合は、本書の評価フェーズのセクションで説明されている回帰テストを実行し、このようにアップグレード プロセスの遅い段階でアップデートを導入することに起因している可能性がある問題を特定します。アップデートにはバグ修正が含まれるため、最新のアップデートへの移行により新たな問題が発生することはないと考えられます。問題が発生した場合は、今後のアップデートでの修正を検討できるよう、MathWorks の Web サイトでサービスリクエストを作成して、MathWorks サポートに問題を報告してください。新規バージョンへのアップグレード後は、MathWorks が新しいアップデートをリリースするたびにいつでもこの回帰テストを CI サーバーですばやく繰り返し、新しいアップデートのリリースが組織に与える影響を評価できます。

3.5 リリース

リリースフェーズの目的は、MATLAB および Simulink の新規バージョン、カスタムツール、および自動化ツールをリリースし、ユーザーにトレーニングを提供することです。

3.5.1 トレーニング

アップグレード時には、2 種類のトレーニングクラスをユーザー向けに用意することを推奨します。1 つ目は、MATLAB および Simulink のアップグレードに伴い利用可能になった主な新機能を要約し、組織に最も適した機能と利益を強調するセミナーやワークショップです。2 つ目は、アップグレード プロセス、特定の問題に推奨される回避策、新しいスタイルガイドライン、カスタムツールの新機能などに焦点を当てます。

3.5.2 リリース

テストが完了すると、カスタムツールをユーザーに公開できるようになります。カスタム ツール ソフトウェアの各バージョンにリリースノートを含めることをお勧めします。また、ツールを社内でのみ使用する場合でも、リリース前にファイルを *P コード化*して MATLAB コードを難読化するのも良い方法です。この方法により、エンジニアが他のユーザーに知らせずに自分だけで問題を解決しようとすることを防ぎ、チームによって異なるバージョンのカスタムツールを使用する可能性や、将来的にアップグレードの問題が発生する可能性を回避することができます。カスタムツールやインストーラーは、組織内のすべてのユーザーがアクセスできる場所に配置します。カスタムツールと MATLAB および Simulink の新規バージョンが利用可能であることをユーザーに通知する際に、インストールとアップグレードの手順も記載します。

特定のグループが移行を行うのではなく、エンジニアが各自で自分の Simulink モデルをアップグレードすることを推奨します。モデルを開発したエンジニアは、妥当性確認を行うために必要な専門知識を持っています。また、生産納期が迫っていることや、将来の運用環境ソフトウェアのバージョンに合わせてモデルのどの部分を修正しなければならないかについても認識しています。モデルを新しいリリースにアップグレードする必要があると判断した場合、エンジニアは妥当性確認済みのそのモデルを使用することについて、他の主要なステークホルダー (影響を受けるプロセスの下流工程を担当する他のエンジニアリング グループなど) から合意を得る必要があります。ここでの目標は、MATLAB および Simulink の新規バージョンへの移行により、運用スケジュールに遅延が生じないようにすることです。すべての新しいモデリング プロジェクトは、MATLAB および Simulink の新規バージョンを使用する必要があります。すべてのユーザーが自分のモデルの MATLAB および Simulink 新規リリースへの移行を完了するよう、期限を設定することをお勧めします。移行が完了するまでの間、複数の MATLAB および Simulink バージョンをサポートすることが必要な場合があります。

広範なテストと MATLAB スクリプトの作成は、プロセスを可能な限り自動化することを目的としています。何らかの理由でエンジニアが自分のモデルをアップグレードすることができない場合は、モデル テスト カバレッジの目標レベル (100% を推奨) を達成するテストケースが必要になります。モデルと生成済みコードを SIL 環境でテストし、シミュレーションとコード生成の出力が以前の MATLAB および Simulink バージョンの出力と一致することを (可能であれば自動化を介して) 検証します。

3.6 サポート

サポートフェーズの目的は、アップグレード後のアクティビティを継続的に実行し、次回のアップグレードのための作業を最小限に抑え、ユーザーに継続的なサポートを提供することです。

3.6.1 アップグレード後のアクティビティ

アップグレードの完了後、次回のアップグレードを容易にするために、いくつかのアップグレード後のアクティビティを実行できます。カスタムツールの移行フェーズで、カスタムツールの要件や機能を詳細に説明した設計ドキュメントを作成しなかった場合は、サポートフェーズで作成する必要があります。

この段階を利用して、組織で使用するカスタムツールや一般的なワークフローのテスト計画およびテストケースを作成します。次に、テスト計画とテストケースを使用して、MATLAB および Simulink の新規バージョンへのアップグレード時にカスタムツールのテストを行うことができます。テストケースには、各テストで期待される出力を含める必要があります。テスト計画が文書化され、テストケースが存在する場合は、次回のアップグレード時にテストを自動化することが容易になります。そのリリースにアップグレードする予定がない場合でも、各プレリリースで自動回帰テストを実行し、MathWorks にフィードバックを提供することができます。このフィードバックにより、ユーザーが MATLAB および Simulink を次のバージョンに移行する前に、MathWorks は製品の改良やユーザーが遭遇した問題の解決を行うことができます。もちろん、アップグレード対象予定の MATLAB および Simulink の新規バージョンや、MathWorks がリリースするあらゆるアップデートで自動回帰テストを実行することもできます。

カスタムツールのテストを自動化した後は、次の手順としてモデルの移行とモデルの妥当性確認プロセスを自動化し、CI 環境においてバッチモードでモデルのテストを実行して、問題のレポートを自動生成できるようにします。シミュレーションと生成されたコードの出力が、MATLAB と Simulink の以前のバージョンからの出力と一致することを検証するために、モデルおよび生成されたコードに対して、目標レベルのテストカバレッジを使用した SIL 環境でテストを行う必要があります。また、この種の SIL テストを自動化することで、次回アップグレード時のワークロードを軽減することができます。以下は、CI 環境で実行できるテストの一部です。

- カスタムツールの出力が MATLAB および Simulink の異なるバージョン間で一致するかどうかの検証
- 複数のモデルの MATLAB および Simulink 新規バージョンへの移行
- MATLAB および Simulink の新規バージョンにおけるモデルの更新、シミュレーションの実行、および複数のモデルのコード生成
- MATLAB と Simulink の異なるバージョン間でのシミュレーション結果の比較
- MATLAB と Simulink の同じバージョンでのモデルと生成コードのシミュレーション結果の比較
- MATLAB と Simulink の異なるバージョン間で生成されたコードのシミュレーション結果の比較

アップグレードを振り返って見ることで、文書化する価値のある教訓が明らかになる場合があります。アーキテクチャ、モデリングスタイル、カスタマイズに関する教訓のうち、アップグレード全般に当てはまるものと特定のバージョンに関連するものを判別するには、ある程度の労力と経験が必要です。これらの教訓を文書化するリポジトリまたは Wiki を作成し、バージョン固有の教訓へのリンクを含めます。新しい教訓があれば適切なバージョン固有のページに記載し、それらのページを定期的にレビューして、より一般的な項目を見つけ (さらにプロモートし) ます。

4 持続: 継続的なアップグレードという考え方

MathWorks では、継続的なアップグレードという考え方の導入を推奨しています。アップグレードのアクティビティを継続的に行うことで、次回のアップグレードを前回のアップグレードよりも簡単に行うことができるようになります。この考え方を導入する際の支援として、プレリリーステストや Industry Model Testing (IMT)、MathWorks のセミナー、Web セミナー、カンファレンスなどの利用を検討してください。

4.1 プレリリース テスト

可能であれば、プレリリースごとにモデルとカスタムツールをテストし、MathWorks の Web サイトでサービスリクエストを作成して MathWorks サポートに問題を報告してください。現行バージョンと将来のバージョンの間には多くの新機能が導入されるため、アップグレード予定のリリースのみでのテストを待たずにモデルをテストすることが最善の方法です。半年ごとにプレリリースをテストした場合、早期のテストフィードバックにより、MathWorks は次回アップグレードまでにユーザーが遭遇した問題を修正することができます。各プレリリースをテストすることで、リリースノートを読むよりもそのリリースに対する徹底した評価を行うことができるだけでなく、MATLAB および Simulink の新規バージョンへの

アップグレードに必要な労力を最小限に抑えることができます。さらに、新機能の学習を促進し、企業の意思決定者にアップグレードの適切な時期を案内することができます。

4.2 Industry Model Testing

Industry Model Testing にモデルを提供することを検討してください。IMT 情報シートからの抜粋を以下に示します。

IMT は、MathWorks の製品品質に関する戦略的イニシアチブの一環です。IMT のシステムは、MathWorks ソフトウェアの使用において、お客様が遭遇するリグレッションを特定、分離、防止するために設計されています。このシステムの一環として、MathWorks は、セキュリティで保護された環境で実際のお客様のモデルを使用してテストを行っています。この環境は、MathWorks ソフトウェアのビルド、テスト、およびリリースのプロセスと統合されています。IMT のプロセスにお使いのモデルを含めることにより、リリース時に非互換性が発生する可能性を大幅に減らすことができます。

IMT が適切でない場合は、類似の手法を内部で考案することもできます。CI 環境にテストの自動化を実装することが最初の手順です。MATLAB および Simulink の特定のバージョンを対象にする柔軟性を高めることで、少なくともプレリリースバージョンおよび一般リリースバージョンに対して MathWorks Industry Model Testing をエミュレートできます。これにより、早期に新規リリースの潜在的な問題に関する通知を受け取ったり、その信頼性を確認できます。

4.3 セミナー、Web セミナー、およびカンファレンス

MathWorks のセミナー、Web セミナー、およびカンファレンスに参加することで、新機能に関する最新情報を得ることができ、組織がアップグレード対象とすべき MATLAB および Simulink のバージョンについて十分な情報を得たうえで決定を行うことができます。

5 MathWorks サポート

MathWorks はアップグレード プロセスに関するセルフサービス サポート オプションを幅広く提供しています。こうしたセルフサービス オプションには、オンラインの製品ドキュメンテーション、新機能や非互換性を絞り込んで検索できるリリースノート、新機能の使用方法の例、MATLAB Answers、バグレポート、およびアクティブなユーザー コミュニティ (MATLAB Central) などがあります。追加のサポートが必要な場合は、製品に関する具体的な質問についてはサポートに、製品トレーニングについてはトレーニング担当に、アップグレードの支援についてはコンサルティング サービスにお問い合わせください。

6 付録

チェックリストの内容
評価: アップグレードの影響を理解する
☐ アップグレードのトリガーを特定
☐ 想定される非互換性も含め、現在の状況を評価
☐ 対象とする MATLAB および Simulink のバージョンを選択
☐ 一般的なワークフローを使用し、いくつかのモデルを新規バージョンでテスト
☐ ユーザーがよく行う作業について回帰テストを実行
☐ アップグレード評価レポートを作成
☐ 実行/中止を判断
計画: アップグレード プロジェクト全体の範囲を定義する
☐ ROI を明示したビジネスケースを作成
☐ 影響を受けるモデル、開発プロジェクト、および組織を特定することにより、範囲を決定
☐ タイミング、リソース、主要なステークホルダーなどについて定めた詳細な計画を作成
移行: 開発プロジェクトを反復的な方法で新規バージョンに変換する
☐ 初期移行: 少数のモデルを移行し、一般的なワークフローをテスト
☐ カスタムツールの移行: カスタムツールの機能を MATLAB および Simulink の組み込み機能に置き換え
☐ 移行の自動化: テスト対象をさまざまなモデリングスタイルのモデルに拡大し、自動化ツールを使用して残りのモデルをアップグレード
テスト: アップグレードモデルの妥当性を確認する
☐ ユーザーがよく行う作業について回帰テストを実行
☐ Simulink 製品を使用し、検証と妥当性確認のためにオープンループ、MIL、SIL、PIL の各シミュレーションをデスクトップで実行
☐ HIL、ラピッド プロトタイピング、オンターゲット ラピッド プロトタイピングの各テストをラボで実行
☐ サードパーティツールをテスト
☐ 新規バージョンにおけるモデルの機能の等価性を検証
☐ 選ばれたユーザーによる、新規バージョンのベータテストを実行
リリース: MATLAB の新規バージョンとカスタムツールをリリースする
☐ トレーニングクラスをスケジュールおよび実施
☐ 新しい MATLAB バージョンとカスタムツールをユーザーにリリース
☐ モデルのエキスパートに使用しているモデルの変換を推奨
☐ 全員が移行を完了する期限を設定
サポート: アップグレード プロセスを自動化し、問題を報告する
☐ CI サーバーを使用してアップグレード プロセスを自動化することを検討
☐ 振り返りを行い、アップグレードから学んだ教訓を文書化
☐ すべてのプレリリースをテストし、問題を MathWorks サポートに報告