博士論文

SysML メタモデルを用いた宇宙機システム運用シナリオ評価

染谷 一徳

主指導教員 青木 利晃

北陸先端科学技術大学院大学
先端科学技術専攻
（情報科学）

令和 8 年 3 月

Abstract

Spacecraft system development faces new operational challenges, such as adapting to new mission concepts and supporting multi-spacecraft constellation operations that require greater autonomy and reduced workload. At the same time, high operational reliability is essential because repairs in space are extremely difficult. Therefore, engineers must share a common understanding of operations from the early development phase by defining and reviewing operational scenarios. However, scenarios represented by activity diagrams often become large and complex, especially when off-nominal states with many variations are included, making them difficult to review and increasing the risk of design inconsistencies and missing requirements.

This study aims to improve the reviewability of operational scenarios by (i) enabling compact representation of large scenarios in activity diagrams and (ii) enhancing review coverage through exhaustive enumeration of state transitions and execution paths for off-nominal scenarios derived from a single nominal scenario. These capabilities are realized within a model-based development framework using three proposed metamodels: the operational layer definition metamodel, the operational stereotype metamodel, and the operational abnormal event metamodel integrated with STAMP/STPA safety analysis.

In the compact representation method, behaviors that have low review value and are already commonly understood are defined as *common behaviors* using stereotypes. By applying these stereotypes to action elements in activity diagrams in an annotation-like manner, known behavioral portions can be omitted. Removing such action elements reduces the number of elements and achieves a more compact representation. As a result, diagrams can be generated without requiring scrolling, improving visibility, enabling an overall view of the scenario, and facilitating intuitive understanding and effective review of essential aspects. However, among the omitted information, elements necessary for review are preserved as tagged values. Furthermore, a method is included to verify stereotypes and tagged values from a review perspective, ensuring that all information required for review is retained.

For the generation of off-nominal scenarios, the nominal scenario is analyzed using STAMP/STPA, and the results are represented in a SysML model and integrated into the MBSE framework. To derive off-nominal scenarios from the nominal scenario, we propose the concept of a *join point*, inspired by aspect-oriented approaches. Join points are defined within the nominal scenario, and unexpected abnormal scenarios (e.g., unsafe control actions and unsafe scenarios) derived from STAMP/STPA analysis are inserted at these points. This enables branching from the nominal scenario to abnormal states, thereby generating off-nominal scenarios.

In addition, to comprehensively review transitions to unexpected abnormal states, execution paths are explored from a single scenario, and all possible state transitions are enumerated and represented as a graph. Execution paths are exhaustively enumerated using a SAT-solver-based approach. Inspired by the path exploration mechanism of bounded model checking, we implement a SAT-solver-based approach to enumerate feasible execution routes and state transitions, which we refer to as Bounded Search.

Experiments using representative JAXA spacecraft scenarios show that the compact representations reduce both the number of elements and the required screen area by approximately 50%, enabling activity diagrams to fit within approximately a single A4 page, thereby improving review visibility. Furthermore, off-nominal scenario analysis revealed previously unanticipated transitions, demonstrating that the proposed approach provides effective design feedback and contributes to improved spacecraft system reliability.

Keyword: Spacecraft operation; Model-based systems engineering; review; activity diagram; compaction; Bounded model checking; SAT solver

概要

宇宙機システム開発は，新しいミッションに伴う新しい運用への対応，複数の宇宙機システムが連携してミッションを遂行するコンステレーション運用に対応した運用の自律化，省力化など，運用に対する新しい課題が発生している．その一方で，宇宙機システムの運用は，宇宙空間で行われるため容易に修理が出来ないことから高信頼性が求められる．そのため，開発の初期段階から各エンジニアで運用イメージを共有し，運用イメージをシナリオとして具体化し，レビューする方法が有効である．しかし，運用シナリオをアクティビティ図で可視化し共有を図ろうとすると，シナリオが肥大化し，レビューが困難なる課題がある．特に異常状態はバリエーションも多く，シナリオ化すると膨大になり，さらにレビュー困難となる．レビューが困難になると，運用イメージが共有できず，認識の齟齬や要求の抜け漏れが発生し，最悪の場合，宇宙機システムの損失に繋がるリスクが高まる．

本研究の目的は，運用シナリオのレビューに焦点を当て，肥大化した運用シナリオのコンパクトなアクティビティ図での表現によるレビュー容易性向上，及び正常状態を示すノミナルシナリオに対して異常状態を含めたオフノミナルシナリオの網羅的な実行ルートの探索と状態遷移の列挙によるレビュー拡充を行う．また，これらはモデルベース開発のフレームワーク上で実現し，情報統一を図るため，UML 及び SysML のメタモデルとして提案する．メタモデルは，コンパクト化を目的とした運用レイヤー定義メタモデル及び運用ステレオタイプメタモデル，安全解析手法 STAMP/STPA と連携しオフノミナルシナリオ評価を目的とした運用異常イベントメタモデルを提案する．

コンパクト化の手法においては，レビュー価値の低く共通認識がとれた振る舞いを *Common behavior* としてステレオタイプで定義する．ステレオタイプをアノテーションのようにアクティビティ図のアクション要素に適用することで，既知の振る舞い部分を削除する．アクション要素を削除することで要素数を削減しコンパクト化を実現する．これによりスクロールを不要とする図を生成し，シナリオの全体俯瞰がしやすく，視認性が向上し，理解しやすく本質的なレビューが行いやすい図を作成する．ただし，削除された情報の内，レビューに必要な情報はタグ付き値として情報を復活させる設計とする．また，レビュー観点からステレオタイプとタグ付き値を検証し，レビュー時に必要な情報が残ることを保証する仕組みを取り入れる．

Off-nominal scenario の生成では，まず Nominal scenario を STAMP/STAP で解析した結果を SysML モデルで表現し MBSE のフレームワークに取り込む．Nominal scenario から Off-nominal scenario を派生させるために，アスペクト指向に着想を得た *join point* を提案する．Nominal scenario に join point を設定し，STAMP/STPA の結果から導かれた想定外の異常状態シナリオ（Unsafe control action（UCA），unsafe scenario）を挿入する．Nominal scenario の join point から異常状態への分

岐が組み込まれ Off-nominal scenario を生成する．

また，想定外の異常状態への遷移を網羅的にレビューするために，１つのシナリオから実行可能なパスを探索し，可能性のある状態遷移を列挙しグラフ化する．シナリオの網羅的な実行ルート探索と状態遷移の列挙には，Bounded model checking の成立するルート探索機能の部分を利用し，SAT Solver で実装した探索方法を提案する．これを Bounded Search と呼ぶ．

実験の結果，コンパクト化においては JAXA 宇宙機システムの一般的な実シナリオを想定した場合，アクティビティ図に含まれる要素数及び画面サイズを最大 50%削減し，視認性向上に伴うレビュー容易性を実現した．特にアクティビティ図を確認する際に，画面スクロールを伴わない 1 つの画面にシナリオの開始から終了までのすべてのフローが収まることで，レビューの有効性が向上した．オフノミナルシナリオの評価においては，STAMP/STPA 安全解析手法によるノミナルシナリオの解析結果に基づきオフノミナルシナリオを生成し，Bounded search により実行可能なルートを探索することで，状態遷移を列挙し，想定外の状態に遷移する可能性の有無を評価した．JAXA の宇宙機システムを想定した実験では，オフノミナルシナリオ上の異常状態から正常状態へ復帰において制約条件があることが判明し，宇宙機システムへの設計フィードバックが可能であることが分かった．特に異常状態への復帰条件は，宇宙機システム信頼性に関わることから，本手法が宇宙機システム開発の信頼性向上に寄与できることが確認できた．

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　目的

本研究の目的は，宇宙機システムの運用シナリオを対象として，(1) 肥大化した運用シナリオのレビュー容易性向上，及び (2) 安全解析と連携した異常状態を含めた運用シナリオに基づく宇宙機システム設計の品質向上の二段階による運用シナリオ評価の改善を図る．宇宙機システムの運用シナリオとは，宇宙機システムをどのように運用するか，すなわち時系列に従いステップ・バイ・ステップでどのような処理を実行していくかを示したものである．この運用シナリオは宇宙機システムへの設計要求でもあり，打ち上げ前の検証シナリオにもなる重要な情報である．そのため，運用シナリオを正しく設計し，レビューにより評価を行い，品質を保つことは宇宙機システム全体の品質向上につながることとなる．

一段階目のレビュー評価改善は，肥大化した運用シナリオのレビュー容易性向上である．運用シナリオを SysML のアクティビティ図で示し，概要レベルからシステム間の詳細なやり取りまでを表現した場合，肥大化してレビューが困難となる課題がある．そこで本研究では，レビュー用にアクティビティ図をコンパクト化する方法を SysML のメタモデルとして提案する．運用シナリオのレビューは，実際の開発現場において，主にウォークスルー方式で行われる．この方式では主にエンジニアなど人の確認であるため，レビュー対象のアクティビティ図が分かりにくい，読みにくい，情報量が多すぎる場合，視認性が低下し，レビューの有効性が低下する懸念がある．レビューにおいて有効的な指摘を行うためには，視覚的な理解容易性の向上が有効な手段の 1 つであることが，先行研究 [28] で示されている．先行研究では，アイトラッキングを用いた実験により，Business process model and notation (BPMN) を対象に複雑なフローや画面スクロールを伴う図では指摘数が減少するという結果がある．BPMN はフローチャートの一種でありアクティビティ図を類似する図である．本研究では，その先行研究の結果を参考にし，レビュー品質の有効性向上として視認性に着目し，アクティビティ図をコンパクト化することが有効と考えた．

アクティビティ図のコンパクト化とは，レビューにおいてレビュー価値の低い情報を削除し，レビューに必要な情報を残すことで図の描画サイズを小さくすることである．レビュー価値の低い要素をまとめたり，削減するため，定量的な評価指標としては要素数として現れる．また，運用シナリオの開始から終了までの一連のフローが 1 つの視界に入り，画面スクロールが不要なアクティビティ図は，運用

シナリオを全体俯瞰をした視点の確認が容易となる．そこでもう一つの定量的指標として，A4 用紙で印刷した場合を単位とした何ページになるか評価する．なお，アクティビティ図の文字サイズを小さくしたり，各要素が重なり合うように配置したりすることでも図の描画サイズを小さくすることもできるが，その場合は視認性が低下し，レビューの有効性は低下する．そのため，指標としては，SysML ツールのデフォルト設定における A4 出力とし，フローが上から下へ時系列に沿って流れるように各要素は重ならず配置するとする．A4 用紙を指標とした理由は，実開発の現場では計算機のディスプレイやプロジェクターを用いることも多いが，環境によりさまざまであることから，印刷機能を用いた共通的な指標になることから選択した．また，スクロールに伴うレビュー時の可視性の改善も狙いであるため，A4 用紙の縦方向に着目し，複数枚に分かれた場合でも全体俯瞰をしてまとまりがあるところでページが変わるようにする．以上を踏まえて，コンパクト化の定義は，アクティビティ図に含まれる要素数の減少であり，SysML ツールのデフォルト設定において A4 用紙で出力した場合のページ数（縦方向）の減少とする．

二段階目のレビュー評価改善は，安全解析と連携した異常状態を含めた運用シナリオ評価方法の改善である．宇宙機システムの運用シナリオは，正常な運用ケースをノミナルシナリオと呼び，異常等の想定外のケースをオフノミナルシナリオと呼ぶ．提案する SysML メタモデルにおいては，ノミナルシナリオに対して安全解析を実施した結果に導出されるオフノミナルシナリオの関係性を定義する．これによりノミナルシナリオ，安全解析結果，オフノミシナリオを Model Based Systems Engineering (MBSE)[14] 上で情報管理が行える．また，レビューの充実を図るため，ノミナル及びオフノミナルシナリオの実行可能なルート探索および状態遷移の列挙する方法を提案する．運用シナリオは，初期の段階からノミナルシナリオとオフノミナルシナリオを検討する必要性がある [30]．しかし，運用シナリオで示せるのは，一例を示したに過ぎず，それ以外の実行順序（すなわち，実行可能なルート）は作成者およびレビューアによる想像に委ねられる．加えて，異常ケースは異常状態，発生タイミングなどパターンが多くアクティビティ図で網羅的に表現するのは困難である．そのため，レビューの品質において，運用シナリオが潜在的に持っている状態遷移の情報を可視化し，レビュー対象として含めることで，網羅性の向上が図れる．網羅性 (coverage）とは，レビューのスコープ [19] に対して充分な情報がすべて提示されているかである．これまでアクティビティ図を人の目が見て，頭の中で状態遷移をイメージしていた方法が，本研究により，運用シナリオが持つ潜在的な状態遷移を列挙し，提示することで，評価可能な情報が増え，レビュー時の評価における網羅性が向上する．なお，ウォークスルーの特性上，完全な網羅性を担保することが難しいため，人が確認できる範囲においての網羅性が向上となる．

本研究における運用シナリオの状態遷移の列挙とは，アクティビティ図の各アクションにおける状態をノードとして，運用シナリオの開始から終了までを実行した際に変化した状態遷移を矢印付きのエッジとして，グラフで示したものである．

ここでの状態とは，宇宙機システムのモード等のシステムや周辺環境などが個々の状態をまとめて，ビットベクターとして，ある時点における状態を示したものである．また，宇宙機システムは，前提条件や制約等により状態遷移に制限をかけており，想定外の状態に陥らないように設計している．モデル検査は運用シナリオの実行による状態遷移が制約に抵触しないかを評価するのが一般的な使い方であるが，本研究ではモデル検査の仕組みを利用して，実行可能なルートを検索すると共に，そのルートに基づいた状態遷移を列挙し，グラフによる可視化を行う．状態遷移の制約に抵触しない範囲において，運用シナリオにあるアクションの実行順序の組換えを総当たりで実施し，可能性のある状態遷移を列挙する．すなわち，運用シナリオに対するモデル検査ではない．

本研究における貢献は，提案するメタモデルを利用者が適用することで，運用シナリオのコンパクト化として，シナリオ内に存在するレビューの価値の低い汎用的な表現を削除し，レビューに必要な情報に絞ることで視認性を向上させ，レビューの有効性を向上させる．また，異常状態も含めた網羅的なシナリオレビューにおいては，提案するメタモデルに従い，安全解析結果と運用シナリオ間の関係性を整理し，異常状態を含む運用シナリオを整理する．さらに，シナリオ内の実行順序を網羅的に組み替えて，可能性のあるシナリオの実行ルート探索し，その結果に基づく状態遷移のグラフをレビューに提供する．これにより想定外の状態遷移の有無，設計や運用上の考慮漏れを見つけ出し，不備があれば宇宙機システム設計にフィードバックすることでシステム品質の向上を図る．

1.2 背景

高い信頼性を得る方法の一つとして,「Test Like You Fly, Fly Like You Test.」という言葉がある．宇宙機システム開発に当てはめると，試験は実際の運用を想定した内容を実施し，実際の運用は試験で実施した内容に沿って行うことを意味している．そして，システムズエンジニアリングのV字モデルに従えば，試験内容は要求の裏返しである．すなわち，宇宙機システムの開発初期段階から，どのようにシステムを運用するのか，想定するシナリオや異常状態に対応したシナリオを早期に検討し，設計や試験で活かすことで，狙った運用が実現できることになる．宇宙機システム開発には多くのエンジニアが関わることから，運用シナリオを関係者間で共有し，十分なレビューを行うことで最終的に実施したい運用が行えるシステムを開発することに繋がる．JAXA 衛星の ASTRO-H「ひとみ」では，運用に起因したミスが１つの要因となり，衛星損失を招いた．ASTRO-H の事故報告書 [24] によると，複数の原因が重なって事故が起こっているが，その中に姿勢異常検知時の使用する制御パラメータの設定ミスがあったと報告されている．このパラメータは，打上げ後に変更する計画であったものの JAXA と運用支援事業者の双方で運用内容の詳細が共有されておらず，セーフティに関わる重要なパラメータにもかかわらず，検証が十分になされずに設定する運用になってい

たという経緯がある．もし運用シナリオが可視化されており，関係者間で十分なレビューがされていれば，もしかしたら未然に防げた可能性も考えられる．

また，昨今の宇宙機システムの特徴として，１つの人工衛星でミッション達成するだけなく，Quasi-Zenith Satellite System（QZSS）[39] や Starlink[45] など複数衛星が連携して，コンステレーション衛星を構築するミッションが増加傾向にある．そして，複数衛星を同時に運用するためには，現在 JAXA で多くとられている 1 衛星を前提とした運用方法から改善が必要であり，省力化，自動化，そして異常時に自律的に対処できるシステムが求めれる．従来の地上システムを運用者が操作する管制方法から，宇宙機システムの自律的な運用にも重点を置き，全体システムとしての最適化が求められる．これは現在の運用方式より高い信頼性と高度で複雑な機能が必要となる．

これらを背景に，運用シナリオと宇宙機システム設計を連携し，関係者間の共通認識を円滑に構築するために，宇宙機システム開発にモデルベース開発を適用することを研究している．特に本研究で着目している運用シナリオは，作業の流れをフローチャートとして図示化し，データの流れや手順を可視化することで関係者間の共通理解が得やすくする．例えば，図 1.1 は，運用シナリオをモデル化した図である．この図は SysML のアクティビティ図であり，角丸四角で表わされた ”*Send X1 command for X2 action* “ 等はアクションと呼ばれる要素であり，振る舞いを示す．図の上部の”: *Ground Sys.*”等はアクターと呼ばれ，システムなどシナリオの登場人物が記載される．アクター毎の縦割りの線はスイムレーンと呼び，スイムレーション上のアクションは，その該当アクターが実行する振る舞いとしてとして解釈する．しかし，フローチャートのような図示化したモデルは，自然言語による文章記述よりアクター間の境界線が明確になり曖昧性が無くなる反面に，記述するスペースを多く必要とし，大きな図になるとレビューしきれないという新たな課題を引き起こす．特に異常状態への遷移や復帰にはバリエーションが多く，シナリオの数も増加するため，益々レビューが困難となる．

1.3 課題

本研究の課題は，異常時も含めた肥大化した宇宙機システムの運用シナリオをレビュー可能にするところにある．肥大化の要因は，先にも述べた通り，宇宙機システムの設計前提となるため詳細なやり取りまで表現する必要があり，安易に削除することできず，記述粒度が細かくなることである．粒度は，宇宙機システム開発のレイヤー毎に徐々に詳細化されると共に，細かくなっていく．宇宙機システムの運用シナリオの場合は，全体概要から徐々に細分化されたシステム，サブシステムと進める．そのため，最初は，衛星システム，地上システム，ユーザー間のやり取りから始まり，次に衛星システム内部の各サブシステム間，されに１つのサブシステム内部の各コンポーネント間と順を追って詳細化させる．

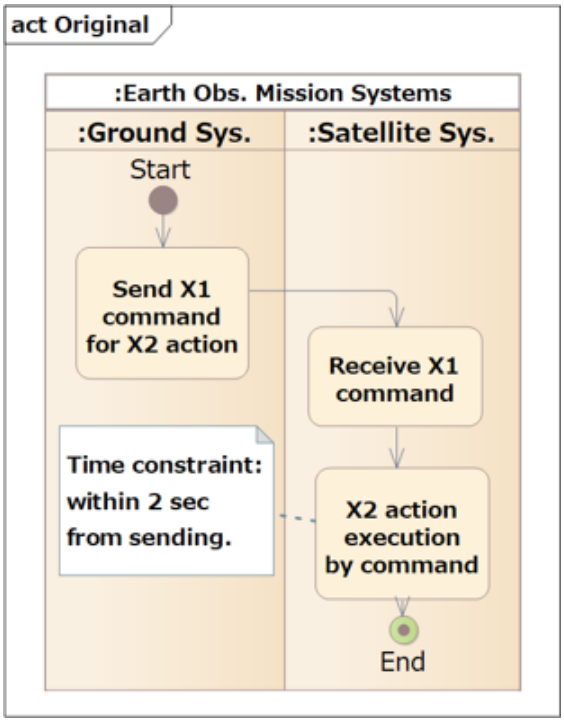


図 1.1: 運用シナリオ（アクティビティ図）

その上に，各レイヤーでの異常状態を検討する必要があり，バリエーションが増えるためシナリオ数は増加する．異常状態を検討する際は，まず正常状態としてノミナルシナリオを検討したうえで，次に異常状態を含むオフノミナルシナリオの作成に着手する．システムエンジニアがいきなり思いつきで異常状態を検討してオフノミナルシナリオを作成してしまうと，網羅性を担保することが難しく，発散してしまう．そのため，オフノミナルシナリオ作成を体系だって作成しており，無尽蔵にシナリオ数が増えることがないように工夫をしている．しかし，異常状態の発生タイミングは予見できず，システムエンジニアが宇宙機システム運用への影響が大きそうなタイミングを想定し作成するため，運用シナリオで示された一連の流れは一例に過ぎない．そのため，実際はもっと悪い状況に陥るオフノミナルシナリオが存在するかもしれないが，それを把握するための術がない．

よって，本研究では大きく２つの課題を扱う．１つ目は運用シナリオのアクティビティ図が肥大化して，そもそもレビューが困難であることを改善する．２つ目はレビューが可能になったアクティビティ図においてオフノミナルシナリオを生成し，異常状態も含めたシナリオ評価を行う．１つ目の課題に対しては，レビュー容易性を向上させるためのアクティビティ図をコンパクトにする方法を提案する．２つ目の課題に対しては，ノミナルシナリオからオフノミナルシナリオを生成すると共に，オフノミナルシナリオに含まれるアクション実行順序を網羅的に組み替えて，可能性のある状態遷移を可視化する方法を提案する．これらはすべてモデルベース開発の一環として，情報の整合性と一貫性を保つために，モデルベー

ス開発を前提としたメタモデルとして設計し，提案する．

1.4 論文の構成

本論文の構成は，第 1 章で，はじめにとして本研究の目的，背景，及び課題を述べる．第 2 章では，本研究で扱う関連する技術の解説，及び関連する研究について関連技術として述べる．第 3 章では準備とし本研究で使用するモデリング技術，MBSE および安全解析手法，運用シナリオのレビュー方法，状態遷移の評価方法について述べる．第 4 章では運用シナリオのレビュー容易性を図るコンパクト化手法について述べる．第 5 章ではオフノミナルシナリオの評価方法を述べる．第 6 章では第 4,5 章で述べた手法に基づき，適用実験を行う．第 7 章は実験結果に基づき考察を行う，最後の 8 章で本研究のまとめを述べる．

- 第 1 章：目的，背景，課題
- 第 2 章：関連研究
- 第 3 章：準備
- 第 4 章：シナリオコンパクト化メタモデル
- 第 5 章：運用異常イベントメタモデル
- 第 6 章：実験・評価
- 第 7 章：考察
- 第 8 章：おわりに

第2章　関連研究

関連研究としては，宇宙機システムのシステムモデル化として各宇宙機関の取り組みを紹介の現在位置を示す．次にアクティビティ図を含むフローチャートを対象としたコンパクト化手法を紹介し，本研究の新規性を説明する．また，本研究で用いるコンパクト化の手法として使用しているステレオタイプをアノテーションとしてアクティビティ図に組み込む方法を関連研究を説明する．オフノミナルシナリオ評価においては，安全解析手法との連携について，すでに公開されている手法と本研究との差異を示す．また，シナリオ実行順序の探索および状態遷移との整合性評価手法については，先行研究と本研究との差異及び新規性を述べる．

2.1　宇宙機システムのモデリング

宇宙機システム開発のモデリングは，大きく分類して Model Driven Development(MDD)，Model Based Development（MBD）の 2 種類のモデルベース開発手法がある．MDD は主にシステムアーキテクチャをモデリングし，要求から設計，検証のトレースを明確にすることを目的とする手法である．MBD は，主に Matlab/Simulink などのシミュレーション技術を駆使して，制御ロジックなどをコンピュータ上でシミュレーションし，最適化や検証を効率的に行うことを目的とする．どちらもモデルベース開発と呼ばれることが多いが，目的に応じて使い分ける．

本研究では，運用シナリオから要求，設計，検証との関係性を明確にする目的として使用することから MDD に着目する．具体的には，宇宙機システム開発におけるシステムズエンジニアリング [30] をモデルベース開発として実施する Model-Based Systems Engineering (MBSE)[14, 12, 15] 手法を対象する．MBSE の詳細は 3.2.1 にて説明する．現在，MBSE を用いた宇宙機システム開発は，JAXA，米国 NASA，欧州宇宙機関 ESA などの各国の宇宙機関及び各宇宙機メーカーなど多岐にわたり研究が行われおり，実開発への適用が試行されつつある [44, 18, 13, 10, 48]．これらの研究においても運用シナリオは，開発の初期段階から登場し，システム要求の抽出に使用される．いわゆる，Concept of Operations（ConOps）としてのモデリングである [30]．ConOps とはシステムの利用や運用イメージ及び方針などのコンセプトを示すものであり，要求の背後要因や根拠になる．ConOps を関係者間で共有することで，開発の方向性に統一が取れたり，要求の解釈を理解を助け

たりする効果がある．そのため，ConOpsが求める運用シナリオの記述粒度は大まかな流れやコンセプトレベルの開発の上位レイヤの内容に留まる．すなわち，サブシステム間のやり取りなどは一般的には含まない．そのため，シナリオの記述が細かくなることにより，モデルが肥大化するというリスクは低い．一方，本研究では，運用シナリオに基づきシステム設計や検証シナリオの前提とすることに加えて，より下位の開発レイヤのやり取りまで可視化し，具体的なデータのやり取りを表現し，最終的には宇宙機システムの運用手順書にまでシームレスに繋げることを構想しているため，シナリオが肥大化する傾向にある．現時点で他機関において，運用シナリオが肥大化してレビュー困難となるという同様の課題が発生している報告は確認できなかったが，それは記述粒度が荒い段階で留めているためと想定する．今後，同様の手法を取る場合は，同じ課題が顕在化すると推測する．

2.2　レビューの容易性向上の手法

運用シナリオは，MBSEにおいてモデル化する場合，アクティビティ図やシーケンス図で示されることが多い[15]．本研究では，運用シナリオをアクティビティ図を用いてモデル化する[44]．アクティビティ図はフローチャットとして表現することが可能であり，必要な機能，実行順序，条件分岐，及び機能間のデータのやり取りを図示化することに適している．加えて，実際のJAXAの宇宙機システム開発現場において，シーケンス図よりアクティビティ図の方が多くのエンジニアに広く馴染みがあり，受け入れられ易い表記方法であった．なお，シーケンス図はネットワークの情報のやり取り（3ウェイシェイクハンド等）を示すのに適しているが，運用の流れを直感的に理解する上ではアクティビティ図の方に優位性があった．

JAXAの宇宙機システム開発におけるアクティビティ図のレビューは，各システム，サブシステム，ソフトウェア担当などの専門分野のエンジニアによる確認が主流にある．毎回，新しいミッションに臨む宇宙機システムに対して，新規要素を各専門分野の視点で多角的に評価を行う．より多くの視点で確認を行い，抜け漏れを防ぐためにも，レビューの容易性向上させ，レビュー品質の有効性を上げることは重要となる．アクティビティ図のレビュー容易性向上には，評価の自動化，階層化，抽象化や不要な情報を省く視認性向上などの方法が考えられる．本研究では，参考文献[28]のアイトラックによる実験において，複雑な図や複数のページに分割されてスクロールを必要とする図はレビューの精度が落ちるという視認性の向上がレビュー品質の向上につながることを参考に，レビューがエンジニアによる目視によって行われることから，レビュー容易性向上の手法として視認性に着目する．よって，本研究では運用シナリオが数ページに記述されるスクロールを必要とするアクティビティ図をスクロールが不要となるA4用紙の高さで

アクティビティ図が区切りよく収めることを目標とするコンパクト化の手法を提案する．

2.3 アクティビティ図コンパクト手法

アクティビティ図のコンパクト化方法としては，オントロジーを利用したモデルの再構築，複数回の出現する部分をまとめる方法，MBSE ツールの機能を使う方法が先行研究としてある．

オントロジーを利用する方法は，モデルの規模と複雑性が増すことにより理解や利用が困難になることを課題に，意味論的構造に基づいて，意味的に整合したビューに分解し，認知しやすいモデルに再構成する [11, 16]．JAXA の運用シナリオにおいても，似て非なる言葉や，同じものを別の用語を用いて記述するなど，オントロジーに対する課題はあるが，運用シナリオは時系列による流れを示す必要があるため，モデルの再構築によりビューが変わり，フローの流れが示せなくなることは許容できない．ただし，オントロジーを用いて整理するという点は，同じ対象物で同じ用語を使うことで，表記の揺れを低減し，レビュー時の負荷低減や誤認識の低減につながるため，有益な方法である．そこで本研究においても，提案するメタモデルを設計の準備として，宇宙機システム開発で使用する用語を木構造で整理し，簡易的オントロジーとして各用語を明確にする方法を取り込む．

複数回出現する箇所を一つにまとめる方法は，コンパクト化においては有効な手法である．参考文献 [1] では，シナリオ単位で同じシナリオを１つのまとめて表現するという方法が提案されている．本研究の課題は１つのシナリオが肥大化するし，レビュー困難となることである．そのため，コンパクト化のターゲットは，運用シナリオ単位ではなく，運用シナリオ内で出現するレビュー価値の低い汎用的な表現である．また，運用シナリオ内で繰り返し出現するフローも，処理の流れは同じであるが，やり取りするデータが異なるなど，完全に同一でないという特徴がある．よって，本研究のコンパクト化は，単純な同じものをグルーピングするのではなく，共通する部分と変化する部分を区別して，まとめる工夫を行った方法を提案する．

同じ内容の振る舞いをまとめるという点においては，アクティビティ図には Call behavior action 要素を使う方法がある [9]．Call behavior action は，モデリング言語 UML で定義されている要素であり，別に定義したアクティビティ図を呼び出す．すなわち，アクティビティ図の一部を別のアクティビティ図として設計し，元のアクティビティ図が呼び出せる．言い換えれば，いくつかの要素をグルーピングすることができる．そして，UML に準拠した MBSE のツールにおいても，Call behavior action 要素は利用可能である．しかし，本研究が対象とする運用シナリオ上に繰り返し出現するフローは，先も述べた通り，大まかな流れは共通しているが，インタフェースされるデータ名称が運用シナリオ毎に異なったり，通信相手が異なったりと，似て非なる振る舞いが何度も繰り返されるのが特徴である．よっ

て，Call behavior action 要素で呼び出すアクティビティ図で共通化が図れず，レビューにおいては Call behavior action 要素で呼び出した先のアクティビティ図も確認する必要があり，手間が増えることになる．そこで，本研究では，共通部分と変化部分を区別し，追加のアクティビティ図を作成することなく，まとる手法を提案する．

2.4 安全解析手法との連携手法

安全解析手法において，一般的に利用頻度が高く，広く認知されいる方法は，Fault Tree Analysis（FTA）と Failure Mode and Effect Analysis（FMEA) である．FTA はトップダウン方式であり，異常事象を木構造のトップに置き，その異常事象を発生させる要因を葉として細分化し，原因究明を行う方法である．宇宙機システム開発や運用の現場においても，不具合が発生した際にその原因究明や対策の決定に用いられる．しかし，設計の考慮漏れや制約の抜けの検出には適さない手法である．

そのため，設計時の考慮漏れや制約の抜けを未然の防止ためには，ボトムアップ方式である FMEA が有効である．FMEA は機器の不具合などの故障モードがシステム全体に対して，どのような影響や不具合を引き起こすかを分析する手法である．しかし，運用シナリオの分析時点においては，開発の初期段階であることから機器の設計までは未完であるケースが多い．そのため，運用シナリオ上の異常状態を未完の機器設計から導くことは困難である．

そこで，本研究では運用シナリオの安全解析手法として STAMP/STPA を用いる．STAMP/STPA は，システム間の相互作用に着目し，データのやり取りが遅れた場合，誤って提供された場合などの振る舞いに関する事象を当てはめて，異常に陥るアクションの検出，及びそこからの導かれる異常となるシナリオを分析する手法である．STAMP/STPA については 3 章で詳しく述べるが，運用シナリオの主要な要素としてはシステム間のデータのやり取りであるため，そのデータが遅れた場合や異常であった場合に，システム全体に異常をきたす場合があるかを評価することは，運用シナリオの評価と整合する．よって，システムの相互作業による影響がシステムにどう波及するかを検討可能な STAMP/STPA を選択する．

これら安全解析を MBSE 上で表現する方法としては，RAAML[37] がある．RAAML では，FTA，FMEA，STAMP/STPA で使用する要素や記号を，SysML や UML として記述するためのステレオタイプが定義されている．SysML を主に用いる MBSE によるモデル作成において，RAAML を適用することで，安全解析を同じ MBSE 上で扱うことができる．しかし，安全解析で検出した異常状態，正常のノミナルシナリオ，オフノミナルシナリオなどと連携する部分がない．本研究では RAAML を利用し，安全解析を MBSE の枠組みで実施しつつ，オフノミナルシナリオを連携する部分をメタモデルとして提案する．

2.5　実行順序の探索手法

運用シナリオの実行順序を探索し，状態遷移との整合性を評価する方法としては，Littel-JIL[49, 25, 7, 38] 手法がある．Little-JIL は，医療分野を対象に医療手順の実行順序と状態遷移の整合性を評価するために考案された手法である．例えば，輸血を考えた場合は，患者の血液型確認，血液パックの選択，患者への輸血などのステップを踏む必要がある．ここでの状態とは，患者の血液情報を得ているか，輸血パックの取得したかなどである．状態遷移は，例えば，患者の血液型確認というアクションを実行すると，患者の血液型情報の有無を示す状態が，情報無しから情報有りに変化する．そして，患者に合致する血液型のパックの選択するときは，事前に患者の輸血情報を入手した状態でなければ実行できないという制約を設定することで，患者の血液型を知らない状態では輸血パックを入手しようとするとエラーとなって結果が出力される．各制約を満たしながら，最後のステップまで実行し，輸血完了という受理状態でシナリオが完了するかを評価し，受理状態で完了しない場合は反例が示される．

本研究においては，シナリオ実行と状態遷移の整合性という観点において Littel-JIL の手法は有効である．しかし，Littel-JIL を実行するシナリオは，並行実行などの選択要素は入れられるものの，基本的には Littel-JIL を実行する解析者が設定した実行順序で検証される．すなわち，実行順序を探索するうえでは，並行実行において，どちらを先に実行するか等の自由度があるが，解析者が作成した線形的な実行順序が前提であり，解析者の実行順序を無視して，その順番を逆にしてまで探索することはない．例えば，先の例を用いると，輸血パックを選択した後に，輸血作業を行うという線形の実行ステップにおいては，輸血した後に輸血パックを取得するという実行順序は評価しない．一方，宇宙機システムの運用シナリオは，複数のシステムが独立して稼働している状態を表現するため，エンジニアが線形的に実行されると想定していても，実際の運用では逆転する可能性がある．例えば，地上システムが故障してから，宇宙機システムがモードを変更することを想定した場合，先に宇宙機システムがモード変更を行い，地上システムが故障発生しても，問題ないかを確認する必要がある．このように，Littel-JIL ではある程度，エンジニアが想定した実行順序を前提として解析されるため，思いもよらない，想定外の未知の事象を見るけるためには網羅的な実行順序の入れ替えによる評価手法が必要となる．

また，MBSE ツールの中には，アクティビティ図と状態遷移図を連携して，シナリオの実行と状態遷移の評価をシミュレーションする機能が備わっている製品もある．しかし，Little-JIL 同様に，並行実行部分は実行順序の選択性があるが，線形で作成された実行部分の実行順序を入れ替えてまで評価することはない．あくまで評価機能の狙いは，設計したシナリオ通りに，矛盾なく状態が遷移するかを確認するためである．本研究でレビュー時に提示したい，シナリオの実行順序で可能性のあるパターンから考えられる状態遷移の列挙する機能ではない．

第3章　準備

準備の項においては，本論文で使用する技術を記述する．具体的には，運用シナリオ，UML，SysML，Model Based Systems Engineering(MBSE)，レビュー技術，STAMP/STPA，Bounded model checking，SAT ソルバーについて述べる．

3.1　モデリング言語

本研究で使用する Unified Modeling Language（UML) 及び UML に備わっている仕組みを述べる．

3.1.1　Unified Modeling Language

Unified Modeling Language(UML)[35] とは，Object Management Group（OMG）が取りまとめ公開しているモデリング言語であり，JAVA 等のオブジェクト指向のプログラムを表現する際に使用されるケースが多い．UML の登場により，三者三様であり，各々が独自の技法で作成していたフローチャートなどのブロック線図を統一の標記を用いて，共通の理解が図れるようにした．UML では，クラス図やオブジェクト図と呼ばれる構造を示す構造図，アクティビティ図やシーケンス図等の振舞いを示す振舞い図等の数種類の図が規定されている．利用者は，表現したい内容や目的に応じて図を使い分ける．本研究においては，UML のサブセットであり，システムアーキテクチャを表現するために拡張した SysML を用いる．SysML の詳細は，3.2.2 項について説明するが，本研究で使用するアクティビティ図及びステートマシン図の基本的な書き方は UML と同じである．

3.1.2　メタモデル

メタモデルのメタとは「高次の」という意味がある．本研究におけるメタモデルは，OMG が定義する UML 等で定義されるメタモデルのことを示す [36]．OMG が定義するモデルは 4 層構造のメタモデルアーキテクチャで示されることが多く，上位レイヤーからメタメタモデル，メタモデル，モデル，インスタンスというレイヤー構造をもつ．ただし，必ずしも 4 層である必要はない．一般的なシステムでは 4〜2 層を用いることが多いが，最低限 2 層あれば良い．重要なのは，Classifier

の Instance の関係性であり，それを繰り返すことで多層構造を構築する．本研究で提案するメタモデルは，4 層構造を前提としたメタモデルである．メタメタモデルは UML 及び SysML で規定されている要素や関係性を用いる．提案するメタモデルは，メタメタモデルに準拠し，アクティビティ図を作成するための骨格及び枠の定義である．そして，利用者は，提案するメタモデルに従い，モデルとしてアクティビティ図を作成する．さらに，インスタンスとして，アクティビティ図に具体的な処理内容及びデータ名を記載することで運用シナリオが表現される．

3.1.3 ステレオタイプ

ステレオタイプは，UML の拡張の仕組みの１つであり，SysML でも備わっており，≪≫記号で示される [40]．ステレオタイプを用いることで，モデル要素の論理的な拡張あるいは意味づけが可能となる．UML は特定の分野に特化したモデリング言語ではないため，利用者の分野に応じたカスタマイズして使用することが想定されている．その１つがステレオタイプであり，UML で予め定義されているステレオタイプに加えて，利用者が新しいステレオタイプを定義することも可能である．例えば,「えいせい」という要素を作成した際に,「衛星」なのか「衛生」なのか名前だけでは区別がつかない．これに対して≪宇宙機システム≫というステレオタイプを独自で設計し付与することで，意味が追加され「≪宇宙機システム≫ えいせい」が「衛星」であることを理解できる．本研究では，本ステレタイプによる拡張機能を用いて，宇宙機システムの運用シナリオで登場するデータおよびそれに依存した振る舞いの区別に使用する．

3.1.4 タグ付き値

UML のタグ付き値は，タグ情報と値の組合せであり，ステレオタイプ同様に利用者が独自に設計し利用できる拡張機能を有する [35]．何らかのパラメータ値の情報を作成した要素に対して付加することが可能である．加えて，タグ付き値はステレオタイプと関連して設計が可能である．例えば，宇宙機システムというステレオタイプを定義した際に，質量というタグを設計することで，宇宙機システムの質量 4000kg の 4000kg を値として記載することが可能となる．本研究では，ステレオタイプと一緒にタグ付き値も設計し，コンパクト化によって削除された情報のうち，レビューに必要な情報を残すための仕組みとして利用する．

3.1.5 パッケージ

パッケージは，独自で設計したステレオタイプ集やタグ付き値を１つにパッケージングする仕組みである．パッケージしたものは，他の利用者に配布し，適用す

ることが可能となる．OMG では，各分野に特化したパッケージがいくつか公開されており，利用者は UML のツールなどを通じてパッケージを読み込むことで自身のモデルで利用できる．組込み系に対応した MARTE[33] や，宇宙機システムの運用をターゲットとした SOLM[32] や，安全解析に特化した RAAML[37] がある．本研究においても，設計したステレオタイプ集及びタグ付き値をメタモデルとしてまとめ，それらをパッケージとして配布可能する．

3.2 モデルベース開発

本研究における宇宙機システム開発に適用するシステムズエンジニアリングを対象としたモデルベース開発を述べる．

3.2.1 Model Based Systems Engineering

Model Based Systems Engineering(MBSE)[14] は，システムズエンジニアリング [30] 手法をモデルベース開発として実施する方法として考案された手法である．MBSE の重要な概念は，Single source of truth である．MBSE の対比する手法としてはドキュメントベース開発がある．ドキュメントベース開発は，要求書や設計書などを文書を中心として開発を進める従前から行われてきた方法である．ドキュメントベース開発の弱点は，文書間の情報の同期や整合性を取ることが容易ではないということである．一方，MBSE は，UML のサブセットである SysML を用いてシステムモデルを構築する．そして，運用シナリオなどのフローチャートはアクティビティ図など記述するが，これはシステムモデルを可視化したビューとして扱われる．システムモデルとビューを区別することで，例えば，2 つの図で同じシステムが登場した場合，システムモデルとしては同じシステムが参照される．つまり，同じシステムであれば複数の図で現れても，根っこは 1 つである．そのため，片方の図でシステム名を変更すると，大元のシステム名も変わるため，もう一つ図にあるシステム名も同期して変更される．これが Single source of truth であり，どちらも同じシステムを見ており，真実は 1 つである．一方，ドキュメントベースでは文書 A と文書 B で同じシステムの記述があっても，それは読み手の解釈に委ねられる．また，文書 A のシステム名を変更しても，もう片方の文書 B のシステム名を変更し忘れるリスクもある．宇宙機システム開発のように，大勢のエンジニアが関わるプロジェクトにおいては，全員が同じものを参照して，同期して開発することは，認識の齟齬や矛盾を防ぐうえで重要となる．

3.2.2 SysML

MBSE のモデリング言語として，一般的に使用されるのが SysML[34] である．SysML は UML のサブセットであり，システムアーキテクチャを表現するために UML を拡張しており，ステレオタイプやダイアグラムが追加されている．本研究で提案するメタモデルは，UML のメタモデルであるが SysML にも適用可能であり，MBSE として使用するため SysML 上での利用を前提に設計した．また，UML を拡張する方式には，ヘビー方式とライト方式の 2 つの方式がある [40] がある．ヘビー方式は，メタクラスを独自に定義し，利用する方法である．メタクラスにはアクティビティやアクション要素などがある．一方，ライト方式は，新規のメタクラスを作成せず，既存のメタクラスも変更しない方法である．既存のメタクラスから派生させたり，独自のステレオタイプを定義する方法で拡張する．本論文では UML や SysML を拡張する方法としてライト方式を採用する．ライト方式を選択する理由は，SysML のメタクラスを利用するため，市販の MBSE ツールが使用でき，実開発への展開のハードルをヘビー方式より下げられるためである．

3.2.3 アクティビティ図

アクティビティ図は，UML 及び SysML の両方で利用可能な図であり，アクションと呼ばれる要素に処理を記載し，アクション同士を矢印（フロー）で繋ぐことで，アクションの実行順序，アクション間のデータのやり取りを示すことができる．また，分岐要素や並行実行要素を使用することで，フローチャートのように条件分岐などの複雑な処理も記述する．また，アクティビティパーティションと呼ばれる要素を配置することで，アクティビティ図内にスイムレーン（縦割りの線）を作成する．スイムレーンには，アクターと呼ばれる宇宙機システムや地上システムの担当が割り当てられる（allocation を行う）．そして，スイムレーン上に配置したアクションは，スイムレーンに allocation されてたアクターの処理として関係性が形成される．そのほかの要素としては，開始，終了などの要素を使用する．また，アクティビティ図で別に作成したアクティビティ図を内部で呼び出すためには，Call behavior action 要素も必要に応じて使用する．Call behavior action 要素は，アクティビティ図の入れ子を作ることができ，複数アクションのグルーピングや，同じフローが複数回登場する場合に使用する．

本研究における運用シナリオのモデル化では，アクティビティ図を用いる．シナリオの開始と終了は，アクティビティ図の開始要素，終了要素で示す．異常終了などのシナリオの目的が達成されずに途中終了となる場合は，フロー終了要素を用いる．また，シナリオ上の登場人物やシステムは，アクターとしてスイムレーンに割り当てる．そのうえで，シナリオに含まれる各処理（ステップ）はアクション要素として，実行主体となるスイムレーン上に配置することで，どのシステムまたは誰のアクションかを明確にする．

UML 及び SysML におけるシナリオの種別として，基本フローに対して，代替フロー（または代替シナリオ）と例外フロー（例外シナリオ）がある [50]．代替フローは基本フローに戻れる場合である．例外フローは基本フローに戻れない場合に用い，シナリオの終端はフロー終了要素を用いる．本研究におけるノミナルシナリオは，基本フローのみで示したフローとなる．一方，オフノミナルシナリオは基本フローに代替フローまたは例外フローが含まれたシナリオとなる．

3.3 マインドマップ

マインドマップは，頭の中の思考を整理する際に，関連する要素を親子関係で結び，木構造で表わせる図である．UML のクラス図，SysML のブロック定義図を用いること同じ図を作成可能であるが，要素間の関係性を定義しない点において厳密性に欠ける．ただし，専用市販のツール等を活用することで，深く考えず，手軽にスピーディーに情報を整理し可視化することができる．ブレインストーミングなどの場面において，情報の可視化や整理で有効である．

本研究においては，メタモデルの設計前提として宇宙機システムのドメイン知識を整理するために使用する．宇宙機システム開発や運用で使われている用語等を体系的に整理し，第三者のレビューと共有する．2.3 節においてもコンパクト化の方法としてオントロジーを用いる方法を紹介した．本マインドマップは簡易的なオントロジーの整理として，厳密な定義は行わないが，用語を予め示すことで，表記の揺らぎを低減するためにも活用する．

3.4 レビュー技術

レビュー技術として，レビューの種類や実施方法などの手法及びレビューにおける品質について述べる．特に宇宙機システムの運用シナリオのレビュー方式として主に用いられるウォークスルーに着目する．

3.4.1 レビュー手法

レビューの目的は，レビュー参加者（レビューアー）によって，レビュー対象に対する欠陥を検出することである [50]．レビュー方式には，技術面に焦点を当てた技術レビュー，インスペクション，ウォークスルー，比較的に管理的な側面に焦点を当てた管理者レビューや監査がある．さらに，インスペクションやウォークスルーは，ピアレビューとも呼ばれ，その分野の関係者や専門家が集まり，技術的な視点で比較的小規模で行われる．インスペクションはレビュー構成が厳密に定義され，ピュアレビューにおける公式なレビューとして扱われ，ウォークスルーは厳密な定義がない非公式レビューと位置付けられる．

宇宙機システムの運用シナリオのレビューは，主にウォークスルーで行われることが多く，レビュー対象の運用シナリオの欠陥を見つけると共に，関係者間で共通認識を得ることも狙いの 1 つとなっている．実際の実施方法としては，レビュー対象である運用シナリオを作成者が説明を行い，レビューアーが質問やコメントを通じて，理解を深めると共に指摘を行う．そして，コメントのやり取りを何回か行い，追加のコメントが出なくなった段階で終了となる．レビューアーは，運用シナリオに登場するシステム及びそのシステムと関連を持つシステム，システム全体のとりまとめを担うエンジニア，及び過去の宇宙開発システムの経験者や関連する技術の専門家がアサインされる．ウォークスルーにおいてレビューの有効性を高めるには，経験豊富なエンジニアをレビューアーとして参画させ，経験的な気づきによる欠陥も発見することが有効である．また，ウォークスルーの実施方法が，質疑応答による確認が主であることから，開発の初期段階から実施できる利点を持つ反面，厳密な網羅的確認には不向きである．

3.4.2 レビュー品質

レビューでは，単にレビューをしただけではなく，レビュー対象の欠陥を是正し，技術的な整合性と品質を保つところにある [19]．ピアレビューの一つであるウォークスルーは，先のレビュー手法と照らし合わせると，レビューの品質としては網羅性，整合性，有効性が重要と考えられる．網羅性は，レビュー観点において網羅的に評価できるかである．ただし，ウォークスルーは経験に基づく問題点の指摘であるため，完全に網羅性を担保することは困難となり，人が評価可能な範囲における網羅性となる．整合性は，運用シナリオと宇宙機システムを含む関連するシステムとの整合性が取れていることの確認である．有効性は，人によるレビューを前提に運用シナリオがレビュー可能であり，有効な指摘ができ，宇宙機システムの設計改善及び要求追加に繋がるかである．運用シナリオ（アクティビティ図）の視認性が悪い場合，レビューの大半の時間を読むことに費やし，本質的な指摘が減ることが懸念される．さらに，大きな図は，全体を俯瞰して捉えることが難しく，論点が局所的になる可能性もあり，これらはレビューの有効性を低下させる．

本研究では，レビューが人の目による確認であることから視認性に着目し，アクティビティ図をレビュー用にコンパクト化することでレビュー容易性を改善させ，レビューの有効性向上を図る．レビューに必要な情報を残し，レビュー価値の低い情報を削除する．どれだけの情報を削減したかは，アクティビティ図のアクション数として現れるため，アクティビティ図の要素数を定量評価の指数とする．また，長い運用シナリオは，アクティビティ図においてスクロールを伴い，視界にスタートとゴールまでが入らず，全体を俯瞰した評価が難しくなる．図のサイズとレビュー時の指摘の関係性について評価した研究 [28] では，スクロールを伴うフローチャートでは指摘数が減少する結果が報告されており，レビューの有効

性が低下する結果であった．この先行研究に基づき，本研究では，視界に１つのアクティビティ図が入ることを目指し，定量的な指標として，A4 サイズで何ページ分に相当するかを評価する．

オフノミナルシナリオを含む運用シナリオの実行可能なルート探索では，運用シナリオが持つ潜在的な状態遷移をウォークスルーにおけるレビュー対象として追加することで，レビューの網羅性の向上を図る．アクティビティ図で示せる運用シナリオは，ある実行の一例に過ぎないことから，想定外の状態遷移が起こらないかは，これまで人が頭の中で想像することで評価をしていた．機械的に探索した結果を示せることで，レビュー時の網羅性の向上が図れる．コンパクト化と合わせて，このようにレビュー品質の網羅性及び有効性を向上させることにより，充実したレビューが実施でき，間接的には運用シナリオと宇宙機システムとの整合性の確認の向上にもつながる．

3.5 安全解析手法

本節では本研究で選択した安全解析手法としての STAMP/STPA 及び，SysML を用いて STAMP/STPA を実行するためのプロファイルである RAAML について説明する．

3.5.1 STAMP/STPA

安全解析手法は，異常状態を含むシナリオを作成する際の異常状態の抽出に使用する．一般的な安全解析手法としては，Fault Tree Analysis（FTA）や Failure Mode and Effects Analysis（FMEA）が有名であり，よく知られている．実際の宇宙機システム開発の現場においても，FTA 及び FMEA を使用しているが，本研究では，運用シナリオにおける異常状態を抽出するため，STAMP/STPA[26, 46] を選択した．その理由は，運用シナリオは複数システム間のやり取りが時系列に並んだものであり，単体のシステム評価より，複数システムのインタラクションに着目した安全解析手法が適切と判断した．特にコントロールストラクチャー図 (図 3.1) の分析から Unsafe Control Action(UCA) 算出，及び UCA に基づくシナリオ生成の部分を利用する．コントロールストラクチャー図は，分析対象の構造と関係性を表現する図である．分析対象はコンポーネントと呼ばれる要素で作成し，コンポーネント間の関係性を線で結ぶ．関係性には，コントロールアクションとフィードバックの 2 種類がある．コントロールアクションは制御元から制御対象に対しての指示であり，フィードバックはその制御によっての反応を示す．例えば，図 3.1 において，コンポーネントは衛星システム，他衛星 1〜3，地上システム，ユーザ，及び衛星システムの自律運用機能が該当する．コントロールアク

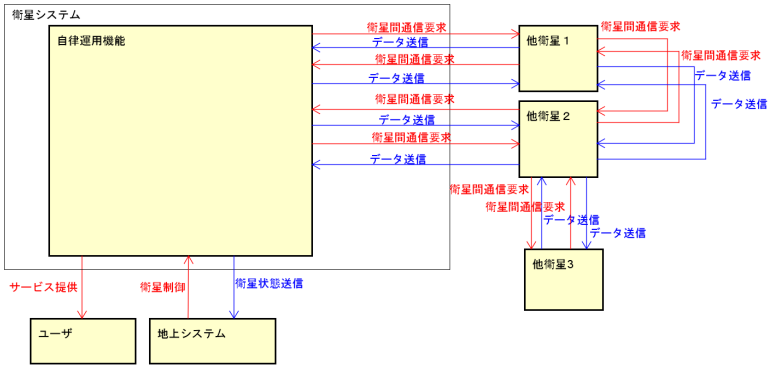


図 3.1: コントロールストラクチャー図の記述例

ションは赤矢印で示され，”衛星間通信要求”などである．フィードバックはコントロールアクションへの反応であり，青矢印で示した”データ送信”等である．

UCA はそのコントロールアクションにおいて，非安全となるアクションを指す．コントロールアクションとしての衛星間通信要求があるが，想定内のタイミングであればシステム全体として問題ないが，通信要求が想定より遅かった場合，軌道情報の予測誤差が増大し，通信確立に至らないなどのシステム全体としては失敗となることもある．STAMP/STPA は本来，システムの安全性を評価するためであり，UCA による非安全となるシナリオの想定と対策を検索する．先に示した例において，通信確立失敗が直接，衛星システムの非安全な状態ではないが，全体システムとして望まない想定外の事態である．よって，本研究では，非安全に拘らず，STAMP/STPA のシステム間の相互作業の評価の手法として，運用シナリオにおける想定外や異常事象の分析に用いている．

3.5.2 RAAML

Risk Analysis and Assessment Modeling Language (RAAML) Libraries and Profiles [37] は，安全解析を UML 及び SysML として実装するためのプロファイルとして OMG から仕様が提供されている [37]．RAAML が対象とする安全解析手法は，STAMP/STPA だけではなく，FTA や FMEA など一般的な主要な手法にも対応している．RAAML の前身は，SafeML であり，SysML の拡張及び MBSE における利用を目的に開発がされており [5, 6]，2022 年 12 月に RAAML として OMG より Verstion 1.0 が公開された．また，一部の市販の SysML ツールでは，追加のプロファイルとして RAAML に対応しており，容易に扱うことが出来る．

本研究では RAAML で利用可能なステレオタイプなどのプロファイルを活用しつつ，不足する部分を追加する方法でメタモデルを提案する．RAAML を活用する理由は，今後の将来性を視野に，他の機関と連携や，モデルの継続的な再利用を考えた場合，独自の定義は互換性が乏しく，メンテナンスに労力がかかる．そ

のため，円滑な利用をするためには，既存の仕組みは有効活用した方が利点が大きいと考えたためである．

3.6 アスペクト指向技術

アスペクト指向技術とは，ソフトウェアにおいて横断的関心事を分離しカプセル化することで，プログラムのモジュール性を高める技術である．横断的関心事とは，オブジェクト指向におけるクラスの場合，クラス間を横断するような機能や各クラスで共通的に利用される機能や処理である．一般的には，ロギング，エラー処理などがあげられ，複数のクラスで共通的に実行される機能で用いられることが多い．そして，アスペクト指向プログラミングとしては，JAVA における AspectJ が有名であり，よく知られた言語である [31]．

AspectJ には，Aspect，Advice，Pointcut，JoinPoint の 4 つの基本的な用語および仕組みにより実装される．

- Aspcet：横断的な機能であり，共通の処理を示す．
- Advice：実際に行われる処理そのものである．メインの処理とは別に記述する．
- Pointcut：Advice を適用する場所を記載する．どこのクラスで呼び出すかなどを指定する．
- JoinPoint：実際に共通的な処理が行われるポイントを指定する．

本研究では，アスペクト指向技術に着想を得て，ノミナルシナリオの実行をメイン処理として，異常状態を引き起こす UCA 及びそれに伴うシナリオを Aspcet として捉える．UCA 及びそれに引き起こされる異常状態のオフノミナルシナリオは，ノミナルシナリオとは独立しており，ノミナルシナリオ内のどこで発生するかも検討の余地がある．これをオフノミナルシナリオの作成に応用する．すなわち，AspectJ と対比すると，UCA 及びそれに関連する異常状態のシナリオによるアクションは Advice であり，Pointcut はノミナルシナリオであり，JoinPoint はノミナルシナリオのどこで異常が発生するかを示すポイントを示す．アスペクト指向技術を応用することで，ノミナルシナリオからオフノミナルシナリオへの派生を実現する．Advice はノミナルシナリオとは別のアクティビティ図で表現する．Pointcut は JoinPoint にて異常状態発生ポイントが示せるため省略する．JoinPoint は，ノミナルシナリオのアクティビティ図内に分岐点として設定する．この場合，アクティビティ図には JoinPoint という識別は無いため，新たにステレオタイプとして提案すると共に，JoinPoint が異常と正常の分岐であることからアクティビティ図上のディジョンノードで示す．

3.7 モデル検査

運用シナリオの実行可能ルート探索および状態遷移の列挙にモデル検査の仕組みを用いる．モデル検査の手法としては，Bounded model checking 手法を SAT ソルバーで実現して利用する．なお，本研究ではモデル検査の仕組みを利用するのであり，運用シナリオのモデル検査を行う研究ではない点には留意いただきたい．あくまでモデル検査の手法を利用して，ルートの探索および状態遷移の列挙を行い，グラフ化する．

3.7.1 Bounded Model Checking

有界モデル検査（Bounded model checking（BMC））は，対象システムの状態遷移に対して，指定した長さの実行パスにおいて性質を満たすかを検証する方法である [2, 3, 8, 4]．ここでの性質とは状態遷移の制約，初期状態，受理状態である．状態遷移と性質はブール論理式に変換することで，次項に示す SAT ソルバーを用いて論理式が充足されるかを判定することが可能となる．これにより特定の長さまでの実行パスにおいて不具合があるかを発見することができる．ただし，基本的には部分的な検証であり，検証結果により不具合があることは断定可能であるが，何も見つからなくても不具合がないこと検証したことにはならない．

BMC を用いたモデル検査は，時間と共にシステム内部状態が変化する動的システムを検査する [51]．システムの状態 s を長さ n ビット列で表わしたとした場合，1 回の状態遷移により別状態 s' に変化する．このような動的なシステムをクリプキ構造 M と呼び，$M = (S, I, T, L)$ で表わされる．集合 S はシステムの取りうるすべての状態の集合である．集合 I は初期状態の集合であり，$I \subseteq S$ となる．二項関係 T は状態遷移を示し，$T \subseteq S \times S$ となり，2 つの状態 $s, s' \in S$ が $(s, s') \in T$ を満たすならば状態 s から s' に遷移可能を意味する．ラベリング関数 L は $L : S \to P(A)$ であり，基本的な命題（原子命題）の集合を A とするとき，$P(A)$ は A のべき集合となる．ラベリングはシステムに観測結果を付加する方法であり，状態 $s \subset S$ に対して，集合 $L(s)$ は状態 s において成立する原子命題から成る．

本研究においては，運用シナリオのループ部分を除けば有限ステップであり，状態遷移は運用シナリオに登場する宇宙機システムや地上システムの状態遷移が対応する．運用シナリオの 1 ステップは，アクティビティ図のアクション要素として表現される [44]．そして，１つのアクション実行は，なんらかの状態変化を引き起こす．例えば，機器を ON にするアクションを実行した場合，機器状態が停止状態から稼働状態へ変化，あるは軌道状態にあれば軌道状態を維持する．宇宙機システムの運用シナリオは，極端に言えば，状態遷移を正しく行うための手順である．ただし，実際はシステムの不具合や外的要因により，想定外の遷移が発生する．問題はその想定外の遷移に対して，予め対策をして，システム設計に反映す

ること共に，異常状態からの復帰シナリオを準備しておくことである．これにより正常な状態を維持したミッション遂行が行われる．

3.7.2 SAT ソルバー

SAT ソルバーとは，充足可能性判定 (SAT) 問題を解くためのプログラムである [2, 3, 8, 4]．SATisfiability の頭文字 3 つを取って SAT と呼ぶ．SAT ソルバーによって解くために，BMC をブール論理式である連言標準形（ Conjunctive normal form（CNF））に変換する．CNF では，論理積，論理和，否定を用いて論理式を示す．複数の節（１つ以上のリテラルを論理和で連結したもの）を論理積で繋げていく形で表現される．本研究では，SAT ソルバーとして Python プログラムの Z3 を用いる．

第4章　レビュー改善方針

宇宙機システム開発における運用シナリオのレビューは，有識者によるエンジニアによって確認及び評価が行われる．各個人で確認する場合もあれば，複数人が集まって会議形式で行うこともある．どちらにせよ，パソコンの画面や印刷物を通じて実施する．開発の進捗と共に運用シナリオが詳細化され肥大化された場合，シナリオを１つの画面や１ページに収まらず，また細かいデータのやり取りも増えて，シナリオの本質が見えにくくレビューが困難となる．宇宙機システムの運用シナリオはシステム設計への要求になるため，開発の初期から正しく設計し，関係者間で認識を合わせることは，開発の後半においての不具合や手戻り防止に効果がある．近年は，複数衛星によるコンステレーション運用を用いたミッションなど，複数の小型衛星を組み合わせる新しい運用方法も増加すると予想される．そのため，今後ますます，１つ１つの宇宙機システムの自動化，自律的な稼働することが求められる．特に異常状態に対して復帰は，状態に応じた対応が求められるため，運用負荷が高い．異常を想定した運用シナリオも設計し，宇宙機システムの自律化として対応できれば運用負荷低減に繋がる．しかし，詳細化されて肥大化した運用シナリオが，異常状態も含めて検討をする場合，さらに運用シナリオが肥大化し，レビューが困難となることが予見できる．

運用シナリオが肥大化し，レビューが困難になる課題に対して，大きく２つのアプローチを提案する．１つ目は，レビュー用のコンパクト化である．異常状態を含まない運用シナリオ（ノミナルシナリオ）に対してアクティビティ図のコンパクト化に実現し，視認性に着目したエンジニアによるレビュー容易性向上を図る．２つ目は異常状態を含むオフノミナルシナリオ生成及び実行ルート探索による状態遷移の列挙である．UCA から異常状態を特定し異常状態を含むオフノミナルシナリオを生成し，BMC を用いてアクションの実行順序を網羅的に探索し状態遷移を列挙する．これらを MBSE 上で実現し，BMC に繋げる運用シナリオのレビュー方法の改善アプローチについて図 4.1 に示す．

図 4.1 上の左にある運用シナリオは，本レビューにおけるスタート地点であり，エンジニアが作成する最初のアクティビティ図である．レビュー用のコンパクト化においては，運用レイヤー定義メタモデルと運用ステレオタイプメタモデルを提案する．それ以外に，運用ステレオタイプメタモデルがコンパクト化において正しい設計となっているかを検証するためにコンパクト化検証モデルを作成する．さらに，それらの前提条件としてドメイン知識モデルを作成する．運用レイヤー定義メタモデルは，運用シナリオの記述粒度を宇宙機システムの開発レイヤーに沿

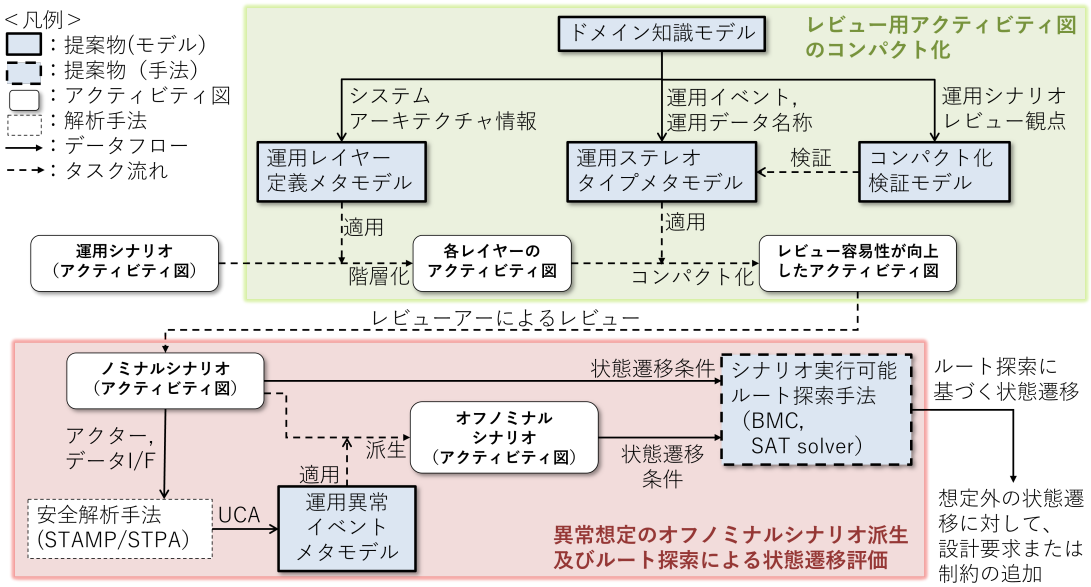


図 4.1: Overview of metamodels

うためのメタモデルである．運用レイヤー定義メタモデルを適用することで，開発レイヤーに沿ったアクティビティ図を生成する．次に，運用ステレオタイプメタモデルを適用する．運用ステレタイプメタモデルがコンパクト化を実現するメタモデルである．コンパクト化によりレビュー価値の低い一部の情報は削除されることとなる．レビューの観点において必要な情報まで削除されないかをコンパクト化検証モデルにて確認する．なお，コンパクト化検証モデルはメタモデルではない．これらの手順により，運用シナリオのレビューの容易性を向上する．

次は，異常状態を正常状態のノミナルシナリオに組込み，異常状態も含めたオフノミナルシナリオのレビューを実現する．前段による出力されたアクティビティ図に対して，STAMP/STPA を用いて UCA を抽出する．運用異常イベントメタモデルは，UCA により正常状態から異常状態への分岐，UCA に伴う異常シナリオ，及び異常シナリオからの復帰アクションの関係性を整理し，正常のノミナルシナリオから異常状態を含むオフノミナルシナリオの生成をガイドする．これにより異常状態への遷移を含むオフノミナルシナリオとしてのアクティビティ図を作成する．異常状態に遷移するパターンは過去の知見や経験則，現状の設計前提等をエンジニアが検討して設定する．そのため，異常時のフローではあるものの，ケーススタディの 1 ケースに過ぎない．実際には，実行パターンの順序，どのアクションの後に UCA が発生するかなど，バリエーションが多く存在し，それらをすべてアクティビティ図で書き出すと，膨大になりすぎて，再びレビュー困難に陥る．そこで，BMC を用いて，運用シナリオの各アクションにおける状態遷移が成立する組合せを探索することで，有限のステップ数における実行可能なアクションの順序を網羅的に入れ替え，１つのシナリオから可能性のあるアクションの実行順序

（ルート）およびその時の状態遷移を網羅的に列挙する．エンジニアは，想定した１つの運用シナリオから，全網羅的な状態遷移を確認することができ，想定外の状態遷移が起こっていないか，そしてその状態遷移はどの順番でアクションを実行した場合に発生するかを評価することができる．想定外の状態遷移は，設計上の考慮漏れである可能性があるため，未然の不具合抽出及び宇宙機システムの信頼性向上へとつなげることができる．

第5章　運用シナリオコンパクト化メタモデル

本章においてはレビュー用に運用シナリオをコンパクト化することを目的としたSysMLのメタモデルを提案する．

5.1　メタモデル設計

コンパクト化のためのメタモデルについて，設計における各モデルの関係性を図5.1に示す．宇宙機システムの運用シナリオへの適用を前提ととして，ドメイン知識(Spacecraft Domain knowledge)として宇宙機システムの設計や運用など用語を可視化する．運用メタモデル（Operational metamodels)は，運用レイヤーメタモデル(Operational layer metamodel)と運用ステレオタイプメタモデル(Operational stereotype metamodel)の2つを設計する．利用者はこれらの運用メタモデルを用いることでコンパクト化したアクティビティ図を作成する．[42, 43]

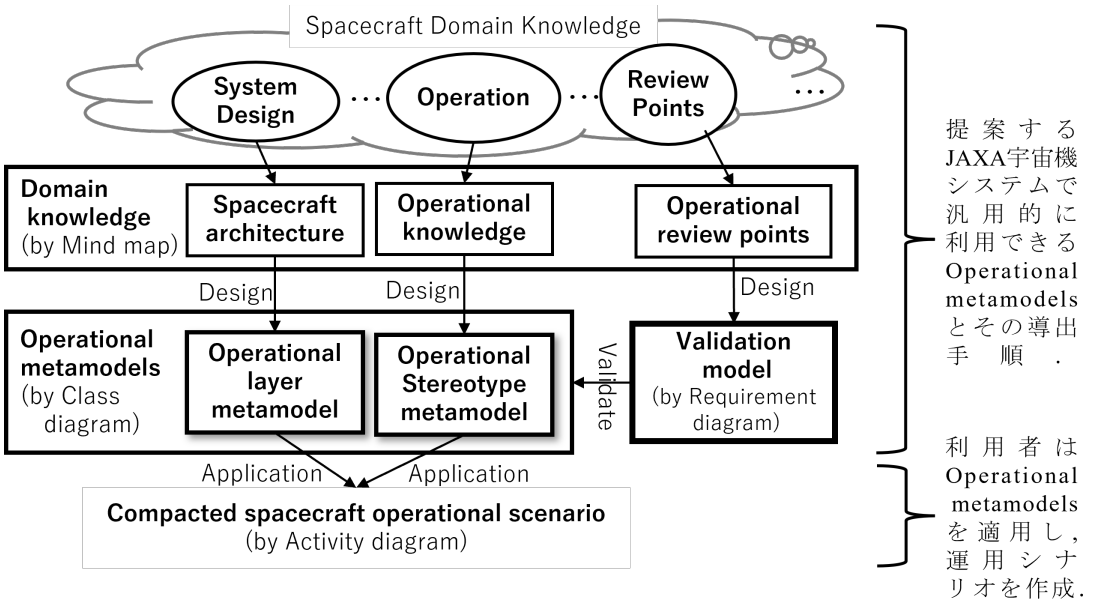


図 5.1: コンパクト化メタモデル設計における各モデルの関係性

運用レイヤーメタモデルは，運用シナリオのレイヤー構造と粒度をメタモデルとして定義する．システム，サブシステムなどの宇宙機システムの開発上のアーキテクチャ（Spacecraft architecture）情報を用いて，宇宙機システムと運用シナリオのレイヤー構造を整合させ，アクティビティ図上に配置可能なアクターとアクションを指定することで粒度を制御する．システムズエンジニアリング [30] によれば，システム開発を階層的に開発を進めるうえで，開発対象のレイヤーは 1 つ下位のレイヤーに対して機能配分を行うため最低限 2 階層あれば良く，それを繰り返すことによりシステム全体が開発される．実際のシステム開発においても，システムレイヤーを担当する設計者はサブシステムへの機能配分を行い，サブシステムの要求仕様を決定するなど，1 つずつ下位のレイヤーへと設計が進められていく．宇宙機システム開発の場合，経験上，ミッション全体からハードウェア及びソフトウェアへの設計要求までを階層化すると，全体では 4 階層程度となる．

運用ステレオタイプメタモデルは，運用シナリオにおいて汎用的で共通認識が取れている省略可能な振る舞いを，SysML のステレオタイプとタグ付き値で設計し，メタモデルとして定義する．汎用的で共通認識が取れている省略可能な振る舞いは，本研究において *Common behavior* と呼ぶ．宇宙機システムはミッションに応じて新規設計の部分もあるが，信頼性の観点から実績のある設計や運用手順を再利用する部分も多い．そのため，アクティビティ図のレビューでは，Common behavior とならない，新規設計や再利用元からの変更箇所に重点が置かれる．変化があった箇所は不具合となることが多いためである．そのため，既存設計を踏襲する箇所や標準化により不変な箇所である Common behavior 部分はレビューの価値が低くく，省略可能の候補となる．このことに着目して，運用中に発生するイベント，データ種別及びデータ名称などの過去の運用知識（Operational knowledge）から Common behavior を抽出し，ステレオタイプとタグ付き値で設計する．

検証モデル (Validation model) は，運用メタモデルがレビューに必要な情報を含むアクティビティ図が生成可能かを検証するために作成する．関係者が持っているレビュー観点（Operational review points）をドメイン知識として収集し，運用メタモデルに対する要求として抽出する．検証モデルでは抽出したレビュー観点である要求と運用メタモデルがその要求に対応した設計になっているかを，SysML の要求図を用いて確認する．もし，要求に対応していない箇所が発見された場合，運用メタモデルの設計不備として改良を行う．このように予め，運用モデルを利用者に提供前に，コンパクト化後もレビューに必要な情報が保たれることを評価する．

5.2 ドメイン知識モデル

ドメイン知識として，運用レイヤーメタモデルのために宇宙機システムのアーキテクチャ設計情報，運用ステレオタイプメタモデルのために運用知識，検証モデルのために運用モデルのレビュー観点を用いる．これら情報は関係者や設計文

書，開発標準文書等から収集し，整理及び分類するためにツリー構造で可視化し，これをドメイン知識モデルと呼ぶ．運用はアクターからアクターへデータを渡し，処理することの連続で構成されることから，ドメイン知識モデルではアクターとしてシステムの呼称とレイヤー構造，受け渡すデータとしてデータの呼称と種別，及びそれらに関わるイベントの呼称等を定義する．また，本研究では，ツリー構造を示すうえで，マインドマップソフトウェアを用いて描いた図を例として示す．

宇宙機システムのアーキテクチャ設計情報は，図5.2に示す通り，最上部をSystem of Systems (SoS) レイヤーとし，System，Subsystem，Component，ハードウェア (H/W)，ソフトウェア (S/W) 等の開発上のレイヤー構造を示すと共に，各階層に含まれる具体的なシステムやサブシステム名称とその親子関係を明らかにする．図 5.2 は 4 層構造としたが，システムの開発規模により増減するため，予め設計前提を可視化により明らかにする．加えて，相対的なシステム，サブシステムという呼び名ではなく，具体的なレイヤー名称を定義することで，全体を統括した際に，一意にどこのレイヤーの議論であるかを定めることが可能となる．例えば，System layer では，親要素が Spacecraft system であり，子要素が Attitude and Orbit control，Data Handling などのサブシステムになる．下位のレイヤーや Ground system，User の別システムも同様に展開する．

運用知識は，運用中のインタフェースに着目したイベント名称，データ名称やデータ種別とし，マインドマップによりモデル化する．例えば，図5.3の Communication events は通信に関するイベントであり，Common Behavior を区別するために，通信相手に応じて通信レイヤーの名称（Downlink，Onboard network 等），データ通信の種別（RF band, Space-Wire 等）で細分化する．Common Behavior は過去の JAXA 宇宙機システムの運用において，個々の宇宙機システムのミッションに関係なく必要な振る舞いとする．Common Behavior の設定は，関係者の知見に加えて，標準文書 [20, 21, 23, 22] への記載の有無を参考にする．これら標準文書にはシステム開発において共通的に適用されるため，この文書への記載内容は JAXA 宇宙機システムに共通的に存在することがほとんどである．例えば，地球観測衛星では，探査機のような惑星着陸ミッション運用は求められないが，それを実現する振る舞いとして地上から宇宙機システムへのコマンドデータによる制御手順があり，コマンドのデータのやり取りは共通の振る舞いとなり，Common behavior となる．

レビュー観点 (図 5.4) は，関係者がレビュー時に着目している点であり，運用上の前提条件や制約，システム設計との整合性確認などである．これらは暗黙知や経験に基づく部分が多いため，ヒアリング等を通じて収集し，マインドマップ上で可視化を行う．詳細は，5.5 節の検証モデルにて説明する．

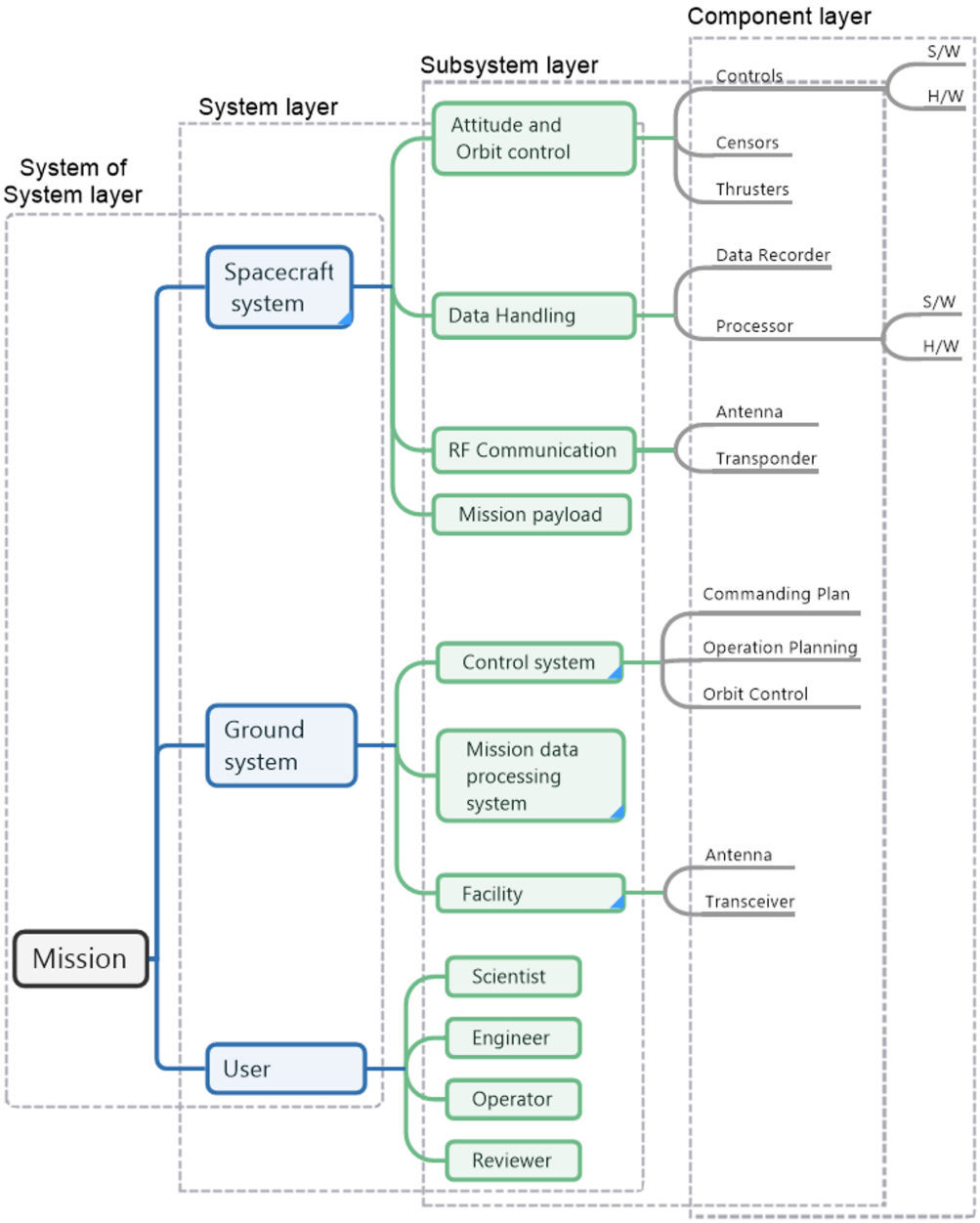


図 5.2: システム構造ドメイン知識

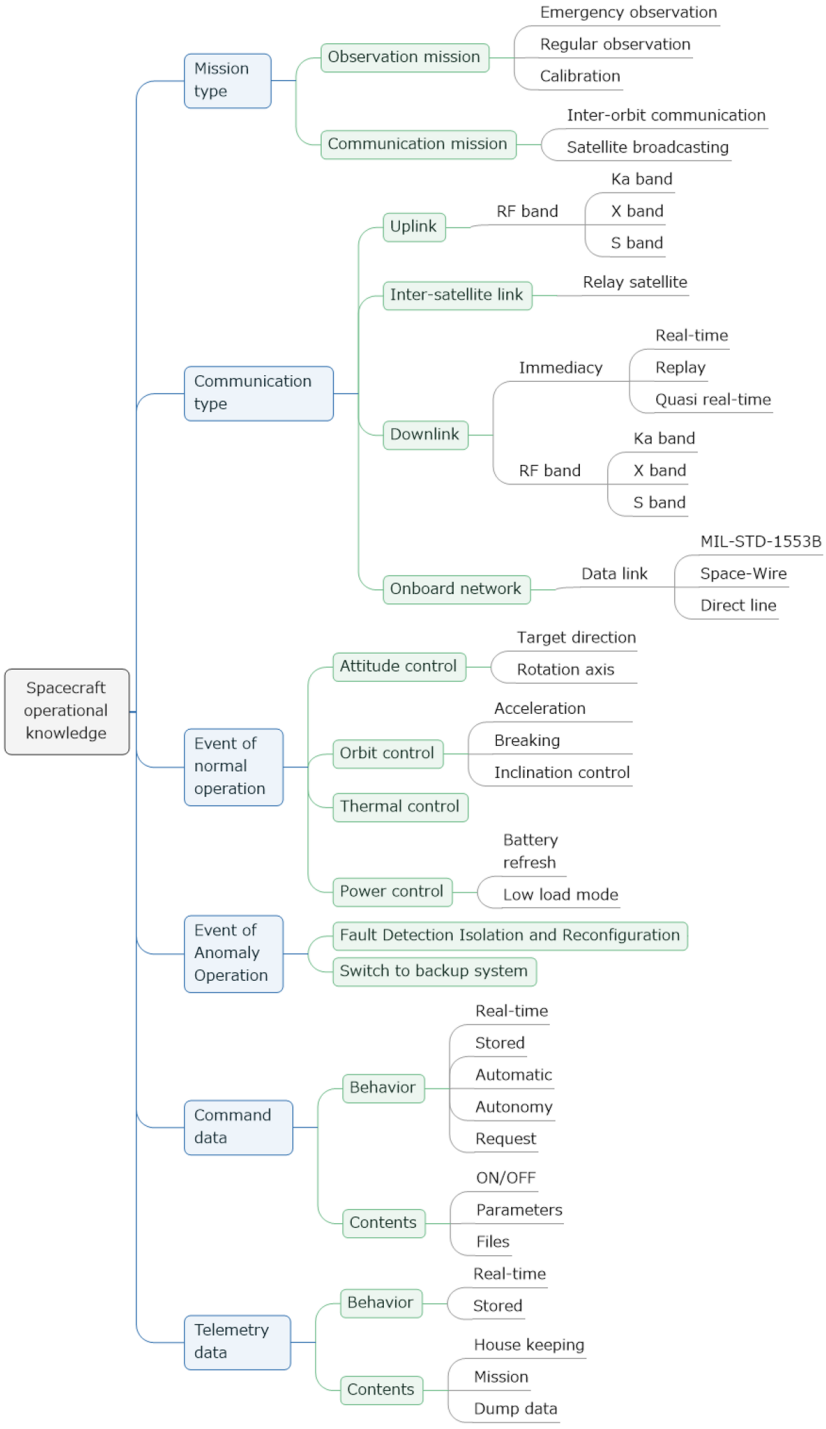

Spacecraft operational knowledge
Mission type
Observation mission
Emergency observation
Regular observation
Calibration
Communication mission
Inter-orbit communication
Satellite broadcasting
Communication type
Uplink
RF band
Ka band
X band
S band
Inter-satellite link
Relay satellite
Downlink
Immediacy
Real-time
Replay
Quasi real-time
RF band
Ka band
X band
S band
Onboard network
Data link
MIL-STD-1553B
Space-Wire
Direct line
Event of normal operation
Attitude control
Target direction
Rotation axis
Orbit control
Acceleration
Breaking
Inclination control
Thermal control
Power control
Battery refresh
Low load mode
Event of Anomaly Operation
Fault Detection Isolation and Reconfiguration
Switch to backup system
Command data
Behavior
Real-time
Stored
Automatic
Autonomy
Request
Contents
ON/OFF
Parameters
Files
Telemetry data
Behavior
Real-time
Stored
Contents
House keeping
Mission
Dump data

図 5.3: 運用知識

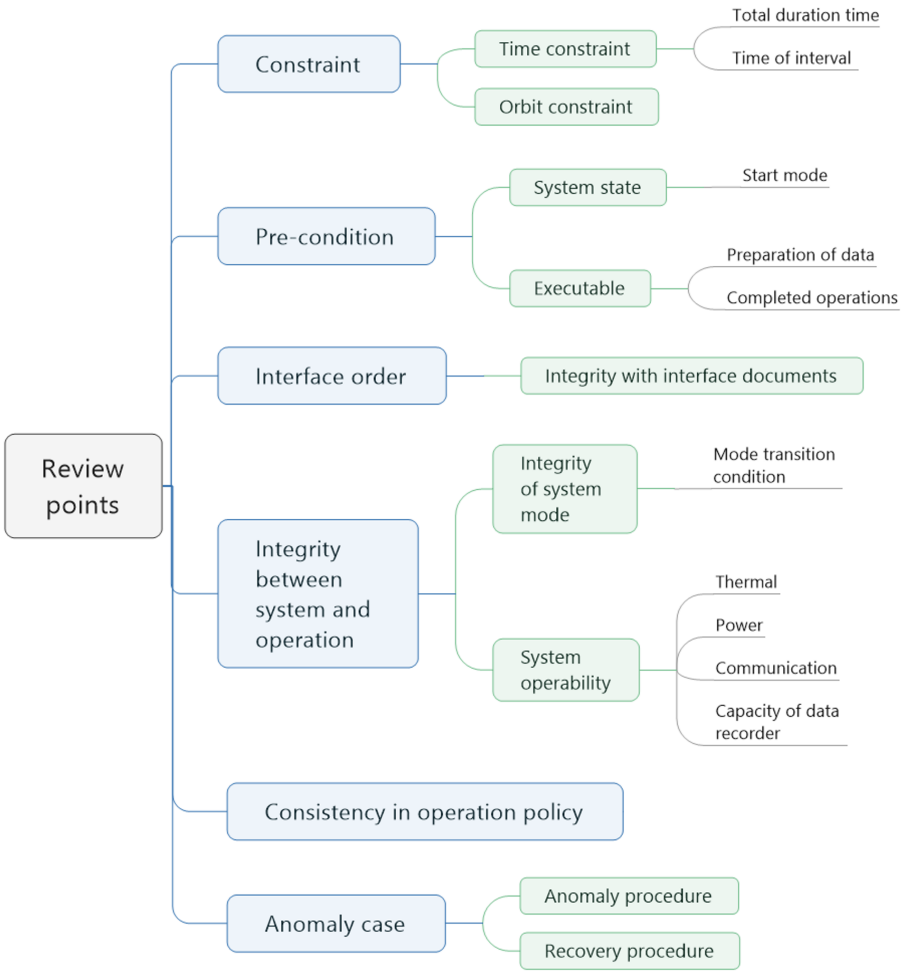

Review points
Constraint
Time constraint
Total duration time
Time of interval
Orbit constraint
Pre-condition
System state
Start mode
Executable
Preparation of data
Completed operations
Interface order
Integrity with interface documents
Integrity between system and operation
Integrity of system mode
Mode transition condition
System operability
Thermal
Power
Communication
Capacity of data recorder
Consistency in operation policy
Anomaly case
Anomaly procedure
Recovery procedure

図 5.4: レビュー観点

5.3　運用レイヤー定義メタモデル

あくてぃび運用レイヤー定義メタモデルは，運用シナリオの階層化のために，各開発レイヤーとアクティビティ図の対応関係を定義する（図 5.5）．各開発レイヤーは，ドメイン知識モデルの宇宙機システムにおけるアーキテクチャ情報を用いて 4 層構造とする．本メタモデルの最上部は，Layered acitivity diagram として，アクティビティ図を開発レイヤーに応じて階層化することを示す．具体的なアクティビティ図は，System of Systems，System，Subsystem，Component に 4 つのレイヤーに対応したアクティビティ図を作成する．各アクティビティ図は，最上位を System of systems として，下位のレイヤーに行くにつれて詳細化されていく関係にある．そして，重要な点は，システムズエンジニアリングに従い，各アクティビティ図には上位レイヤーと下位レイヤーの 2 つのアクターが Swim lane として設定する部分である．下位のアクティビティ図と上位のアクティビティ図では，必ず共通するアクターが含まれる構造となる．この共通するアクターを通じて，各レイヤー間をつなぐことで，システム全体の整合性を取る．このことを，メタモデル上では各アクティビティ図間に trace の関係性を持つことで示している．また，各アクティビティ図は，設計対象となるレイヤーと 1 つ下の階層のレイヤーが登場し，それに対応したアクションを配置する．アクション要素を 0 個以上持つとして，運用シナリオの 1 ステップ毎の処理内容が記載される．

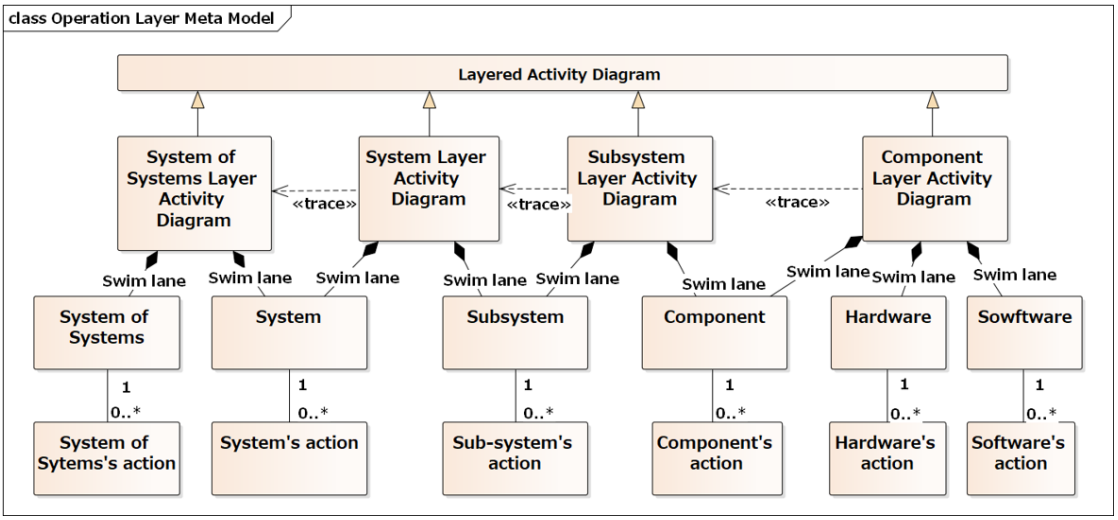


図 5.5: 運用レイヤー定義メタモデル

表 5.1: 運用レイヤー定義メタモデルにおける各項目説明

項目	カテゴリ	説明
Layered activity diagram	Diagram	宇宙機システムの各開発レイヤーに対応する階層化のアクティビティ図.
System of Systems layer activity diagram	Diagram	総合システムレイヤーのアクティビティ図．システム間のやり取りを示す.
System layer activity diagram	Diagram	システムレイヤーのアクティビティ図．サブシステム間のやり取りを示す.
Subsystem layer activity diagram	Diagram	サブシステムレイヤーのアクティビティ図．コンポーネント間のやり取りを示す.
Component layer activity diagram	Diagram	コンポーネントレイヤーのアクティビティ図．ハードウェア，ソフトウェア間のやり取りを示す.
System of Systems	Actor	総合システム．複数システムをまとめた総称.
System	Actor	単体のシステム．例えば，宇宙機システム.
Subsystem	Actor	サブシステム．システムの構成要素.
Component	Actor	コンポーネント．サブシステムの構成要素.
Hardware	Actor	ハードウェア．物理的な実装部品.
Software	Actor	宇宙機システム搭載のソフトウェア.
Action	Element	アクション要素．個々の振る舞いを示す.
Swim lane	Element	アクティビティ図で使用する縦割りのパーティションであり，アクターを割り当てる.
Trace	Relationship	下位レイヤーが上位レイヤーのアクティビティ図に対してトレースが取れることを示す

このようにレイヤーに対応したアクテビティ図と，アクティビティ図上のアクターを予め定義することにより，記述粒度を揃える．これによりCommon behaviorとなる一連の振る舞いの記述粒度も揃い，後段の運用ステレオタイプメタモデル適用の効率を上げ，コンパクト化の効果を上げる狙いがある．提案するコンパクト化の手法は，一連の繋がったアクション要素を置き換えるため，コンパクト化の効率を上げるためには，粒度が揃っており，同じアクションの並びが複数回出現することが望ましい．一連のアクション要素の途中に下位や上位のレイヤーである粗すぎるまたは細かすぎる内容のアクションが挿入された場合，似て非なる一連のアクション要素は，予め設計したCommon behaviorと異なり，置き換えが困難となる．すなわち，要素数の削減効果は低減する．コンパクト化の第一歩としては，アクティビティ図の粒度の整理が必要である．

5.4 運用ステレオタイプメタモデル

運用ステレオタイプメタモデルでは，Common behaviorに対応するステレオタイプを定義し，コンパクト化により消えたレビューに必要な情報をタグ付き値として設計する [44]．提案するメタモデルを図5.6，本メタモデル含まれる各要素を表5.2に示す [42, 53, 43]．カテゴリーは，Meta-calss，Stereotype，Tagged value，Enumerationがある．Meta-classはUMLで定義されている要素である．Stereotype及びTagged ValueはUML拡張機能を用いてコンパクト化のために本研究で追加した要素である．Enumerationは列挙要素であり，ドメイン知識からエンジニア間で共通認識が取れている用語を設定し，選択制にすることで言葉の揺らぎを抑制し，記述粒度を制御する．

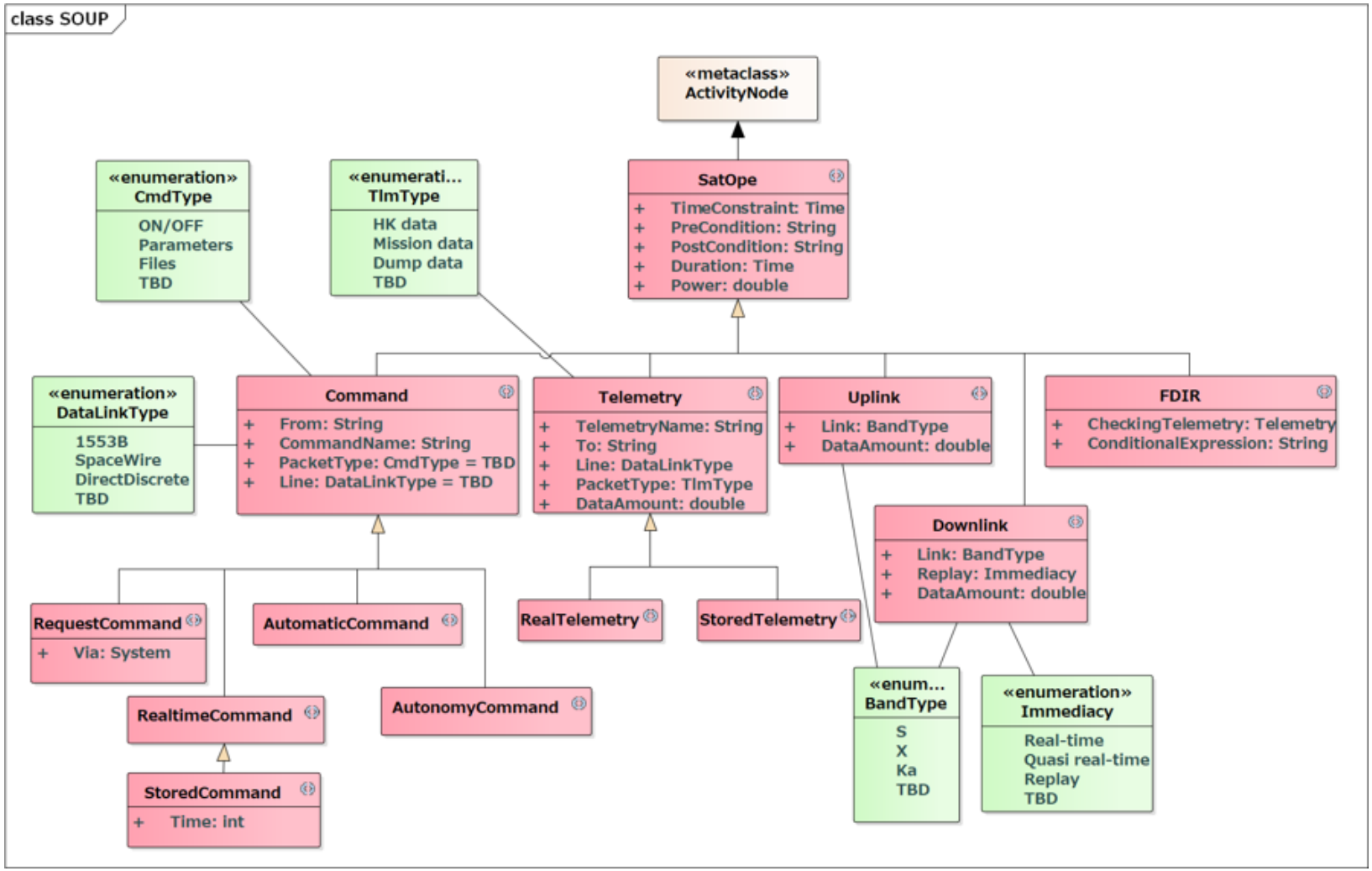


図 5.6: 運用ステレオタイプメタモデル

表 5.2: 運用ステレオタイプメタモデルにおける各項目の説明

項目	カテゴリ	説明
ActivityNode	Meta-class	UML のメタクラスであり，アクティビティ図で使用する要素の総称である.
SatOpe	Stereotype	各ステレオタイプで共通的に使用されるタグ付き値を集約.
Command	Stereotype	地上システムから宇宙機システムに送信する指令である.
RealtimeCommand	Stereotype	コマンド種別の一つであり，地上システムから受信したコマンドは宇宙機システムで即時実行される.
StoredCommand	Stereotype	コマンド種別の一つであり，地上システムから受信したコマンドは，時限付であり，指定した時間に実行される.
AutomaticCommand	Stereotype	コマンド種別の一つであり，予め宇宙機システムに設定した一連のコマンドが実行される.
AutonomyCommand	Stereotype	コマンド種別の一つであり，宇宙機システムが機器状態を評価し，自律的に実行する.
RequestCommand	Stereotype	コマンド種別の一つであり，宇宙機システム内で他のコマンドを呼び出すコマンドである.
Telemetry	Stereotype	宇宙機システムから地上システムに送信されるデータや機器情報.
RealTelemetry	Stereotype	テレメトリ種別の一つであり，リアルタイムで宇宙機システムから送信される.
StoredTelemetry	Stereotype	テレメトリ種別の一つであり，宇宙機システム内のレコーダーに一時的に保存され，地上システムからの指示等で送信される.
Uplink	Stereotype	地上システムから宇宙機システムへの通信回線であり，コマンド等を送信する.

項目	カテゴリ	説明
Downlink	Stereotype	宇宙機システムから地上システムへの通信回線であり，テレメトリやミッションデータを受信する.
FDIR	Stereotype	Fault Detection, Isolation, and Recovery の略であり，宇宙機システムが自身の機器状態等を確認し，閾値を越えたタイミングで予め設定していたシーケンスを実行する.
TimeConstraint	Tagged value	シナリオ上の時間制約.
PreCondition	Tagged value	事前条件であり，アクションやシナリオを実行時の事前条件.
PostCondition	Tagged value	アクションやシナリオの実行後にあるべき状態の姿.
Duration	Tagged value	アクションの実行に係る時間
Power	Tagged value	アクション実行時に消費または発生の電力.
CommandName	Tagged value	実行するコメントの名称.
From	Tagged value	アクションにおけるデータの送信元
PacketType	Tagged value	コマンドやテレメトリに含まれるデータ分類.
Line	Tagged value	通信回線の種別.
Via	Tagged value	経由するアクター.
Time	Tagged value	ストアードコマンド等の実行タイミング.
TelemetryName	Tagged value	テレメトリの名前.
To	Tagged value	テレメトリの送付先のアクター.
DataAmount	Tagged value	データ量.
Link	Tagged value	無線通信で使用する周波数帯.
Replay	Tagged value	ダウンリンクにおいて経由箇所がある場合のアクター.
CheckingTelemetry	Tagged value	FDIR において確認した機器の情報.
ConditionalExpression	Tagged value	FDIR において障害復旧のトリガー条件.

項目	カテゴリ	説明
CmdType	Enumeration	コマンド内容の種類：ON/OFF, パラメータ値, ファイル形式, TBD.
TlmType	Enumeration	テレメトリ内容の種別：ハウスキーピング（HK），ミッションデータ，ダンピングデータ, TBD.
DataLinkType	Enumeration	データリンクの通信手段：1553B, Space-wire, TBD.
BandType	Enumeration	周波数帯：S, X, Ka, TBD.
Immediacy	Enumeration	ダウンリンクにおける即時性: リアルタイム real-time, 準リアルタイム quasi real-time, 衛星間通信経由 replay, TBD.

5.4.1 Common behavior のステレオタイプ化

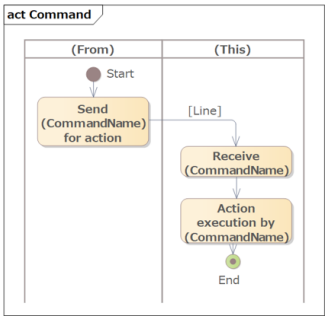
Common behavior は，関係者間で共有認識が取られた振る舞いであり，ドメイン知識モデルの運用イベント名やデータ名により特定でき，かつシナリオ上で複数出現し，イベント名が同じでデータ名だけが異なるような特徴を持つ共通する一連のアクションとする．ステレオタイプ名称には Common behavior の名称，すなわちをドメイン知識モデルの運用イベント名やデータ名から命名する．Common behavior の具体的な振る舞いである一連のアクションの並びは図 5.7 に示す通り，アクティビティ図として別途定義しステレオタイプと関連付けす．運用ステレオタイプメタモデルの適用時は，Common behavior のアクティビティ図上のアクター “This” に対応したスイムレーン上の重要なアクションのみを残し，他のアクションは削除する．ステレオタイプに付与するタグ付き値には，ステレオタイプ適用により削除した情報などのレビューに必要な情報を設定する．

例えば，図 1.1 はコマンドを使った Common behavior が含まれており，送信（send），受信（receive），実行（execution）の 3 ステップで示される一連のアクションで表わされている．レビューの観点においては，コマンドが実行されることによって，どのような動作が発生するかであるため，重要な要素は “X2 action exectution by command” アクションとなる．この一連のコマンドの振る舞いに対して，Common behavior を定義したのが図 5.7a である．Common behavior としては，Start 要素から始まり，End 要素で終わる．アクションは，Send, Receive, Action であり，図 1.1 と同じ 3 ステップとなる．スイムレーンは From アクターと This アクターを割り当てる．From アクターは Command の送信アクターを示し，

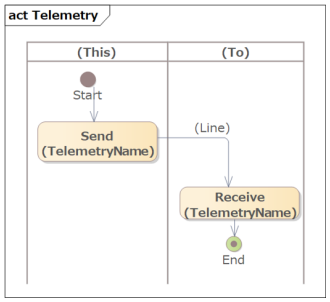
This アクターはこのステレオタイプを適用するアクションのアクターであり，レビュー観点から重要な要素となるアクション要素である．具体的に，図 1.1 に適用する場合は，Command ステレオタイプは “X2 action exectution by command” 要素に適用し，From アクターには “Ground Sys.”，This アクターは “Satellite Sys.” となる．そして，Command ステレオタイプを適用することにより，図 1.1 の send アクションと receive アクションを削除し，シナリオ上，重要な実行の execution に関わるアクションのみを残す．send アクションの削除により，コマンドの送信元の情報も失われる．運用シナリオをレビューする際において，コマンドの送信元情報は必要である．そのため，Command ステレオタイプの From タグにより送信元のアクター名を記述することで情報を復活させる．Command name タグはインタフェースのデータ項目を強調するために設計したタグである．他のステレオタイプも Command ステレオタイプ同様に，Common behavior をアクティビティ図で定義し，レビューに必要な情報等をタグ付き値として設計する．

Telemetry common behavior（図 5.7b) は，Send と Receive の 2 つのアクション要素であり，Send アクションが重要なため Send アクションのアクターが This となり，受信側のアクターが To となる．Stored command common behavior(図 5.7c) は，メタモデル上，Command ステレタイプの下位要素である．そのため，基本的な振る舞いは似通っているが，スタート要素の次にタイマーの要素がある点で差異がある．Stored command は，時限付コマンドとも呼ばれ，時間指定で実行されるコマンドである．そのため，Common behavioer に砂時計マークの時間に関する要素を追加する．Request command の common behaivor(図 5.7d）は中継のアクターが存在する振る舞いを行うため，スイムレーンを 3 分割として中継役のアクターである Via が command common behavior と比較した差異となる．Uplink common behavior（図 5.7e は，command を地上システム（Ground system）から地上のアンテナ局（Tracking station）を経由して，宇宙機システムへの送信を示す振る舞いである．This はシナリオの粒度に応じて可変となるため固定アクターせず，例えば，宇宙機システムや宇宙機システムのサブシステムに置き換えることが可能とする設計とした．Donwlink common behavior（図 5.7f）は，telemetry を地上システムで受信する振る舞いであり，Uplink common behavior と逆の情報の流れとなる．FDIR common behavior（図 5.7g) は，宇宙機システムの状態を示す telemetry を受信し，その値に基づきチェックを行い異常と判定した場合は，異常状態からの切り離し及びリカバリー処理を行う．なお，FDIR とは Fault Detection, Isolation and Recovery の略であり，宇宙機システムに備わっている安全装置の 1 つである．異常を検知すると予め定められたリカバリー対処を自律的に実行する機能である．

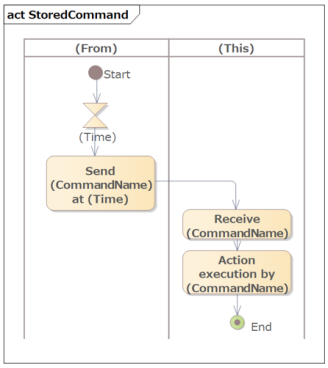
運用ステレオタイプ定義メタモデルのステレオタイプのうち，図 5.7 と同じ名称の Common behavior を持たないステレオタイプもある．各ステレオタイプは継承関係を持っており，上位のステレオタイプで定義された Common behaivor と同様の Common behaivor を持つステレオタイプは新たに Common behaivor を定義し



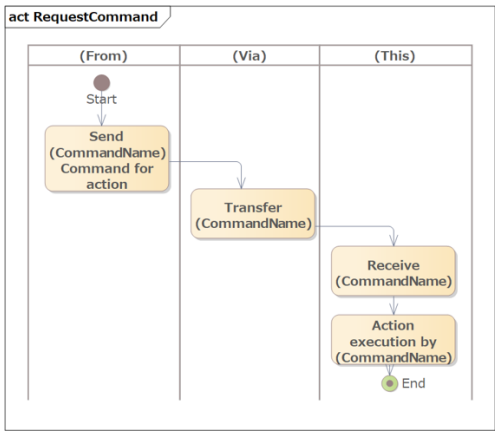
(a) Command common behavior



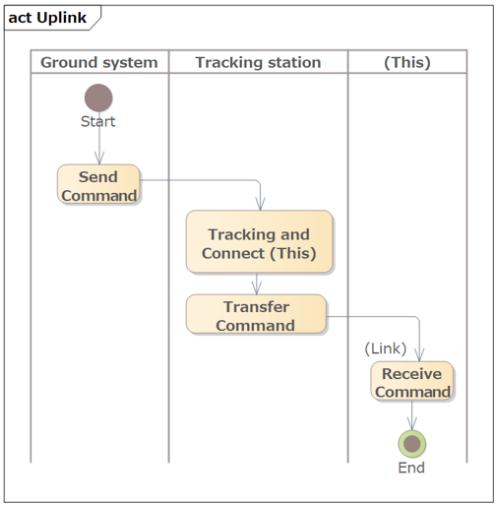
(b) Telemetry common behavior



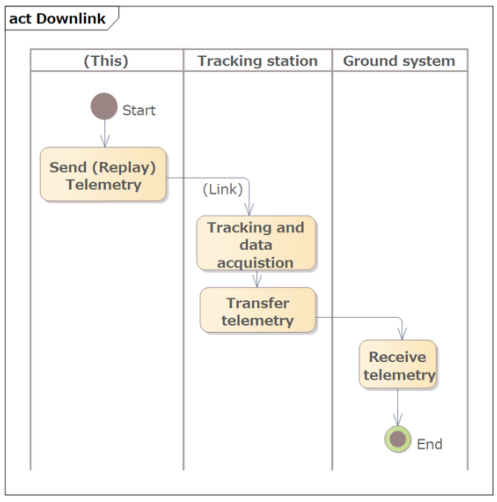
(c) Stored command common behavior



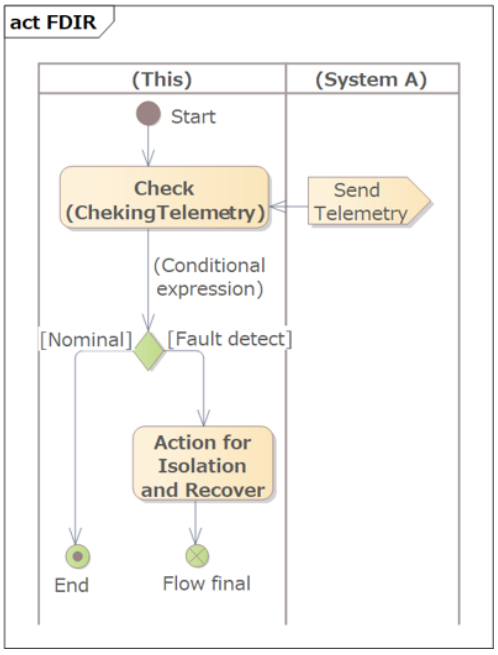
(d) Request command common behavior



(e) Uplink common behavior



(f) Downlink common behavior



(g) FDIR common behavior

図 5.7: 各 Common behavior の振る舞い

ない．その場合は，上位のステレオタイプを辿って，初めて出てきたCommon behaviorが継承される．また，上位で定義したタグ付き値は下位の全ステレオタイプで使用できる．そのため，全ステレオタイプで共通するTagged Valueは，SatOpeステレオタイプに集約した．そのため，SatOpeステレオタイプにはCommon behaviorを持たない．例えば，Automatic commandステレオタイプは，commandステレオタイプと同様のCommon behaviorと設計したため，Automatic command common behaviorとしてはcommand common behavior(図5.7a）を適用する．上位のCommond behaviorを参照するステレオタイプの特徴は，振る舞いが上位のステレオタイプと同一なため，独自のタグ付き値を持たない．

5.4.2　運用用語一覧化

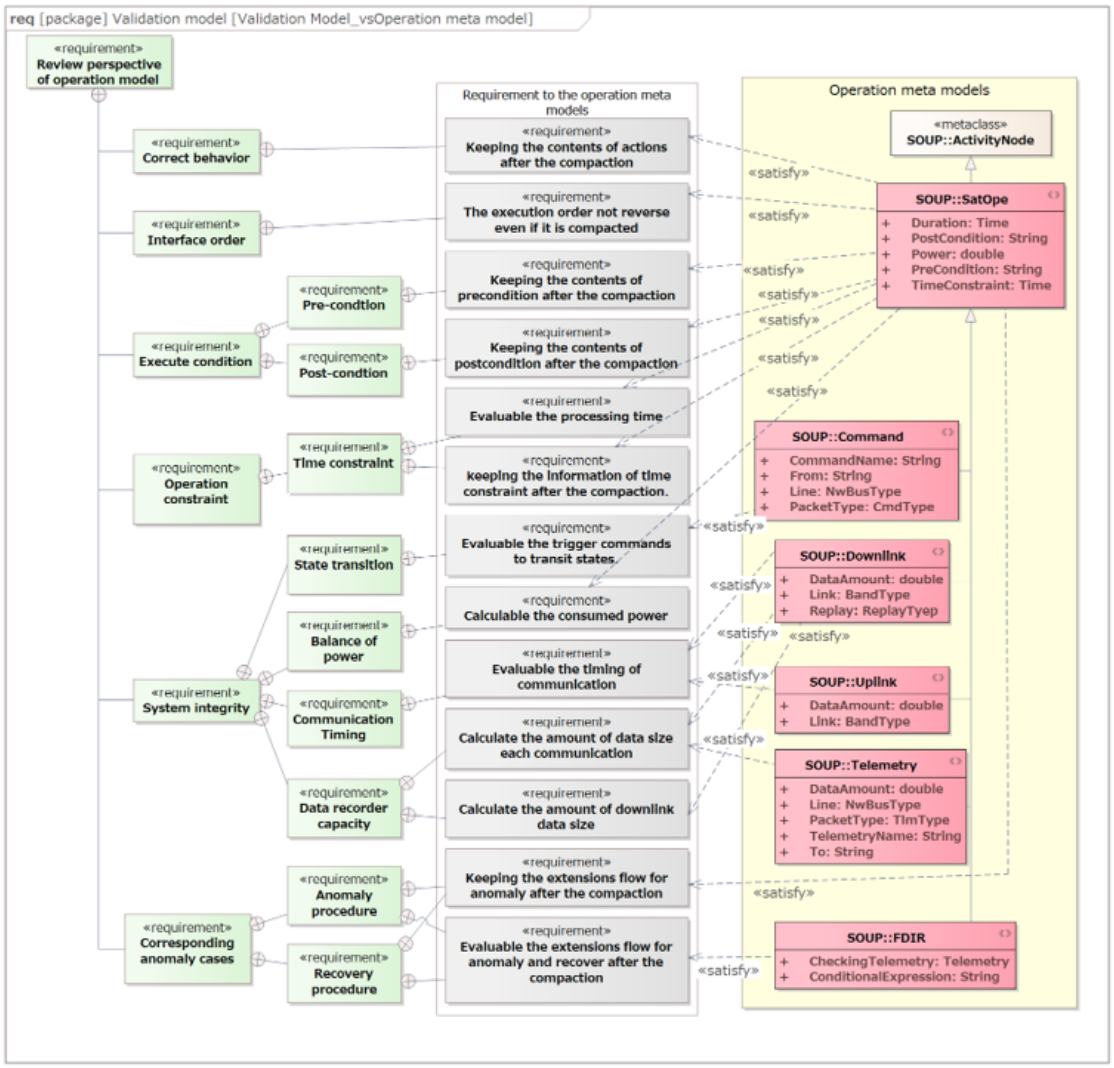
列挙型enumerationは，タグ付き値で設定される値をリストとして予め定義する．開発において一貫した用語を使用し，似て非なる用語による誤認を低減するため，マインドマップより選定する．また，本研究としては，TBD (To Be Determinate)を選択肢の一つとして加え，開発が未完であることが強調可能な工夫をする．例えば，Data link typeはドメイン知識モデルの運用知識（図5.3）のCommunictation typeのOnbord network配下のData linkに基づく通信規格にTBDを加えてリスト化する．CmdTypeはコマンドタイプのことであり，運用知識（5.3）のCommand dataのContentsをリスト化した．TlmTypeはテレメトリタイプであり，運用知識（5.3）のTelemetry dataのContentsをリスト化した．それ以外のBand typeはRF band，ImmediacyはDonwlinkのImmediacyであり，運用知識のマインドマップより選択肢が予め定まっているものを設計する．

5.5　メタモデル検証

コンパクト化検証モデルは，ドメイン知識モデルのシナリオレビュー観点に基づき，運用メタモデルがコンパクト後もレビューに必要な情報が保持できる設計であるかを評価するモデルである [42, 53, 43]．コンパクト化は，ステレオタイプの適用により情報が削除され，タグ付き値で情報が復活する仕組みである．よって，検証では，運用メタモデルのうち，主に運用ステレオタイプメタモデルを対象に，レビューに必要な情報を記述できるタグ付き値が存在するかを評価する．不足が見つかれば運用ステレオタイプメタモデルを修正する．利用者に対しては，本評価を経ることで，保証済みの運用ステレオタイプメタモデルとして使用することができる．

コンパクト化検証モデルは，図5.8に示す通り，SysMLの要求図を用いる．マインドマップのレビュー観点（図5.4）に基づき運用ステレオタイプメタモデルへの要求を要求要素（requirementステレオタイプを付けた要素）として配置する．

要求は，ネスト関係により詳細化する．最下層の要求は，それに対応する運用ステレオタイプメタモデルのタグ付き値が存在するかを確認し，存在する場合は充足（Satisfy）の関係性を取る．すべての最下層の要求に対して Satisfy の関係が得られることで，レビュー観点に対応した運用ステレオタイプメタモデルであると評価する．



SOUP：運用ステレオタイプメタモデルを示す.

図 5.8: 運用メタモデルの検証

例えば，図5.4 の System Integrity 要求は，図5.4 の Integrity between system and operation 要素及び子要素をまとめた要求である.「トリガーとなるコマンドとそれに伴うシステム状態の変化がシナリオ上，評価可能であること（Evaluable the trigger commands to transit states）」は System Integrity を詳細化した要求である．このトリガーとなるコマンドに対して，Command ステレオタイプに CommandName タグがあり，コマンド名称により把握可能として Satisfy の関係性を引く．他の要

求についても同様に詳細化と Satisfy の関係を確認する．

5.6 適用方法

アクティビティ図をコンパクトするには，運用レイヤー定義メタモデル及び運用ステレオタイプメタモデルの順に適用する．準備段階として，運用レイヤー定義メタモデルを参照し，運用シナリオが対象とするアクティビティ図のレイヤーを選択する．そのレイヤーに対応した上位と下位のアクターをスイムレーンに配置し，スイムレーン上にはそのアクターに関係のあるアクション要素を配置する．これによりアクションは，運用シナリオのレイヤーと整合した記述粒度に統一することを図る．

次に，運用ステレオタイプメタモデルを用いて，Common behavior になっている一連のアクション要素をステレオタイプ付きのアクション要素１つに置き換える．各ステレオタイプで定義した Common behavior（Command ステレオタイプの例では図 5.7a が該当）は，同一粒度を前提に設計しているため，一連のアクションが粒度を揃えてあることが重要となる．

図 5.9 は System of Systems レイヤーの運用シナリオ上にあるコマンドを用いて，一連のアクションをコンパクト化した例である．図 5.9(a) の運用レイヤーメタモデルに従い，System of Systems レイヤーのアクティビティ図として，アクターは System of Systems として “Earth Obs. Msssion Systems”，System レイヤーとして “Ground Sys.” および “Satellite Sys.” を設定する（図 5.9(b)）．運用ステレオメタモデルの Command ステレオタイプを図 5.9(b) に適用し，Send, receive, execution の各アクションを図 7(d) に示す通り，Execution を担う “Satellite Sys.” のアクションとして集約する．また，Command ステレオタイプは，SatOpe クラスを継承しているため，図 5.9(b) の吹き出しである時刻制約を Time Constraint タグとして更なる集約を図る．これにより 3 つのアクションと 1 つの吹き出し要素を１つのアクションにまとめることが出来た．運用シナリオのレビューアーは事前に Command ステレオタイプの Common Behavior を認識している前提で，図 5.9(d) からコマンドの提供元，時間制約，および “Satellite Sys.” がコマンドの内容が Execution されるシナリオとして評価する．

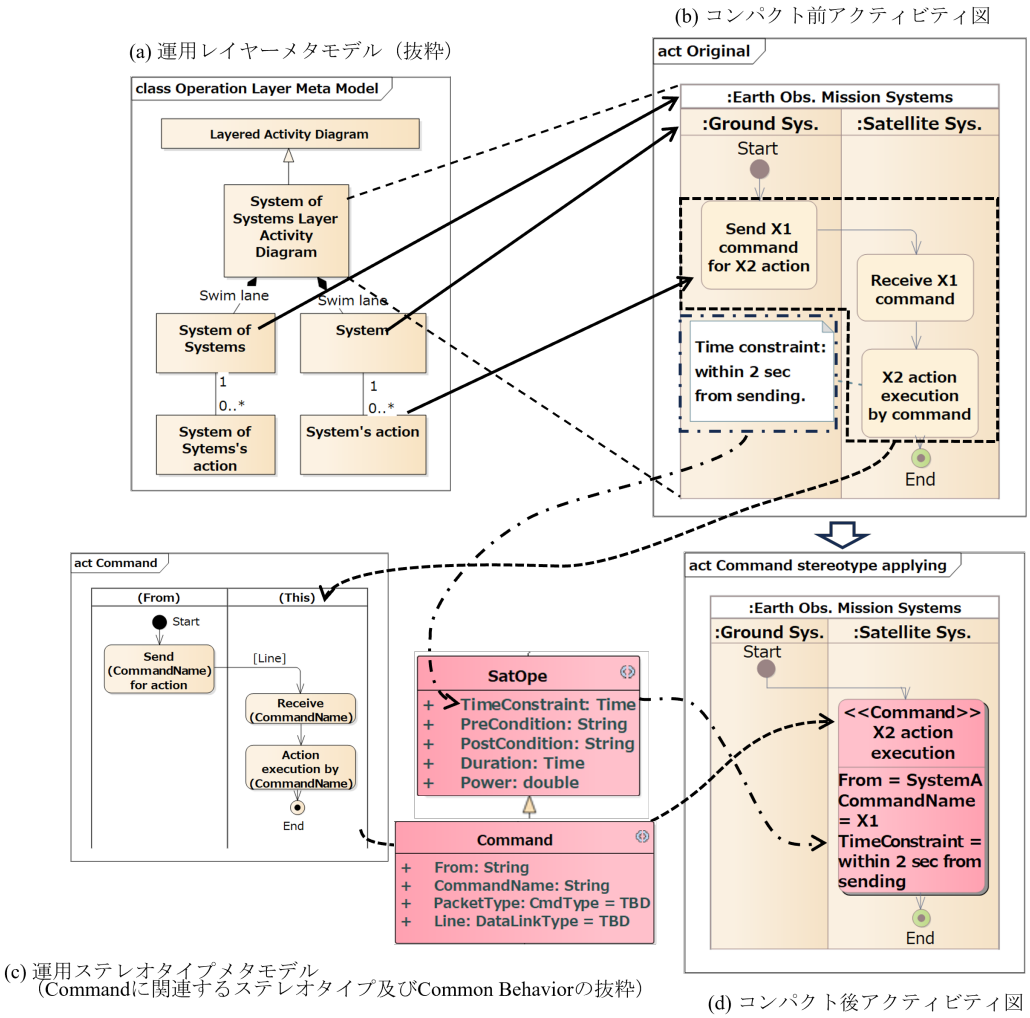


図 5.9: 運用メタモデル適用手順

第6章　運用異常イベントメタモデル

本章では運用シナリオのレビューの網羅性を向上させるためにオフノミナルシナリオに着目し，シナリオが潜在的に持つ状態遷移の列挙方法を述べる．まず，ノミナルシナリオとオフノミナル，および各シナリオが持つ状態とその遷移の関係性を明確にし，MBSE 上で一元管理するための SysML の運用異常イベントメタモデルを提案する．次にモデル検査の仕組みを利用して，シナリオ実行順序の探索と状態遷移の列挙の方法について述べる．

6.1　オフノミナルシナリオ評価方法

本研究では運用シナリオのうち，想定通りに実行されるシナリオをノミナルシナリオと呼び，異常への対応を含めた想定外のシナリオをオフノミナルシナリオと呼ぶ．運用シナリオの評価は，最初にノミナルシナリオ，次にオフノミナルシナリオを対象として実施する．オフノミナルシナリオのバリエーションは，異常等のイベントがいつ発生するかに自由度があるため，それらを 1 つずつ表現するとシナリオ数が増大する．また，シナリオ数を抑えるために，アクティビティ図の並行要素及び分岐要素による表現を多用すれば，複雑となり視認性が低下し再度レビュー困難となる．例えば，複数の地上局が故障するシナリオを考えた場合，実際は故障発生には順番およびタイミングがあるが，それらを１つ１つ表現するとシナリオ数が膨大になる．そのため，シナリオとしては，順序を問わず地上局の故障を並行実行として表記するか，決め打ちの順序で地上局の故障を順次発生として線形で表現する．実行順序の問題については，オフノミナルシナリオに限った事象ではなく，独立した複数システムがあった場合にどこから順に立ち上げるか等，正常時のイベントも同様であるため，実はノミナルシナリオの段階から，すでに実行順序のルートに複数存在することになる．オフノミナルシナリオはノミナルシナリオをベースに作成するため，ノミナルシナリオの段階からバリエーションがあることは，さらなるシナリオ数や実行可能なルートの増加が見込まれることになる．さらに実行可能なルートが複数存在するということは，状態遷移の観点においても，そのルート分だけの状態遷移が存在することとなり，エンジニアが想定していない状態遷移や状態の組合せに陥る可能性がある．この想定していない状態の組合せを見逃すと，宇宙機システムの設計不備となり不具合に繋がることになる．さらに状態遷移は事前条件や事後条件，ガード条件など遷移自体に

制約がある場合があり，これらの制約も含めて目視で確認しながらエンジニアがレビューすることは困難である．

オフノミナルシナリオの実行順序に起因した状態遷移の評価を行うためのアプローチを 6.1 に示す．Nominal scenario は最初に設計する正常系の運用シナリオであり，アクティビティ図である．Off-nominal scenario は Nominal scenario から派生で生成されるとして，Derive の関係性を持つ．Off-nominal scenario を作成するためには，Nominal scenario を対象に STAMP/STPA による安全解析を行う．Unsafe control Action(UCA) は，STAMP/STPA の結果から抽出される非安全となる振る舞いであり，アクション要素である．Unsafe scenario は，UCA が発生することで異常状態時に行われる振る舞いを示すシナリオである．Unsafe scenario は技術者が UCA を手掛かりに検討し，アクティビティ図で作成する．UML における代替フロー及び例外フローに該当する．Recovery action は，Unsafe scenario から正常状態への復帰するために必要な振る舞い，または宇宙機システムを非安全から安全な状態に移行させるための振る舞いであり，アクション要素である．Recovery Action も技術者が検討し，復帰機能として宇宙機システム設計に反映する．これら UCA，Unsafe scenario，Recovery action の三点セットでノミナルシナリオに挿入することでオフノミナルシナリオとなる．State transition は，各シナリオの任意の地点における状態を入力として，運用シナリオに基づく状態遷移であり，ステートマシン図で表現する．状態には宇宙機システムの動作モード（例えば，正常モード/自律モード）などのアクターが持つ状態や，システム間の通信状況などのシナリオ上の環境に関する状態など，宇宙機システム開発で考慮すべき状態として各シナリオからステートマシン図にインプットされる．この状態遷移は，Bounded model checking 手法のルート探索の機能のみを活用した Bounded search を利用して評価する．Bounded search は SAT ソルバーで実装し，シナリオの実行可能ルート探索と状態遷移の列挙出力に使用する．SysML のメタモデルとしては，Nominal scenario, off-nominaol scenario, および State transition を対象範囲とする．Bounded search の SAT Solver 及び STAMP/STPA はメタモデルと連携を持つ，外部の解析手法とする．

本アプローチを，地上局の停止をトリガーに宇宙機システムが自律的に相互の衛星間通信を開始してサービスを継続する運用を例題に説明する．Nominal secnario は，最終的に双方向の衛星間通信が成立することが目的である．それに対して，Off-nomial scenario では，衛星間通信が成立しない UCA が含まれ，UCA に対する Recovery action により最終的には衛星間通信が成立するシナリオである．UCA は，Nomial scenario に対して STAMP/STPA により安全解析した結果から導かれるとする．衛星間通信を行うためには通信相手先の軌道情報が必要であるが，その情報取得が遅延した場合に UCA となる可能性がある．UCA に対応した Unsafe scenario は，相手の軌道情報が遅延により，有効な軌道情報をリアルタイムで正しく取得できず，衛星間通信が成立しないというシナリオとなる．そして，正常に復帰するための Recovery action は遅延を想定して，別のシステムから提供を受ける，

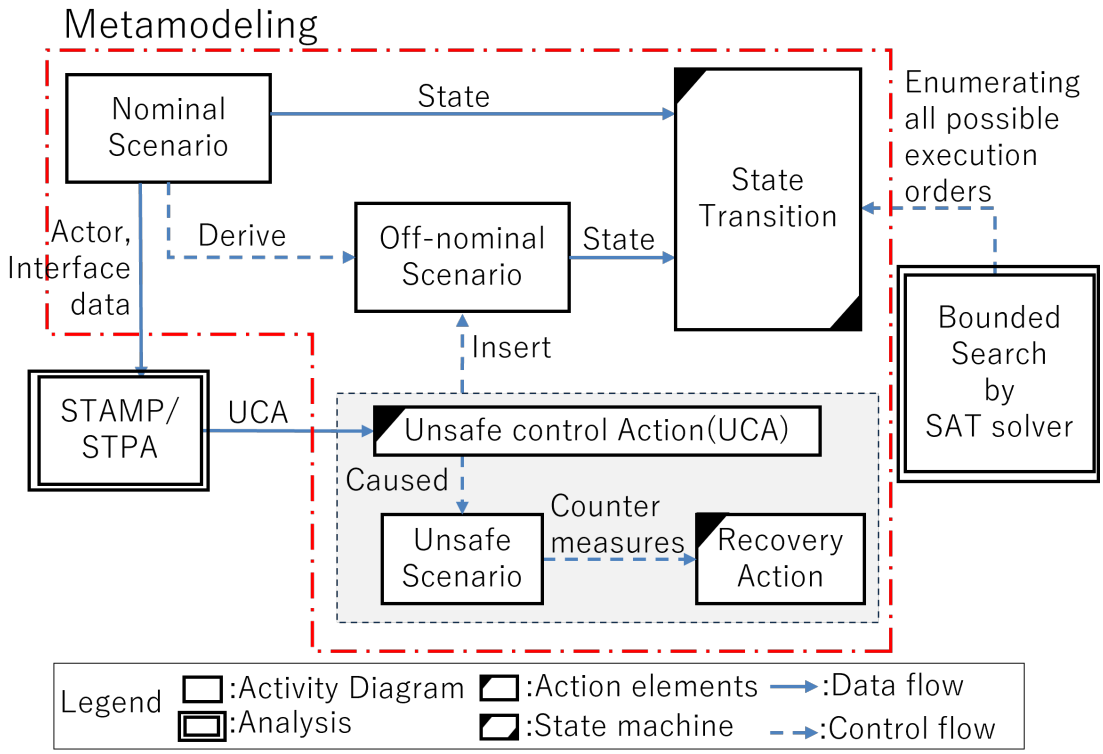


図 6.1: オフノミナルシナリオ評価方法の全体概要

または軌道情報の有効期限を長くするなどが考えられる．Off-nominal scenario は，Nominal scenario に対して代替フローとして，これら UCA と Recovery action を追加することで完成する．State transition は，地上局の稼働状況，宇宙機システムの軌道情報の有効性，衛星間通信状況が考えられ，それらを 0,1 に割り当てる．状態を表す 0, 1 は各アクションの実行により変化を示す．地上局の正常稼働を 1 として割り当てた場合，地上局を停止するというアクション実行により状態を 1 から 0 に変更する．Bounded search では，State transition で作成された状態遷移を SAT Solver を用いて，初期状態から受理状態まで変化するルートを算出する．その際，宇宙機システムの設計からガード条件を設定し，0,1 の状態遷移に制約を課す．本研究では，運用シナリオのレビューの場に，SAT solver により求めらえた状態遷移のグラフを提示することで，それまでエンジニアが頭の中で考えていた状態遷移を可視化されたグラフを見て，想定外の実行ルートや状態遷移に陥っていないかを評価する．そして，想定外の遷移遷移が検知された場合は，遷移条件を追加するとともに，それに整合するように宇宙機システムの設計にフィードバックする．なお，本例題を用いた実験は 7.2.3 項で詳しく述べる．

6.2 安全解析手法連携

本研究における安全解析手法としては，複数の独立したシステムの相互作用に起因した解析に適切な STAMP/STPA を用いる．STAMP/STPA 自体は専用ツールも一般公開されているが，本研究では一貫した MBSE メソッドに沿った情報集約及び分析を進めていることから，STAMP/STPA も SysML 上で構築する．STAMP/STPA 手法において，本研究で主に参照するのが UCA 抽出部分である．そして，STAMP/STPA 解析の準備として作成するコントロールストラクチャー図も SysML で表現する．コントロールストラクチャー図は，安全制約の実現に関するコンポーネント（サブシステム，機器，組織等）及びコンポーネント間の相互作業を分析した制御構造図である [55].

本研究におけるコントロールストラクチャー図は，運用シナリオ上のアクター間のデータインタフェースを入力とすることから，コンポーネント間の相互作用においてデータのインタフェースに観点を置く．STAMP のコントロールストラクチャー図は，コンポーネント間のやり取りを可視化することが狙いであることから，SysML の内部ブロック図を用いてシステム間の双方向のやり取りを表現する．本研究におけるコントロールストラクチャー図を図 6.2 に示す．STAMP/STPA のコンポーネント図にはコントロールアクションとフィードバックデータの種別があるが，内部ブロック図では，コントロールアクションに対応して，データのやり取りである itemFlow を使用して運用シナリオ上のデータインタフェースを示す．

図 6.2 に示した内部ブロック図は，衛星間が自律的に衛星間通信を行うことを題材としたコントロールストラクチャー図である．コンポーネントとしては衛星システム 2 機（Sat-1，Sat-2）及び地上局 2 基（Gnd-1,Gnd-2）を作成する．衛星間通信を行うためには，双方と位置情報である軌道情報を保持しておく必要がある．そのため，地上局から衛星に対して，各々の軌道情報を提供する．データの提供は内部ブロック図の itemFlow を用いる．そして，衛星間通信を行うためには，双方向で通信要求を出す必要がある．通信要求を受け取り，通信の準備が出来次第，通信開始となる．また，衛星間同士にある軌道情報は，衛星間通信が成立している間，双方向に軌道情報をやり取りし，最新化を図ることを意図している．

UCA の抽出は，STAMP/STPA 手法に従い，内部ブロック図で示した itemFlow に対して，Not providing causes hazard，Providing causes hazard，Too early，Too late，Order causes hazard, Stopped too soon，Applied too long[26] の通常とは異なる振る舞いの動きを当てはめ，非安全となるアクション（UCA）になりうるかを検討する．例えば，図 6.2 において，Gnd-1 から Sat-1 への Sat-2 軌道情報 (Orbital Information) の itemFlow に対して，Too late（遅すぎ）が発生した場合，Sat-1 で保有している Sat-2 の軌道情報の更新が遅れ，軌道情報の精度が劣化することで衛星間通信を行うための精度が保てず，衛星間通信が成立しないという異常状態のシナリオが考えられる．分析により抽出した UCA 及び異常状態のシナリオについては，次項に示す運用異常イベントメタモデルにおいてシナリオ評価全体におけ

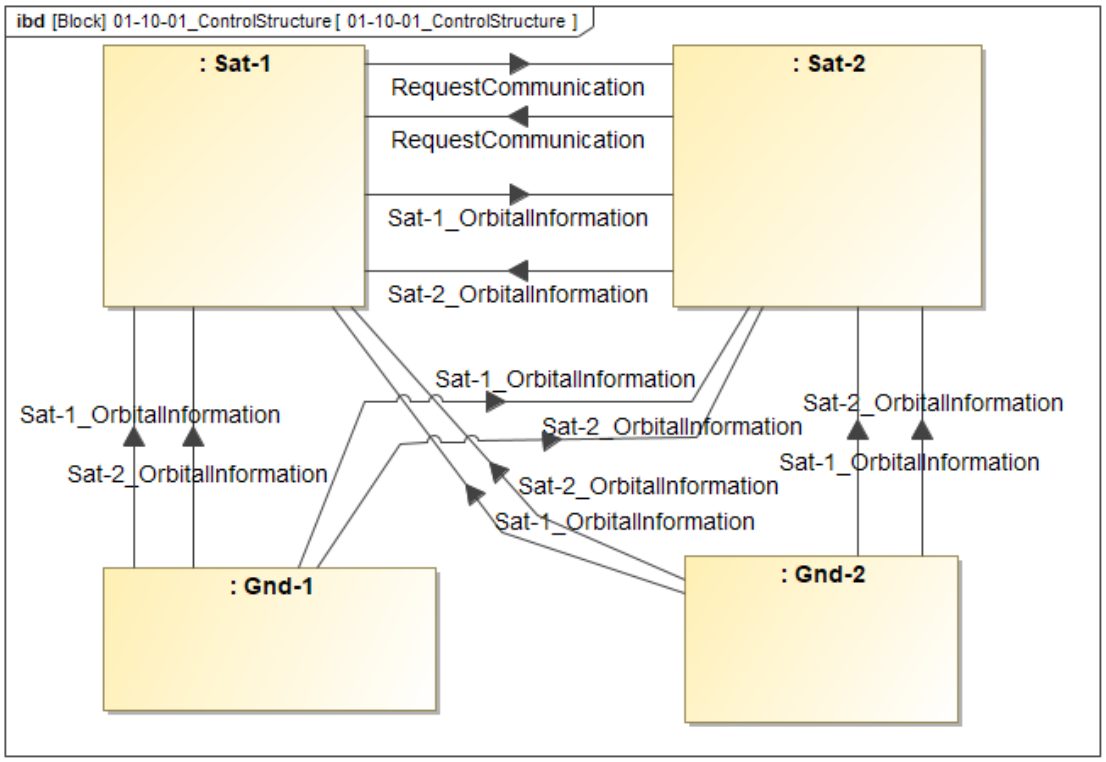


図 6.2: SysML 内部ブロック図を用いたコントロールストラクチャ

る関係性を明確に定義する．

6.3　運用異常イベントメタモデル

本節では，運用異常イベントメタモデルの設計及びメタモデルによるノミナルシナリオからオフノミナルシナリオを派生する方法について述べる．

6.3.1　メタモデル設計

提案するメタモデルは，STAMP/STPA と連携した Off-nominal scenario の生成及び，シナリオの実行ルート探索と状態遷移の列挙のために Bounded search と連携する範囲を対象して，図 6.3 に示す．メタモデルは，SysML のメタモデルとしてクラス図で表現する．また，STAMP/STPA との連携，および Bounded search と SAT Solver との連携のために 2 つのステレオタイプを提案する．

- JoinPoint ステレオタイプ：Nominal scenario から異常状態への分岐ポイントを示す．UCA により発生する unsafe behavior 及び recovery actions の挿入ポイントを示す．

- SAT solver ステレオタイプ：SAT solver による状態遷移と連携するためのステレオタイプである．アクティビティ図上の各アクションに適用し，アクションの実行前後による状態遷移をタグ付き値として示す．

なお，SAT solver ステレオタイプは，オフノミナルシナリオ生成とは異なる性質であるため，運用異常イベントメタモデルには含めない．SAT solver によるアクティビティ図の状態遷移は，宇宙機システムの運用シナリオに限らず汎用的に利用可能と考え，別のパッケージとして SATSolverProfiles パッケージとして設計する．それ以外に使用しているステレオタイプは，利用者の利便性を考慮し，OMG にて規定されている RAAML などの既知ステレオタイプを流用する．Metaclass のステレオタイプは，UML または SysML で定義されている要素であり，Activity, Activity Group, Activity Partition, Decision Node, Actoin が該当し，(図 6.3 においてオレンジ色で示される．Block ステレオタイプは SysML で定義されているステレオタイプであり，オブジェクトとして捉えられる要素であり，宇宙機システムや地上局システムなどのアクターが該当する．Loss Scenario 及び Unsafe Control Action のステレオタイプは，安全解析のプロファイルである RAAML から引用する．

運用異常イベントメタモデルの Top は運用シナリオをアクティビティ図で示すことから Activity 要素とする．運用シナリオの種別は Nominal scenario，異常状態を含む派生シナリオの Off-nominal scenario がある．Off-nominal scenario は RAAML の Loss scenario のステレオタイプを付与し，Nominal scenario と区別可能とする．また，Nominal scenario と Off-nominal scenario の多重度は１対多の関係性とする．１つの Nominal scenario に対して，異常状態に対応した複数の Off-nominal scenario が生成される．Join point は，アスペクト指向に着想を得て，異常状態への分岐ポイントとして Nominal scenario に含まれ Decision Node 要素である．Unsafe Control Action（以下，UCA）は，Nominal scenario を STAMP/STPA の解析結果より導き出される．UCA 発生に伴う異常状態時の振る舞いは，Unsafe scenario として定義し，Off-nominal scenario に含まれる．なお，Unsafe scenario の振る舞いは，STAMP/STPA の解析手法の中でエンジニアが検討して設定する．加えて，Recovery Action は異常状態から正常に復帰するためのアクションして検討する．異常状態から復帰までの一連の振る舞いは，UCA，Unsafe scenario, Rcovery action のセットで成立する．そして，Off-nominal scenario は，この一連の振る舞いを Nominal scenario の Join point に挿入することで実現する．

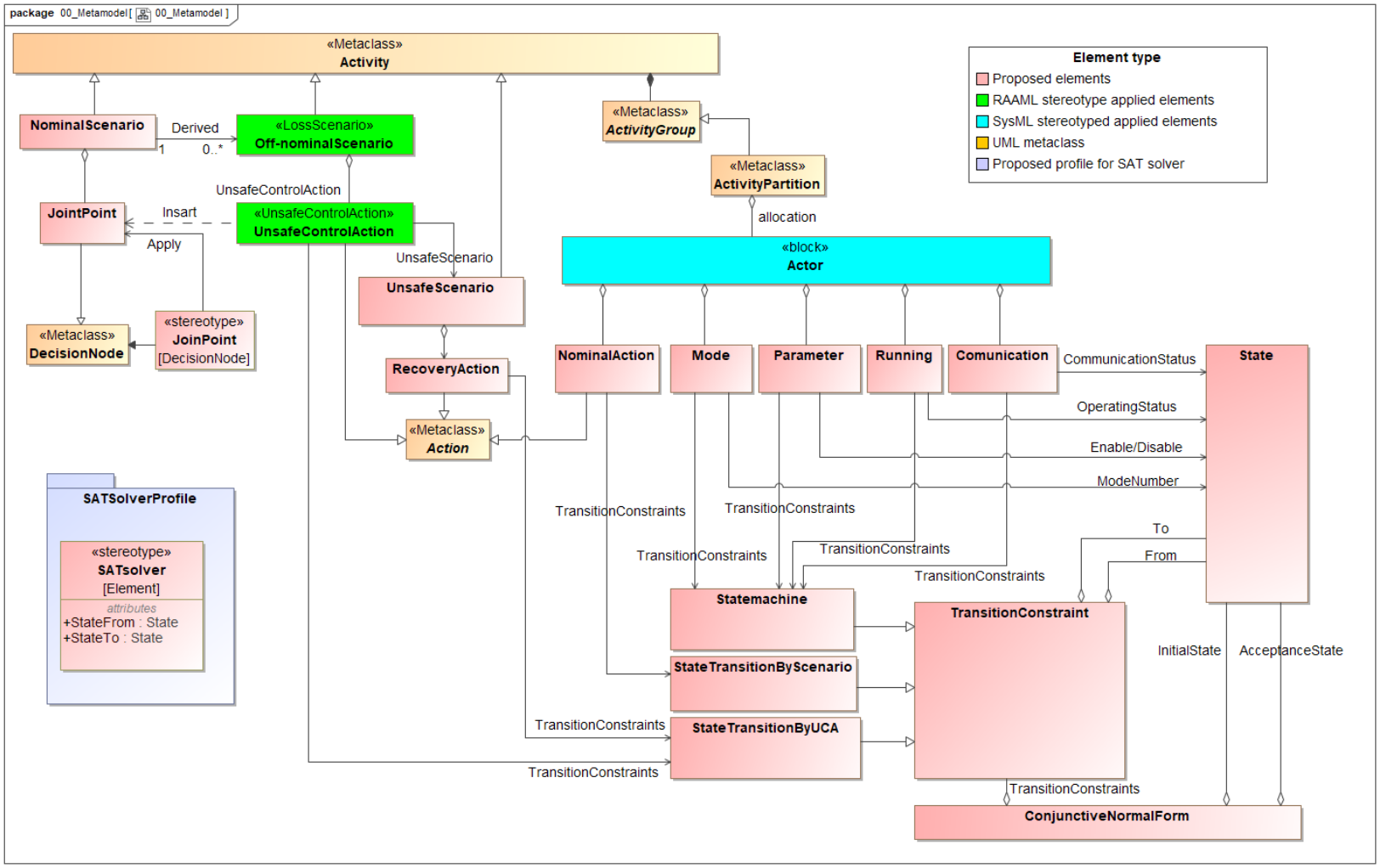


図 6.3: 運用異常イベントメタモデル

Bounded search によるシナリオの状態遷移検証は，シナリオの開始時点を初期状態とし，シナリオ実行に沿った状態遷移により，シナリオの最終地点の状態を受理状態として評価する．シナリオの登場人物である Actor は，例えば，宇宙機システムや地上局であり，アクティビティ図上のスイムレーンに Allocation する．メタモデルの Actor は，スイムレーンを示す ActivityPartition に対して Allocation の関係性を持つ．また，Actor は，下記の通り，SysML のアクション要素である Nominal Action，及びシナリオの状態を示す Mode，Parameter，Running，Communication を持つ．

- Nominal Action：Nominal scenario に含まれる Action 要素である．同じアクション要素である UCA や RecovryAction と区別する．
- Mode：システム設計上のモードである．具体的には，宇宙機システムでの通常モード，自律モード，縮退モードなどの動作モードに該当する．
- Parameter：運用シナリオ上の Actor に関連して変化する要素である．例えば，本研究における Parameter は，宇宙機システムが保有する必要な軌道情報が該当する．
- Running：システムの稼働状況を示す．例えば，地上局が稼働状況にあるか，故障などで使用できない状況かを示す．
- Comunication：Actor 間の通信状況を示す．例えば，本論文における Comunication は，宇宙機システム同士，及び衛星 - 地上間があり，2 拠点間の通信が可能か不能状態かを示す．

メタモデル上の State 要素は，上記の Mode,Parameter,Running,Comunication の各要素が持つ状態を 1 つにまとめたものである．例えば，動作モードが 1 で自律モード，軌道情報が有効で 1，衛星間通信が確立されており 1，地上局との通信が確立されておらず 0 とした場合，State は (1,1,1,0) のようにあるシナリオ時点における状態を 0,1 の配列で表現する．Transition constraint は，State に対して From，To として要素を持ち，Action の実行前の状態を From，実行後を To とする．例えば，衛星間通信が途切れた場合は，対応する 3 つ目の状態が 1 から 0 に変化することを（1,1,1,0),(1,1,0,0) と表現し，前半が From, 後半が To である．さらに Transition Constraint は 3 つに具体化される． 1 つは State machine であり，宇宙機システムの設計に基づく動作モード遷移や，地上局が正常から故障するなどの Running の状態遷移を示す．State Transiton by Secnario は，シナリオ上のアクションにより変化する状態であるため，Nominal action と関係性を持っている．State transition By UCA は，UCA が発生することにより状態変化を示すため，UCA 及び Recovery action と関係性を持つ．Conjunctive Normal Form（CNF) は，ブール論理における論理式である．Transition Constraint の 1 つ 1 つの遷移を論理和で連結したものであり，SAT Solver に設定することで Bounded search を

実行する．また，Bounded search におけるシナリオの初期状態，Initial Statet として CNF に連結する．受理状態は，State の Acceptance Status として CNF に連結する．

運用異常イベントメタモデルに含まれる各項目の内容を表 6.1 に示す．

表 6.1: 運用異常イベントメタモデルの要素一覧

項目	カテゴリ	説明
Nominal Scenario	Activity	ノミナルシナリオは正常状態の想定通りであり，アクティビティ図で作成する．
Off-nominal Scenario	Activity	オフノミナルシナリオは，ノミナルシナリオからの派生シナリオとして生成し，想定外の異常状態を含むシナリオである．アクティビティ図で作成する．また，RAAML の LossScenario ステレオタイプを付与することでノミナルシナリオを区別する．
Join Point	Desion Node	ジョインポイントはノミナルシナリオからオフノミナルシナリオへの分岐ポイントを示し，異常イベントの発生タイミングとなる．UML,SysML のディジョンノードを使用する．
Unsafe Control Action	Action	非安全となるアクション（UCA）であり，本研究では異常状態を引き起こすアクションとし，STAMP/STPA 解析結果から抽出する．また，ノミナルシナリオのジョインポイントに挿入することでオフノミナルシナリオを生成する．また，UnsafeControlAction ステレオタイプは，RAAML の STAMP/STPA 用として定義されている要素である．
Unsafe Scenario	Action	非安全シナリオは，UCA をトリガーに発生する異常状態時のシナリオである．STAMP/STPA の解析の中で検討する．
Recovery Action	Action	復帰アクションは，非安全シナリオから正常状態へ復帰するためのアクションであり，STAMP/STPA 解析の中で検討する．

項目	カテゴリ	説明
Actor	Block	アクターはシナリオの登場人物やシステムであり，アクティビティ図のスイムレーンに割り当てる.
Nominal Action	Action	ノミナルアクションはノミナルシナリオにおいてアクターが実行する処理であり，UCA などの他のアクションと区別する.
Mode	State	モードはアクターが持つモードである．状態の一つである．例えば，稼働モードや停止モードなどのシステム状態を示す.
Parameter	State	パラメータはアクターが持つ，シナリオ上のパラメータである．状態の一つである．例えば，軌道情報の有効，無効などの状態を示す.
Running	State	ランニングはアクターが持つシステムの稼働状況である．状態の一つである．例えば，稼働中，故障中などの状態である．モードとの違いは，ランニングが機器の電源 ON/OFF などハードウェアの状態であり，モードはシステム内部で持つ論理的な動作モードをソフトウェアの状態を示す.
Communication	State	コミュニケーションは，通信状態であり，特定にアクター間の通信中か，停止中かを示す．状態の１つである.
State	State	状態はモード，パラメータ，ランニング，コミュニケーションの状態の行列であり，シナリオ上のある時点における状態を示す.
Transition Constraint	State transition	状態遷移の集合であり，遷移前の状態と遷移後の状態を示す行列である
Statemachine	State transition	ステートマシンは，アクターが持つ状態遷移の設計結果及び制約であり，ステートマシン図で示す.
Sate Transition By Scenario	State transition	アクティビティ図上のアクション実行により変化する状態遷移である．遷移前と遷移後の状態を行列で示す.

項目	カテゴリ	説明
State Transient By UCA	State transition	UCA 及びリカバリーアクションにより変化する状態遷移である．遷移前と遷移後の状態を行列で示す.
Conjunctive Normal Form	Logical expression	連言標準形は，１つ１つの状態遷移のまとめて，論理積で連結したブール論理における論理式である.
SAT solver	Stereotype	SAT ソルバーに入力する元情報となる Trainsitoin constraint を作成するためのステレオタイプである．アクティビティ図上のアクションに適用し，タグ付き値によりアクション実行前の状態を State From，実行後の状態を State to を示す．また，本ステレオタイプは異常状態イベントに限らず利用できるため，SAT Solver Profile として別に設計する

6.3.2 派生シナリオ生成

オフノミナルシナリオは，Aspect 指向 [27, 31, 29] を参考にし，ノミナルシナリオに UCA を入れ込むジョインポイントを設け，異常状態のシナリオを挿入することで生成する．ジョインポイントを起点に異常状態への分岐し生成することから派生シナリオとなる．異常状態へ分岐後は，非安全なシナリオ（Unasfe scenario）を実行し，復帰アクション（Recovery action）により正常状態に復帰する．なお，異常によっては復帰アクションを実施しても，ノミナルシナリオと同じ終了状態に達成しない場合もある．その場合は，アクティビティ図のフロー終了としてシナリオを終了する．

冒頭で述べたコンステレーションによるミッションを想定した自律化を目指したシステムを例題に，図 6.4 で示したノミナルシナリオを用いて派生シナリオの生成方法を述べる．図 6.4 は運用異常イベントメタモデル上の Nominal scenario をモデル化したものである．アクティビティ図で示されたシナリオの見方は，Start 要素から各アクションを矢印に順に従い実行し，End 要素まで各アクションを順次実行する．アクターは，2 機の衛星システム Satellites（Sat-1, Sat-2）と 2 局の地上局 Ground Stations（Gnd-1，Gnd-2) を割り当てる．シナリオの流れは，Gnd-1 が停止し，Gnd-2 も続いて稼働を停止する．次に Sat-1 は，地上局からの電波が途切れことを検知し，地上局が停止したと判断する．そして，衛星システムの動作モードを通常モードから自律モードに切り替える．Sat-2 も Sat-1 に続き，Ground Station を停止と判断し，動作モードを自律モードに切り替える．最後に，自律モードとなった Sat-1 は運用を継続するために，Sat-2 と衛星間通信を行い，通信を確立する．Sat-2 も自律モードに入り，Sat-1 と衛星間通信を開始する．シナリオは双方向で衛星間通信が行われて正常終了となる．

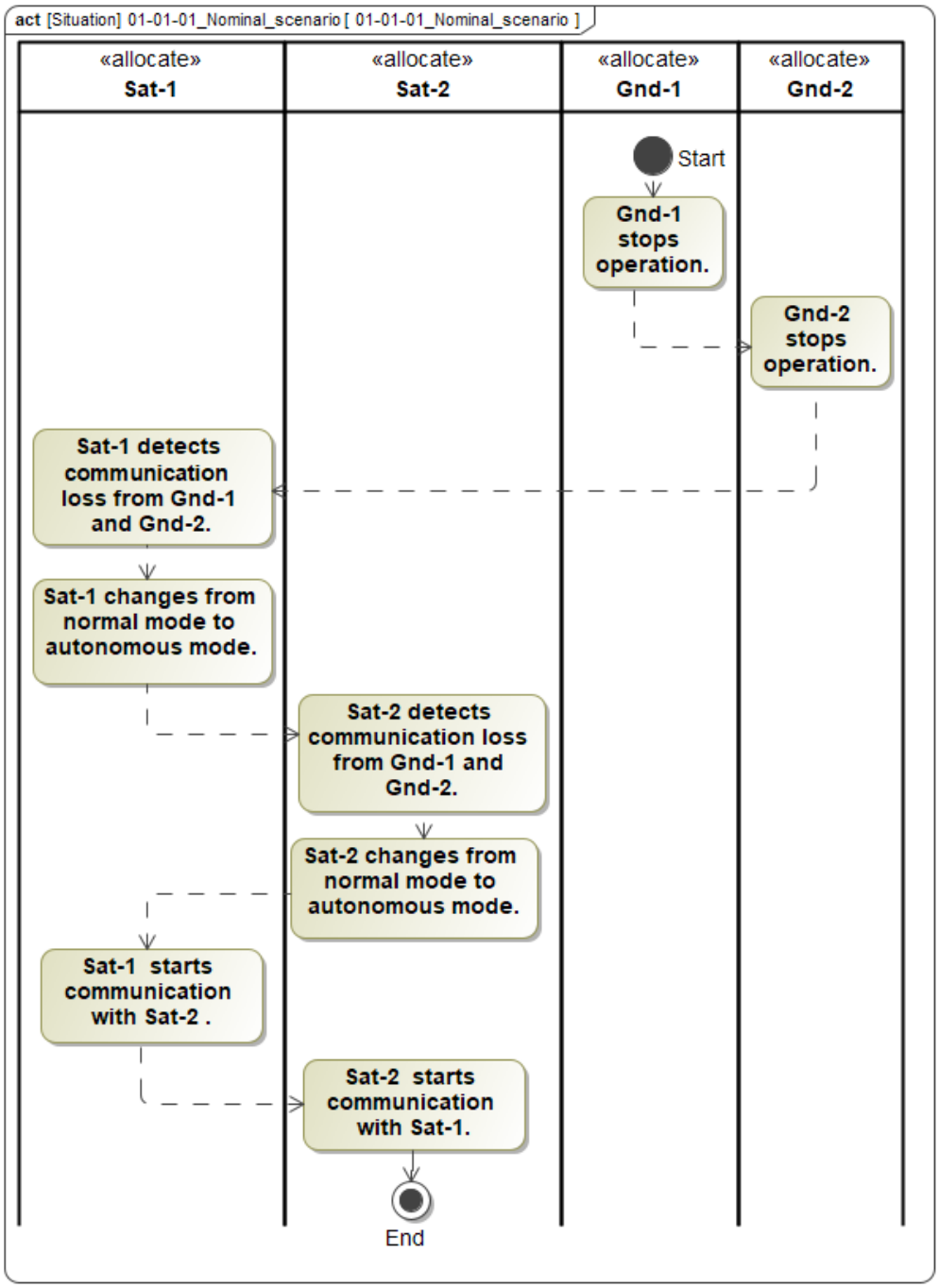
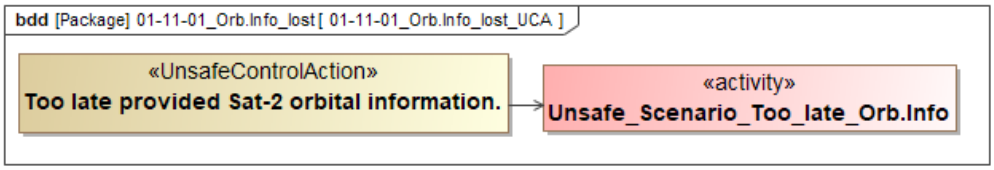
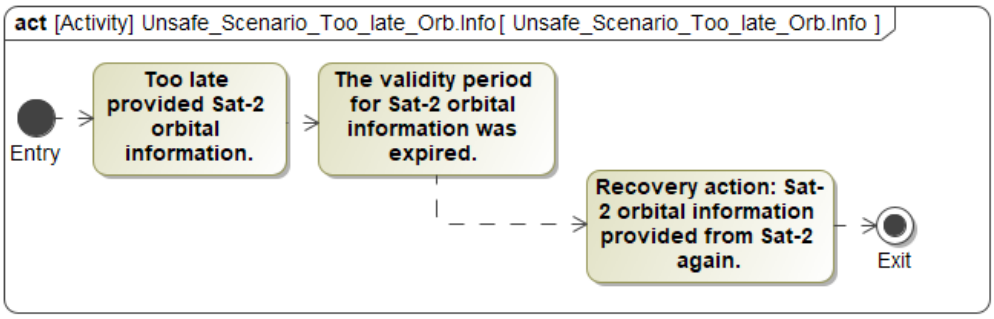


図 6.4: 派生シナリオ生成におけるノミナルシナリオ例



(a) Unsafe control action の例



(b) UCA に伴う非安全シナリオ

図 6.5: STAMP/STPA により抽出した UCA 及び非安全シナリオ

ノミナルシナリオに対して，STAMP/STPA による安全解析結果の一例として集出した UCA を図 6.5 に示す．検知された UCA は，アクション要素として Unsafe control action ステレオタイプを付与する (図 6.5a)．アクションの内容に非安全となるアクションを記述し，本例では “Too late provided Sat.2 orbital informatio” である．Sat-2 の軌道情報の提供が遅れた場合に異常となることを示している．軌道情報は将来を予測した軌道であるため，時間と共に誤差が増大するため賞味期限がある．そのため，図 6.5b に示す通り，UCA によって発生する非安全なシナリオは，提供が遅れると賞味期限切れとなり使い物にならないということが考えられる．非安全シナリオはノミナルシナリオからの分岐により実行されるため，アクティビティ図上，Entry 要素から実行し，Exit 要素で終了する．また，復帰アクション（Recovery action) は，再度軌道情報が提供されることで正常復帰可能として，Exit 要素の前に配置する．

ノミナルシナリオに Join point を設定し，オフノミナルシナリオとして派生させたシナリオを図 6.6 に示す．ノミナルシナリオに配置したディションノード（ひし形マーク）は，通常の分岐要素とは区別するために Join point ステレオタイプを適用する．本例では，ジョインポイントを Sat-1 が Sat-2 と衛星間通信を開始アクションの直後に設定した．ジョインポイントでは，ガード条件で Nominal と Off-nominal が分岐される．Nominal を選択した場合はノミナルシナリオと同等である．Off-nominal を選択した場合は，図 6.5b で作成したアクティビティ図が挿入される．なお，アクティビティ図に activity 要素を組み込むために，Call behavior action 要素を用いるか，非安全シナリオの内容が短い場合は Call behavior action

要素を使用せずに，直接，非安全シナリオに含まれるアクション要素を配置する．どちらの方式を選択するかは，視認性等を考えて選択する．本論文では紙面の関係上，コンパクトに示すため Call behavior action 要素を用いた．一方，実開発において印刷等によるレビューを前提とする場合は，非安全シナリオのアクション要素を別ページで示す必要があるため，同じアクティビティ図上に記載した方がレビューしやすいと考える．非安全シナリオの実行後は，復帰アクションにより異常状態から正常状態に復帰する前提であるため，マージノード（2 重ひし形マーク）を配置し，Nominal 側に分岐したルートと合流する．マージノードは，図 6.6 では “Sat-2 starts communiction with Sat-1” アクションの前に設定する．

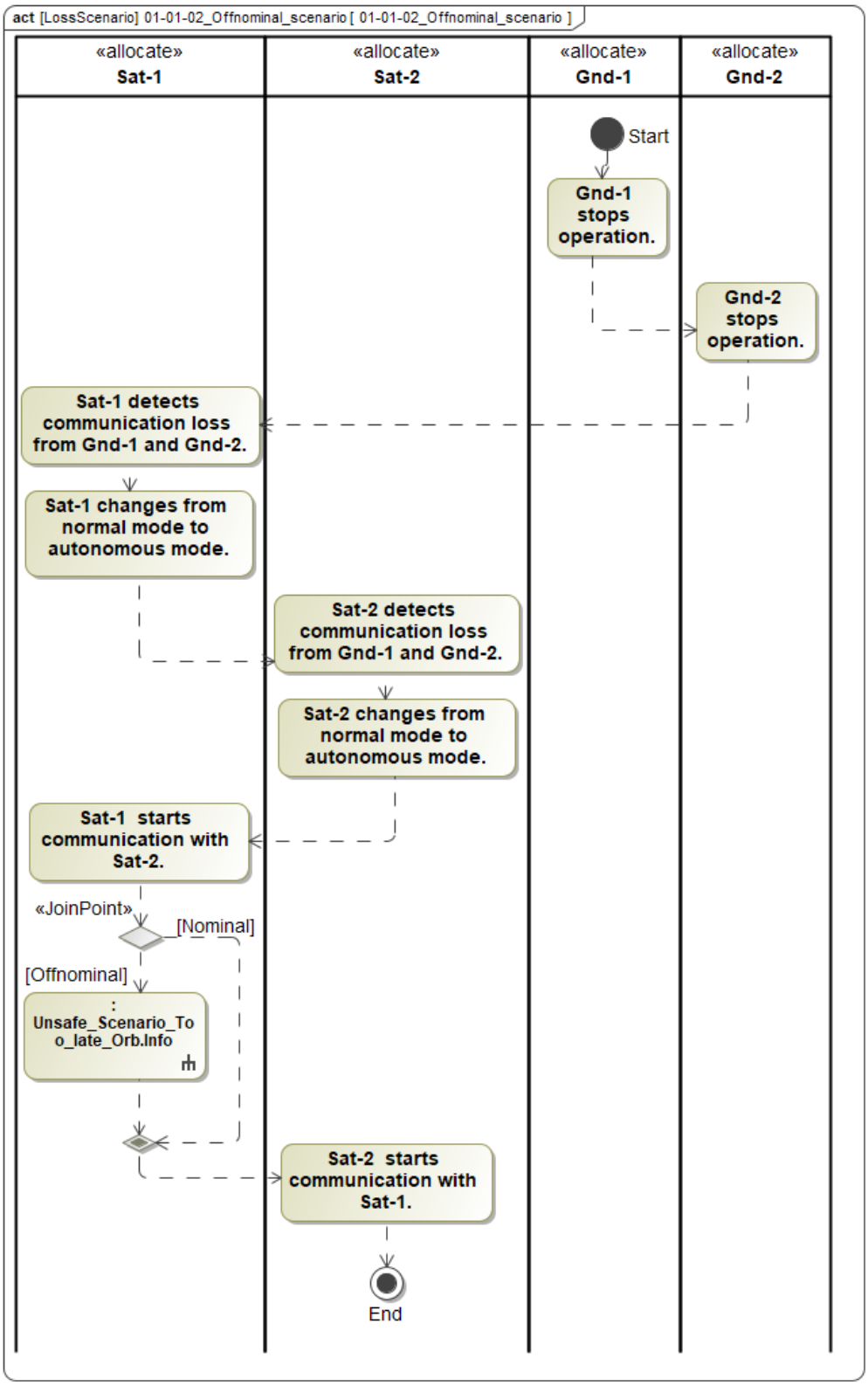


図 6.6: 派生シナリオとして作成したオフノミナルシナリオ

6.4　シナリオ実行順序列挙

アクティビティ図で示すシナリオは，ある実行の一例に過ぎない．ノミナルシナリオでもオフノミナルシナリオでも，任意のアクションの実行順番を入れ替えても成立する場合もある．また，オフノミナルシナリオにおいては，ジョインポイントが別の箇所に設定されても成立する可能性がある．その逆に，ある実行順序においてはエンジニアの想定外の状態遷移が発生し，シナリオが成立しないパターンが潜在している可能性もある．その場合，実運用において想定外の状態に陥り，宇宙機システムに悪影響を及ぼす可能性がある．そこで，シナリオ上で行われる個々のアクションを抽出し，実行可能なアクションの順序を探索することで，1つのシナリオから可能性のある実行ルートを洗い出し，そのルートにより辿る状態遷移をグラフとして可視化する方法を提案する．

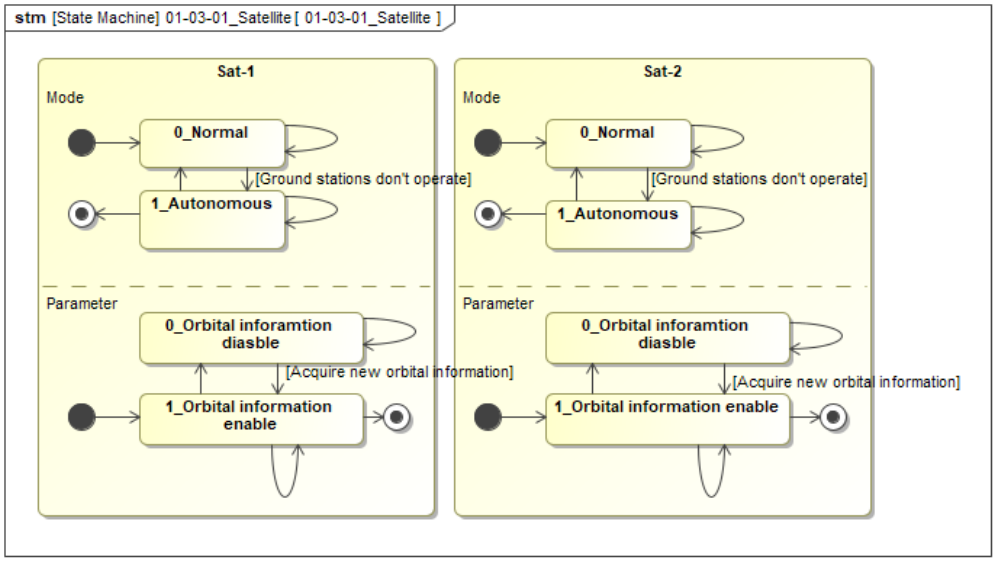
状態遷移を探索には，Bounded model checking（BMC）手法を参考に，BMCがもつ成立する解を探索する機能部分を用いる．これを本研究ではBounded searchと呼んでおり，SAT solverを用いて解く．BMCの状態遷移は，クリプキ構造 M, $k \geq 0$

$$[[M]]_k := I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i, s_{i+1}) \tag{6.1}$$

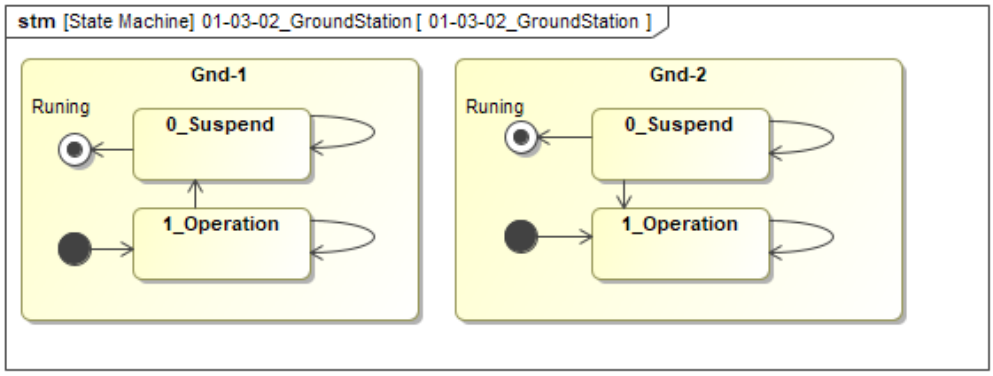
で表わされる．状態の集合 S に対して s は状態を示し，$s_i \subseteq S$ は遷移前，$s_{i+1} \subseteq S$ は遷移後の次の状態を示し，具体的には図6.3で示したState要素である．$I(s_0) \subseteq S$ は初期状態を示し，図6.3のState要素におけるInitial Stateとなる．$T \subseteq S \times S$ は状態遷移（Transfaer）であり，$T(s_i, s_{i+1})$ は状態 s_i から状態 s_{i+1} へ遷移を可能を表す．なお，T は図6.3のTransiton Constraintに対応する．初期状態及び受理状態，各状態の遷移をandで結合し，Conjunctive Normal FormとしてSAT solverで評価を行う．状態遷移は大きく分けて，システム設計に基づく状態遷移と，シナリオの実行に伴う状態遷移がある．システム設計に基づく状態遷移はガード条件として働くため状態遷移の制約となり，シナリオ全体を通じて影響する．シナリオ実行に伴う状態遷移は，アクティビティ図上のアクションを実行するたびに変化する状態遷移である．

6.4.1　システム状態遷移

システムの状態遷移は，運用異常イベントメタモデルのStatemachineが該当する．Statemachineは，Transiton Constraintの１つであり，図6.7に示す通りActorがもつ状態遷移である．図6.7aは宇宙機システムのMode及びParameterとしての軌道情報の状態遷移を示したものである．Modeは，NominalとAutonomousが双方向で遷移が可能である．ただし，Nominalモードから Autonomousモードの移行にはガード条件として地上局がすべて運用停止していることを付与している．また，Nominalモードは状態の番号として0を割り当て，Autonomuousには１を



(a) 宇宙機システムのステートマシン



(b) 地上局のステートマシン

図 6.7: State Machine

割り当てている．Parameter も同様であり，各状態に 0,1 を割り当てると共にガード条件を示す．図 6.7b は地上局の状態であり，Running を示している．0 が停止状態であり，1 が運用中である．停止状態から運用状態への遷移は，運用者が状況を確認して稼働させるため，シナリオ上の自動復帰は認めていないとした．この 2 つ状態遷移図より State は，Sat-1 の Mode が 1bit,Parameter で 1bit, Sat-2 の Mode も 1bit, Parameter も 1bit, Gnd-1 の Runnging が 1bit，Gnd-2 の Running も 1bit で構成され，状態 s は $(1, 1, 1, 1, 1, 1, 1)$ の合計 6bit で表わされる．

例えば，すべて 1 の場合は Sat-1 及び Sat-2 共に Autonomous モードであり，軌道情報は Enable，Gnd-1 及び Gnd-2 は共に稼働中であることを意味する．そして，Gnd-1 が稼働中から稼働停止に状態遷移した場合は，Gnd-1 の Running に対応し

た bit が 1 から 0 に変化する．この場合は

$$s_i = (1,1,1,1,1,1), s_{i+1} = (1,1,1,1,0,1) \tag{6.2}$$

となり，状態遷移 T は

$$((1,1,1,1,1,1),(1,1,1,1,0,1)) \in T \tag{6.3}$$

であり，状態 $(1,1,1,1,1,1)$ から状態 $(1,1,1,1,0,1)$ に遷移可能であることを示す．ガード条件は，$T(s_i, s_{i+1})$ の否定として設定する．図 6.7a の Sat-1 における *Ground stations don't operate* は，Gnd-1 と Gnd-2 がどちらも Suspended の場合に遷移して良いとする．その場合，Gnd-1 の Running に対応する bit が Operation を意味する 1 ならば，Sat-1 の Mode に対応する bit を 0 から 1 に遷移してはならない，かつ Gnd-2 が Running に対応する bit が Operation の 1 であるならば遷移してならい，かつ Gnd-1 及び Gnd-2 の両方の Running に対応する bit が Operation の 1 ならば遷移してはならないという制約を加えることで実現する．これにより Gnd-1 及び Gnd-2 の両方の Running に対応する bit が Supend の 0 の時のみ，Sat-1 はモードを 0 の Normal から 1 の Autonomous に遷移可能となる．

6.4.2 シナリオ状態遷移

シナリオの実行に伴う状態遷移は，運用異常イベントメタモデルの State Transition by Scenario と State Transiotn by UCA が該当する．

State Transition by Scenario は，シナリオの実行に伴いに変化する状態遷移を示す．アクティビティ図のアクション要素に対して，状態遷移が発生する．各アクションに関連する状態遷移は，図 6.3 の SAT solver ステレオタイプを適用することで実現する．SATsolver ステレオタイプは，状態の遷移前後を記述するタグ付き値がある．From タグにはアクションの実行前の状態を指定し，To タグにはアクション実行後に変化した状態を指定する．Nominal scenario に対して，SAT solver ステレオタイプを適用したアクティビティ図を図 6.8 に示す．なお，本研究では状態遷移の制約として，1 つのアクションについて変化する状態は 1 つとする．例えば，“Gnd-1 stops operation” アクションは，地上局が停止する内容であり，図 6.7b において Gnd-1 の Operation から Suspend に変化する．アクションには高々１つの状態遷移のみ発生するとしているため，例えば Gnd-1 と Gnd-2 の同時に停止するという遷移は考慮しない．実世界においても厳密には同時ではなく，時間的間隔は短くとも順序があると考える．そして，Gnd-1 と Gnd-2 の停止の順序によっては，シナリオの成立性に影響が出る可能があることを否定できない．実行順序については Bounded search の中で評価することで網羅性に確認できる．そのため，評価対象のアクティビティ図では，並行実行を用いず，決め打ちで順序を設定する．なお，実際の開発現場においても，どこを並行実行しようか，順序は

どちらが最初が良いかを悩み始めると，時間ばかり過ぎてしまい，作業効率が悪いという点もある．よって決め打ちで順序を設定し，評価の際に確認する方が効率的でもある．なお，他のアクションも同様に，ステートマシンで示した状態を引用し，タグ付きと連携付けさせる．

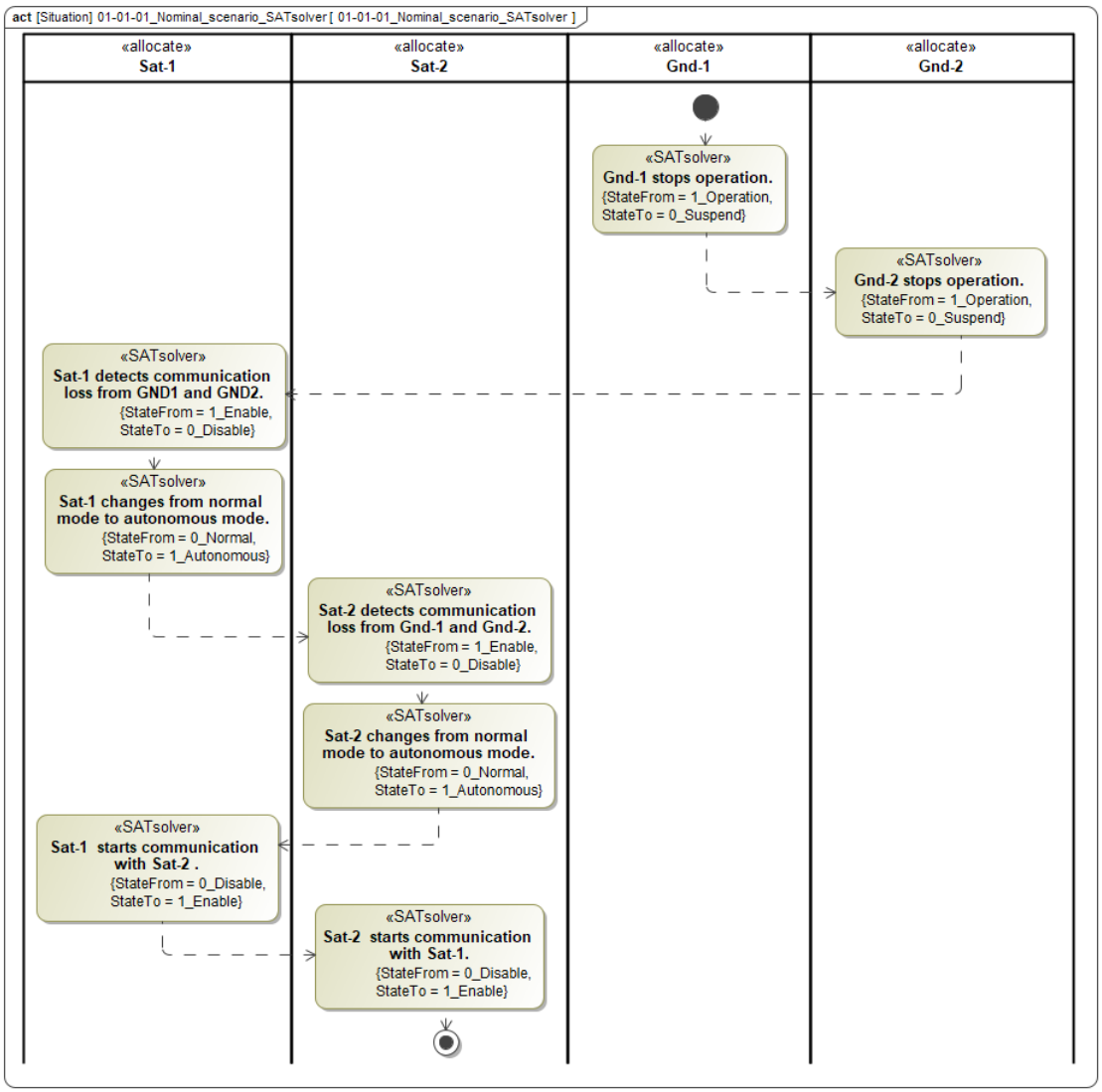


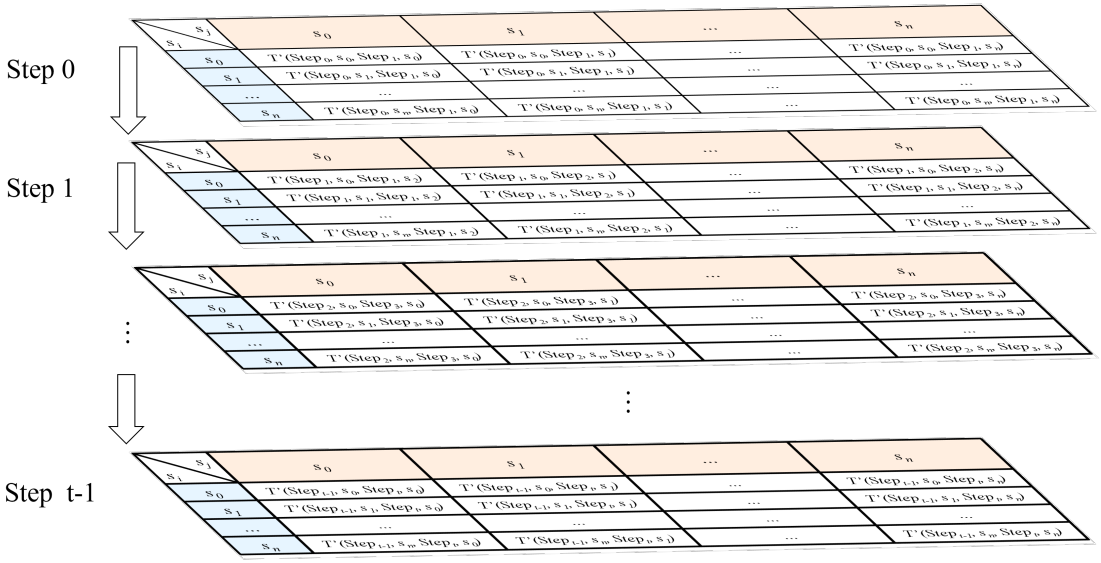
図 6.8: Applied SATsolver stereotype

また，State Transition by Scenario においては，網羅的な状態の組合せを探索するため，式6.1の状態遷移$T(s_i, s_{i+1})$を改良する．まず，アクテビティ図上の実行ステップを$k = 0, 1, 2, 3, ..., t$と定義する．そのうえで，ステップ集合$K = \{0, 1, ..., t-1\}$を導入し，状態遷移関係を

$$T' \subseteq K \times S \times S \tag{6.4}$$

s_i \ s_j	s_0	s_1	...	s_n
s_0	$T'(Step_k, s_0, Step_{k+1}, s_0)$	$T'(Step_k, s_0, Step_{k+1}, s_1)$	...	$T'(Step_k, s_0, Step_{k+1}, s_n)$
s_1	$T'(Step_k, s_1, Step_{k+1}, s_0)$	$T'(Step_k, s_1, Step_{k+1}, s_1)$	...	$T'(Step_k, s_1, Step_{k+1}, s_n)$
...	...	...	...	...
s_n	$T'(Step_k, s_n, Step_{k+1}, s_0)$	$T'(Step_k, s_n, Step_{k+1}, s_1)$	...	$T'(Step_k, s_n, Step_{k+1}, s_n)$

(a) 1 ステップにおける状態遷移



(b) ステップ毎に時間展開

図 6.9: Bounded Search における状態遷移

と定義する．ここで $T'(Step_k, s_i, Step_{k+1}, s_j)$ は，ステップ k 時において状態 s_i から s_j への遷移が可能であることを表す．すわわち，アクティビティ図のアクションの実行前が s_i，実行後が s_j となる．状態遷移 $T(s_i, s_{i+1})$ では，運用シナリオ作成者が想定した状態遷移のみが許可される遷移となる．そこで s_i と s_{i+1} の添え字を i, j に分け，遷移先の状態を任意の状態から探索できるようにする．図 6.9a は，1 ステップ当たりの状態遷移 T' を網羅的に示した図である．あるステップ k における s_i から s_j の遷移の一覧である．図 6.9b は，図 6.9a で示した 1 ステップ当たりを各ステップに時間展開したものである．なお，各ステップに置いて，1 つの T' が選択される．これを全ステップに渡って行うことでアクティビティ図の最初から最後までのルートが選択される．

T' に基づく，クリプキ構造 M' は，

$$[[M']]_k := I(s_0) \wedge \bigwedge_{k=0}^{t-1} \left(\bigvee_{i,j=0}^{n} T'\left(Step_k, s_i, Step_{k+1}, s_j\right) \right) \tag{6.5}$$

であり，$k \geq 0$, $i \geq 0$，$j \geq 0$，$Step_{i+1} = Step_i + 1$ となる．ステップ数を付与する理由は，Bounded search の中で各アクションの実行順序を変えながら評価する．そのため，同じ状態でも何番目に出現するかにより，シナリオの成立性が変わってくる．例えば，地上局が停止が最初のステップで出現した場合，そのあとの衛星モード切替はガード条件に抵触しない．しかし，地上局の停止がシナリオの最後の場合，その前に衛星モード切替が行われているとガード条件に抵触し，不成立なシナリオとなる．

State Transiotn by UCA は，UCA の発生とそれに伴う Unsafe Scenario，Recovery action の実行による状態遷移である．各アクションに SATSolver ステレオタイプを適用し，状態の前後を示す点においては State Transition by Scenario と同じである．

第7章　実験・評価

5章及び6章で示した各メタモデルの適用実験から運用シナリオのレビュー品質向上が得られたことを示す．なお，実験はコンパクト化とオフノミナルシナリオ評価の大きく2つに分けて実施する．

7.1　コンパクト化実験

コンパクト化の実験では，最初に5.4節に示した運用メタモデル評価を行ったうえで，次に実際に運用メタモデルを適用するコンパクト化の実験を行った．後半のコンパクト化の実験では，サンプルシナリオを用いた手法の比較による手法の優位性評価，実際の宇宙機システムの運用シナリオを想定し実開発上での効果を評価した．最後に，提案手法の詳細な分析としてコンパクト化の傾向とレビュー容易性の向上を確認するためインタビューを実施した．本実験結果は参考文献[43]で発表した内容である．

7.1.1　運用メタモデル検証実験

宇宙機システムの運用シナリオへのレビュー観点は，図5.8より16個（緑色の要求要素）である．16個のレビュー観点に基づき，13個の運用メタモデルへの要求仕様を導いた（灰色の要求要素）．メタモデルへの要求仕様を設定する上では，1つのレビュー観点から複数の要求が導かれたり，複数のレビュー観点をまとめて１つの要求仕様にすることもある．例えば，宇宙機システムのデータレコーダ容量（Data recorder capacity）を確認する上では，レコーダに対するデータ量の収支バランスを確認する必要があり，発生するデータ量（Calculate the amount of data size each communication）とレコーダから出力するCalculate the amount of data sizeの両情報が運用モデル上で表現されている必要があるためである．

検証の結果，運用メタモデルに対する要求仕様の13個のうち，対応関係が取れたのは12個であり，1個は未対応との判定になった（7.1）．未対応となった部分は，シナリオ全体が所定の時間内に完了するかというレビュー観点において，各アクションにおける所要時間を表現することが出来なかった．そこで，運用ステレオタイプメタモデルに，Durationタグを追加する改良を実施した．Durationタグは，すべてのアクションに共通することから，運用ステレオタイプメタモデル

の最上位の SatOpe ステレオタイプのタグ付き値として設計した．なお，図 5.8 は，すでに Duration タグ追加後の改良版を示している．

表 7.1: 運用メタモデル検証の結果

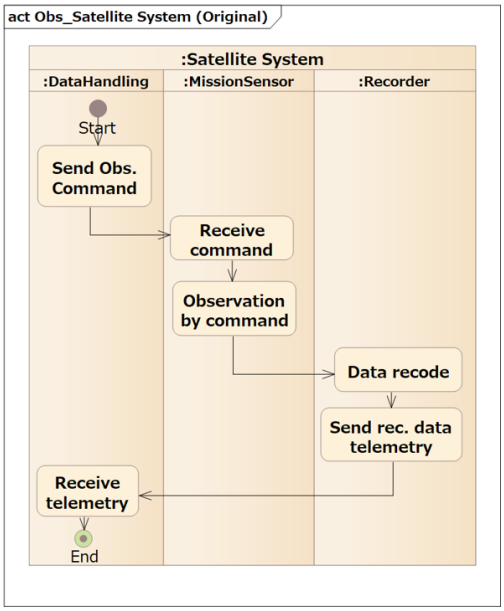
項目	要素数
レビュー観点	16
レビュー観点に基づく，運用メタモデルへの要求仕様	13
要求仕様に対する充足数	12
運用モデルの欠陥数 （要求仕様が満たせていない数）	1

7.1.2 手法比較

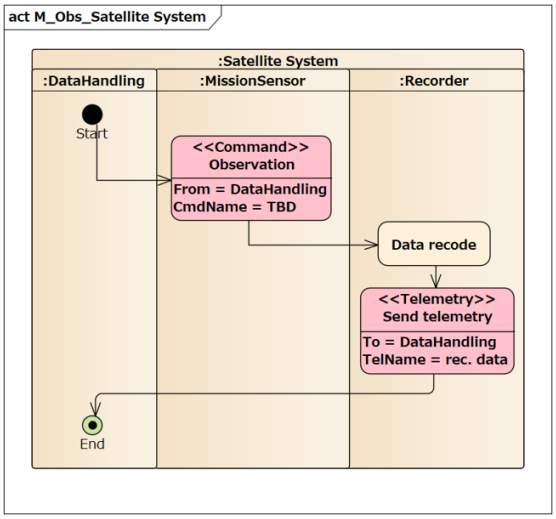
サンプルシナリオを用いて，運用メタモデルによるコンパクト化手法と実際の開発現場で使用していたコンパクト化手法，MBSE ツールの機能を用いた 3 つの方法の比較を行った．評価観点としては，レビュー品質の有効性向上を狙って，視認性に着目したレビュー容易性向上が目的であることから，要素の削減数，描画サイスの縮小，新たな作図が不要であること，およびシナリオレビューの観点からアクターとアクション関係が維持できることの 4 点を設定した．新たな作図が必要なことを観点として含めた理由は，図が分かれると全体像が把握しにくくなること，および合計のダイアグラム数が増加するためである．アクターとアクションの関係性維持については，シナリオレビューの本来の目的を果たすためである．

コンパクト化前の元となるサンプルシナリオは，図 7.1a である．評価を実施するうえで，宇宙機システムの運用で必須となるコマンドとテレメトリのシンプルな振る舞いが含まれており，前半の 3 つのアクションが Command，後半の 2 つアクションが Telemetry の Common behavior に対応する．なお，実際の開発現場で使用した手法は，運用メタモデルを用いた本手法と比較して，Common behavior の該当部分を 1 つにアクションにまとめる点が同じであるが，ステレオタイプを使わずに複数のスイムレーンを跨った横長のアクションで表す点が特徴である．MBSE ツールの機能を使う方法は，common behavior を UML に標準で備わっている Call behavior action 要素に置き換える方法である [9].

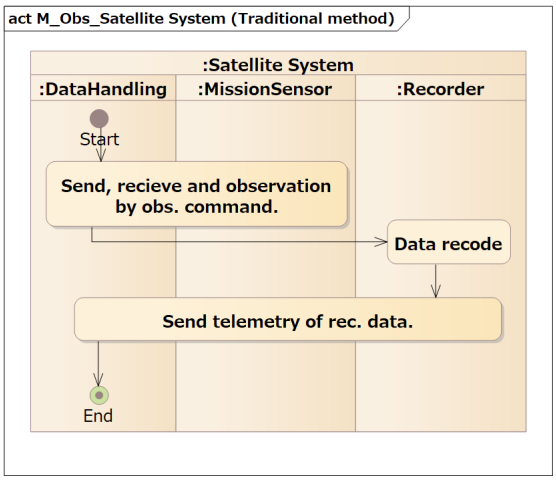
実験の結果として，運用メタモデル適用を使用した方式を図 7.1b 及び現場の従来方法を図 7.1c に示す [43]．コンパクト化の範囲は，どちらの手法もコマンドとテレメトリに関する Common behavior の部分が 1 つの要素にまとまり，要素数も同等の 5 個であった．しかし，開発現場で使用した手法（図 7.1c）では，アクションとアクターの関係性が崩れる結果となった．元のシナリオである図 7.1a では Mission sensor アクターは，テレメトリの運用には関与しないが，図 7.1c にお



(a) 元シナリオ



(b) 運用シナリオメタモデル適用



(c) 従来の現場で使用していた手法

図 7.1: コンパクト化手法比較

いては Mission sensor のスイムレーン上に，アクションが表示されているため，誤解を招く可能性が高く，レビューには適さないことが分かった．

MBSE ツールの機能を使う方法の結果は，Call behavior action 要素の呼び出し先として，common behavior の部分に対応した新規のアクティビティ図が必要となる．MBSE ツールの中には Call behavior action をクリックすることで呼び出し先のアクティビティ図が表示され，確認することができる．要素数も増えることとなる．よって，評価観点での NG となるのが明らかとなった．

これらの 3 つ手法の評価観点における結果を図 7.2 に示す．3 つの手法においては，提案手法の運用メタモデルの適用方法がコンパクト化において優位性があることが分かった．

表 7.2: コンパクト化手法の比較結果

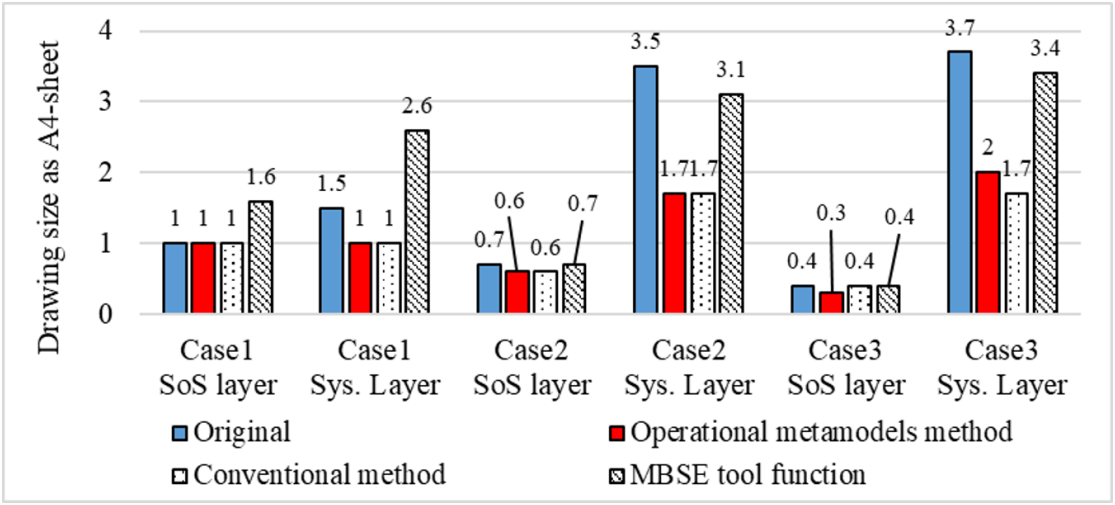
評価観点	運用メタモデル適用	現場の従来方法	MBSE ツールの機能
削減した要素数	○: 元の要素数より削減	○: 元の要素数より削減	×: 元の要素数より増加
描画サイズの縮小	○: 元の描画サイズより縮小	○: 元のの描画サイズより縮小	○: 元の描画サイズより縮小
新たな作図が不要	○: アクティビティ図の総数が同じ	○: アクティビティ図の総数が同じ	×: 追加のアクティビティ図によりレビュー対象が増加
アクターとアクションの対応付け	○: アクターとアクションの対応関係を維持	×: ケースのよりアクターとアクションの対応関係が不明確となる	○: アクターとアクションの対応関係を維持
総評	○: コンパクト化手法として有能	×: アクションとアクターとの対応関係が崩れる	×: 新たな図が必要となり，総要素数が増加

7.1.3 実シナリオ適用評価実験

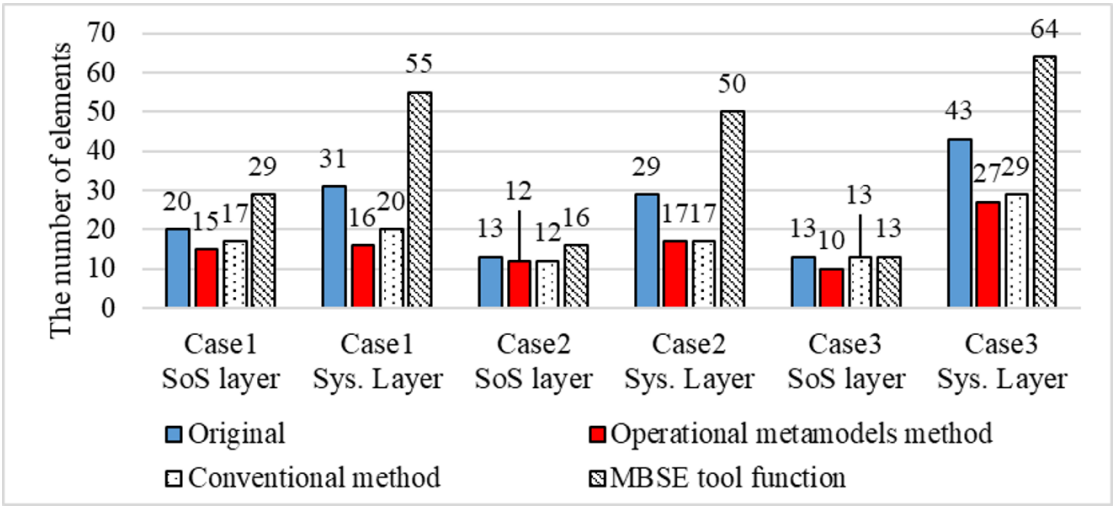
実際の JAXA の運用シナリオを想定した題材にコンパクト化行い，実開発での効果を評価した．手法は，7.1.2 節で示した 3 つの手法で評価する．実験では，代表的な 3 つのシナリオを選定し実施した．ケース 1 としては，地球観測のシナリオであり，地球観測衛星における代表的なシナリオである．ケース 2 としては，ロケット分離直後のシナリオであり，宇宙機システムにとって初めての宇宙環境での稼働となるクリティカルなシナリオである．ケース 3 は，レビューを困難にさせる要因である長いシナリオであり，本実験では宇宙機システムの軌道制御に関わる一連の作業を対象とした．また，開発レイヤーよる効果の差異も比較するため，System of systems レイヤーと System レイヤーの 2 つを用意し，合計 6 ケースとした．

実験の結果は，提案手法の運用メタモデルを用いた方法が，全ケースにおいてコンパクト化効果が高かった（図 7.2）[53]．MBSE ツールにおける Call behavior action を用いた方法では，適用前より要素数及び描画サイズが増えるケースが存在した．原因は，Call behavior action に置き換え後，Call behavior action が呼び

出す先として新たなアクティビティ図を作成する必要があり，合計の要素数及び描画サイズで増加したためであった．ちなみに，呼び出されるダイアグラムが共通であり，再利用が多くなれば効果的であるが，宇宙機システムの運用シナリオの特徴として，同じ振る舞いに見えても，コマンド名だけが異なるなど似て非なる部分が多い．そのため，少しでも違う場合には，新たなダイアグラムが必要となった．その点において，提案手法する手法は，コマンド名の違いはタグ付き値を変えることで吸収できるため，新たなアクティビティ図を必要とせず優れている．提案手法は，共通部分をステレオタイプで，差分をタグ付き値で表現することで，効率的なコンパクト化に寄与できた．



(a) 描画サイズ比較



(b) 要素数比較

図 7.2: コンパクト化結果

7.1.4 コンパクト化効果測定

提案する手法は，表7.3に示す通り，7.1.3節で使用したケースにおいて，System of systemsレイヤーでは80%程度，Systemレイヤーで50%程度のサイズに変換できた．System of systemsレイヤーに比べて，Systemレイヤーのコンパクト化効果が高いのは，運用メタモデルの前提とするドメイン知識がSystemレイヤーに重きがあったことが考えられた．例えば，コマンドの振る舞いはSubsystem間のやり取りを前提としてCommon behaviorを定義していた．それ以外の理由としては，System of systemsレイヤーのシナリオの方がSystem レイヤーのシナリオと比較して，最初から小さいためコンパクト効果が出にくいという点もあった．

表 7.3: コンパクト化率

Layer	Case 1	Case 2	Case 3	**平均**
要素数				
System of systems layer	75%	92%	77%	**81%**
System layer	52%	59%	63%	**58%**
描画サイズ（A4ページ換算）				
System of systems layer	100%	86%	75%	**87%**
System layer	67%	49%	54%	**56%**

ステレオタイプ毎に適用数とコンパクト化の効果を表7.4，7.5に示す．Commandステレオタイプは，運用シナリオ上，宇宙機システムを地上から制御するうえで必須の手順であるため出現頻度が高く，表7.4に示す通り適用数も一番多い結果となり，かつコンパクト化の効果に一番寄与した．加えて，Command ステレオタイプは3つの要素を1つにまとめると共に，吹き出しで示した関連情報もタグ付き値として取り込むため，表7.5の削減要素数の平均より2要素以上の削減効果となった．一方，FDIR （Fault Detection, Isolation and Recovery）ステレオタイプは，表7.5より要素の削減効果がなかった．これはもともと実運用シナリオ上でFDIRのフローを1つの要素で表現していたため，ステレオタイプを付与するだけに留まり, 削減には寄与しなかったためである．

レビューの容易性について，描画サイズが小さい方がレビューしやすい[28]という前提でコンパクト化の方法を提案したが，運用シナリオをレビューする立場のJAXA職員3名に簡易的ではあるが，インタビューにより確認した．インタビュー項目は，コンパクトの前後でレビューが良くなったか，レビューに必要な情報がコンパクト後も含まれているか，コンパクト化に伴う懸念があるかであった．概ね，コンパクト化したほうがレビューしやすいという回答が得られた．また，コンパクト後に情報が不足しているという回答もなかった．それ以外には，タグ付

き値の TBD 表記がレビュー時の着目ポイントになるとのコメントももらえた．しかし，描画サイズはアクションの配置などにも左右されるため，より厳密さが必要との指摘もあった．

表 7.4: ステレオタイプの適用数

	Case 1		Case 2		Case 3	
Stereotype	Sos	Sys	Sos	Sys	Sos	Sys
SatOpe	2	2	1	0	3	0
Command	0	2	0	4	0	6
Request command	0	1	0	0	0	0
Telemtry	0	0	0	0	0	1
Stord Telemtry	0	1	0	0	0	0
Uplink	1	1	0	1	0	0
Downlink	2	2	2	2	0	0
FDIR	0	0	1	1	1	2
Total	5	9	4	8	4	9

表 7.5: ステレオタイプ毎のコンパクト化効果

		削減要素数	
Stereotype	適用総数	合計	平均
SatOpe	8	8	1.0
Command	12	26	2.2
Request command	1	3	3.0
Telemtry	1	2	2.0
Stord telemtry	1	1	1.0
Uplink	3	4	1.3
Downlink	8	8	1.0
FDIR	5	0	0
Total	39	52	1.3

7.2 オフノミナルシナリオ評価実験

本項では Bounded model checking を SAT solver で実装し，運用シナリオの実行可能なルート探索とそれに伴う状態遷移の列挙について実験を行い，レビュー

対象として状態遷移の列挙を加えるで宇宙機システム設計へのフィードバックが可能かを評価する．また，SAT solver で実装においては，基礎的な性能評価として，運用シナリオの長さ，UCA 追加，状態空間が増えることによる処理時間への影響を確認した．次に実際の宇宙機システムを模した題材を対象にノミナルシナリオ，およびオフノミナルシナリオの実行可能ルートの探索および状態遷移のグラフ作成を行い，運用シナリオ評価から宇宙機システムへの設計への反映を評価した．[41]

7.2.1 SATsolver への実装

SAT solver としては，Python の Z3 プログラムを用い，探索により発見された状態遷移ルートは PyVis library を用いて有向グラフを作成した．評価対象となる状態は，運用異常イベントメタモデルより Mode, Parameter, Running Comunication を設定し，各状態を 0 か 1 で示し，ある時点における状態の組合せをビットベクターとしてビット列で表わす．次に，Transiton Constraint として，シナリオの 1 ステップで変化可能な状態配列をすべてを Statemachine，State Transition by Scenario 及び State Transition by UCA から作成する．Transiton Constraint は，式 6.1 であり，1 ステップの遷移のみである．シナリオ検証するために，式 6.1 を時間方向に展開する．すなわち式 6.5 で示す通りステップ数を付与することで，各ステップの状態配列を区別する．なお，本プログラムでは，1 ステップで変化する状態を 1 箇所に限定する．

状態遷移に基づく Conjunctive Normal Form（CNF）は，アクターの増加に伴い状態配列が増えること，および今後 0,1 以外の値にも対応する拡張性を想定し，状態配列の何番目が 0 か 1 かというプログラム実装を取った．なお，実験時のステップ数は，評価対象の運用シナリオに依存して Start から End まで含まれるアクション数で設定する．

また，検証する運用シナリオのアクションの実行順序は，一例であるため，実際はアクションの実行順序が入れ替わっても成立するルートが存在する．例えば，地上局の 1 と 2 の停止を考えた場合，どちらが先に停止するかは実運用では分からないため，どちらでも成立するかを探索する．よって，本プログラムは，実現可能なルートが 1 つ見つかるたびに，制約として SAT solver に追加し，実行可能なルートがこれ以上見つからないまでループで回すことで，すべてのルートを検出した．

7.2.2 SATSolver の性能評価

SAT solver の性能を図るために，ノミナルシナリオを想定した衛星及び地上局数増加に伴う状態空間の増加による影響評価，及びオフノミナルを対象に衛星及び地上局を固定し，UCA を追加した場合の影響評価を行った．なお，計測に用いる PC

のスペックは Intel Core i7-10810U CPU, 16GB メモリ, 512GB SSD, Windows11 である．

7.2.2.1 状態数による処理時間評価

衛星数及び地上局数を増やすことによる処理時間及び実行可能なルーツ数を指標に SAT Solver の性能評価を評価した．処理時間は，CNF を作成する時間 (CNF Time)，と実際の SAT Solver で解析している時間（Solver time）を分けて計測した．状態数が多い場合，対応した論理式を作成するプログラム部分も処理負荷が高くなる．そのため，実際の SAT を解く時間は短期間でも論理式を作成する部分で処理時間がかかっていることがある．それらを区別して評価するために，計測時間を区別して評価した．他の指標としては，探索により発見されたルート数を評価した．本評価で用いたアクティビティ図は Start 要素から End 要素まで一本道とするシンプルな運用シナリオとする．例えば，地上局 1 と地上局 2 は決め打ちで地上局 1 が停止後に地上局 2 が停止するとした．SAT Solver では，状態遷移のガード条件（システム設計上の制約）を満たす範囲で，アクションの実行順序を入れ替えるため，プログラム上は地上局 2 が先に停止して地上局 1 が停止することも可能であるため，2 つルートがあるとしてカウントする．なお，見つかるルート数が多ければ，処理時間は長くなる．

解析ケースは，アクターである衛星と地上局を共に 1 つずつ増やしながら，処理時間と発見されたルート数を評価した．衛星と地上局は基本的に 1 対 1 で通信するため，本評価では同時に 1 つずつ増やすこととした．また，処理時間の評価が目的であることから，シンプルに 1 つのアクターに 1 つのアクションと 1 つの状態を持つ必要最低限の構成とした．ガード条件は，地上局がすべて停止してから衛星のモード変化を許可するという制約だけ設定した．

解析結果を表 7.6 に示す．1 アクター 1 状態を持つ設定とした解析であるため，アクターの合計数は状態数と一致した．ステップ数は，シナリオの Start 要素から End 要素までステップ数であり，ループを持たない一本道のシナリオであるためアクション数と一致する．衛星数及び地上局数が 2,3 までは 1 分もかからず処理が完了していた．4 衛星 4 地上局の場合は CNF 作成に 2 分を要し，処理時間の 2/3 を占めていた．5 衛星 5 地上局の場合は，全体として 90 分程度かかっており，CNF 作成より SAT を解く時間の割合が多くなってきたことが分かった．5 衛星 5 地上局の場合における発見されるルートは 4 衛星 4 地上局と比較して 25 倍の 14400 ルートであり，探索の処理負荷が高くなったと推測した．6 衛星 6 地上局のケースでは，CNF の作成だけ 2 時間以上を要した．そこから SAT の処理が実行されるが，事前の予測では 518400 ルートあると考えており，2 時間以上経っても処理が終わらなかった．なお，実際の評価作業においては，処理時間が長くなることが予想された場合，順次発見されるルートをリアルタイムで確認し，想定外のルートや状態遷移が確認された段階で，処理を一旦停止し，制約等を増やすことでルート数を

減らすことを行う．そのため，評価として１ルートの算出平均時間も項目として追加した．その結果，ルート数増加に伴う１つ当たりの発見時間も増加しており，1 ルートの発見が遅くなってきた段階で，処理を停止するという解析上の工夫を取ることとした．

7.2.2.2　UCA による処理時間評価

UCA を追加に伴う SAT Solver の処理時間を評価した結果を表 7.7 に示す．評価項目は表 7.6 と同じであるが，入力条件として UCA を追加した．UCA は，実際のシナリオにおいて複数発生する可能性もあるため，１つ，２つと増やした場合の性能を評価した．UCA の追加に伴い，Recovery Action と合わせて 2 アクションを追加した．そのため，ステップ数は UCA 追加の度に，異常への遷移と復帰の 2 アクション分の追加とした．なお，UCA 追加の影響を評価するため，衛星及び地上局は，7.2.2.1 項を参考に作業効率を考えて，3 システムずつで固定した．結果は UCA 追加に伴い，シナリオの任意の箇所で異常への遷移と復帰が発生するためルート数が増加した．しかし，それに対して処理時間は大幅に増えていなかった．例えば，同じステップ数 10 である Table7.6 の 5 衛星 5 地上局と 7.7 比較ししても，90 分と 1 分未満である．このことから処理時間は UCA の追加によるステップ数より，状態数が増えることによる探索範囲が広がる方が，影響が大きいと分かった．

表 7.6: 衛星数，地上局数による処理時間

Case No.	解析条件				結果			
	衛星数	地上局数	状態数	ステップ数	CNF Time (min)	Solver Time		抽出ルート数
						合計 (min)	平均１ルート (min)	
1	2	2	4	4	0.023	0.001	0.013	4
2	3	3	6	6	0.243	0.020	0.031	36
3	4	4	8	8	1.866	0.682	0.071	576
4	5	5	10	10	13.646	54.309	0.226	14400
5	6	6	12	12	89.235	Not completed	54.309†	-

†: 処理終了していないため 100 ルート発見時点での平均値.

表 7.7: UCA 数による処理時間

Case No.	解析条件					結果			
	衛星数	地上局数	**UCA 数**	状態数	ステップ数	CNF Time (min)	Solver Time		抽出ルート数
							合計 (min)	平均１ルート (min)	
2	3	3	**0**	6	6	0.243	0.020	0.031	36
2a	3	3	**1**	6	8	0.357	0.077	0.038	120
2b	3	3	**2**	6	10	0.484	0.973	0.043	1344

7.2.3　実シナリオ評価

オフノミナルシナリオ評価においては，コンステレーション衛星によるミッションを想定した題材を用いて実験を行う．昨今の宇宙機システムは，従来の大型衛星で 1 つのミッションを達成する方法から，複数の衛星を連携させてミッション達成するトレンドがある．複数の衛星を制御するためには，従来の地上局から常に制御を行う運用から，衛星を自律化させた省力化の運用及び地上局に依存しない継続可能なシステムが求められる．日本の測位衛星システムである QZSS においても，地上局の障害に対応して宇宙機システムが自律的にサービス継続を行う Autonomous navigation (AUTONAV)[47] が検討されている．本実験のシナリオの概要図を図 7.3 に示す．問題を簡易的にするために宇宙機システムを 2 機，地上局を 2 基とする．通常状態では，1 つの宇宙機システムに対して 1 つの地上局が常時リンクを取っている前提（静止衛星等の運用）である．そのうえで，地上局が全停止しても継続的な運用が可能なシステムを目指す．具体的には，地上局からリンクが途絶えた場合に，衛星間通しで通信を行いミッションを継続する．自律的な宇宙機システムは地上の障害時に強く，運用の省力化に繋げることが期待できる．

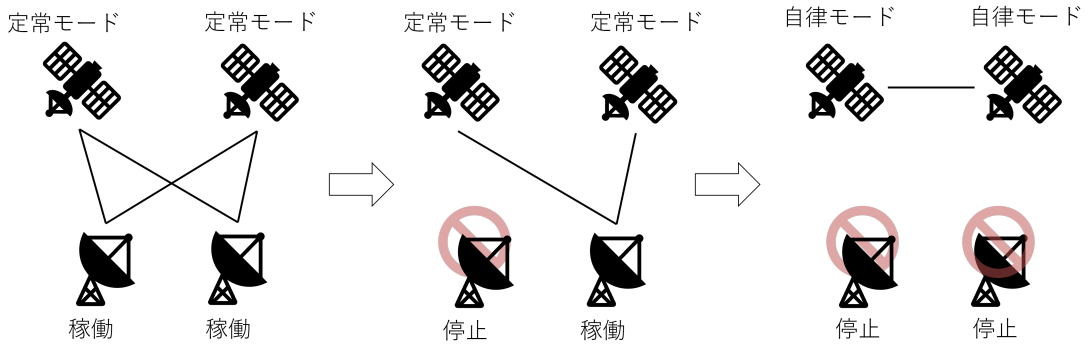


図 7.3: 自律による複数衛星運用

7.2.3.1　状態定義

本実験では，図 6.8 をノミナルシナリオとして対象とした．宇宙機システムは 2 機，地上局も 2 機とする．この場合における解析対象の状態定義を図 7.4 に示す．Sat-1 の状態は 4bit で示される．最初の bit は Sat-1 がもつモードであり，0 が Normal で定常モード，1 が Autonomous で自律モードである．2bit 目は Sat-1 に関するパラメータであり，ここでは衛星間通信に必要な Sat-2 の軌道情報を示す．0 が Disable で無効（有効期限切れ），1 が Enable で有効（有効期限内）を意味する．3bit 目は衛星間通信の状態を示しており，0 が Disable で未接続，1 が Enable で接続（通信中）を示す．4bit 目は衛星と地上局間の通信状態を示す．0 が Disable で未接続，1 が Enable で接続（通信中）を示す．なお，Sat-1 は地上局からの通信

がまったくなくなった場合をトリガーとするため，地上局は Gnd-1 および Gnd-2 をまとめて 1bit で表現した．Sat-2 の状態 4bit も Sat-1 と同様であり，Sat-2 用には 5bit 目から 8bit 目までを割り当てる．9bit 目は Gnd-1 の運用状態であり，0 が Suspend で停止中を示し，1 が Opearting で稼働中を示す．10bit 目は，Gnd-2 の運用状態であり，内容は 9bit 目と同じである．よって，本シナリオでは状態を表すビットベクターあは 10bits とした．なお，図 7.4 は，便宜上，状態がアクター毎に分かりやすいように中括弧で区切って示している．

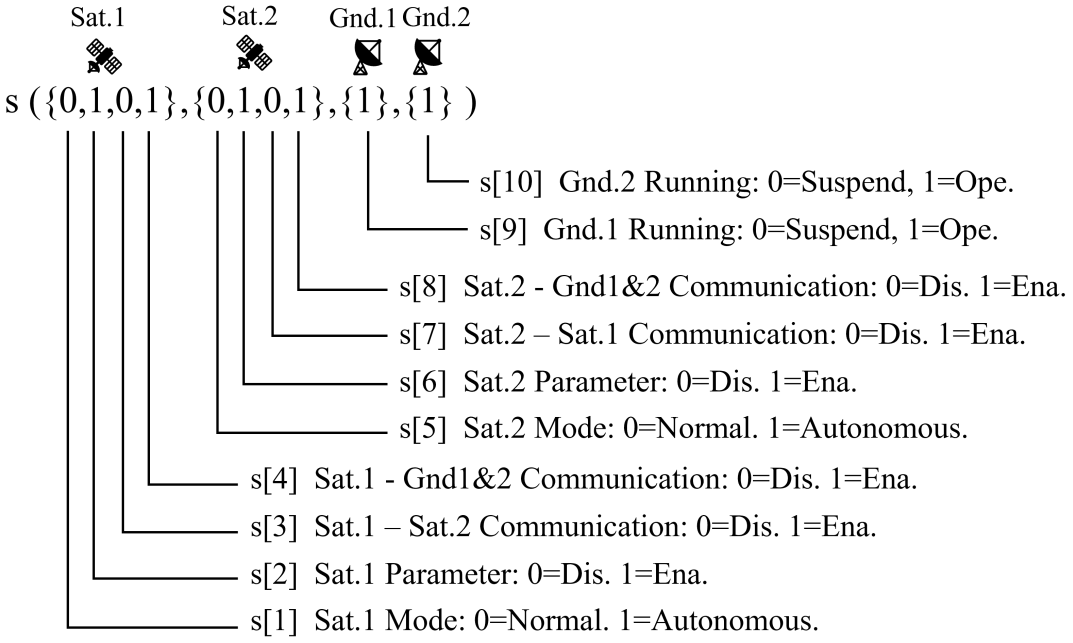


図 7.4: State definition for actual scenario.

7.2.3.2 システム状態遷移設定

システム状態を示すステートマシンは宇宙機システムのステートマシン，地上局のステートマシン，各システム間の数珍状況を示すステートマシンの 3 つを設計した．各ステートマシンにおける SAT solver への設定を述べる．

宇宙機システムのステートマシンを図 7.5 に示す．宇宙機システムの状態は，宇宙機システムの内部状態を示す mode 及び軌道情報に関わる parameter の 2 つで構成した．Mode について，初期値は 0 で示された Nominal（定常モード）であり，受理状態を 1 の Autonomous（自律モード）とする．すなわち，シナリオに従い状態遷移が起こり，定常モードから自律モードに移行し完了することを評価する．シナリオの各ステップで同一状態への遷移を許すため，各状態に対して自己ループを設定する．Parameter は，通信相手先の軌道情報が有効化無効化を示す状態を持つ．初期状態は 1 の有効であるが，シナリオに基づき無効化された場合，有効状態に戻らなければ受理されない設計とした．

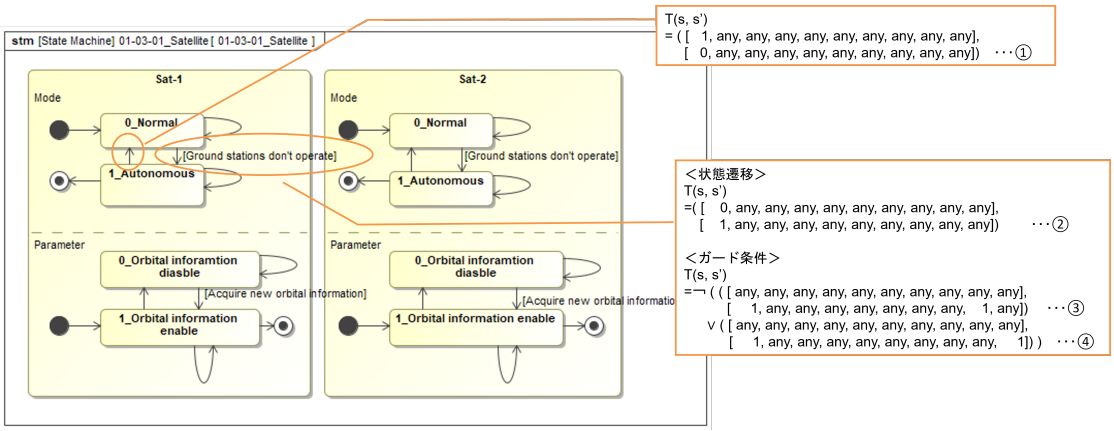


図 7.5: 宇宙機システム状態遷移設定

状態遷移を SAT Solver に設定する場合は，各状態は図 7.4 で示した通り，対応する bit が存在する．例えば，図 7.5 の Sat-1 の Mode において，1 の Autonomous から 0 の Nomal に遷移する場合は，状態の 1bit 目が対応する．そのため，状態遷移 $T(s, s')$ は下記の式として設定する．(図 7.5 の①式)

$$T(s, s') := ([1, any, any, any, any, any, any, any, any, any], \\ [0, any, any, any, any, any, any, any, any, any]) \tag{7.1}$$

any は無関係の bit 部分であるため，0,1 を問わない．最初の 1bit 目のみが 1 から 0 に変更となる．一方，0 の Nomal から 1 の Autonomous に遷移する場合は，“Ground station don’t operate” というガード条件が付与される．そのため，通常の状態遷移に加えて，ガード条件としての状態遷移 $T(s, s')$ として下記のガード条件を付与する．

$$\begin{aligned} T(s, s') := \neg(&([any, any, any, any, any, any, any, any, any, any], \\ &[1, any, any, any, any, any, any, any, 1, any]) \\ \vee&([any, any, any, any, any, any, any, any, any, any], \\ &[1, any, any, any, any, any, any, any, any, 1])) \end{aligned} \tag{7.2}$$

遷移前はすべて any で状態を問わないが，遷移後において，1bit 目と Gnd-1 が稼働中の 1 を示す 9bit 目が 1 となる組合せは否定する．また，地上局 Gnd-2 もあるため，同様に 10bit 目が 1 となる組合せも否定をする．そして，どちらか一方が否定すればよいので，両式の論理和の否定を取る (図 7.5 の③④式)．他の状態遷移も同様に，通常の状態遷移に加え，ガード条件がステートマシン図で示されているものを SAT solver に設定する．

次に地上局のステートマシンを図 7.6 に示す．地上局は稼働と停止の 2 つの Runing を示す状態とした．初期状態は稼働中を示す 1 の Operation であり，シナリオを実行することで受理状態を 0 の Suspend とした．地上局は，宇宙機システムと

は独立して動作するためステートマシンも別で定義した．状態遷移については，各地上局に対応する 9bit 目または 10bit 目が変化する．なお，停止から稼働への遷移，すなわち「0 → 1」は定義していないため，$T(s, s')$ も設定しない．図 7.6 では，Gnd.1 の 1 の Operation から 0 の Suspend への $T(s, s')$ を例示として示す．他の遷移も同様である．

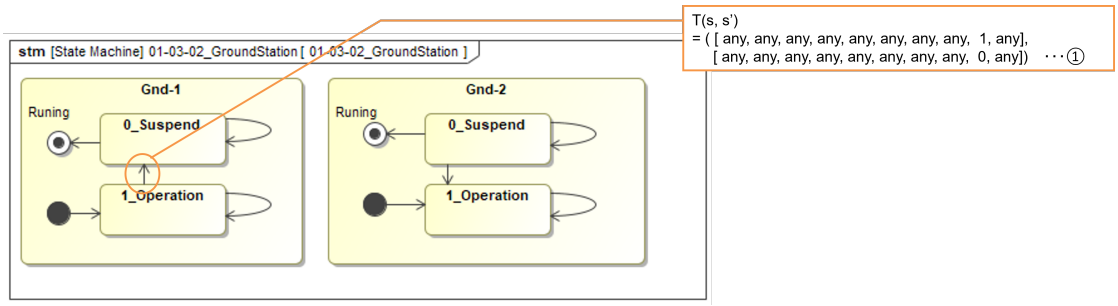


図 7.6: 地上局状態遷移設定

宇宙機システムと地上局間，宇宙機システム間の通信状態は，Communication のステートマシンとして図 7.7 に示す．本シナリオにおいては図 7.4 と整合するように 4 つを設計した．宇宙機システム間通信は Sat-1 から Sat-2 への方向と，Sat-2 から Sat-1 への方向の 2 つを分けて状態管理する．これは Sat-1 から通信要求を行い，Sat-2 から応答がなく，Sat-1 は通信中であるが，Sat-2 は未接続と認識している状態を表現するためである．地上局との通信状態は，Sat-1 向けと Sat-2 向けで区別して状態設計した．地上局はどちらか片方が稼働していれば，宇宙機システムと通信可能であるため，地上局は Gnd-1 と Gnd-2 で合わせて 1 つとした．一方，宇宙機システムは Sat-1 の地上局に対する稼働状況の把握と，Sat-2 の地上局の把握は，各々で独立して認識するため，分けて状態を設定した．

通信状況の SAT solver への設定については，他のステートマシン同様に，通常の状態遷移とガード条件を設定する．図 7.7 の①は，Sat-1 が Sat-2 へ衛星間通信の要求を行い，未接続から通信中となる遷移である．この時，Sat-1 が所持している Sat-2 の軌道情報が有効期限内でなければならないというガード条件がある．宇宙機システムは，宇宙空間を飛行しており，時々刻々と位置が変わるため，相手の位置情報が分からないと通信が成立しないという制約から設計した．ガード条件に対しては，図 7.7 の②に示す通り，Sat-1 が所有する Sat-2 の軌道情報有効無効を示す 2bit 目が無効の場合は，通信状態を示す 3bit 目が Enable になることを否定する．他の遷移についても同様に，SAT solver に設定する．

これらのシステム状態遷移設定は，システム設計やシナリオ実行上の制約から発生するものである．よって，SAT Solver によりステップごとにシナリオの状態を探索するが，常に上記で設定した遷移条件が考慮されていることとなる．

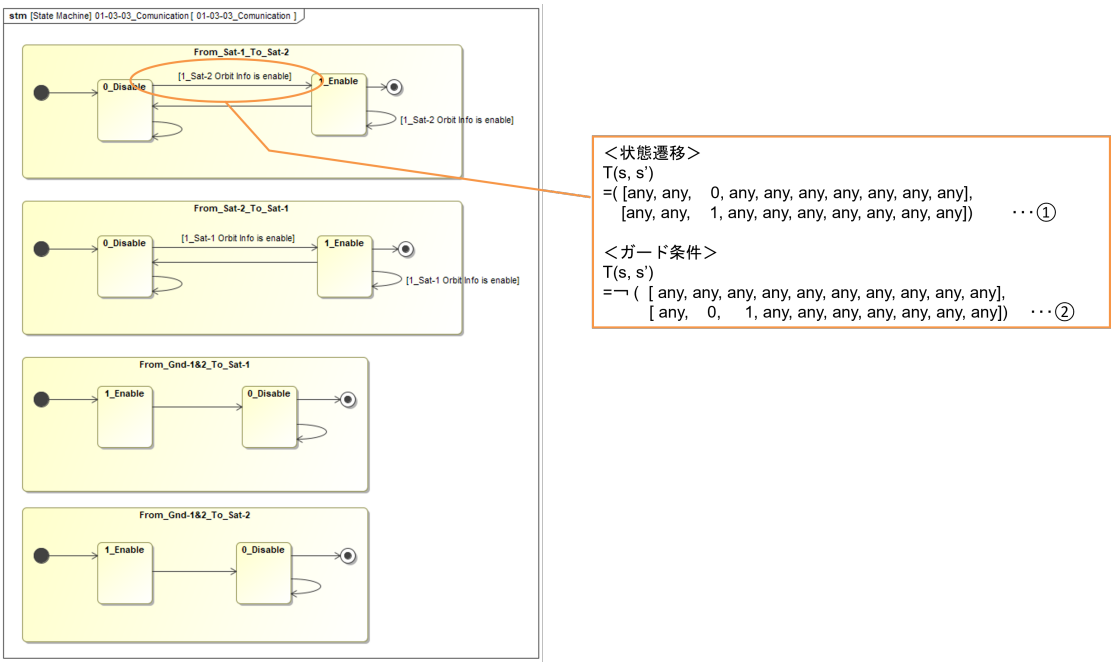


図 7.7: 通信状態遷移設定

7.2.3.3 シナリオ状態遷移設定

シナリオ実行に伴う状態遷移は，アクティビティ図上のアクションが実行されるたびに，高々１つの状態 bit が変化する．各アクションにおける状態遷移は，SAT solver ステレオタイプのタグ付き値で示しす（図 7.8）．タグ付き値に設定する状態は，前項で示したステートマシンの状態と整合性が取る．StateFrom タグ付き値には遷移前の状態，StaeTo タグ付き値には遷移後の状態を指定する．なお，MBSE ツールを用いれば，タグ付き値に設定する値をステートマシンで使用されている状態から参照することができる．すなわち，ステートマシンの状態名を変更した場合，アクティビティ図上の状態名も同期して変わるため，情報の一元管理が出来る．

アクションに伴う状態遷移の SAT Solver への設定については，図 7.8 の “Sat-1 changes from normal mode to autonomous mode.” というアクションを例に説明する．このアクションは，Sat-1 のスイムレーン上にあるため，Sat-1 が実行するアクションである．そして，SAT solver ステレオタイプにより，StateFrom = 0_Normal, StateTo = 1_Autonomous というタグ付き値から，状態の１ bit 目の Sat.1 の Mode 部分が「0 → 1」に変化することを意味していると読み解く．また，Start 要素から考えて，4 番目のアクションであるため，ステップ 4 で実行され，ステップ 5 で状態が変化すると考える．これらを踏まえると状態遷移は下記となる．

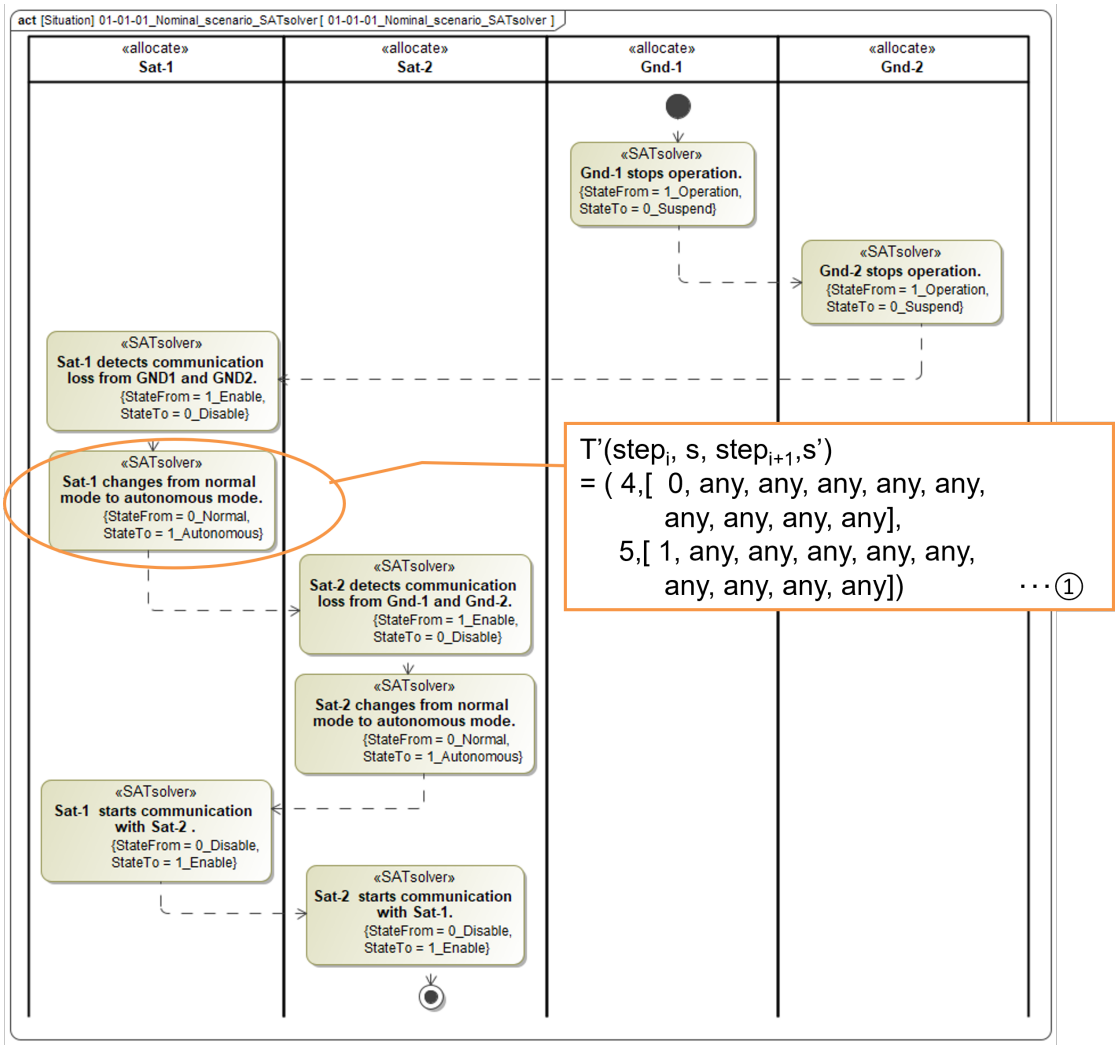


図 7.8: シナリオ状態遷移設定

$$T\text{'}\,(step_i, s, step_{i+1}, s\text{'}) \\ := (4, [0, any, any, any, any, any, any, any, any, any], \\ 5, [1, any, any, any, any, any, any, any, any, any]) \tag{7.3}$$

なお，実際の SAT solver の網羅探索時は，ステップ数を可変パラメータとなり，例えば，ステップ番号 1 で実行可能か，ステップ番号 7 で実行可能かなどを探索することで，図 7.8 で示された実行順序はある一例に過ぎず，アクションの実行順序を入れ替えても成立する場合のルートを探索する．その際，ガード条件として設定した式 7.2 が制約となり，実行可能かが判断される．それ以外のアクションも同様であり，SAT solver ステレオタイプによって示された遷移に従い設定する．

このように，SAT solver がアクションの実行順序を決めるステップ数を変更させて成立解を見つけることで，有限のステップ数における実行可能なルート探索

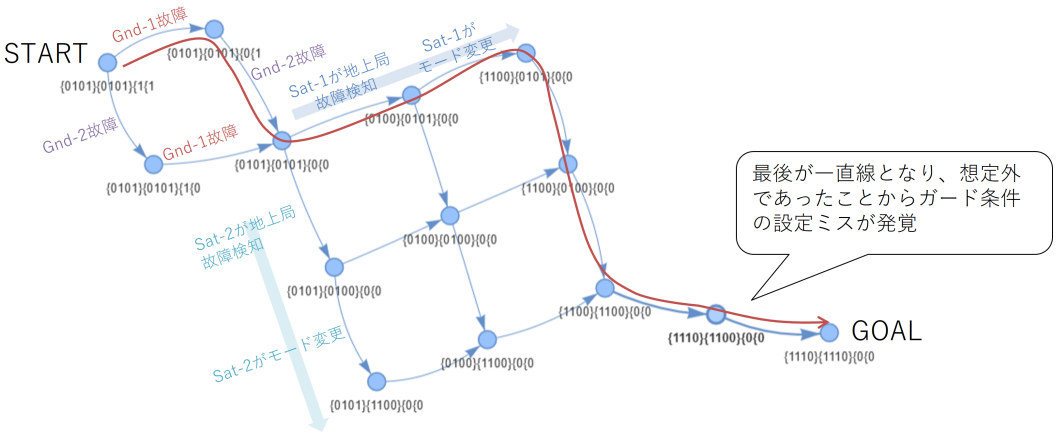


図 7.9: SAT Solver によるノミナルシナリオ解析結果（ガード条件不備）

結果から，どのような状態遷移を辿ってきたかを記録してグラフ化することで，状態空間の網羅的な遷移について可視化し，エンジニアに対して想定外への状態がなかったことレビュー情報として示すことができる．

7.2.3.4　ノミナルシナリオ評価

ノミナルシナリオを対象に SAT solver により評価した結果を図 7.9 に示す．各ノードは State を示し，１つのアクションを実行することで，次のノードに遷移する．ガード条件として，地上局が停止後に衛星がモード変更を設定しているため，すべての地上局が停止した状態を示す ({0,1,0,1},{0,1,0,1},{0},{0}) ノードを必ず経由するルートとなった．Gnd-1 と Gnd-2 は，Gnd-1 から停止するルートと Gnd-2 から停止するルートが算出されていることが分かる．({0,1,0,1},{0,1,0,1},{0},{0}) ノード以降も，衛星の地上局停止の検知，モード変更，衛星間通信については，Sat-1 と Sat-2 のどちらが先に実行するかは複数ルートがあることが分かる．しかし，GOAL 付近において図 7.9 は一直線となってしまった．

本来は Sat-1 と Sat-2 の衛星間通信の開始に順序性がない．図 7.9 の実行時は誤った制約を課していたことに気づき，Sat-1 と Sat-2 の順序を固定となった．誤りを是正したのが図 7.10 である．このように 1 つのシナリオにおいても複数ルートが存在するが，グラフとして可視化することにより，エンジニアに対してエラー発見の気づきを与えてくれることが分かった．図 7.10 は想定通りのルートが算出された結果である．赤色の線は，図 6.8 通りに実行した場合である．オレンジの線はシナリオとしては成立する別のルートが示されている．具体的には，Sat-1 が Sat-2 と通信開始してから，Sat-2 が地上局の停止を検知するケースとなっている．複数のルートが検出されても，グラフとして整理することで，例えば ({0,1,0,1},{0,1,0,1},{0},{0})

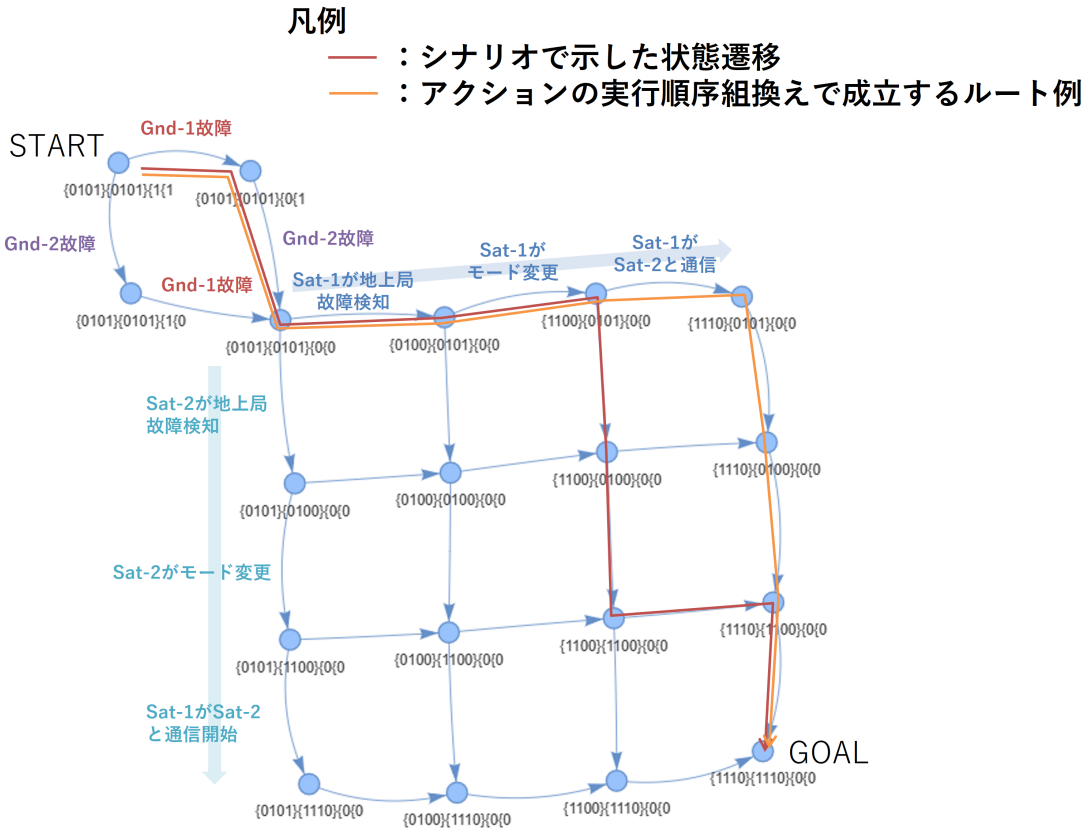


図 7.10: SAT Solver によるノミナルシナリオ解析結果（ガード条件修正後）

ノード以降は，右に進めばSat-1 のアクションが進行し，下に進めばSat-2 のアクションが進行すると整理することができる．よって，大量にルートが検出された場合も，想定したグラフの形や，矢印の方向性を確認することで，問題があるかを評価することができるようになる．そして，必要以上のルートや想定外のルートがある場合は，ガード条件を増やすと共に，対応した衛星システム設計に反映することが可能となる．

7.2.3.5　オフノミナルシナリオ評価

本項で評価するオフノミナルシナリオは，図6.5a で示したUCA を図7.10 のノミナルシナリオに追加して作成した派生シナリオである．実行結果を図7.11 に示す．UCA は，Sat-1 のパラメータであるSat-2 の軌道情報の提供が遅れて，それに伴い衛星間通信が確立できないという流れである．図7.11 では，Step7 にUCA を挿入した場合のケースを示した．Step7 のUCA の実行により赤矢印の状態遷移が発生した．そして，赤枠で囲った両機の状態は，Sat-1 がSat-2 の有効な軌道情報を保持していない状態を示している．この領域からRecovery Action による軌道情報をSat-2 から入手するというアクションが実行される．Recovery Action に対応するフローは

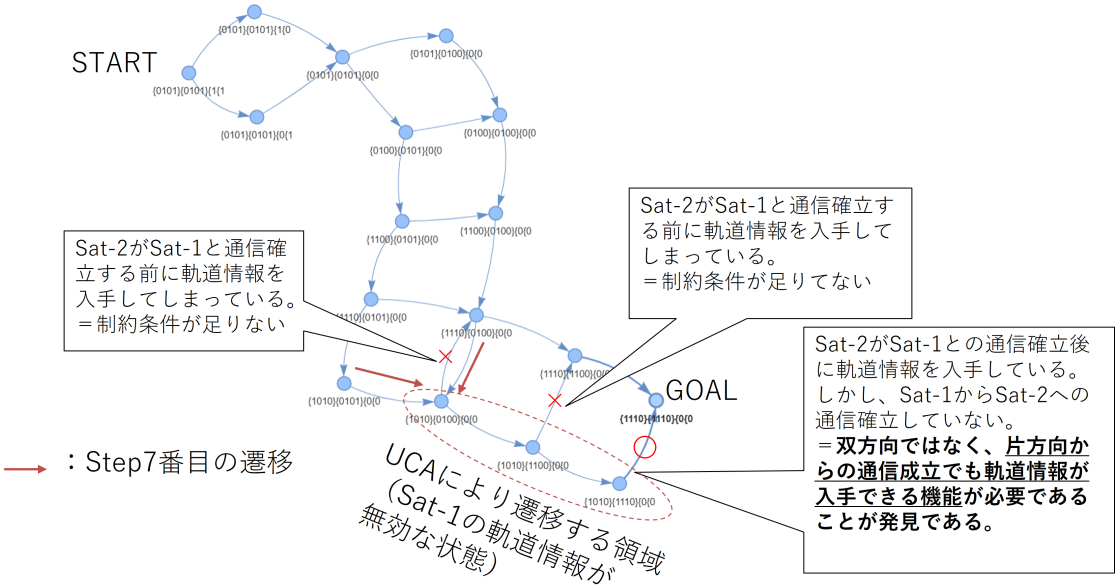


図 7.11: SAT Solver によるオフノミナルシナリオ解析結果

３つあった．状態 ({1,0,1,0},{0,1,0,0},{0},{0}) から ({1,1,1,0},{0,1,0,0},{0},{0}), ({1,0,1,0},{1,1,0,0},{0},{0}) から ({1,1,1,0},{1,1,0,0},{0},{0}) の 2 つの遷移については，実現不可でありガード条件の不足が検知できた．Recovery Action は Sat-1 が Sat-2 の軌道情報を Sat-2 の衛星間通信で取得する処理であるが，どちらも 7 番目の値が 0 を示しており，Sat-2 は衛星間通信を実施していない状況であった．({1,0,1,0},{1,1,1,0},{0},{0}) から ({1,1,1,0},{1,1,1,0},{0},{0}) の遷移は，Sat-2 が衛星間通信を開始しており，実現可能な遷移であることが分かった．

本解析結果から UCA に対しては，Sat-1 が所有する Sat-2 の軌道情報が無効になっても，Sat-2 からの衛星間通信で提供できればノミナル状態に復帰できることが分かった．ただし，この遷移から，Sat-1 から Sat-2 への衛星間通信が確立していない状況であることから，一方的に Sat-2 からの情報提供を受け取れる衛星システムの設計が必要であることの知見を得ることができた．また，ガード条件は，運用シナリオ実行において必要な要求であり，ガード条件に対応した衛星システム設計になっていることを確認することも実開発において有益だと判明した．

第8章　考察

本章では，(1) 肥大化した運用シナリオのレビュー容易性向上，(2) 安全解析と連携したオフノミナルシナリオ評価の目的別，及びそれらを連携した全体について，各節に分けて考察を述べる．

8.1　アクティビティ図のレビュー容易性向上

肥大化した運用シナリオのレビューにおいて，本研究では視認性に着目したレビュー容易性向上を行い，有益な指摘及び正しい理解ができる有効なレビューが出来るように，レビュー対象であるアクティビティ図のコンパクト化を実施した．

8.1.1　階層化効果

本項では運用レイヤー定義メタモデルを中心とした考察を行う．運用レイヤー定義メタモデルの適用により，アクティビティ図を開発レイヤーに対応した階層化を行い，記述粒度を揃えたため，多少の視認性向上の効果はあったと考える．一般的に複雑な図を読みやすくする手法として階層化の手法が取られるためである．過去の経験から運用シナリオの作成時，作成者の心理として，知っていることは記述粒度をないがしろにしても多く書くという傾向があるように思える．そのため，担当や経験したシステムの運用シナリオ部分は，他のシステムより細かく記述する傾向があった．その結果，アクティビティ図のスイムレーン毎に，アクションの要素を確認すると，1つのスイムレーンのみ，詳細なステップや情報が含まれることがあった．運用レイヤー定義メタモデルでは，運用シナリオが対象とする開発レイヤーを予め定義し，スイムレーンには上位，下位の2つまでしか設定できないように制約をかけた．そして，スイムレーンに配置されるアクションはアクターがもつ内容に限定をするため，例えば，システムレイヤーのスイムレーンに，コンポーネントレイヤーの内容を記述することに違和感が出て，記述粒度が異なることに気づくことができる．これにより作成者自らが，アクションの記述粒度に着目でき，アクティビティ図作成の習熟度があがったと感じている．

なお，今回の実験では，運用レイヤー定義メタモデル適用は直接的なコンパクト化効果を狙ったものではないため，定量的な評価は行わなかった．運用レイヤー定義メタモデルは，コンパクト化における事前準備であり，アクションの記述粒度を

ガイドすることで，運用ステレオタイプメタモデルで設計したCommon behaviorの一連のアクションが一致しやすくした．

本メタモデルにおける一般性については，本メタモデルを設計するうえで宇宙機システムの多層構造を前提として名称を付けるなどの設計をしたが，1つ下の階層に機能分配をして設計を進めるという方法はシステムズエンジニアリング[17]の手法に沿ったものである．システムズエンジニアリングは宇宙機システムに限らず，他の分野でも適用可能である．よって，システムを多層構造に分けて開発する場合において，運用シナリオを作成する際は他の分野でも適用可能と考える．

8.1.2 コンパクト化効果

本項では運用ステレオタイプメタモデルを中心とした考察を行う．運用シナリオのコンパクト化においては，運用シナリオのウォークスルー形式のレビューを前提に，レビューの有効性を向上させるために，視認性に着目したレビュー容易性向上を図った．視認性向上がレビューにおいて有益な指摘が出来るという先行研究に基づき，スクロールを不要とするA4一枚に収まる図のサイズを目指し，定量的な指標として図に含まれる要素数とサイズを評価した．サイズの図にすることで視認性を，運用ステレオタイプメタモデルを適用することにより，レビューに適したコンパクト化したアクティビティ図を作成した．本手法は，従来の現場で実施していた方法及びMBSEツールを用いてCall behavior actionを適用する方法と比べても，レビューとして情報を保ちつつ，高いコンパクト化効果が得られた．コンパクト化前は3〜4ページに渡って，多くのスクロールを必要とするシナリオが，図7.2及び表7.3の結果より，コンパクト化によりが最大で2ページに収めることができ，レビューの容易性に関わる視認性向上が確認できた．また，補足ながらもインタビューを通じて同様のコメントが得られた．他の宇宙機システムの運用シナリオにおける効果は，実シナリオを想定した代表的なシナリオで実験しており，同様のコンパクト効果がが得られると考えている．特にシステムレイヤーの運用シナリオは，主にコマンドやテレメトリの振る舞いが多くを占める．実シナリオを想定したシナリオにおいても，表7.4の結果よりコマンドステレオタイプが適用多く，表7.5の結果よりコマンドステレオタイプのコンパクト化効果は高いことが分かる．よって，他の宇宙機システムの運用シナリオにおいてもコマンドを用いたcommon behaviorがあり，コンパクト化の効果が得らると考える．

本メタモデルにおける一般性について，他分野での利用は難しいことが予想されるが，他の宇宙機システムにおける運用シナリオへの適用は可能と考える．宇宙機システムの基本的な運用は，地上システムからコマンドを送信し，宇宙機システムからテレメトリを受信するの繰り返しである．そして，宇宙機システム開発は，信頼性の観点から過去の宇宙機システムの設計を再利用するケースが多く，運用においても，すでに共通認識があるフローが何回も出現すること予想されるためである．ただし，他の宇宙機システムに適用する際は，事前準備として用語

の定義が図 5.3 の運用知識と整合しているか，図 5.7 の Common behavior が一致するかを確認する必要がある．もし差異がある場合は，その宇宙機システムに合うように修正したうえで使用することで対応する．

8.2 オフノミナルシナリオ評価

安全解析と連携したオフノミナルシナリオの作成及び，各シナリオが持つ状態遷移を列挙したグラフをレビュー対象とし付加することで，これまで頭の中の想像した範囲で評価していた状態遷移を可視化し確認できることで，レビュー品質の網羅性向上を図った．

8.2.1 安全性解析との連携

本項で STAMP/STPA の連携について考察を行う．7.2 節のオフノミナルシナリオ評価の実験においては，題材が宇宙機システムと地上システム間のやり取りであり，レイヤー定義メタモデル（図 5.5）において System of Sysytems のレイヤーの運用シナリオである．FTA により UCA 相当の異常を検討する場合，トップダウン方式であるため，安全性解析にて洗い出したい System of Sysytems の視点における異常状態が先に必要となり，適用が難しい．FMEA はボトムアップ方式であるため，System of Sysytems より下位のシステムの異常が，上位である System of Sysytems の視点でどのような異常（故障モード）があるかを探索する方法であり，思考の順序としては適している．しかし，本研究で題材とした「地上システムの稼働が停止した場合に宇宙機システムが自律的に正しく動作するか」という評価においては，地上システムが停止するという前提がそもそも異常とも考えられ，故障モードの設定が難しい．

STAMP/STPA では，コントロールストラクチャー（図 6.2）を用いて各システム間の相互作用関係を可視化し，その相互作用に対して想定外の振る舞い（例えば，Too late，Not Providing 等）が与えるシステム全体への影響を評価できた．分析手法としては FMEA のボトムアップ方式であるが，着眼点がシステム間の相互作用であり，運用シナリオもシステム間のインタラクションであることから，コントロールストラクチャーで示したフローがアクティビティ図のフローとも対応するため，安全分析の結果が運用シナリオ上で特定しやすく，相性の良い評価方法であった．

想定外の振る舞いとしては，Hazop のガイドワード（[52]) の適用も検討した．Hazop は化学産業における危険減の特定にあたる連続プロセスが対象である．そのため，Hazop のガイドワードには，設計意図の否定，論理的反対，完全な転置という論理的な内容に加えて，量的増加や減少，質的増加や減少など物理現象に主眼を置いた内容が含まれている．運用シナリオにおけるシステム間のインタラク

ションは，無線による通信であり，電気的なインタフェースであることから，物理的な変化を対象としたガイドワードはUCA検出には適さないと考えた．また，設計意図の否定，論理的反対，完全な転置は，STAMP/STPAにおいて類似の振る舞いがありカバーされていると評価した．例えば，設計意図の否定は,「設計で意図したことが全く起こらない」という解説からSTAMP/STPAの “Not Providing” が対応する．よって，本研究においては，新たな想定外の振る舞いの追加検討を行ったが，現時点においてはSTAMP/STPAで十分との結論となった．

8.2.2 オフノミナル生成

本項では運用異常イベントメタモデルを中心に考察を行う．オフノミナルシナリオの生成においては，運用異常イベントメタモデルの役割は情報の整理であり，MBSE上での情報の一元管理を図るためである．運用異常イベントメタモデルにより，安全解析手法STAMP/STPAの結果に基づきに洗い出されたUCAとそのUCAに伴う異常状態のシナリオ，及びその異常から復帰するためのアクションの関係性を示した．これらをセットにして，ノミナルシナリオに提案したJoin Pointのステレオタイプを付与したディションノードにUCAを挿入することでオフノミナルシナリオが構成する．オフノミナルシナリオやUCAは既存のRAAMLで定義されたステレオイプを活用することで，STAMP/STPAも含め，すべての情報をMBSEのフレームワークで管理でき，情報の一元管理が可能となった．

また，運用異常イベントメタモデルには，運用シナリオの実行可能ルートを探索するうえで必要な情報として，対象とする状態のビットベクターの定義，アクション実行に伴う状態遷移，システム設計による状態遷移の制約の関係性も整理した．これより，ルート探索時の実行条件やシステム制約も含め，MBSE上で一元管理できる環境を整えた．ただし，構築したMBSEのSysMLモデルを自動でSATソルバーが読み込めるツール連携までは至っていない．将来的には自動変換ツールを開発し，作業効率の向上を図りたい．

運用異常イベントメタモデルの一般性について，オフノミナルシナリオはUMLおよびSysMLの代替フロー及び例外フローをベースとしており，STAMP/STPA及びRAAMLも宇宙機システムに限った技術ではないため，ノミナルシナリオとオフノミナルシナリオをMBSE上で整理するという観点においては，他の分野でも利用可能と考える．一方，運用シナリオの実行可能ルートを探索するうえで必要な情報としての状態は，宇宙機システムを前提とした設計であるため，同じ宇宙機システムにおいては利用可能であるが，他の分野で利用する場合は状態定義から行う必要がある．また，対象は連続した状態変化ではなく，ステートマシンで表現可能な離散的な状態変化に限る．

8.2.3　状態遷移の列挙

本項ではBounded model checking手法の探索機能をBounded SearchとしてSATソルバーで実装し，運用シナリオの実行可能なルート探索と状態遷移の列挙について考察を行う．１つの運用シナリオに対してBounded Searchを用いることで，実行順序の複数パターンの検出に成功した．本来，エンジニアが１つの運用シナリオを見ながら，頭の中でイメージして評価していたことを可視化したことになる．また，表7.6より衛星数，地上局数の増加による状態空間が増えると大量の複数ルートが検出される．ガード条件を付与することで，ルートの数を絞り込み，かつ図7.10で示す通りグラフとして整理することで，運用シナリオの状態空間の遷移を俯瞰して評価することができる．1つの運用シナリオに隠されていたルートを可視化させ，整理することにより，より網羅的な視点によるシナリオ評価が可能となったと考える．

また，運用異常イベントメタモデルを適用する利点としては，安全解析手法に基づくオフノミナルシナリオの作成において，関連する情報がMBSEフレームワーク上で実現できることで，情報の統制を図ることが出来た点である．これに伴いノミナルシナリオもオフノミナルシナリオも区別なく，Bounded Searchによる解析手法を適用することが出来た．特にUCAの挿入位置は，自由度があるため，すべて網羅的にシナリオを作成することは困難であった．UCAの分析の結果をオフノミナルシナリオとして作成し，Bounded Searchにより状態遷移を評価することで，UCAが発生した後，およびそこからの復帰において必要な設計要求を抽出することができた．Unsafe scenarioもRecovery actionもエンジニアに依存した設計であるため，客観的な評価と網羅的な評価が困難であった．本論文で示した通り，UCAによるノミナルシナリオでは到達しない状態に遷移することが可視化され，それを元にした議論ができるようになった．また，UCAの同時発生も考慮した解析を行うことを考えた場合，表7.7の結果から処理時間への影響は軽微であることがことが分かったため，今後は複数のUCAを含んだノミナルシナリオに対して評価を実施し，より複雑で技術者の目では気づきにくい不具合の早期発見を期待している．

今後の改善ポイントとしては，SAT solverの結果より網羅的な状態遷移について，エンジニアが想定外の状態遷移をレビューする際に，意味のある領域で解釈できるように表示を工夫する点である．例えば，7.11においては，UCAによりSat-1の軌道情報が無効な状態は，複数のノードを１つのグループとして領域として捉えた．シナリオが複雑となれば，ルート数も増えて，状態遷移のグラフもより複雑かつノードもエッジ数も増加する．その場合，1つ1つのノードで捉えて評価することは困難になると想定する．すなわち，ある特性の状態を持つノードを領域と捉える考え方が必要である．今回の実験では宇宙機システム数2，地上局2で実施した．例えば，大型人工衛星のコンステレーションは，日本の測位衛星「みちびき」で7機[39]であり，Star linkのような小型衛星によるコンステレーション

の場合は数千にも達する [45] ことがある．コンステレーションに対応した解析を進めるうえで，状態遷移を領域として捉えて表示する改善を今後の課題とする．

一般性について，他のシステムでの適用は，他の JAXA 衛星においてもノミナルシナリオ及びオフノミナルシナリオという考え方はあり，どちらもアクティビティ図で表現可能であることから，他の宇宙機システムの運用シナリオのレビュー評価としても利用可能と考える．従来からアクティビティ図とステートマシンの整合性評価方法は多く研究されており，本研究の提案手法はその応用問題である．

8.3 レビュー方法改善

本節ではアクティビティ図のレビュー性向上及びオフノミナルシナリオ評価の連携した総合的なレビュー方法の改善について考察を述べる．最初の運用シナリオは，詳細を記載するがゆえに，アクティビティ図が肥大化したため，レビュー困難という課題があった．そこで本研究ではレビュー品質の向上として，コンパクト化により視認性の向上を図り，レビューが出来る状態にすることでレビューの有効性向上を図った．次に，異常状態を含む想定外のオフノミナルシナリオを対象に，運用シナリオが持つ潜在的な状態遷移をグラフにより可視化し，レビュー時の評価材料として加えることで，レビュー品質の網羅性の向上を図った．有効性と網羅性の向上を図ることで，間接的に運用シナリオのレビュー評価の改善を図った．

本研究において，上記の 2 つの改善を 1 つの題材での一気通貫で実験が出来ていない点が今後の課題である．オフノミナルシナリオ評価の実験においては，題材としてレビュー可能なシンプルな運用シナリオを用意したため，前半のコンパクト化が終わっている状態であった．特に System of Systems レイヤーの運用シナリオは，前半の実験のおいても記述粒度が荒いため，複雑になりにくい性質を持っていた．なお，一気通貫で実施する場合において，運用ステレオタイプメタモデルのステレオタイプが適用されていた場合，コマンド送信の Too late などは Common behavior の中に含まれてることになる．そのため，Join point を挿入する際は，その部分についてはステレオタイプを適用しない方法を取ることで対応する．

第9章　おわりに

新しいミッションに挑み，信頼性の高い宇宙機システムを開発するためには，最終的な利用イメージである運用シナリオを各専門のエンジニアで共有しレビューをすることで，システム設計への要求抜け低減，想定外も考慮したロバストな宇宙機システムに仕上がる．特にエンジニアが気づけていなかった事象を，実際の運用前の開発段階で気づけることは重要である．そのためにも，より有効的な運用シナリオのレビューが実施できることが望ましい．本研究では，エンジニアによる運用シナリオレビューに着目して，コンパクトなアクティビティ図を生成するためのメタモデル，オフノミナルのレビューを充実させるために状態遷移と網羅的な実行ルートの探索の自動化を図った．コンパクトなアクティビティ図では，概ね縦方向に A4 一枚のスクロール不要のアクティビティ図を作成することに成功した．オフノミナルのルート列挙においては，想定外の遷移を見つけることで宇宙機システム設計へのフィードバックが行えた．

本研究で提案する手法は，MBSE をベースとしており，他の宇宙機システムの運用シナリオでも適用可能であることから，今後は他のプロジェクトにも適用し，知見及び実績を貯める計画である．それに向けて状態遷移のグラフを領域として捉えて利用者に提示する方法の改善に取り組んでいく．

謝辞

本研究に取り組むにあたり，長い間，主指導教員としてご指導いただいた青木利晃教授に感謝いたします．モデル検証，ソフトウェア工学などを中心に基礎から実社会での応用まで含め，ゼミ等を通じて幅広い専門知識をご教授頂き，より広い世界への視野を持つことが出来ました．副指導教員の鈴木正人准教授には，研究を進めるうえで要所でのご指導を賜りました．平石邦彦教授には，副テーマ研究，本論文の審査員としてご指導いただき，2.5 節の関連研究として比較検討をすることが出来ました．

冨田尭准教授，岸知二教授（早稲田大学），岡本圭史教授（仙台高等専門学校）には，本論文の審査員をお引き受けくださり，本研究に対して有益なご意見をくださいました．青木研究室の皆様とも，ゼミ等を通じて，活発な議論や意見交換をさせて頂き，良い刺激となり感謝いたします．

宇宙航空研究開発機構 研究開発部門 第三研究ユニットの皆様にも感謝いたします．すでに異動された方もおりますが，特に石濱直樹様，和田恵一様，舟生豊朗様にはさまざまなご尽力・ご助言，そして活発な意見交換をさせて頂きました．本研究を通じて得られた技術や知識は，今後の宇宙開発に活かせるように引き続き精進します．

最後になりますが，妻及び息子に感謝いたします．研究や論文執筆等に集中できたのは家族の協力あってのことです．そして，引き続き，今後ともよろしくお願いします．ご支援くださった皆様に重ねて御礼を申し上げ，謝辞とさせていただきます．

本研究に関する業績

論文誌（査読有り）

- [43] K. Someya, T. Aoki, and N. Ishihama. Metamodel for compaction of spacecraft operational scenario models. IEEE Access, 2025.
- [53] 染谷一徳, 青木利晃, 石濱直樹. 宇宙機システムにおける運用シナリオのコンパクト化メタモデル. 電子情報通信学会論文誌 D, 107(12):532–543, 2024.

国際学会（査読有り）

- [42] K. Someya, T. Aoki, and N. Ishihama. Compaction of spacecraft operational models with metamodeling domain knowledge. In Proceeding 18th Intl. Conf. on Evaluation of Novel Approaches to Software Engineering (ENASE2023), Prague, Czech Republic, 2023.
- [41] K. Someya and T. Aoki. A metamodel for enumerating off-nominal scenarios in operational scenario review through bounded model checking. In Proceeding 14th International Conference on Model-Based Software and Systems Engineering (MODELSWARD2026), Marbella, Spain, 2026.

国際学会（アブストラクト査読のみ）

- [44] K. Someya, N. Ishihama, K. Wada, and T. Aoki. Compaction of spacecraft operation model using domain knowledge based stereotype. In Proceeding 32nd Intl. Symp. on Space Technol. and Sci., Fukui, Japan, 2019. No. 2019-t-05.

国内研究会

- [54] 染谷 一徳, 平石 邦彦. Little-jil を用いた宇宙機システム運用シナリオのモデル検証. 信学技報, MSS2021-78:121–126, 3 2022.

参考文献

[1] M. Beckmann, V. N. Michalke, A. Schlutter, and A. Vogelsang. Removal of redundant elements within uml activity diagrams. In *2017 ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 334–343, Texas, USA., 2017. IEEE.

[2] A. Biere, A. Cimatti, E. Clarke, and Y. Zhu. Symbolic model checking without bdds. In *International conference on tools and algorithms for the construction and analysis of systems*, pages 193–207. Springer, 1999.

[3] A. Biere, A. Cimatti, E. M. Clarke, M. Fujita, and Y. Zhu. Symbolic model checking using sat procedures instead of bdds. In *Proceedings of the 36th annual ACM/IEEE Design Automation Conference*, pages 317–320, 1999.

[4] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu. Bounded model cheking. Vol. 58 of Advances in Computers, 2003.

[5] G. Biggs, T. Juknevicius, A. Armonas, and K. Post. Integrating safety and reliability analysis into mbse: overview of the new proposed omg standard. In *INCOSE International Symposium*, volume 28, pages 1322–1336. Wiley Online Library, 2018.

[6] G. Biggs, K. Post, A. Armonas, N. Yakymets, T. Juknevicius, and A. Berres. Omg standard for integrating safety and reliability analysis into mbse: Concepts and applications. In *INCOSE International Symposium*, volume 29, pages 159–173. Wiley Online Library, 2019.

[7] B. Chen. *Improving processes using static analysis techniques*. University of Massachusetts Amherst, 2011.

[8] E. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. *Formal methods in system design*, 19(1):7–34, 2001.

[9] Nesting of activity diagrams. `https://www.sparxsystems.eu/languages/uml/diagrams/activitydiagram/` [Accessed on Nov. 10, 2025].

[10] S. Feo-Arenis and J. Teraillon. The savoir road map of model-based avionics. In *ESA Workshop on Avionics Data Control. and Software Systems (ADCSS2017)*, Noordwijk, Netherlands, 2017.

[11] G. Figueiredo, A. Duchardt, M. M. Hedblom, and G. Guizzardi. Breaking into pieces: An ontological approach to conceptual model complexity management. In *2018 12th International Conference on Research Challenges in Information Science (RCIS)*, pages 1–10. IEEE, 2018.

[12] E. Fosse. Model-based systems engineering (mbse) 101, January 2014. INCOSE International workshop.

[13] E. Fosse, A. Devereaux, C. Harmon, and M. Lefland. Inheriting curiosity: Leveraging mbse to build mars2020. In *AIAA SPACE 2015 Conference and Exposition*, page 4617, CA, USA., 2015.

[14] S. Friedenthal, A. Moore, and R. Steiner. *A practical guide to SysML: the systems modeling language (3rd. Ed.)*. Morgan Kaufmann, 2014.

[15] S. Friedenthal and C. Oster. *Architecting Spacecraft with SysML A Model-Based Systems Engineering Approach*. Createspace Independent Pub, 2017.

[16] G. Guizzardi, G. Figueiredo, M. M. Hedblom, and G. Poels. Ontology-based model abstraction. In *2019 13th International Conference on Research Challenges in Information Science (RCIS)*, pages 1–13. IEEE, 2019.

[17] C. Haskins, K. Forsberg, M. Kureger, D. Walden, and R. Hamelin. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. International Council on Systems Engineering (INCOSE), v3.2.2 edition, 2011.

[18] S. J. Herzig, D. Velez, B. Nairouz, B. Weatherspoon, R. Tikidjian, T. M. Randolph, and B. Muirhead. A model-based approach to developing the concept of operations for potential mars sample return. In *2018 AIAA SPACE and Astronautics Forum and Exposition*, page 5389, FL, USA, 2018.

[19] S. R. Hirshorn. Expanded guidance for nasa systems engineering. volume 1: Systems engineering practices. Technical report, National Aeronautics and Space Administration Office of the Chief Engineer", 2016.

[20] Japan Aerospace Exploration Agency. *Spacecraft design standard*, 2013. JERG-2-000A.

[21] Japan Aerospace Exploration Agency. **システム設計標準**, 2016. JERG-2-100.

[22] Japan Aerospace Exploration Agency. **運用準備標準**, 2020. JERG-2-701.

[23] Japan Aerospace Exploration Agency. *Spacecraft design standard*, 2021. JERG-2-610B.

[24] X 線天文衛星 astro - h「ひとみ」 異常事象調査報告書, 2016.

[25] B. Lerner. *Getting Started with Little-JIL Case Study: Measuring Stream Discharge*, May 2010.

[26] N. G. Leveson. *Engineering a Safer World : Systems Thinking Applied to Safety*. The MIT Press, London, 2012.

[27] M. Lippert and C. V. Lopes. A study on exception detection and handling using aspect-oriented programming. In *Proceedings of the 22nd international conference on Software engineering*, pages 418–427, 2000.

[28] D. Lübke, M. Ahrens, and K. Schneider. Influence of diagram layout and scrolling on understandability of bpmn processes: an eye tracking experiment with bpmn diagrams. *Information Technology and Management*, 22(2):99–131, 2021.

[29] S. Madadpour, S.-H. Mirian-Hosseinabadi, and V. Abdelzad. Testing aspect-oriented programs with uml activity digarams. *International Journal of Computer Applications*, Volume 33(No.8):pp.4–11, November 2011.

[30] National Aeronautics and Space Administration Office of the Chief Engineer. *NASA Systems Engineering Handbook*, nasa sp-2016-6105 rev2 edition, 2016.

[31] N. Noda. *Aspect-Oriented Design for Embedded Software*. PhD thesis, Japan Advanced Institute of Science and Technology, Ishikawa, Japan, September 2008.

[32] Object Management Group. *OMG Satellite Operations Language Metamodel (SOLM)*, ver 1.0 edition, 2012.

[33] Object Management Group. *A UML Profile for MARTE: Modeling and Analysis of Real-Time and Embedded Systems*, ver 1.1 edition, 2012.

[34] Object Management Group. *OMG System Modeling Language (OMG SysML)*, ver 1.4 edition, 2015.

[35] Object Management Group. *OMG Unified Modeling Language (OMG UML)*, ver 2.5 edition, 2015.

[36] Object Management Group. *Meta Object Facility (MOF) 2.0 Core Specification*, ver 2.5.1 edition, 2019.

[37] Object Management Group. *Risk Analysis and Assessment Modeling Language (RAAML) Libraries and Profiles*, ver 1.0 edition, 2022.

[38] L. J. Osterweil, M. Bishop, H. M. Conboy, H. Phan, B. I. Simidchieva, G. S. Avrunin, L. A. Clarke, and S. Peisert. Iterative analysis to improve key properties of critical human-intensive processes: An election security example. *ACM Transactions on Privacy and Security (TOPS)*, 20(2):1–31, 2017.

[39] Qzss is becoming a seven-satellite constellation, 2023. `https://qzss.go.jp/en/overview/services/seven-satellite.html` [Accessed on Sep 27, 2025].

[40] M. Seidl, M. Scholz, C. Huemer, and G. Kappel. *UML@ classroom*. Springer, 2015.

[41] K. Someya and T. Aoki. A metamodel for enumerating off-nominal scenarios in operational scenario review through bounded model checking. In *Proceeding 14th International Conference on Model-Based Software and Systems Engineering*, Marbella, Spain, 2026.

[42] K. Someya, T. Aoki, and N. Ishihama. Compaction of spacecraft operational models with metamodeling domain knowledge. In *Proceeding 18th Intl. Conf. on Evaluation of Novel Approaches to Software Engineering*, Prague, Czech Republic, 2023.

[43] K. Someya, T. Aoki, and N. Ishihama. Metamodel for compaction of spacecraft operational scenario models. *IEEE Access*, 2025.

[44] K. Someya, N. Ishihama, K. Wada, and T. Aoki. Compaction of spacecraft operation model using domain knowledge based stereotype. In *Proceeding 32nd Intl. Symp. on Space Technol. and Sci.*, Fukui, Japan, 2019. No. 2019-t-05.

[45] Starlink network update, 2025. `https://www.starlink.com/updates` [Accessed on Oct 4, 2025].

[46] An stpa primer, 2013. `http://psas.scripts.mit.edu/home/wp-content/uploads/2015/06/STPA-Primer-v1.pdf` [Accessed on Sep 27, 2025].

[47] I. Takahashi, K. Akiyama, T. Katsuyama, K. Someya, Y. Yoshimura, E. Myojin, A. Matsumoto, H. Maeda, T. Sawamura, and T. Shibata. Feasibility

study of an autonomous navigation system using optical inter-satellite ranging technology. *Journal of Evolving Space Activities*, 2(140), 2024.

[48] D. Tamakoshi, Y. Hirayama, H. Kubota, and T. Inoue. Application of model-based systems engineering to satellite system development. In *Procceeding 32nd Intl. Symp. on Space Technol. and Sci.*, Fukui, Japan, 2019. No. 2019-t-07.

[49] A. Wise. Little-jil 1.5 language report. department of computer science, university of massachusetts. Technical report, Amherst UM-CS-2006-51, 2006.

[50] 岸知二 and 野田夏子. **ソフトウェア工学**. 近代科学社, 2016.

[51] 戸田貴久. モデル検査における反例空間の構造解析. In **人工知能学会研究会資料 人工知能基本問題研究会** *107* **回** *(2018/8)*, page 01. 一般社団法人 人工知能学会, 2018.

[52] 高木伸夫. 非定常 hazop の基本手順と進め方. **安全工学**, 53(4):244–251, 2014.

[53] 染谷一徳, 青木利晃, and 石濱直樹. 宇宙機システムにおける運用シナリオのコンパクト化メタモデル. **電子情報通信学会論文誌** *D*, 107(12):532–543, 2024.

[54] 染谷一徳 and 平石邦彦. Little-jil を用いた宇宙機システム運用シナリオのモデル検証. **信学技報**, MSS2021-78:121–126, 3 2022.

[55] 独立行政法人情報処理推進機構. **はじめての** *STAMP/STPA* **〜システム思考に基づく新しい安全性解析手法〜**, ver.1.0 edition, 2016.