# 人工衛星のソフトウェア IV&V における D-Case を利用した短期間でのフォーマルメソッド適用

大森 洋一[1]　日下部 茂[1]　林 信宏[1]　荒木 啓二郎[1]　神戸 大輔[2]　寺西 誠[2]　川口 真司[2]　梅田 浩貴[2]

**概要**：人工衛星の開発は軌道制御ハードウェア，基本ソフトウェア，ミッション固有機器など複数の専門家組織の分担により行われ，特にソフトウェア開発の初期段階における検証手段の確立が課題となっている．このように独立した組織間のコミュニケーションには仕様書が決定的な役割を果たす．フォーマルメソッドは，数理的なモデルにより対象を記述し開発の初期段階からの検証を可能とする．しかし，対象の仕様を理解するのに適した仕様書の記述項目や優先順位は，各専門家組織，たとえばソフトウェア開発者と検証者の間で必ずしも一致しない．このとき，関係者の間でフォーマルメソッドによるモデルの仕様全体における目的や位置づけを共有する必要がある．本研究では，検証目的に限定した情報のみを抽出することで、短期間でフォーマルなモデルを記述する手法を提案する．情報を限定するための手段として，検証すべき目的をゴールとするディペンダブルケース（D-Case）を作成・利用する．提案手法を実際の仕様書に対して適用したところ， D-Case の個別衛星へのカスタマイズ，仕様書とのマッピングにより，検証に必要な仕様理解に必要な時間を短縮できる，また不足している情報を明確に指摘できるといった成果が得られた．

**キーワード**：フォーマルメソッド, ディペンダブルケース, 仕様理解, モデリング

# Rapid Application of A Formal Method with D-Case in software IV&V for Artificial Satellites

***Abstract:*** Development of an artificial satellite is collaboration of multiple specialist teams, such as orbit control hardware, basic software, and mission inherent instruments. Therefore, establishment of the verification method especially in the early stage of software development is still a problem. Specification plays decisive role in the communication among such independent teams. Formal methods can describe the target as a mathematical model and verify it in early development stage. However, the contents and priority of the specification for understanding the target is not necessarily in accordance among different teams, for example, between software developers and verifiers. Stakeholders of the development team need to share the purpose and importance of the formal model in whole the specification because all aspect can not be expressed in the model. We propose a method to compose a rapid formal model with a dependable case (D-Case) which conducted goal analysis. As case studies, applying the proposed method achieved that required time for specification understanding could be shortened and the lacked information could be clearly pointed out.

***Keywords:*** Formal method, Dependable Case, Specification understanding, Modeling

## 1. はじめに

我々は， 2013 年度より「運用シナリオのモデル開発管理及び型式検証の先端技術に関する共同研究」をテーマに，フォーマルメソッドの具体的な人工衛星の開発および運用への適用について共同研究を実施してきた．本稿は，その成果の一部として， D-Case を用いたフォーマルメソッド適用のための仕様書理解およびその結果に基づく仕様のトレーサビリティ確保について述べる．最初に，人工衛星のソフトウェア開発における問題として，製品ごとの違いの大きさおよび，要求と実装手段の乖離について説明する．

[1] 九州大学 大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University
[2] 宇宙航空研究開発機構 情報・計算工学センター
JAXA's Engineering Digital Innovation Center, Japan Aerospace Exploration Agency

### 1.1 人工衛星の開発手順

人工衛星の用途は通信衛星，気象観測衛星，測地衛星，実験衛星などさまざまであり，その達成すべき目的であるミッションによって搭載する機器などが異なる．また，同じミッションの人工衛星であっても，搭載する機器の性能・能力により搭載する機器の大きさなども違い，それら各種機器に加えて，姿勢・軌道制御装置，電源系，地上との通信装置などの効率的な配置をしなければならず，慎重な設計が不可欠である．安全係数の見積りや推進剤の量などさまざまな決定に影響する衛星の寿命も，用途により 1 年程度から 10 年以上とさまざまである．このような事情により，各衛星は共通する部分もあるものの，独自に設計がなされており，形状や機能がそれぞれ異なる．

その一方で，人工衛星の打ち上げには重量に応じた高額の費用がかかることや，人工衛星に載せる機能が高度化・複雑化していることから，極限までの軽量化や個々の機能ハードウェアの小型化が求められている．このため，小型かつ軽量なハードウェアと実績のある動作アルゴリズムを組み合わせて，重量のないソフトウェア制御により実現される機能が増えており，ソフトウェア開発の初期段階における検証手段の確立が課題となっている．

人工衛星の標準的な設計手順として，National Aeronautics and Space Administration (NASA) はフェーズに分割されたプロジェクト計画および管理を確立している [8]．日本における宇宙航空研究開発機構 (Japan Aerospace Exploration Agency, JAXA) および各国の標準的な衛星開発手順もこれに準じたものである．しかしながら，具体的なプロジェクトについては，個々の衛星について事情が大きく異なることから開発の共通化は部品あるいはせいぜいサブシステム単位でしか行われておらず，ハードウェアも含めた複数回のウォーターフォールプロセスによる開発によりシステム全体の検証が行われている．このウォーターフォールプロセスは一回の実施が高価であり，開発対象のシステム仕様の明確化およびハードウェアが完成する前の仕様に対する検証および妥当性確認は，費用対効果の観点から重要である．

### 1.2 専門家集団の分離

人工衛星の打ち上げは高価かつ開発期間が長く，費用および開発期間の制約は非常に厳しい．このため，人工衛星開発においては信頼性や安全性が最優先され，独自のハードウェアや制御アルゴリズムが存在し，ソフトウェアも含めた部品やサブシステムの開発はそれぞれの専門家が分担して行う場合が多い．その一方で，ミッションを提案するそれぞれの分野の専門家は，それぞれのミッションがどのようなものかはよく知っているものの，それらを人工衛星にけるタスクとして詳細化し実装する部分についての理解は必ずしも十分でない．この結果，ミッションの達成に必要な要求を現在のシステム仕様が満たしているかどうかの確認が困難となる．

例えば，人工衛星の実用的なアクティブ姿勢変更手段は

- スラスターによる噴射
- 慣性ホイールによるスピン
- 地磁気を利用した磁気トルク発生

だけである．ただし，全ての衛星がこれら 3 つの手段を備えているわけではない．ミッションの性質による姿勢変更の頻度や耐故障性などを考慮した上で，できるだけ少ないハードウェアで実現するのが望ましいが，その判断には専門的知識が必要となる．

我々は，フォーマルメソッドの適用により，人工衛星開発の初期段階からシステム仕様を数理モデルにより記述・検証し，その後の工程で問題なく利用できることを保証できるような手法の開発を目標とする．本稿では，自然言語による仕様とフォーマルメソッドによるモデル化の観点を明確化する手法として，ディペダブルケース (D-Case) を併用する手法について検討する．

以下，第 2 章でフォーマルメソッドと D-Case によるソフトウェア検証ついて記述し，第 3 章ではこれらを組み合わせた仕様理解および理解した結果を明示する手法を提案する．さらに，第 4 章では事例研究について，第 5 章では関連研究について，第 4 章ではモデルによる検証結果の日本語仕様書へのフィードバックについて述べる．

## 2. 上流工程における検証

### 2.1 ソフトウェア独立検証

ソフトウェア独立検証 (Software Independent Verification and Validation, SW IV&V) は， NASA で 1993 年より本格的に実施されたミッションクリティカルなソフトウェアに対して最高の安全性・良好な費用効果を達成するために，開発とは独立した機関・組織が検証 (Verification) および妥当性確認 (Validation) を実施する「方法」「体制」「枠組み」である [13]．従来の，開発者による V&V も有効かつ重要な手順であり，テストやレビューをはじめ，フォーマルメソッドも含めた種々の技法が活用されてきた．しかしながら，開発者の検証では，開発業務のオーバヘッドとみなされてしまい十分な検証ができない場合が多くみられ，開発者と検証者を分業する体制が望ましい．JAXA においても 1996 年 の宇宙ステーション共同開発において NASA から要請があり，SW IV&V を開始し，衛星やロケットの開発に順次展開している．第 1 章で述べたように，人工衛星開発において，特に上流工程における検証は仕様記述者，システム作成者のいずれかだけでは困難であり，SW IV&V による検証は重要である．

しかし，実験衛星ではフォーマルメソッドの適用も含めた信頼性コストをできるだけ省略したい場合もあるなど，SW IV&V の観点では，すべての衛星のすべての側面へ

フォーマルメソッドを適用するのは適切ではない．実際，JAXA では開発フェーズ，検証の観点に合わせた評価の方法を図 1 のように整理している．

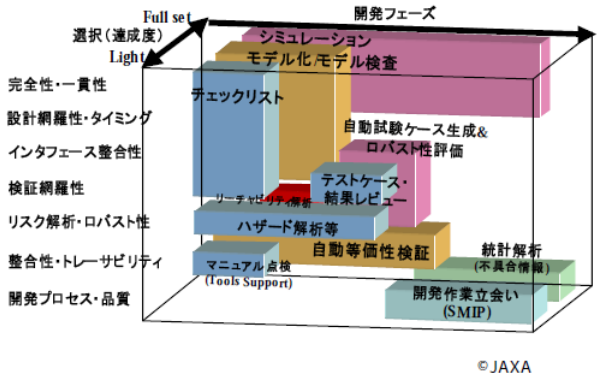


図 1　開発フェーズと SW IV&V 手法 [13].
**Fig. 1**　Map of SW IV&V Methods and Development Phases

SW IV&V においてフォーマルメソッドを利用する場合，検証者が対象の仕様を理解する時間が必要となる．また，複数の手法を組み合わせることもしばしばあるので，フォーマルなモデルの仕様全体における位置づけを明示し，関係者の間で共有する必要がある．

### 2.2　VDM++ の利用

フォーマルメソッドは，計算機システムの仕様を数理的に表記することおよび，その仕様を設計の検証基盤として利用する手法である [2]．フォーマルメソッドを適用し，ソフトウェアの仕様書を数理的なモデルに変換することで仕様書中の矛盾や曖昧さを取り除くことが可能となる．

一方，フォーマルメソッドによる仕様記述は，記述言語の習得が必要であり，背景となる数理的な知識が必要とされる．そのため，日常使用している自然言語による記述と比較して，読み手・書き手とも限られている．

本研究では，フォーマルなモデルの記述には VDM++を採用した．VDM++ はフォーマルメソッドの一種である VDM を構成する仕様記述言語であり，一階述語論理および集合論に基づく意味論をもつ [5].

仕様のモデル記述にあたっては，仕様書からどのような情報を読み取って，どのようなモデルとして実現するかを整理する必要がある．その過程は必ずしも明確でないため，モデル記述者の能力に依存する部分が非常に大きい．

### 2.3　ゴール分析の適用

組織的に独立した SW IV&V では短期間でのドメイン知識の獲得が求められる．しかしながら，検証のためにはフォーマルなモデルを書き始める前に，関係者の間でモデル記述の目的を共有しておかなければ，大きな手戻りが生じたり，いつまでも検証モデルが完成しなかったりする恐れがある．検証のゴールが仕様記述者と仕様検証者の間で共有できて，必要なサブゴールおよび関連した仕様およびドメイン知識のみを理解すればよいのであれば，検証開始時点で各領域の専門家ほどにはドメイン知識を持たない SW IV&V も実用的な期間で適用できる．

本研究で利用する D-Case はディペンダビリティの分野に適用した Assurance Case の一種である．Assurance Case は，想定している環境下において，システムが正しく動作することを，構造化することによって，体系立てて保証する方法または，そこで表記された議論の構造である [3]．Assurance Case はさまざまな規格において，具体的な記述法が定義されており，また記述のためのプロセスも定義される場合がある．

他の Assurance case と比較した D-Case の特徴は，Goal Structuring Notation (GSN) [7] に基づく分析を行う点と，運用に関する記述が考慮されている点である．これらの特徴は，ゴール分析の適用および明快な記述，検証に必要な情報が開発工程に限定されない，といった点で，本研究で想定する仕様書の理解および理解した結果の関係者による共有に適している．

## 3.　D-Case による検証目的共有の提案

本研究は，ソフトウェア開発の上流工程でのフォーマルメソッドを利用した短期間でのモデル記述および検証の実現を目的とする．

短期間での検証のポイントは

- 検証対象の明確化によるドメイン知識および仕様書理解範囲の限定
- 各種ツールの活用およびそれらの連携による検証作業の省力化

である．具体的には，前者はフォーマルメソッド適用によるモデル記述範囲を，クリティカルな部分に限定することで検証コストを下げ， D-Case により検証結果が仕様全体でもつ位置づけの明示，後者はフォーマルなモデルに対するツールを利用した検証およびその仕様書へのフィードバックである．

本研究では次の手順により，検証方針を D-Case として表現し，フォーマルなモデル化と検証を行う手法を提案する．

(1)　D-Case 記述者は，自然言語により記述された対象システムの仕様とドメイン知識に基づき，検証用 D-Case を作成する．

(2)　関係者の間で検証用 D-Case そのものおよび検証対象としての適切なゴールを選択して合意し， GSN の階層に基づいて対象システムの範囲を限定する．

(3)　仕様検証者は，限定されたシステムを，指定されたゴールの観点からフォーマルモデルとして記述し，モ

デル上の検証を行う．

(4) 仕様検証者は，検証用 D-Case で表現された議論の構造に従って，ゴールが達成されたか否かを判別する．

(5) 仕様記述者は，達成されなかったゴールを検証用 D-Case で表現された議論の構造に従って，仕様を改善する．

(6) 目標のゴールが達成されるまで繰り返す．

この結果，従来のフォーマルメソッド適用では，図 2 のような作業フローが，図 3 のような作業フローへ変更される．従来のフローでは，全ての関係者が全ての情報に関わっていたのに対し，仕様検証者はドメイン知識の獲得をしない，もしくは非常に小さいものにできる．一方で D-Case 記述者はフォーマルメソッドに関する知識を持っている必要はなくなるので，お互いの役割が明確になる．しかしながら，検証用 D-Case は，自然言語を用いるとともに図の要素の種類も少ないので，記述のための学習コストがフォーマルメソッドの習得と比べてかなり低い．したがって，ドメインに依存した議論の構造を表現するために仕様記述者が兼任することも容易である．

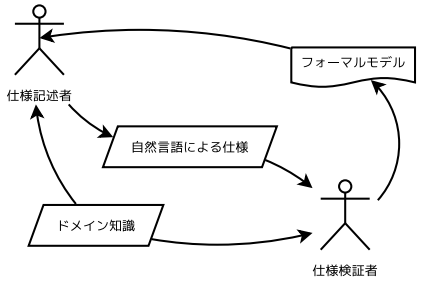


図 2 従来のフロー
**Fig. 2** Existing verification process

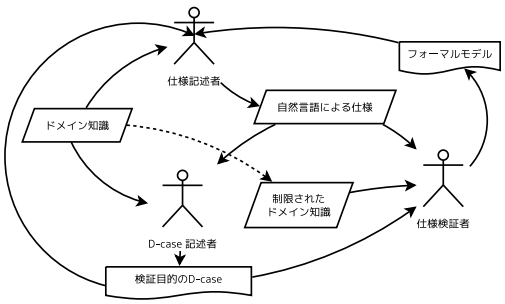


図 3 提案手法のフロー
**Fig. 3** Proposing verification process

## 4. 事例研究

### 4.1 人工衛星の姿勢制御

本節では，人工衛星の姿勢制御について，提案手法を適用した事例について具体的に述べる．それぞれの人工衛星には固有のミッションがあり，それぞれのミッションの性格により適切な姿勢が異なる．しかし，太陽電池による発電を利用する人工衛星では，蓄電量が減少した時に電池パネルを太陽方向へ向ける，逆に電池パネルの温度が上がりすぎた時に太陽方向を外すといった共通の処理もある．

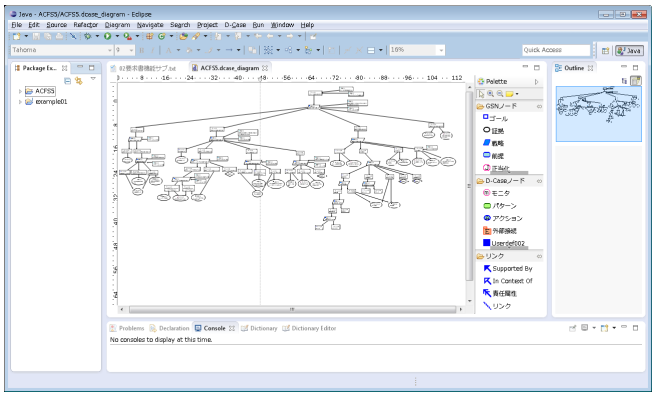

図 4 D-Case エディタ
**Fig. 4** User Interface of D-Case Editor

我々は、姿勢制御系について，参考文献 [1][4][9] によりドメイン知識を獲得し， D-Case を作成した．D-Case 記述には，図 4 に示す D-case Editor を用いた [12]．記述した D-Case は 40 の goal, 27 の evidence をもつ最大 7 階層の構成となった．

対象となる人工衛星の姿勢制御ソフトウェアに関する仕様書は 5 章， 218 ページ + 付録という構成になっているが，仕様書中にはモデル化の前提となる衛星のミッションや，姿勢制御の目的は記述がなく，外部資料を利用して補完した．作成した D-Case の goal から，状態遷移についての検証を行うことを選択し，フォーマルなモデルを作成した．VDM++ モデルおよび D-Case の記述および検証に要した時間はおよそ 5 人月である．対象の性質を検証するフォーマルなモデルは，表 1 のようになり，コメントも含めた全体で 278 行になった．

表 1 VDM++ 適用結果
**Table 1** The VDM++ model

| 項目 | 個数 | 備考 |
|---|---|---|
| 定数 | 6 | |
| 型 | 37 | ユーザ定義のもの |
| 状態変数 | 22 | |
| 操作 | 15 | 状態を変化させる機能 |
| 関数 | 2 | 状態を変化させない機能 |

検証モデルは対象システムの状態遷移について，VDM++の実行可能なモデルとして記述し，状態を変化させる手続きの事前条件および事後条件の正しさを確認した．確認にはモデルのレビューおよび特定の値に対する状態遷移を評価するツールによるアニメーション，すなわち仕様に対するテストを用いた．

### 4.2 仕様のモデル化時間

仕様をモデルとして記述するために必要な作業は問題や関係者のスキルなど複雑な要素に依存する．しかし，モデル化に共通した作業として次のような手順がある [10].

a 要求抽出

関係者とともに，すべての要求を明らかにする．

b 要求分析

要求項目間の衝突を解消し，関係を整理して適切にまとめる．

c 仕様検証

要求を明確な仕様として記述し，仕様の検証と妥当性確認を行う．

d 仕様管理

開発中のシステムをとりまく変化に適切に対応する．要求やシステム制約に起因する仕様の変更を管理する．

これらの作業は繰り返しや手戻りもある．

本研究で対象としているのは，主に (a) および (b) の段階である．

大きな時間削減効果が期待できるのは， D-Case によるドメイン知識の限定である．この事例ではフォーマルメソッドの適用対象を，開発初期の検証項目のひとつであり，姿勢制御の最もクリティカルな側面であるモード遷移解析 [13] に限定した．正しくモード遷移が行われることは，姿勢制御の大前提であるからである．VDM++ による有限状態機械としてのモデル化はこのような状態遷移の検証に適している．さらに，この VDM++ モデルにより保証される検証目的，前提，他の証拠との関係については，D-Case により明示される．

本事例では，作成した一般的な姿勢制御 D-Case に対して，

- 実験衛星なので冗長系はもたない
- 姿勢変更用のハードウェアは 3 軸ホイールと磁気トルカのみ
- ミッションの性質上，地球方向を向く制御は不要

といった固有の制約を反映し，ゴール数を 34 まで縮小することで，さらに仕様の理解に必要な時間を削減できた．

モデル化および D-Case の記述および検証作業にかかった 5 人月のうち 3 人月程度はドメイン知識の獲得であった．たとえば，文献 [4] の場合，姿勢制御の理解に必要な情報は 5 章のうち 2 章，ページ数にして 72/221 でありおよそ 1/3 程度であった．それ以外の部分は，いわば「不要であることを判断するために理解した」ので今回の目的には不要な作業で合った．仮に先に D-Case が完成し，必要な情報が限定されていたならば，およそ 2 人月程度は節約できていたことになる．もちろん，一見不要な情報がモデル化に影響することはままあるので，フォーマルメソッド適用期間を 1/3 にできるとは言い切れないが，他の文献参照についても同様の傾向であった．D-Case に検証目的や一般的な場合との違いが明示され，検証に必要な情報とその目的が関係者で確認可能かつ蓄積可能な形で共有される効果は熟練者にとっても有用である．特に，人工衛星の事例では，ミッションからの要求とソフトウェア仕様の間に，別組織によるハードウェア作成のフェーズがあるため，ソフトウェアの検証に必要なミッションの要求についての情報を明示的に共有することに意味がある．

## 5. 関連研究

D-Case とフォーマルメソッドの関係でもっとも疎なものは， evidence としてフォーマルモデルを利用する場合である．このような関係はいつでも実現可能であるが，一般的な D-Case は数理的な意味論をもたないので，証拠から goal を導出できるかどうかは保証できないからである．

D-Case の人工衛星における姿勢制御ソフトウェアへの適用は，文献 [11] がある．この先行事例では D-Case による分析とレビューを行ない，トレーサビリティの保証について成果があったことを報告している．本研究は，フォーマルメソッドの適用により，さらに厳密かつツールを利用した仕様記述が可能であることを示した．

D-Case とフォーマルメソッドのより密な結合を目指した研究として， D-Case/Agda がある [6]．D-Case/Agda は定理証明系であり，対象となる性質の保証に必要な定義を導出する．ただし，最終的な数学的証明は人間に依存するので，本研究よりも多くの数理的な知識や経験が必要となる．

## 6. 日本語仕様書へのフィードバック

本稿では，自然言語で記述された仕様書の D-Case を利用した分析手法を検討した．

ただし，表 2 の対応度の凡例は以下のとおり:

- — はモデル化の対象外
- ◎は 仕様書の情報だけで完結している
- ○は 仕様書の情報が主だが，補足情報がある
- △は 仕様書の情報が従で，補足情報が主である

本事例における小型衛星の仕様書の章構成と D-Case の対応は，表 2 のようになっていた．この衛星は姿勢制御に

表 2　仕様書と D-Case 階層の対応
**Table 2**　the map between indexes of a specification and D-Case levels

| 仕様書の節番号 | D-Case の階層 | 対応度 | 仕様書の節番号 | D-Case の階層 | 対応度 |
|---|---|---|---|---|---|
| 1 | 1 | ◎ | 5.4 | 5, 6 | ○, ○ |
| 2.1 | — | | 5.5 | — | |
| 2.2 | — | | 5.6 | — | |
| 3.1 | 5, 6 | ○, △ | 5.7 | 4 | ◎ |
| 3.2 | 5, 6 | △, △ | 5.8 | 4 | ○ |
| 3.3 | 4 | △ | 5.9 | 4 | ○ |
| 3.4 | 5, 6 | △ | 5.10 | 5 | ○ |
| 4.1 | 2, 3 | ◎, ○ | 5.11 | 5 | △ |
| 4.2 | 3, 5, 6 | △, ○, ○ | 5.12 | 5 | △ |
| 5.1 | 3 | △ | 5.13 | 6 | △ |
| 5.2 | 4, 5, 6 | △, △, △ | 5.14 | — | |
| 5.3 | 3 | ◎ | 5.15 | — | |

関する機能が少ないので，ゴール数 34 の縮小版の D-Case を用いたが，階層数は 7 で同じである．

D-Case の階層に沿った仕様書が，D-Case で表現する目的についての検証のための理解に適した仕様書と言える．しかし，この表から，仕様書の情報だけでは検証が難しいこと，また仕様書に含まれるモデル作成に必要な情報も順序や出現位置は系統的でなく，全体を理解するまでどの部分の情報が必要か判断できないことが分かる．このような D-Case と仕様書，補完情報を保持する参考書籍，他の仕様書などの対応表を管理することで，設計のための仕様書を検証のための仕様書に読み替えることが可能となる．

仕様書の全情報がフォーマルメソッドを適用した姿勢制御モデル記述に必要というわけではなく，必要部分の限定による検証時間の削減が予想されること，検証用のモデル構築に必要な情報を理解するために読むべき順序と，システム設計および実装を目的とした仕様書の章構成が一致しておらず， D-Case による順序整理が有効であることが分かる．

一方で， D-Case を利用した対応付けにより，姿勢制御に関する状態遷移を確認するために必要な情報が，仕様書あるいは仕様書以外の文献のどこにあるかを明示できる．また，複数の人工衛星の仕様書間の対応付けについて，フォーマルな検証モデルや D-Case として表現された検証のためのドメイン知識の再利用性の向上やトレーサビリティ向上が可能である．

## 7. おわりに

本研究では， SW IV&V を前提として，開発の初期段階で検証すべき目的をゴール分析した結果の D-Case に基づいて，該当する検証目的に必要な情報抽出にみに仕様書の分析作業を限定することで，短期間でフォーマルなモデルを記述する手法を提案した．

提案手法を実際の仕様書に対して適用したところ，D-Case と仕様書とのマッピングにより，検証に必要な仕様理解に必要な時間を短縮できることが確認できた．

より大規模な適用や他のドメインへの応用が今後の課題である．

**謝辞**　本研究で使用した辞書ツール開発にご協力いただいた吉村康晴氏をはじめ，九州ビジネス株式会社のみなさま， D-Case Editor をご提供いただいた DEOS プロジェクトのみなさまに感謝する．本研究の一部は， JSPS 科研費 24220001 ，基盤研究 (S) 「アーキテクチャ指向形式手法に基づく高品質ソフトウェア開発法の提案と実用化」および JAXA と九州大学の共同研究「運用シナリオのモデル開発管理及び型式検証の先端技術に関する共同研究」の成果による．

### 参考文献

[1] Eickhoff, J.: *Onboard Computers, Onboard Software and Satellite Operations*, Springer (2012).
[2] Jones, C. B.: Software Development based on Formal Methods, *Proceedings of the CRAI Workshop on Software Factories and Ada*, LNCS, Vol. 275, Springer-Verlag, pp. 153–172 (1987).
[3] Kelly, T. and Weaver, R.: The Goal Structuring Notation – A Safety Argument Notation, *Proceedings of Dependable Systems and Networks 2004 Workshop on Assurance Cases* (2004).
[4] 木田　隆，小松敬治，川口淳一郎：人工衛星と宇宙探査機，コロナ社 (2001).
[5] Larsen, P. G.，Mukherjee, P.，Plat, N.，Verhoef, M.，Fitzgerald, J.，酒匂寛 (訳)：VDM++によるオブジェクト指向システムの高品質設計と検証，翔泳社 (2010).
[6] Norell, U., Danielsson, N. A. and A., A.: The Agda Wiki, http://wiki.portal.chalmers.se/agda/pmwiki.php.
[7] Goal Structuring Notation Working Group: GSN Community Standard Version 1 , http://www.goalstructuringnotation.info/documents/GSN_Standard.pdf.
[8] Office of the Chief Engineer: NASA Space Flight Program and Project Management Requirements, Technical report, NASA Procedural Requirements, http://fpd.gsfc.nasa.gov/NPR71205/NID.pdf (2012).
[9] 坂井真一郎：”れいめい”衛星における小型衛星向け姿勢制御系開発の実例，日本航空宇宙学会誌， Vol. 56, No. 652, pp. 130–138 (2008).
[10] Sommerville, I. and Sawyer, P.: *Requirements Engineering: A Good Practice Guide*, John Wiley & Sons (1997).
[11] 田中康平，松野　裕，中坊嘉宏，白坂成功，中須賀真：アシュアランスケースにおける品質到達性と トレーサビリティを考慮した記述ルール提案と超小型衛星開発への適用評価，第 10 回クリティカルソフトウェアワークショップ (2012).
[12] DEOS プロジェクト：D-Case Editor , http://www.jst.go.jp/crest/crest-os/tech/D-CaseEditor/index.html.
[13] 経済産業省プロセス改善研究部会：ベストプラクティス調査報告　究極の高品質ソフトウェア開発プロセスをめざして　独立行政法人宇宙航空研究開発機構 (JAXA)，技術報告，独立行政法人情報処理推進機構，http://sec.ipa.go.jp/reports/20070514/JAXA.pdf (2007).