複雑システムの信頼性を向上させる開発手法

―― アーキテクチャ設計手法とモデル検査の融合 ――

加藤 淳*、浦郷 正隆、狼 嘉彰

本稿では、システムの仕様を、システムを構成する要素間による協調動作が整合している構成要素の仕様および構成要素間のインタフェース仕様に分解する開発手法を示す。本開発手法は、システム開発において既に有効性が認められているシステムエンジニアリング標準におけるアーキテクチャ設計手法およびモデル検査を、ブリッジ技術で融合して構築される。産業用ロボットの開発に対して本開発手法を適用した結果を示す。適用結果から、産業界の複雑システムに対して本開発手法が有効であることを示す。

キーワード：開発手法、システムエンジニアリング、アーキテクチャ設計手法、モデル検査、ブリッジ技術、複雑システム、信頼性

A methodology for improving reliability of complex systems

– Synthesis of architectural design method and model checking –

Atsushi Katoh*, Masataka Urago and Yoshiaki Ohkami

This paper describes a methodology for decomposing a system specification into component specifications and interface specifications whose cooperative behavior is consistent with each component. The methodology is constructed by a bridge method of combining architectural design method in systems engineering standards and model checking, which have already been confirmed to be effective in developing systems. As a trial, the methodology was applied to develop an industrial robot system. The result demonstrates that the proposed methodology is effective for complex industrial systems.

Keywords: Developing methodology, systems engineering, architectural design method, model checking, bridge method, complex systems, reliability

1　はじめに

定義された目的を実現するために、相互に作用する要素を組み合わせた集合体をシステム[用語1]という[1]。科学技術の進歩により、電子機器システム[用語2]、情報システム[用語3]などの技術システム（以降、システムという）は、社会に深く浸透している。一方で、求められる機能の高度化、目的の異なる複数のシステムが結びつき新たなシステムを形成するsystem of systems[用語4]の登場により、システムはますます複雑になっている。近年、システムの複雑化により、システムの不具合が多発している。放射線照射装置の事故[2]、アリアン５の爆発事故[3]、航空管制システムの障害[4]など、複雑システムの不具合が社会に大きな影響を与えている。複雑システムの信頼性を向上させることは、安心・安全な社会を実現する上で重要な課題である。

複雑システムではシステムの構成要素が連携しながら結びついている。例えば、５章で述べる不定形剛体運搬ロボットシステムの場合、測定サブシステムの測定結果を基に統合制御サブシステムが周辺状況を判断し、ロボットサブシステムが動作することなどである。これを本稿では構成要素の協調動作[用語5]と呼ぶ。より詳しくは、システムの機能を実現するために、システムの構成要素における処理がインタフェースを介し、他の構成要素の処理と連携することである。複雑システムにおいては、システム仕様に対して構成要素による協調動作が正しく行われる(整合[用語6]する)ことが重要である。しかし、その複雑性のために、構成要素による協調動作の仕様に誤りが混入し、協調動作が正しく行われない（不整合である）場合がある。通常、システムの構成要素による協調動作は、システム開発の終盤に実施される構成要素の実プロダクトを組み合わせたシステム試験において確認される。システム試験において協調動作の不整合を検出した場合、システム開発の上流工程に立ち戻り、構成要素による協調動作を再設計する必要がある。その際、協調動作の不整合に関する改修に多くのコストを要する。また、開発終盤における協調動作の設計変更

慶應義塾大学大学院システムデザイン・マネジメント研究科　〒223-8526 横浜市港北区日吉 4-1-1
Graduate School of System Design and Management, Keio University　4-1-1 Hiyoshi, Kohoku-ku, Yokohama 223-8526, Japan
＊ E-mail: katoh.atsushi@z7.keio.jp

Original manuscript received January 19, 2010, Revisions received June 4, 2010, Accepted June 7, 2010

により、システムの信頼性が低下する可能性がある。システムの構成要素による協調動作に対しては、システム開発の上流における確実な設計および確認が必要であるものの、これらを実現する開発手法は提案されていない。理由の一つ目は、システムにおける信頼性という観点において、システムの構成要素による協調動作が着目されることがなかったためである。理由の二つ目は、システム開発の上流において、システムの品質を作りこむこと自体が比較的新しい概念であるためである。そこで、私達は、システムの仕様を構成要素の仕様および構成要素間のインタフェース仕様に分解し、それらの協調動作が整合していることをシステム設計段階において確認する開発手法を研究している[5][6]。協調動作が整合している仕様を基にして構成要素の開発を進めることで、複雑システムにおける信頼性の向上が期待できる。本開発手法は、システムエンジニアリング[1]用語7におけるアーキテクチャ設計手法用語8およびモデル検査[7]用語9を融合することで構築される。

システムエンジニアリングとは、与えられた費用、期間内で、必要な品質を満たすシステムを実現する技術体系である。システムエンジニアリングは主に軍事、航空・宇宙分野において研究が始まり、システム開発のBest Practiceを蓄積・反映しながら発展してきた。システムエンジニアリング・プロセスは、技術分野に依存しないノウハウ・ルールとして標準化されている[8]-[10]。システムエンジニアリング・プロセスの一部に、アーキテクチャ設計手法が規定されている。アーキテクチャ設計とは、システムの構成要素に対し、システムに要求される機能・性能を配分し、構成要素の仕様および構成要素間のインタフェースを明確化する技術である。標準に規定されるプロセスに従い、アーキテクチャ設計を実施することで、複雑システムを円滑かつ確実に構成要素に分解することができる。本稿では、標準化されたアーキテクチャ設計手法を単にアーキテクチャ設計手法と呼ぶことにする。

モデル検査とは、システムの状態遷移を表すモデルに対し、モデルが実現しうるすべての状態遷移において与えられた性質が成り立つか否かを計算機により網羅的に検証する技術である。モデル検査は形式手法[11]用語10の一つに位置付けられる。モデル検査は検証技術として確立され、近年、ソフトウェア開発において普及が進んでいる。また、機能安全規格であるIEC61508[12]用語11において、開発における形式手法の適用が推奨されたことにより、システムの高信頼化を実現する技術としても注目されている。構成要素間の協調動作に関する仕様に対してモデル検査を適用し、協調動作が満たすべき性質が成り立つかどうかをしらみつぶしに確認する。それにより、複雑な状態で発生する協調動作の不整合を検出することができる。

アーキテクチャ設計手法は、システム工学に基づく、システム設計領域におけるBest Practiceを結集した体系的知識である。モデル検査は、数理論理学および計算機科学に基づいた、システム検証領域における信頼性の向上に貢献する研究成果である。この研究では、複雑システムの高信頼化を目指し、アーキテクチャ設計手法とモデル検査を融合し、それぞれの特徴を活かした開発手法を創造する。私達の研究活動は、社会・経済のニーズへ対応するために異なる分野の知識を幅広く選択し、それらを融合する第2種基礎研究に値する。

本稿では、システムの仕様を協調動作が整合している構成要素の仕様および構成要素間のインタフェース仕様に分解する開発手法を述べ、また、本開発手法の研究プロセスについても述べる。構成は次のとおりである。2章は研究目標および研究シナリオ、3章はアーキテクチャ設計手法およびモデル検査について述べる。4章はアーキテクチャ設計手法およびモデル検査の融合に関するプロセスを述べ、5章では産業事例への適用結果を述べる。6章で本開発手法の有効性および課題を考察し、7章において本稿のまとめと今後の展望を述べる。

2　研究目標および研究シナリオ

この研究の目標は、システムの仕様を協調動作が整合している構成要素の仕様および構成要素間のインタフェース仕様に分解する、特定の技術システムに特化しない開発手法の確立である。図1に研究シナリオを示す。研究目標を達成するための研究シナリオとして、システム開発に関する技術領域から、既に有効性が認められている技術を選択する。理由は、システム開発において有効性が認められている技術を活用することで、高品質な開発手法を効率良く確立することができるためである。そして、選択した技術について、それぞれの特徴を活かして融合することで、開発手法を確立する。理由は、各領域の技術を融合することで新規技術が生まれ、新たな研究領域の創出が期待でき

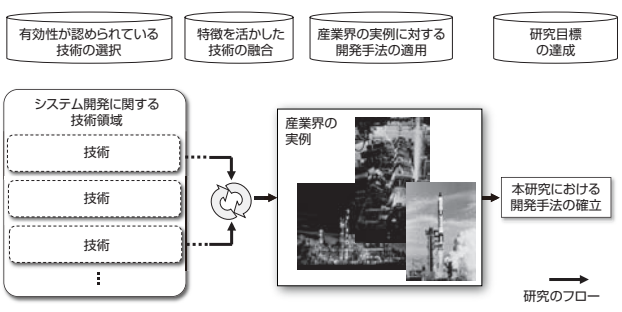


図1　研究シナリオ

るためである。また、産業界の実例に対してこの研究での開発手法を適用することで、開発手法としての有効性を確認する。適用対象として産業界の実例を選択した理由は二つある。理由の一つ目は、本開発手法が産業界において実用可能であることを確認するためには、サンプル的なケースではなく、安全性などの考慮が必要な、機能的に複雑な実例を適用対象とする必要があるためである。理由の二つ目は、産業界の実例に対する開発手法の有効性を社会に発信することにより研究活動と、研究成果の社会貢献との間のギャップ、いわゆる死の谷を越えられる可能性があるためである。

3　技術の選択

この研究における開発手法を確立するにあたり、本開発手法が満たすべき機能を次の二つに分解する。

a.システム仕様を構成要素の仕様および構成要素間のインタフェース仕様に分解する機能
b.構成要素の仕様および構成要素間のインタフェース仕様における協調動作が整合していることを検証する機能

システム開発技術の中から、機能 a. を満たすシステム設計技術を選択する。一般にシステム設計とは、ユーザーの要求を分析してシステム仕様にまとめること、さらにシステム仕様に基づいてシステムを構成する要素の機能および実現方法そして構成要素間の関係を仕様化する作業を指す。アーキテクチャ設計手法以外の代表的なシステム設計技術として、構造化分析・設計手法[13] 用語 12 がある。構造化分析・設計手法とは、実現すべきシステムのデータの流れに着目し、システムを構成要素に分解する設計手法である。構造化分析・設計手法は、要求変更や技術の進歩に影響を受けやすい機能や処理ではなく、システム環境の変化に対して安定している業務情報などのデータに着目しシステム設計を行う。それにより、保守性および拡張性を有するシステムを構築することができる。しかし、構造化分析・設計手法は情報システムを念頭において開発された手法であることから、制御の流れや処理タイミングに関する設計を考慮していない[14]。したがって、組込みシステムなど情報システム以外のシステム設計には不向きな側面を持っている。一方、アーキテクチャ設計手法は、例えば前述のデータに着目するといった特定のシステム設計を行う場合、その設計に特化した手順や作業が規定されていないため、特定の設計手法に比べ労力を要する。しかし、アーキテクチャ設計手法は、システムの機能および実現方法を明確にするプロセスが規定されている、特定の技術システムに依存しない汎用的な設計手法である。したがって、この研究の目標としている特定の技術システムに特化しない開発手法を実現するという点を踏まえ、機能 a. を満たすシステム設計技術として、アーキテクチャ設計手法を選択する。また、アーキテクチャ設計手法が規定される代表的なシステムエンジニアリング標準として、ISO 15288[9] 用語 13、ANSI/EIA 632[10] 用語 14、IEEE 1220[11] 用語 15 が挙げられる。ISO 15288 は、システムの概念検討から利用および廃棄というシステムライフサイクルプロセスの全般を適用範囲としているものの、アーキテクチャ設計における作業および手順が詳細には規定されていない。ANSI/EIA 632 は、システムの概念検討から利用移行というシステムライフサイクルプロセスの広範を適用範囲としているものの、アーキテクチャ設計における作業および手順が詳細には規定されていない。一方、IEEE 1220 は、適用範囲がシステムの要求分析・定義から試験に限定されているものの、アーキテクチャ設計における作業および手順が詳細に規定されている。したがって、本開発手法として IEEE 1220 に規定されるアーキテクチャ設計手法を選択する。

また、システム開発技術の中から、機能 b. を満たすシステム検証技術を選択する。一般にシステム検証とは、開発するシステムがシステムの仕様を満たすか否かを確認する作業を指す。モデル検査以外の代表的なシステム検証技術として、テスト手法[用語 16] およびシミュレーション手法[15] 用語 17 がある。テスト手法とは、テストケースに対する実プロダクトの動作を確認する検証手法である。実プロダクトに関する実際の動作を確認することができるものの、起こりうるすべてのケースをもれなく抽出し、それらすべてのケースにおける動作を確認することは困難である。シミュレーション手法とは、検証対象およびその周辺環境を計算機上にモデルとして模擬し、テストケースに対する検証対象モデルの動作を確認する検証手法である。実プロダクトおよび周辺環境が存在しない開発早期の段階で、検証対象の動作を確認することができるものの、テスト手法と同様に、起こりうるすべてのケースをもれなく抽出し、それらすべてのケースにおける動作を確認することは困難である。一方、モデル検査は、検証対象の状態遷移しか確認することができないものの、状態遷移については満たすべき性質が成り立つかどうかを網羅的に検証することができる。システムの状態遷移にデッドロック[用語 18] などが存在する場合、システムの運用時に致命的な事象を引き起こす可能性がある。したがって、本開発手法として、モデル検査を選択する。

次に、IEEE 1220 に規定されるアーキテクチャ設計手法およびモデル検査を詳述する。

3.1　IEEE 1220におけるアーキテクチャ設計手法

図 2 に、アーキテクチャ設計のプロセスを示す。アーキテクチャ設計は、機能設計[用語 19] と物理設計[用語 20] で構成される。機能設計とは、システム仕様として定義される機能を分割・詳細化し、分割・詳細化した機能に対しシステム仕様として定義される性能を配分する作業である。物理設計とは、システムの構成要素を具体化し、機能設計において分割・詳細化した機能・性能を構成要素に配分する作業である。アーキテクチャ設計のアウトプットは、構成要素の仕様および構成要素間のインタフェース仕様である。

図 3 に、IEEE 1220 で定められる機能設計のプロセスを示す。機能設計のプロセスは、IEEE 1220 第 6 章 3 節 Functional analysis[用語 21] で定義されている。図 4 に、IEEE 1220 で定められる物理設計のプロセスを示す。物理設計のプロセスは、IEEE 1220 第 6 章 5 節 Synthesis[用語 22] で定義されている。図 3 および図 4 の番号に従い、作業を実施することによって、円滑かつ確実に複雑システムを構成要素に分解することができる。IEEE 1220 におけるアーキテクチャ設計手法はさまざまな産業ドメインにおいて適用され、成果を上げている。そのため、一定の有効性が保証される [16]。

3.2　モデル検査

図 5 にモデル検査のプロセスを示す。モデル検査のプロセスは、モデルの作成、検査式の作成、モデル検査の実施、検査結果の分析という四つの作業に大別される。まず、適

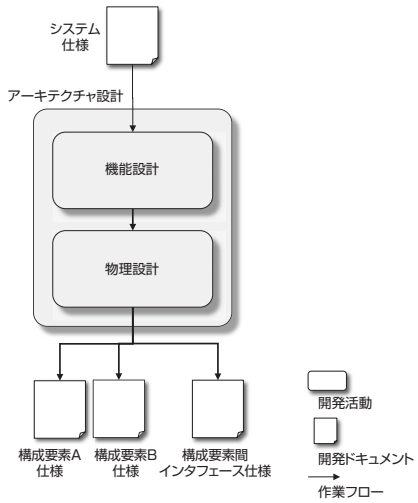


図 2　アーキテクチャ設計のプロセス

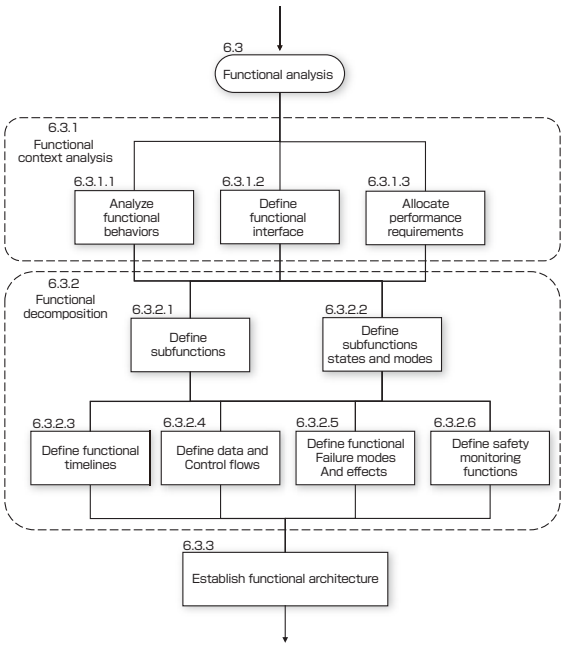


図 3　IEEE 1220 における機能設計のプロセス [11]

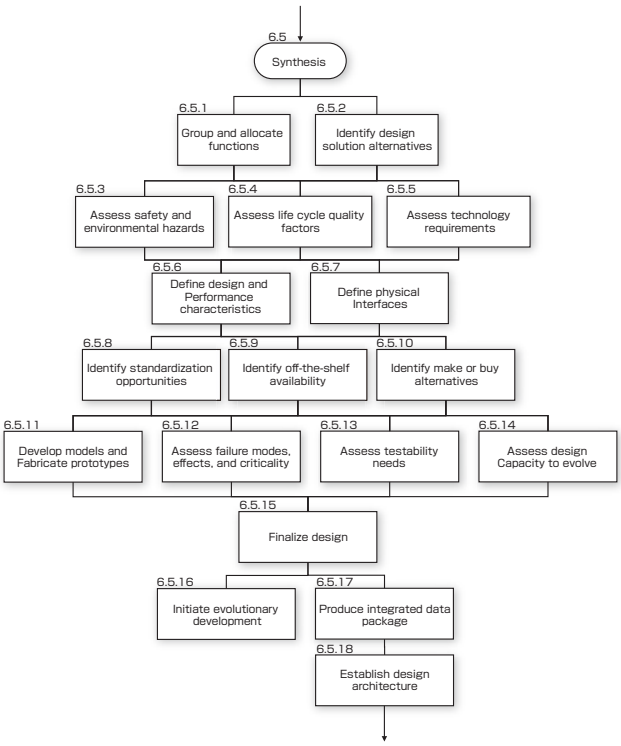


図 4　IEEE 1220 における物理設計のプロセス [11]

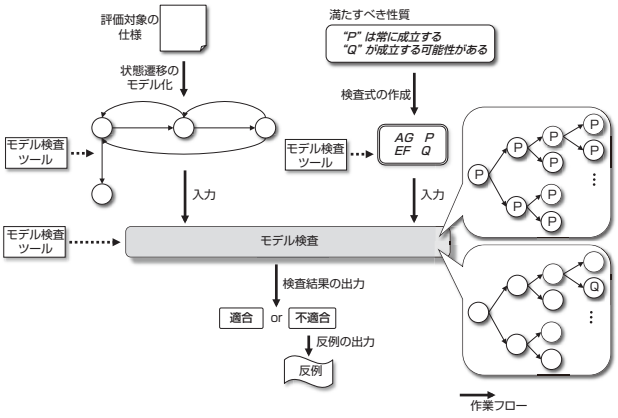


図 5　モデル検査のプロセス

用するモデル検査ツールの表現形式に従い、検査対象となるシステムの仕様などを基に、検査対象の状態遷移をモデル化する。次に、検査対象が満たすべき性質を検討する。適用するモデル検査ツールの表現形式に従い、性質を表す検査式を作成する。そして、計算機上のモデル検査ツールに対して、モデルと検査式を入力し、モデル検査を実施する。モデル検査では、モデルが実現しうるすべての状態遷移において、検査式で表現する性質をモデルが満足するか否かを網羅的に検証する。最後に、モデル検査の出力結果を基に、検査式で表現する性質とモデルとの適合結果を分析する。モデルに対して検査式が適合する場合、それはモデルの基になった仕様が性質を満足することを意味する。モデルに対して検査式が適合しない場合、反例として性質が成り立たない条件に至るまでのモデルの状態遷移が出力される。反例を分析し、モデルに誤りがない場合、モデルの基になった検査対象の仕様に誤りがあることを意味する。

モデル検査を適用することにより、次に示す二つの効果が期待できる。一つ目は、反例を用いることで、仕様の誤りを検出する際の工数を削減できる可能性がある。テスト手法やシミュレーション手法を適用し、確認内容が不適合となった場合、不適合に結びつく原因を複数仮定し、原因を分析する必要がある。その際に原因特定までに大きな労力を要することがある。モデル検査を適用する場合、自動で出力される反例を用いて不適合の発生過程を遡ることができる。それにより、不適合を引き起こす原因を効率よく特定することが可能である。二つ目は、モデル検査を実施する場合、モデル化という仕様の形式化を通して、検査対象である仕様の誤りを検出できる可能性がある。

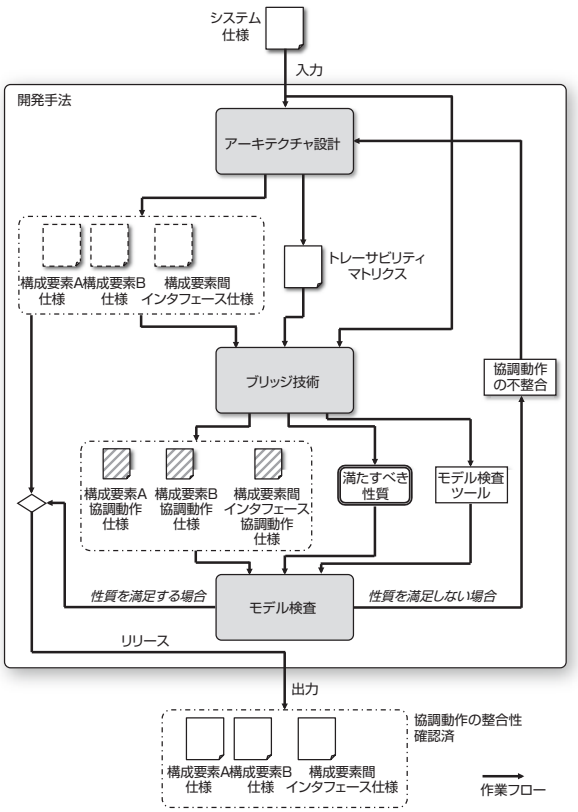


図 6　この研究における開発手法

4　技術の融合

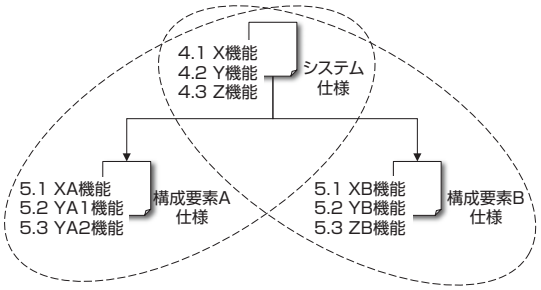
アーキテクチャ設計手法とモデル検査について、それぞれの技術の特徴を活かしながら融合し、この研究における開発手法を構築する。本章では、この研究の開発手法における作業フローを詳述することで、アーキテクチャ設計手法とモデル検査を融合するプロセスを示す。

4.1　提案する開発手法

図 6 に、アーキテクチャ設計手法とモデル検査を融合したこの研究で提案する開発手法を示す。本開発手法は、アーキテクチャ設計、モデル検査、それら二つを繋ぐブリッジ技術[用語 23]で構成される。

まず、本開発手法のインプットとして、システム仕様を入力する。システム仕様を基に、IEEE 1220 に従い、機能設計および物理設計を含むアーキテクチャ設計を実施する。アーキテクチャ設計を実施することで、システム仕様を構成要素の仕様および構成要素間のインタフェース仕様に分解する（図 6 の左上における破線の仕様）。また、アーキテクチャ設計の過程において、IEEE 1220 で規定されるトレーサビリティマトリクス[用語 24]を作成する。図 7 にトレーサビリティマトリクスを示す。システムの仕様および構成要素の仕様には、仕様の一つ一つに項番が付与される。トレーサビリティマトリクスには、システム仕様とそれをブレークダウンした構成要素仕様との対応関係がまとめられる。

次に、構成要素の仕様、構成要素間のインタフェース仕様、トレーサビリティマトリクス、システム仕様に対し、こ



システム仕様−構成要素A仕様
トレーサビリティマトリクス

システム仕様	構成要素A仕様
4.1 X機能	5.1 XA機能
4.2 Y機能	5.2 YA1機能
	5.3 YA2機能
4.3 Z機能	N/A

システム仕様−構成要素B仕様
トレーサビリティマトリクス

システム仕様	構成要素B仕様
4.1 X機能	5.1 XB機能
4.2 Y機能	5.2 YB機能
4.3 Z機能	5.3 ZB機能

図 7　トレーサビリティマトリクス

の研究で開発するブリッジ技術を適用する。ブリッジ技術を適用することで、協調動作に関連する構成要素の仕様および協調動作に関連する構成要素間のインタフェース仕様を抽出する（図 6 の左下における縞模様の仕様）。協調動作が満たすべき性質を導出する。また、開発手法において適用するモデル検査ツールを選定する。ブリッジ技術のアウトプットはモデル検査を実施する上で必要なインプットとなる。

そして、適用するモデル検査ツールの表現形式に従い、協調動作に関連する構成要素の仕様および協調動作に関連する構成要素間のインタフェース仕様をモデル化する。また、適用するモデル検査ツールの表現形式に従い、協調動作が満たすべき性質からモデル検査の検査式を作成する。適用するモデル検査ツールに対し、作成したモデルおよび検査式を入力し、モデル検査を実施する。モデル検査の結果、モデル検査ツールから反例が出力される場合、反例を分析し協調動作の不整合内容をアーキテクチャ設計にフィードバックする。フィードバックした協調動作の不整合内容を基に、再度、作業フローにしたがってアーキテクチャ設計を実施する。モデル検査の結果、モデル検査ツールから反例が出力されない場合、協調動作が整合している構成要素の仕様および構成要素間のインタフェース仕様をリリースし、本開発手法のアウトプットとして出力する。

4.2　ブリッジ技術

この研究における開発手法では、システム設計の段階で構成要素による協調動作に対しシステム検証を行う。そのためには、協調動作に着目し、アーキテクチャ設計のアウトプットおよびモデル検査のインプットをシームレスに繋ぐ必要がある。そこで、構成要素の仕様、構成要素間のインタフェース仕様、トレーサビリティマトリクス、システム仕様を基に、協調動作に関連する構成要素の仕様、協調動作に関連する構成要素間のインタフェース仕様、協調動作が満たすべき性質、適用するモデル検査ツールを導出する技術を開発している。この技術は、アーキテクチャ設計手法とモデル検査の橋渡しをすることからブリッジ技術と呼ぶ。ブリッジ技術は、協調動作に着目し IEEE 1220 といったシステムエンジニアリング標準とモデル検査を融合する具体的な方法を明確にしたところに新規性がある。図 8 に、アーキテクチャ設計とモデル検査のブリッジ技術を示す。図 8 は、図 6 の中央部分にあるブリッジ技術の詳細に相当する。ブリッジ技術のインプットは、構成要素の仕様、構成要素間のインタフェース仕様、トレーサビリティマトリクス、システム仕様である。

まず、協調動作に関連する構成要素の仕様および協調動作に関連する構成要素間のインタフェース仕様を抽出する。図 9 に、協調動作に関連する構成要素の仕様および協調動作に関連する構成要素間のインタフェース仕様を示す。協調動作に関連する構成要素の仕様を抽出する際、トレーサビリティマトリクスを用いる。トレーサビリティマトリクスにおいて、システムの仕様が複数の構成要素における仕様に対応付けられる場合、これら構成要素は協調動作することで、システムの機能を実現することになる。図 7 の場合、構成要素 A 仕様 5.1 XA 機能および構成要素 B 仕様 5.1 XB 機能は、システム仕様 4.1 X 機能を実現するために協調動作する。図 9 では、構成要素 A 仕様および構成要素 B 仕様における縞模様部分に相当する。協調動作に関連する構成要素間のインタフェース仕様については、協調動作に関連する構成要素の仕様内に存在するインタフェース情報を基に、構成要素間のインタフェース仕様から抽出する。図 9 では、中央のメッセージ部分に相当する。

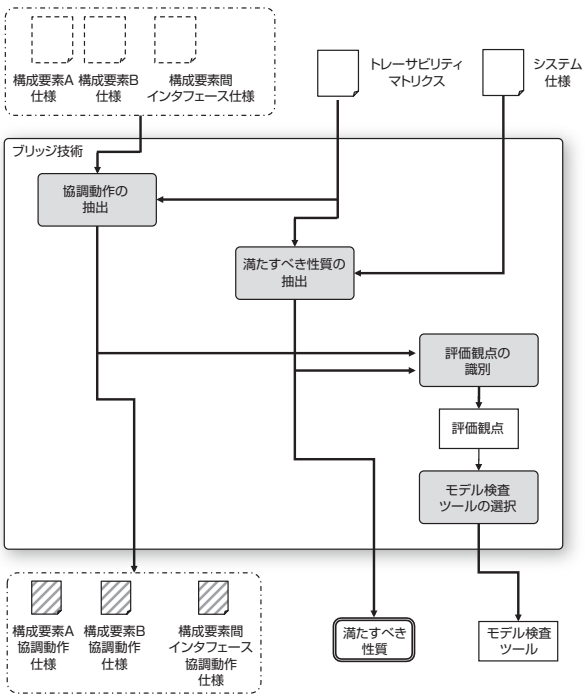


図 8　アーキテクチャ設計とモデル検査のブリッジ技術

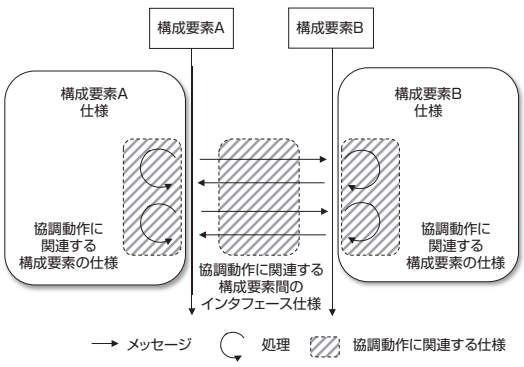


図 9　協調動作に関連する仕様

次に、トレーサビリティマトリクスを基に、システム仕様から、構成要素における協調動作が満たすべき性質を抽出する。その理由は、トレーサビリティマトリクスを用いて抽出した構成要素における協調動作は、協調動作することにより実現するシステムの仕様を満足する必要があるためである。図 7 の場合、構成要素 A 仕様 5.1 XA 機能および構成要素 B 仕様 5.1 XB 機能における協調動作は、システム仕様 4.1 X 機能の性質を満たす必要がある。

そして、抽出した協調動作に関する仕様および協調動作が満たすべき性質を基に、協調動作として評価すべき観点を設定する。協調動作に関する評価観点は、以下に大別することができる。

①構成要素間におけるメッセージおよびメッセージに関する処理に抜けや齟齬が無いこと
②構成要素間におけるメッセージおよびメッセージに関する処理のタイミングが正しいこと

識別した評価観点に応じて、適用するモデル検査ツールを選定する。モデル検査ツールの代表的な種類として、有限オートマトン [17] 用語 25 および有限オートマトンを拡張した時間オートマトン [18] 用語 26 がある。評価観点①を評価する場合、有限オートマトンに対応するモデル検査ツールを選定する。有限オートマトンの代表的なモデル検査ツールとして SPIN[19] 用語 27 がある。評価観点②のようにタイミングなどの時間的な制約を評価する場合、時間オートマトンに対応するモデル検査ツールを選定する。時間オートマトンは有限オートマトンを拡張したオートマトンである。時間オートマトンに対応するモデル検査ツールは評価観点①についても評価することができる。時間オートマトンの代表的なモデル検査ツールとして UPPAAL[20] 用語 28 がある。また、それぞれの構成要素は並行して動作する。したがって、並行システムをモデル化することが可能なモデル検査ツールを選定する必要がある。前述した SPIN および UPPAAL は並行システムをモデル化することができる。

5　産業事例への適用

近年、産業用ロボット [21] 用語 29 は、機能の高度化が進んでいる。また、産業用ロボットの多くは、対象とする作業の性格上、機械的出力が大きく、運用者に対する安全性を考慮する必要がある。産業用ロボットはこの研究における開発手法の適用に適するシステムの一つである。本稿執筆現在、私達は産業用ロボットメーカーと共同で、不定形な剛体を運搬する産業用ロボットシステムを開発している。この研究における産業事例として不定形剛体運搬ロボットシステムを選定し、本開発手法を適用する。

以下に、不定形剛体運搬ロボットシステムを説明し、本開発手法を適用した結果を述べる。

5.1　不定形剛体運搬ロボットシステム

不定形剛体運搬ロボットシステムとは、形状・寸法が一定ではない重量のある剛体の把持、運搬、据置を行う産業用ロボットシステムである。不定形剛体運搬ロボットシステムに関する開発要求の特徴は、不定形剛体の把持および据置作業に対し、次に示す強い自律性が求められる点である。不定形剛体を把持する領域は限定されているものの、剛体の形状・寸法、剛体を把持する位置、剛体の姿勢は不定である。システムは、剛体を把持するにあたり、剛体の形状・寸法、位置、姿勢を正確に判断する必要がある。また、不定形剛体を据え置く領域は限定されているものの、その領域の中で剛体を据え置く位置は不定である。システムは、剛体を据え置くにあたり、剛体が存在しない位置、もしくは剛体が敷き詰められた領域において、最も標高が低い位置を正確に判断する必要がある。

不定形剛体運搬ロボットシステムを開発するにあたり、まず、開発要求を基にシステムの要求分析を実施した。システムの要求分析を実施することでシステム仕様を確定した。確定したシステム仕様を入力として本開発手法を適用した。

5.2　この研究における開発手法の適用

本節では、4 章で述べたアーキテクチャ設計、ブリッジ技術、モデル検査の技術ごとに、この研究における開発手法の具体的な適用内容を述べる。

5.2.1　アーキテクチャ設計

不定形剛体運搬ロボットシステムのシステム仕様を入力として、アーキテクチャ設計を実施した。アーキテクチャ設計の結果、不定形剛体運搬ロボットシステムの仕様を測定サブシステム、ロボットサブシステム、統合制御サブシステムの仕様およびサブシステム用語 30 間のインタフェース仕様に分解した。図 10 に不定形剛体運搬ロボットシステムのアーキテクチャ設計結果を示す。アーキテクチャ設計の際、COTS（Commercial Off The Shelf）用語 31 製品および既存の技術を最大限に活用できるように、あらかじめサブシステムを構成するコンポーネント用語 32 を想定し、各サブシステムを設計した。

測定サブシステムは、3 次元形状を測定するレーザスキャナおよびレーザスキャナ上下動機構、それら二つを制御する測定制御計算機で構成される。測定サブシステムは、剛体を把持する際の剛体の形状・寸法、位置、姿勢を測定する。また、剛体を据え置く際の据置領域の凹凸状況を測定する。

ロボットサブシステムは、ロボットアームおよびロボットハンド、それらを制御するコントローラで構成される。また、運用者がロボットアームに関する動作のプログラミングや、ロボットアームの緊急停止を行う際に用いるティーチペンダント[用語33]を有する。ロボットサブシステムは不定形剛体の把持、運搬、据置を行う。

統合制御サブシステムは、測定サブシステムおよびロボットサブシステムを制御する統合制御計算機、運用者が作業の指示やシステムのステータス確認を行う際に使用するコンソール[用語34]で構成される。統合制御サブシステムは、測定サブシステムからの計測結果を基に、ロボットサブシステムを制御する。

また、アーキテクチャ設計を実施する際、測定サブシステム、ロボットサブシステム、統合制御サブシステムに関するトレーサビリティマトリクスを作成した。

5.2.2　ブリッジ技術

不定形剛体運搬ロボットシステムにおける各サブシステム仕様、サブシステム間のインタフェース仕様、トレーサビリティマトリクス、システム仕様に対し、ブリッジ技術を適用した。ここでは測定サブシステムおよび統合制御サブシステムを取り上げ、ブリッジ技術の具体的な適用内容を述べる。

まず、トレーサビリティマトリクスを基に、各サブシステム仕様およびサブシステム間インタフェース仕様から協調動作に関連する仕様を抽出した。協調動作に関連する仕様の具体的な抽出方法は 4.2 節に示す方法に従う。測定サブシステム仕様においては仕様 39 項目から 6 項目、統合制御サブシステム仕様においては仕様 78 項目から 6 項目を抽出した。また、測定サブシステムおよび統合制御サブシステム間のインタフェース仕様においては仕様 26 項目から 22 項目を抽出した。

次に、トレーサビリティマトリクスおよび不定形剛体運搬ロボットシステムのシステム仕様を基に、サブシステムの協調動作が満たすべき性質を抽出した。協調動作が満たすべき性質の具体的な抽出方法は 4.2 節に示す方法に従う。測定サブシステムおよび統合制御サブシステムにおいては、協調動作が満たすべき性質として 23 項目を抽出した。表 1 に抽出した 23 項目の性質の中から 2 項目を示す。

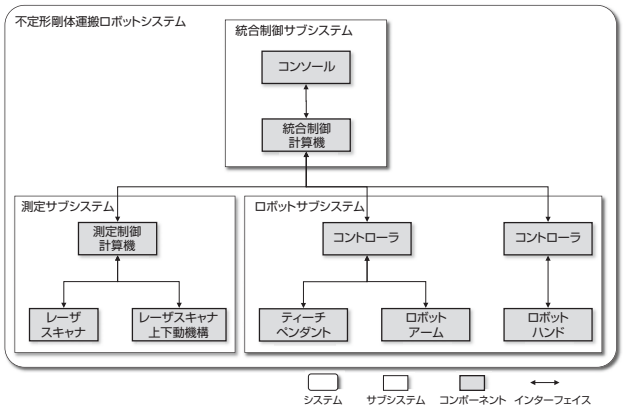


図 10　不定形剛体運搬ロボットシステムのアーキテクチャ設計結果

表 1　測定サブシステムおよび統合制御サブシステムの協調動作が満たすべき性質（23 項目のうちの 2 項目）

No.	性質
1-5	統合制御サブシステムは、測定サブシステムに対する剛体測定要求に対し、測定サブシステムから剛体測定結果もしくは剛体測定失敗の応答を受信すること。
3-3	測定サブシステムは、統合制御サブシステムによる測定停止要求の送信後100 ms 以内に測定処理を停止すること。

そして、抽出した協調動作に関する仕様および協調動作が満たすべき性質を基に、協調動作として評価すべき観点を設定した。測定サブシステムおよび統合制御サブシステムの協調動作においては、メッセージおよび処理の抜けや齟齬が懸念された。100 ms 以内などの時間制約やメッセージおよび処理のタイミングに関する仕様が存在した。したがって、4.2 節で示す①および②の二つを評価観点として設定した。

また、識別した評価観点を基に、適用するモデル検査ツールを選定した。測定サブシステムおよび統合制御サブシステムの協調動作については、時間的な制約やメッセージおよび処理のタイミングを評価する必要がある。また、測定サブシステムおよび統合制御サブシステムは並行して動作するため、並行システムのモデル化が可能なモデル検査ツールを選定する必要がある。したがって、本適用においては、これらの条件を満足するモデル検査ツールとして UPPAAL を選定した。

5.2.3　モデル検査

前項と同様に、本項においても、測定サブシステムおよび統合制御サブシステムを取り上げ、モデル検査の具体的な適用内容を述べる。

まず、モデル検査ツールの表現形式に従い、ブリッジ技術において抽出した協調動作に関連する仕様をモデル化した。図 11 から図 13 に、UPPAAL を用いて測定サブシステムおよび統合制御サブシステムの協調動作に関連する仕様をモデル化した結果を示す。作成したモデルは、協調動作に関連する仕様において、測定サブシステムに対応するモデル（図 11）、統合制御サブシステムに対応するモデル（図 12）、測定サブシステムおよび統合制御サブシステムのインタフェースに対応するモデル（図 13）の 3 モデルで構成される。

表 2　測定サブシステムおよび統合制御サブシステムにおける協調動作の検査式（23 ケースのうちの 2 ケース）

No.	検査式
1-5	A[] (bing_syscont_sendreq == bing_req_rod) imply (bing_syscont_req == bing_req_rod)
3-3	A[] P_BING_SCAN.BING_SCAN_REQ_STOP imply (bing_stopreq_time <= 10)

次に、協調動作が満たすべき性質を基に、モデル検査時の検査式を作成した。測定サブシステムおよび統合制御サブシステムにおいては、UPPAAL の表現形式に従い、協調動作が満たすべき性質を基に 23 ケースの検査式を作成した。表 2 に、表 1 に示す 2 項目に対応する検査式を示す。

そして、作成したモデルおよび検査式を基にモデル検査を実施した。測定サブシステムおよび統合制御サブシステムにおいては、図 11 から図 13 に示す協調動作モデルおよび表 2 の 2 ケースを含む 23 ケースの検査式を基に UPPAAL によるモデル検査を実施した。モデル検査の実施後、検査結果を分析した。測定サブシステムおよび統合制御サブシステムにおいては、検査式のいくつかについてモデルに対して検査式が適合しない結果となった。UPPAAL により出力された反例を分析したところ、協調動作として表 2 に示す検査式の結果を含む 6 ケースの不整合を検出した。表 3 に、表 2 の 2 ケースに対応するモデル検査結果を示す。

その後、アーキテクチャ設計に対し、協調動作に関する不整合内容をフィードバックした。測定サブシステムおよび統合制御サブシステムにおいては、表 3 の検査結果を含む 6 ケースの協調動作における不整合を基に、再度、アーキテクチャ設計を実施した。アーキテクチャ設計の結果、協調動作の不整合が解決された測定サブシステム仕様、統合制御サブシステム仕様、測定サブシステムおよび統合制御サブシステム間のインタフェース仕様を作成した。

表 3　測定サブシステムおよび統合制御サブシステムの協調動作に関するモデル検査結果（6 ケースのうちの 2 ケース）

No.	モデル検査結果
1-5	メッセージおよび処理のタイミングによっては、n 回目に発行された剛体測定要求に対し、n 回目より前に発行された剛体測定要求、もしくは据置領域測定要求、もしくは測定停止要求に対する応答が返却される場合がある。
3-3	測定停止要求に対し、100 ms 以内に測定処理が停止しない場合がある。また、測定停止要求に対し、測定サブシステム自体が停止する場合がある。

5.3　適用結果

不定形剛体運搬ロボットシステムのシステム仕様に対し、この研究における開発手法を適用した。その結果、不定形剛体運搬ロボットシステムのシステム仕様から、測定サブシステム仕様、統合制御サブシステム仕様、ロボットサブシステム仕様および各サブシステム間のインタフェース仕様を見出した。また、これらの仕様を固める前に、測定サブシステムおよび統合制御サブシステムから、表 3 に示す協調動作の不整合を検出することができた。その後、協調動作が整合している構成要素の仕様および構成要素間のインタフェース仕様を作成することができた。したがって、産業用ロボットにおける実際の製品開発に本開発手法を適用し、産業用ロボットの高信頼化に貢献することができた。それにより、この研究が第 2 種基礎研究に値することを再確認した。本稿執筆時点において、不定形剛体運搬ロボットシステムは、産業界のニーズに応えるべく、協調動作が

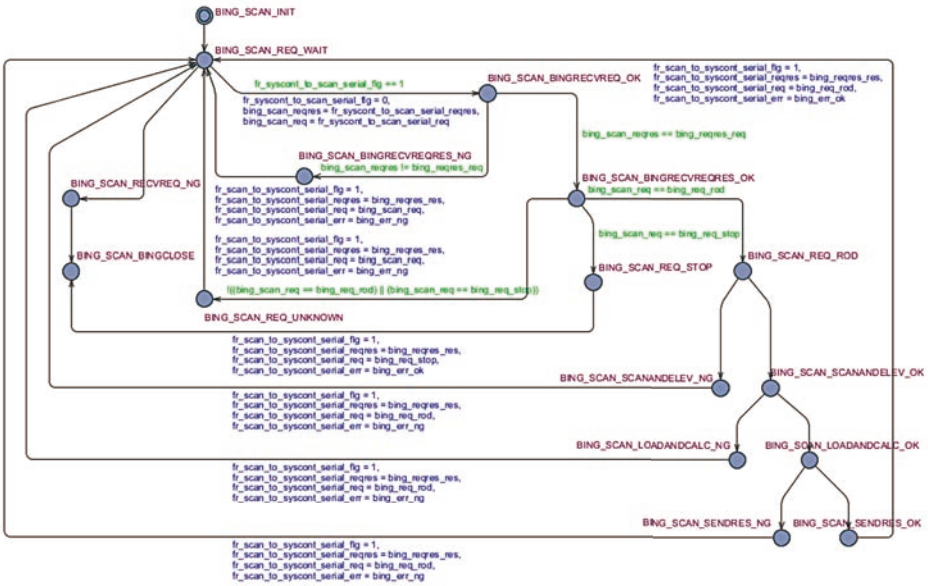


図 11　測定サブシステムにおける協調動作モデル

確認された各仕様を基に開発が進められている。

6　考察

本章では、この研究における開発手法の有効性を示す。また、本開発手法の課題を述べる。

6.1　開発手法の有効性

この研究における開発手法について、5 章で示した産業事例への適用結果を基に有効性を示す。有効性を示すにあたり、適用事例における QCD（Quality：開発対象の品質、Cost：開発コスト、Delivery：開発期間）[用語 35] に着目する。

まず、開発対象の品質という観点から、本開発手法を考察する。今回、不定形剛体運搬ロボットシステムという機能的に複雑な産業事例に対し、この研究における開発手法を適用した。その結果、アーキテクチャ設計に対して、測定サブシステムおよび統合制御サブシステムにおける協調動作の不整合をフィードバックし、1 回の反復で協調動作が整合している測定サブシステム仕様、統合制御サブシステムを作成することができた。特に、表 3 に示す協調動作の不整合については、満たすべき性質が成り立つかどうかをしらみつぶしに確認するモデル検査ならではの、人手では検出することが困難な不具合といえる。そのような不具合を開発の初期段階で検出できたことは、本開発手法の有効性の現れである。

表 4　産業事例に対する本開発手法適用時の工数

作業	工数（人時間）
アーキテクチャ設計	189
ブリッジ技術	5※
モデル検査	12※

※測定サブシステムおよび統合制御サブシステムにおける実績値

次に、開発コストおよび開発期間という観点から、本開発手法を考察する。表 4 に産業事例に対して本開発手法を適用した際の工数を示す。ブリッジ技術およびモデル検査については、5 章で述べた測定サブシステムおよび統合制御サブシステムにおける実績値を示す。アーキテクチャ設計については、189 人時間の工数を要した。システム開発において、アーキテクチャ設計を含むシステムエンジニアリング手法を適用する効果については、多くの先行研究が存在する [22]-[25]。これらの先行研究において、システムエンジニアリング手法を適切に適用することで、システム開発における開発コストおよび開発期間の短縮が可能であることが示されている。本稿においては、今回の産業事例に対し、アーキテクチャ設計手法を適用したことによる開発コストおよび開発期間に関する短縮の程度を示すことはできない。しかし、アーキテクチャ設計において、確実な成果が得られたことを考慮した場合、システム開発における開発コストおよび開発期間を短縮できた可能性が高い。ブリッジ技術

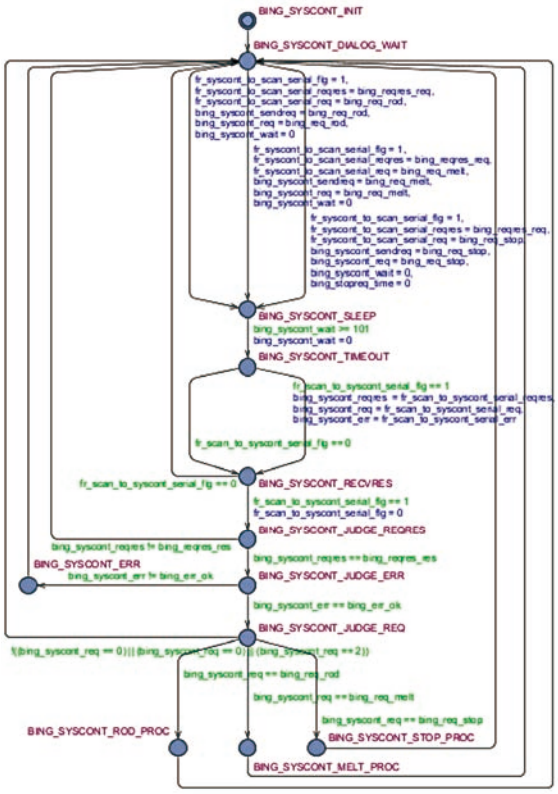


図 12　統合制御サブシステムにおける協調動作モデル

およびモデル検査については、表 4 に示すとおり、測定サブシステムおよび統合制御サブシステムにおいて、それぞれ 5 人時間および 12 人時間の工数を要した。ここで、ブリッジ技術を介し、モデル検査を実施することで検出した協調動作の不整合について考察する。サブシステム間における協調動作の不整合について、通常、それらを検出することが可能な開発フェーズは、システム開発の終盤に実施されるサブシステム同士を組み合わせたシステム試験である。システム試験においてサブシステム間における協調動作の不整合を検出した場合、その改修に多くの改修コストおよび改修期間を要する。文献 [26] の中で Boehm は、要求仕様が確定した段階で要求の誤りを検出し修正する際のコストを 1 とした場合、試験において検出された場合には小規模システムで 2、大規模システムで 20 ものコストを要すると分析している。表 3 に示す協調動作の不整合は、要求仕様を確定する段階において、モデル検査を適用しない限り見逃される可能性が高い不具合といえる。システム開発全体で考えた場合、ブリッジ技術およびモデル検査に要する工数は費用対効果の高い工数であると考える。

6.2　開発手法の適用性

この研究における開発手法は、特定の技術システムに特化したものではなく、技術システム全般の開発に適用することが可能な手法である。それは、システムを設計する際にシステム仕様を基にシステムを構成する要素を見出して構成要素を正しく協調動作させることでシステムの機能を実現することは、技術システム全般において共通的な概念であるためである。また、本開発手法は、適用対象システムにおける協調動作に関する固有の問題に対処することが可能な手法である。その理由は、協調動作の特徴に応じて評価観点およびモデル検査ツールを選択するブリッジ技術を有するためである。

ただし、本開発手法の適用には注意すべき点がある。本開発手法では、構成要素の協調動作が整合していることを確認するために、モデル検査を採用している。モデル検査は、モデルが実現しうるすべての状態遷移において、与えられた性質が成り立つか否かを計算機により網羅的に検証する技術である。モデル検査におけるモデルの状態数が多い場合、状態の組み合わせが膨大になることでモデル検査が終了しない状態爆発という現象が生じることがある。つまり、アーキテクチャ設計の結果、構成要素の協調動作が極端に複雑になると、モデル検査による協調動作の検証が終了しない可能性がある。その場合、構成要素による協調動作が極端に複雑にならないように、再度、アーキテクチャ設計を実施する方策や協調動作に関する仕様のモデルの状態数を減らす方策などが必要になる。

6.3　課題

この研究における開発手法について、産業事例に対する有効性を確認することができた。しかし、本開発手法には今後に解決すべきいくつかの課題がある。

課題の一つ目は本開発手法におけるブリッジ技術の課題である。ブリッジ技術においては協調動作が満たすべき性質を抽出する。一般に、モデル検査の対象が満たすべき性質には活性（liveness）および安全性（safety）が存在する [27]。活性は“検査対象がいつかは良い状態に到達する”という性質である。安全性は“検査対象が決して悪い状態に到達しない”という性質である。検査対象の活性に関する性質については、検査対象が実現すべき仕様そのものである場合が多い。そのため、検査対象の活性は検査対象の仕様もしくは検査対象仕様の基となった仕様から抽出することができる。しかし、検査対象の安全性に関する性質について、その多くは検査対象仕様および検査対象仕様の基になった仕様中に規定されることはない。したがって、検査対象における安全性の抽出については、本開発手法を適用するエンジニアの経験およびスキルに依存する可能性が高い。特に、安全性に関する協調動作が満たすべき性質の抽出については、協調動作の複雑性のため、適用するエンジニアに依存する傾向が強い可能性がある。

課題の二つ目は本開発手法におけるモデル検査の課題である。モデル検査においては、構成要素の仕様および構成要素間のインタフェース仕様から抽出した協調動作に関

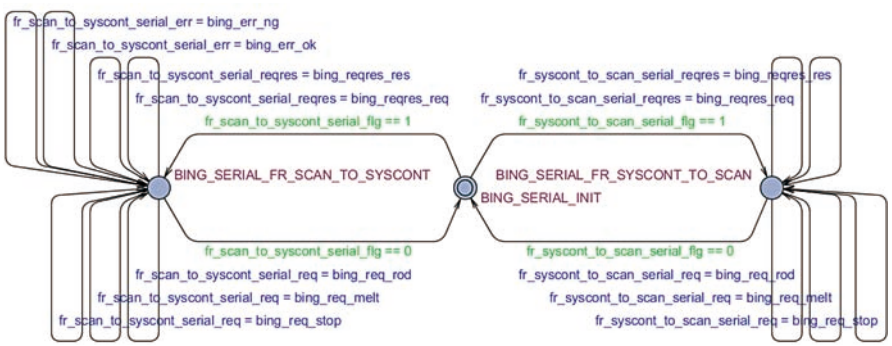


図 13　測定サブシステムおよび統合制御サブシステム間インタフェース仕様における協調動作モデル

する仕様を基にモデルを作成する。構成要素の仕様およびインタフェース仕様は自然言語で記述される場合が多い。そのため、抽出した仕様を人手でモデル化する場合が多く、モデルに誤りが混入しやすい。本課題はモデル検査全般における課題でもある。

課題の三つ目も本開発手法におけるモデル検査の課題である。モデル検査においては、モデル検査ツールの表現形式に従い、協調動作が満たすべき性質を基に検査式を作成する。モデル検査における検査式は、時相論理[用語 36]と呼ばれる時相演算子[用語 37]（G：常に、F：いずれ）およびパス限量子[用語 38]（A：すべてのパスで、E：あるパスで）、論理演算子[用語 39]（OR、AND、NOT）を組み合わせて表現される。しかし、時相論理を扱うには専門的な知識が必要であることから、協調動作が満たすべき性質を基に検査式を作成するハードルは高い。本課題についてもモデル検査全般における課題である。

7　まとめと今後の展望

本稿では、システムの仕様を、システムを構成する要素間による協調動作が整合している構成要素の仕様および構成要素間のインタフェース仕様に分解する開発手法を示した。アーキテクチャ設計手法およびモデル検査を融合するブリッジ技術を見出し、ブリッジ技術の実現例を示した。また、産業用ロボットに対する本開発手法の適用結果を示した。適用結果から、産業界の複雑システムに対し本開発手法が有効であることを示した。本開発手法については、筆者らが所属する慶應義塾大学大学院システムデザイン・マネジメント研究科[28]（以降、慶應 SDM 研究科という）をとおして、産業界に広めることを検討している。慶應 SDM 研究科は文理を問わずさまざまなフィールドの社会人が多く学ぶ研究科である。慶應 SDM 研究科には、航空・宇宙、情報システム、ロボット、電子機器などの製品開発の第一線で活躍するエンジニアが在籍している。慶應 SDM 研究科において、それらのエンジニアに対して本開発手法を展開することで、本開発手法を広めて産業界にて貢献することができると考える。ただし、本開発手法には改善の余地もある。今後、本稿で挙げた課題のうち、課題 1 の解決に取り組む。そして、さらに品質の高い開発手法の実現を目指す。

用語説明

用語 1:　システム：定義された目的を実現するために、相互に作用する要素を組み合わせた集合体。

用語 2:　電子機器システム：取り扱う情報をデジタル処理する複数のプロセッサが搭載されたシステム（例：携帯電話など）。

用語 3:　情報システム：企業活動においてデータ処理を行う複数の計算機がネットワークで接続されたシステム（例：業務システムなど）。

用語 4：system of systems：目的の異なる複数のシステムが結びついた複合システム。

用語 5：協調動作：システムの構成要素における処理がインタフェースを介し、他構成要素の処理と連携すること。

用語 6：協調動作の整合：システム仕様に対し構成要素による協調動作が正しく行われること。

用語 7：システムエンジニアリング：与えられた費用、期間内で、必要な品質を満たすシステムを実現する技術体系。技術分野に依存しないノウハウ・ルールとして標準化されている。

用語 8：アーキテクチャ設計手法：システムの構成要素に対し、システムに要求される機能・性能を配分し、構成要素の仕様および構成要素間のインタフェースを明確化する設計技術。

用語 9：モデル検査：システムの状態遷移を表すモデルに対し、モデルが実現しうるすべての状態遷移において、与えられた性質が成り立つか否かを、計算機により網羅的に確認する検証技術。

用語 10：形式手法：数理論理学に基づく言語を用いて仕様を表現し、仕様が正しいことを保証する開発および検証技術。

用語 11：IEC 61508：プロセス産業、機械製造業、交通輸送、医療機器などの機能安全を電気／電子／プログラム可能な電子系をもって構成する場合の遵守事項を定めた国際規格。

用語 12：構造化分析・設計手法：実現すべきシステムのデータの流れに着目し、システムを構成要素に分解する設計技術。

用語 13：ISO 15288：システムエンジニアリング標準のひとつ。システムの概念検討から利用および廃棄という、システムライフサイクルプロセスの全般を適用範囲とし、各プロセスにおける作業および手順が規定されている。

用語 14：ANSI/EIA 632：システムエンジニアリング標準の一つ。システムの概念検討から利用移行というシステムライフサイクルプロセスの広範を適用範囲とし、各プロセスにおける作業および手順が規定されている。

用語 15：IEEE 1220：システムエンジニアリング標準の一つ。システムの要求分析・定義から試験というシステムライフサイクルプロセスを適用範囲とし、各プロセスにおける作業および手順が規定されている。

用語 16：テスト手法：実プロダクトを用いて、テストケースに対する実プロダクトの動作を確認する検証技術。

用語 17：シミュレーション手法：検証対象およびその周辺環境を計算機上にモデルとして模擬し、テストケースに対する検証対象モデルの動作を確認する検証技術。

用語 18：デッドロック：二つ以上の処理単位が互いの処理終了を待ち、結果としてどの処理も先に進めなくなってしまう状態。

用語 19：機能設計：システム仕様として定義される機能を分割・詳細化し、分割・詳細化した機能に対し、システム仕様として定義される性能を配分する作業。

用語 20：物理設計：システムの構成要素を具体化し、機能設計において分割・詳細化した機能・性能を、構成要素に配分する作業。

用語 21：Functional analysis：IEEE 1220 第 6 章 3 節に規定される機能設計に該当するプロセス。

用語 22：Synthesis：IEEE 1220 第 6 章 5 節に規定される物理設計に該当するプロセス。

用語 23：ブリッジ技術：本論文で明らかにした、アーキテクチャ設計とモデル検査の間をシームレスに繋ぐ技術。

用語 24：トレーサビリティマトリクス：上位仕様と下位仕様の対応関係をまとめた表。

用語 25：有限オートマトン：有限個の状態、遷移、動作の組み合わせからなる振る舞いのモデル。

用語 26：時間オートマトン：有限オートマトンに時間変数を導入した振る舞いのモデル。遷移条件として時間経過をモデル化することができる。

用語 27：SPIN：有限オートマトンベースのモデル検査ツール。PROMELA（Process Meta Language）というＣ言語に似た言語を用いて、システムの状態遷移をモデル化する。次よりダウンロードが可能。<http://spinroot.com/>

用語 28：UPPAAL:時間オートマトンに対応したモデル検査ツール。GUI（Graphical User Interface）を用いて、直感的に、システムの状態遷移をモデル化することができる。次よりダウンロードが可能。<http://www.uppaal.com/>

用語 29：産業用ロボット：自動制御によるマニピュレーション機能または移動機能を持ち、各種の作業をプログラムによって実行できる、産業に使用可能な機械。

用語 30：サブシステム：システムの一部であるものの、それ自体が局所的な一つのシステムとしての構造を持つもの。

用語 31：COTS：Commercial Off The Shelf の略。ソフトウェア製品やハードウェア製品などの既製品。

用語 32：コンポーネント：サブシステムを構成する要素または部品。

用語 33：ティーチペンダント：産業用ロボットに対する動作のプログラミングや産業用ロボットの緊急停止に用いる装置。

用語 34：コンソール：システムを操作する際に利用する入出力装置。キーボードなどの入力装置、ディスプレイなどの表示装置で構成される。

用語 35：QCD:Quality（開発対象の品質）、Cost（開発コスト）、Delivery（開発期間）の略。

用語 36：時相論理：時間との関連で問題を理解し表現するための規則と表記法の体系。時相演算子、パス限量子、論理演算子を組み合わせ、“常に P が成立する”、“いずれ Q が成立する”などの性質を表現することができる。

用語 37：時相演算子：時相論理における“G：常に”や、“F：いずれ”などを表現する演算子。

用語 38：パス限量子:時相論理における“A:すべてのパスで”や、“E：あるパス”を表現する演算子。

用語 39：論理演算子：論理演算を表す記号。“NOT：否定”、“AND：論理積”、“OR：論理和”がある。

参考文献

[1] International council on systems engineering (INCOSE): *INCOSE Systems Engineering Handbook version 3.1*, 1.5 of 6, INCOSE, USA (2007).

[2] N. G. Leveson: *SAFEWARE: System Safety and Computers*, 515-553, Addison-Wesley Professional, USA (1995).

[3] 清水久二: アリアン5の爆発事故とソフトウェア安全性に関する国際規格, *安全工学会安全工学誌*, 41 (1), 39-42 (2002).

[4] 国土交通省: FDPシステムの障害の原因調整の結果, 国土交通省(オンライン), 入手先< http://www.mlit.go.jp/kisha/kisha03/12/120312_.html>(参照2009-09-22).

[5] 加藤淳, 神武直彦, 春山真一郎, 狼嘉彰: モデル検査を用いて組込みシステムにおけるソフトウェアとハードウェアの協調動作に関する要求仕様の不整合を検出する手法, *IPSJ Symposium Series*, 2009, 65-70 (2009).

[6] 加藤淳, 狼嘉彰: FPGAとソフトウェアにおける協調動作の整合性に関する評価手法の提案, *情報処理学会第163回ソフトウェア工学研究会研究報告*, 2009 (31), 105-112 (2009).

[7] E. M. Clarke, O. Grumberg and D. E. Long: Model checking and abstraction, *ACM Transactions on Programming Languages and Systems*, 16 (5), 1512-1542 (1994).

[8] Institute of Electrical and Electronics Engineers (IEEE): *IEEE standard for system and software engineering - System life cycle processes*, IEEE 15288-2008 (2008).

[9] American National Standard Institute (ANSI)/ Electronic Industries Alliance (EIA): *ANSI/EIA Standard for Process for Engineering a System*, ANSI/ EIA 632-1999 (1999).

[10] Institute of Electrical and Electronics Engineers (IEEE): *IEEE standard for application and management of the systems engineering process*, IEEE 1220-2005 (2005).

[11] E. M. Clarke and J. M. Wing: Formal methods: State of the art and future directions, *ACM Computing Surveys*, 28 (4), 626-643 (1996).

[12] International Electrotechnical Commission (IEC): *IEC standard for functional safety of electrical/electronic/ programmable electronic safety-related systems,* IEC 61508-SER Ed. 1.0 (2005).

[13] T. DeMarco: *Structured Analysis and System Specification,* Yourdon Press, USA (1978).

[14] 有澤誠, 斉藤鉄也: *モデルシミュレーション技法*, 16-17, 共立出版 (1997).

[15] V. K. Rompaey, D. Verkest, I. Bolsens and D. H. Man: CoWare – A design environment for heterogeneous hardware/software systems, *Proceedings of the European Design Automation Conference*, 252-257 (1996).

[16] T. Doran: IEEE 1220: For practical systems engineering, *IEEE Magazines Computer*, 39 (5), 92-94 (2006).
[17] M. Sipser: Course Technology Ptr(SD), *Introduction to the Theory of Computation*, 29-90, The Netherlands (1996).
[18] R. Alur and D. L. Dill: A theory of timed automata, *Theoretical Computer Science*, 126 (2), 183-235 (1994).
[19] G. J. Holzmann: The model checker SPIN, *IEEE Transaction on Software Engineering*, 23 (5), 279-295 (1997).
[20] K. G. Larsen, P. Pettersson and W. Yi: UPPAAL in a nutshell, *International Journal on Software Tools for Technology Transfer*, 1 (1-2), 134-152 (1997).
[21] 日本工業標準調査会: *産業用マニピュレーティングロボット−用語*, JIS B 0134 (2008).
[22] J. P. Elm: A study of systems engineering effectiveness – Initial results, *Proceedings of the Systems Conference 2008 2nd Annual IEEE*, 1-7 (2008).
[23] B. Boehm, R. Valerdi and E. Honour: The ROI of systems engineering: Some quantitative results for software-intensive systems, *Systems Engineering*, 11 (3), 221-234 (2008).
[24] E. C. Honour: Understanding the value of systems engineering, *Proceedings of the INCOSE International Symposium*, 1-16 (2004).
[25] A. K. Kludze: The impact of systems engineering on complex systems, *Proceedings of Conference on Systems Engineering Research* (2004).
[26] B. W. Boehm: *Software engineering economics*, 38-40, Prentice-Hall, USA (1981).
[27] B. Alpern and F. B. Schneider: Defining liveness, *Information Processing Letters*, 21, 181-185 (1985).
[28] 慶應義塾大学大学院システムデザイン・マネジメント研究科: 研究科ホームページ, 慶應義塾大学大学院システムデザイン・マネジメント研究科(オンライン), 入手先<http://www.sdm.keio.ac.jp/>(参照2010-04-18).

執筆者略歴

加藤 淳（かとう あつし）

2000 年熊本大学大学院自然科学研究科電気システム専攻修了。同年より、電機メーカーにおいて、ユビキタス環境における小型ネットワークデバイスの研究・開発業務などに従事。2006 年より、宇宙開発領域において宇宙機ソフトウェアの独立評価業務に従事。また、慶應義塾大学大学院システムデザイン・マネジメント研究科において、システムエンジニアリングに関する研究に取り組む。2009 年情報処理学会研究賞受賞。情報処理学会会員。本論文では、研究計画、技術の選択・融合、産業事例への適用、考察に関する部分を担当した。

浦郷 正隆（うらごう まさたか）

1998 年東京工業大学大学院理工学研究科機械物理工学専攻博士課程修了。博士（工学）。同年より、東京工業大学で助手を務める。2008 年より、慶應義塾大学大学院システムデザイン・マネジメント研究科准教授となる。システムズエンジニアリングおよび工学問題のコンピュータモデリング、数値計算に関する研究に取り組む。日本機械学会会員。INCOSE 会員。本論文では、技術の選択・融合に関する部分を担当した。

狼 嘉彰（おおかみ よしあき）

1968 年東京工業大学大学院理工学研究科電気工学専攻修了。工学博士。NASA 国際フェロー、東京工業大学教授、慶應義塾大学教授、宇宙開発事業団技術研究本部研究総監を経て、2008 年より慶應義塾大学大学院システムデザイン・マネジメント研究科委員長を務める。複雑システムのダイナミクス・制御および戦略的システムエンジニアリングに関する研究に取り組む。日本機械学会フェロー。INCOSE フェロー。日本航空宇宙学会会員。IEEE 会員。AIAA 会員。本論文では、研究戦略の立案および研究統括を担当した。

査読者との議論

議論1　課題の新規性と成果

質問（上田 完次：産業技術総合研究所）

2 章で、各領域の技術を融合する理由として、融合により新規技術が生まれて新たな研究領域の創出が期待されるためとされていますが、この研究でどのような結果が得られたのでしょうか。

コメント（赤松 幹之：産総研ヒューマンライフテクノロジー研究部門）

本論文の内容に新規性がうたわれていますが、専門外の読者にとっては新規であるかがにわかには理解できません。これまでもシステムの信頼性を向上させるような手法が提案されているのではないかと推察されますが、この研究が行なわれる前の状況を少し解説していただくことで、新規性が浮き彫りになると思われます。また、同様に、このような必要な技術がなぜこれまで着手されていなかったのか、なぜそれが困難であったかなどを説明していただくと良いと思います。

回答（加藤 淳）

システムの構成要素による協調動作に着目し、この研究の背景などをご説明させていただきます。通常、システムの構成要素による協調動作は、システム開発の終盤に実施されるシステム試験では確認されますが、ここで協調動作の不整合が検出されるとシステム開発の上流工程に立ち戻る必要があり、改修に多くのコストを要します。また、開発終盤における設計変更は、システムの信頼性を低下させる可能性があります。したがって、システムの構成要素による協調動作は、システム開発の上流における確実な設計および確認が必要です。しかし、これらを実現する開発手法は提案されていませんでした。それは、システムにおける信頼性の観点から、システムの構成要素による協調動作が着目されることがなかったことと、システム開発の上流ではシステムの品質を作り込むこと自体が比較的新しい概念だったことに起因します。本論文の 1 章では、これらを加筆いたします。

また、この研究を行った結果、次に示す四つの成果が得られたと考えます。一つ目は、システム仕様を協調動作が整合している構成要素の仕様および構成要素間のインタフェース仕様に分解する本開発手法を確立したことです。二つ目は、アーキテクチャ設計手法とモデル検査を融合するためにはブリッジ技術が必要であることを明らかにし、協調動作に着目した場合のブリッジ技術の実現例を挙げたことです。三つ目は、主にソフトウェア開発に適用されるモデル検査をシステム開発に適用したことにより、モデル検査の適用領域および研究領域を拡大したことです。四つ目は、ロボット産業界に対しこの研究における開発手法を提案したことです。

議論2　協調動作

コメント（上田 完次）

「構成要素の協調動作」の表現が多数用いられていますが、一般読者にとってはその意味を理解することが困難と思われます。また、「協調動作の整合、不整合」も自明のように記述されています。協調動作の定義ないし意味を明確に示して、わかりやすい記述にしてください。

コメント（赤松 幹之）
協調動作に関する評価の観点の設定方法、協調動作が満たすべき性質の抽出方法、安全性の考慮の方法などが書かれていませんので、例えばロボット技術者がこれを使えるかどうかが分からないと思います。ロボット関係者がこれを使ってみようと思う手掛かりとなるような説明があることが望まれます。

回答（加藤 淳）
本論文では構成要素の協調動作を＂システムの構成要素における処理がインタフェースを介し、他構成要素の処理と連携すること＂と定義いたします。協調動作の整合を＂システム仕様に対し構成要素による協調動作が正しく行われていること＂と定義いたします。また、協調動作の不整合を＂協調動作の整合の定義を満足しないこと＂と定義いたします。本論文の1章では、これらを加筆いたします。

協調動作に関する評価観点については、協調動作に関連する仕様および協調動作が満たすべき性質を基に設定します。適用事例における測定サブシステムおよび統合制御サブシステムについては、協調動作に関連する仕様からメッセージおよび処理の抜けが懸念されました。また、協調動作に関連する仕様および協調動作が満たすべき性質の中に、100 ms 以内などの時間的な制約やメッセージおよび処理のタイミングに関する仕様が存在しました。したがって、4.2 節で示す二つを評価観点として設定しました。5 章 2 節 2 項では、これらを加筆いたします。

協調動作が満たすべき性質は、システム仕様とそれをブレークダウンした構成要素仕様との対応関係がまとめられるトレーサビリティマトリクスを用いて抽出します。

また、協調動作における安全性の確認については、この研究の課題です。安全性とはシステムが決して悪い状態に到達しない性質を指します。＂システムとして悪い状態＂をもれなく識別することが困難であるため、協調動作における安全性の確認に漏れが生じる可能性は否定できません。本開発手法に対し Fault Tree Analysis（FTA）などのような安全性解析手法を組み合わせることで、本課題の解決策を検討する予定です。

議論3　技術の選択

質問（上田 完次）
「IEEE 1220 に従い・・・」のような表現が多数見かけられます。この研究で開発されている手法との関係はどうなのでしょうか。また、なぜ IEEE 1220 を採用したのか、根拠も明確ではありません。

コメント（赤松 幹之）
システム設計技術にアーキテクチャ設計以外に何があるのか、同様にアーキテクチャ設計として IEEE 1220 以外に採用しなかった技術についても記載してください。さらに、数あるアーキテクチャ設計手法の中から IEEE 1220 を選択したシナリオを明確にするために、他の手法にはどのような欠点があるのかが記載されていると、理解しやすいと思います。

モデル検査とは、システムの遷移状態を表すモデルに対して網羅的に検証する技術と 1 章に記載されていますが、ここで用いられたモデル検査手法はこの研究で独自開発されたものか、既に開発されていたものを適用したのかが分かるように記述をお願いします。

モデル検査ツールとして有限オートマトンベースの検査ツールを選定したとあり、そのメリットが書かれていますが、それ以外のツールとして何があって、それらの欠点が何であるかも書いてください。同様に UPPAAL の選定についても、もし他の候補があったのでしたら、その点も記載してください。

回答（加藤 淳）
この研究では既に有効性が認められている技術を選択し、高品質な開発手法を効率よく確立する研究方針を採用しています。本開発手法を構築するにあたり、有効性が認められ標準化されているアーキテクチャ設計手法を適用しています。アーキテクチャ設計手法以外の代表的なシステム設計技術として、構造化分析・設計手法があります。アーキテクチャ設計手法はデータやサービスなど特定の技術要素を中心とするシステム設計には不向きであるものの、特定の技術システムには依存しない汎用的な設計手法です。したがって、この研究ではシステム設計技術としてアーキテクチャ設計手法を選択しました。また、アーキテクチャ設計手法が規定される代表的なシステムエンジニアリング標準として、ISO 15288、ANSI/EIA 632、IEEE 1220 がありますが、IEEE 1220 は各プロセスにおける作業および手順が詳細に規定されていることから IEEE 1220 を採用しました。本論文の 3 章では、これらの比較について記載いたします。

モデル検査は検証技術として既に確立されている技術です。1980 年代初頭より研究が開始され、近年、ソフトウェア開発では普及が進んでいます。モデル検査は＂モデルを用いた検査＂なのですが、検証技術の固有名詞になっています。本論文の 1 章ではこれらを加筆いたします。モデル検査以外の代表的なシステム検証技術としてテスト手法およびシミュレーション手法がありますが、モデル検査は状態遷移については満たすべき性質が成り立つかどうかを網羅的に検証することができます。したがって、この研究ではシステム検証技術としてモデル検査を採用しました。本論文の 3 章ではこれらの比較について記載いたします。

また、形式手法の中には定理証明という検証技術があります。定理証明とはシステムの仕様や設計を意味論が数学的に定義された言語で記述し、厳密な証明を与える手法です。定理証明はシステムに対する厳密な検証を可能としますが、人間と対話的に証明を進める部分があり多くの労力を要します。定理証明について、現状、筆者らは産業界では適用され有効性が認められているとはいい難い検証技術と考えます。したがって、この研究における検証技術の候補からは除外しました。

モデル検査の種類には代表的なものとして、有限オートマトンおよび有限オートマトンを拡張した時間オートマトンがあります。適用するモデル検査およびツールは、評価対象の特徴に応じて選定する必要があります。外部からのイベントをトリガとして状態が遷移するような一般的な状態遷移を評価する場合には、有限オートマトンに対応したモデル検査ツールを選定します。代表的なモデル検査ツールとして SPIN があります。時間制約を含む状態遷移を評価する場合には、時間オートマトンに対応したモデル検査ツールを選定します。代表的なモデル検査ツールとして UPPAAL があります。有限オートマトンに対応したモデル検査で確認できる対象は、時間オートマトンに対応したモデル検査ツールを用いても確認することができます。しかし、その逆は成り立ちません。本論文の 4 章では、これらを加筆いたします。また、今回の適用事例では、協調動作として時間的な制約を評価する必要がありました。また、時間オートマトンに対応したモデル検査ツールについて、産業界の実例に対して適用可能な品質を有するのは、現状は UPPAAL のみと筆者らは判断しました。

議論4　成果の産業応用

コメント（上田 完次）
この研究の実例の成果が、実際の産業応用にどのように使われたのか、あるいは、使われる見通しがあるのか、その際の適用限界などについて言及されると、第 2 種基礎研究として意義が高まると思います。

コメント（赤松 幹之）
この研究の重要なことは、この手法が産業界などで実際にシステムを開発している人達に使われることだと思います。その意味で、この成果を活用できる対象はどういったシステムなのか、この手法の適用範囲についての説明が望まれます。また、この手法を広めていくための方策などについてもお考えのことがあれば記載してください。

回答（加藤 淳）
この研究における開発手法は、特定の技術システムに特化しない、

技術システム全般の開発に適用することが可能な手法であると考えます。システムを設計する際、システム仕様を基にシステムを構成する要素を見出し、構成要素を正しく協調動作させることで機能を実現しますが、これは技術システム全般では共通的な概念であるためです。また、本開発手法は、適用対象システムにおける協調動作に関する固有の問題に対処することが可能な手法です。本開発手法は適用対象システムにおける協調動作の特徴に対し、評価観点およびモデル検査ツールを選択するブリッジ技術を有しているためです。

現在、筆者らは産業用ロボットメーカーと共同で、産業用ロボットに関する新製品の企画・開発を行っています。本論文の 5 章における産業事例がそれにあたります。本論文執筆現在、この研究における開発手法を適用し、協調動作が整合していることが確認された仕様を基に、産業用ロボットの開発が進んでいます。産業用ロボットにおける実際の製品開発に本開発手法を適用し、産業用ロボットの高信頼化に貢献した実績から、この研究が第 2 種基礎研究に値すると考えています。本論文の 5 章にこれらを加筆いたします。

ただし、本開発手法の適用には注意すべき点があります。本開発手法では、構成要素の協調動作が整合していることを確認するために、モデル検査を採用しています。モデル検査におけるモデルの状態数が多い場合、状態の組み合わせが膨大になることでモデル検査が終了しない状態爆発という現象が生じることがあります。つまり、アーキテクチャ設計の結果、構成要素の協調動作が極端に複雑になると、モデル検査による協調動作の検証が終了しない可能性があります。その場合、アーキテクチャ設計の際に構成要素による協調動作が極端に複雑にならないように再度アーキテクチャ設計を実施する方策や、協調動作に関する仕様のモデルの状態数を減らす方策などが必要になります。6 章に " 開発手法の適用性 " という節を設け、これらを加筆いたします。

本開発手法については、筆者らが所属する慶應義塾大学大学院システムデザイン・マネジメント研究科をとおして、産業界に広めることを検討しています。本論文の 7 章では、これらを加筆いたします。

議論5　選択した技術のメリット/デメリット

質問（赤松 幹之）

3 章では技術の選択を記述していただきましたが、機能 a を実現する手法として、アーキテクチャ設計手法と構造化分析・設計手法とを挙げ、アーキテクチャ設計手法は、データやサービスなど特定の技術要素を中心とするシステム設計には向かないが、汎用的な設計技術であることから、アーキテクチャ設計手法を選択したとあります。要素技術の選択は研究のゴールに依存して行なわれることから、そのゴールを明記して、それとメリット / デメリットを比較検討した結果としてアーキテクチャ設計手法を選択したという記述ができますでしょうか。

この研究のゴールは汎用性のある手法を構築することにあるからとも推察できますが、汎用性をゴールとすると実際の個別問題には適用しにくくなるという一般的な問題はあると思います。それを解決するのがブリッジ技術かとも推察しますが、もしそうでしたら、（議論 3 とも関係しますが）その点を明記してください。

回答（加藤 淳）

この研究の目標（研究のゴール）を明確にいたします。この研究の目標はシステムの仕様を構成要素の仕様および構成要素間のインタフェース仕様に分解し、それらの協調動作が整合していることを確認する、特定の技術システムに特化しない開発手法の確立です。本内容について 2 章に加筆いたします。本論文 3 章の機能 a. を持つシステム設計技術として、構造化分析・設計手法およびアーキテクチャ設計手法が挙げられます。構造化分析・設計手法はシステム環境の変化に対して安定している業務情報などのデータに着目しシステム設計を行います。それにより保守性および拡張性を有するシステムを構築することができます。しかし、情報システムを念頭におき開発された手法につき、情報システム以外のシステム設計には不向きな側面を持っています。一方、アーキテクチャ設計手法はその設計に特化した手順や作業が規定されていないため、専用の設計手法に比べ労力を要します。しかし、アーキテクチャ設計手法は特定の技術システムに依存しない汎用的な設計手法です。したがって、この研究の目標における特定の技術システムに特化しない開発手法である点を踏まえ、機能 a. を持つシステム設計技術としてアーキテクチャ設計手法を選択しています。本論文の 3 章では、システム設計技術の中からアーキテクチャ設計手法を選択したプロセス、構造化分析・設計手法およびアーキテクチャ設計手法のメリット、デメリットを修正いたします。

開発手法について、ご指摘のとおり一般には特定の技術システムに特化しないことで個別の問題に適用しにくくなる問題はあると考えます。本開発手法ではブリッジ技術により、適用対象システムにおける協調動作の特徴に応じて評価観点およびモデル検査ツールを選択しています。それにより、協調動作に関する適用対象システム固有の問題に対応しています。これらについて、本論文 6 章 2 節に加筆いたします。

議論6　ブリッジ技術

コメント（赤松 幹之）

ブリッジ技術が必要であることを明らかにしたことがこの研究の成果の一つであると述べていますが、なぜブリッジ技術が必要になったのか、またブリッジ技術が満たすべき要件は何であると判断したのか、なぜブリッジ技術と名付けたかなど、ブリッジ技術の研究シナリオについても記載をお願いします。

アーキテクチャ設計手法とモデル検査法がそれぞれ独立した考え方に基づいて開発されてきたために、モデル検査法を適用するためにはアーキテクチャ設計手法からの出力では不十分であったものと推察しますが、そもそもコンセプトが異なるものをつなげるためには、互いを変換する技術が必要になるのは当然ともいえます。したがって、それが単なる変換技術なのか、それとも、例えば、協調動作を検証するという視点が一つのポイントとなって開発した技術なのかなど、オリジナリティを明確にするために、こういった点についても言及してください。

回答（加藤 淳）

この研究では、システム設計の段階で構成要素による協調動作に対してシステム検証を行います。そのためには協調動作に着目し、アーキテクチャ設計のアウトプットおよびモデル検査のインプットをシームレスに繋ぐ必要があります。そこで協調動作に関連する構成要素の仕様および構成要素間のインタフェース仕様、協調動作が満たすべき性質、適用するモデル検査ツールを導出する技術を開発しました。この技術はアーキテクチャ設計手法とモデル検査の橋渡しをすることからブリッジ技術と呼びます。ブリッジ技術は協調動作に着目し IEEE 1220 といったシステムエンジニアリング標準とモデル検査を融合する具体的な方法を明確にしたところに新規性があると考えます。これらについて、本論文 4 章 2 節に加筆いたします。