創発的性質の導出，記述及び検証について

林　　信宏[†1]　大森洋一[†1]
日下部　茂[†1]　荒木啓二郎[†1]

ソフトウェアの複雑化に伴ってソフトウェアの品質を保証するための検証が重要である．検証するには，検証のための性質が必要であり，適切な性質を用意することが課題である．しかし，重要な性質，例えば，安全性は，創発的性質であり，対象ソフトウェアと環境（関連のある他のオブジェクトなど）を含むシステム全体を視点にすることで認識される性質である．本稿は，システム論に基づいて創発的性質を導き出し，この性質を形式検証手法で検証する方法について議論し，可能な方向性を示す．

Discussing the Derivation, Description, and Verification of Emergent Properties

Hsin-Hung Lin,[†1] Yoichi Omori ,[†1] Shigeru Kusakabe [†1]
and Keijiro Araki[†1]

The growing complexity of software makes software verification much more difficult. One of the challenges in software verification is specifying appropriate properties for especially safety critical software.　owever, safety properties are emergent properties which are not recognizable unless we put together the target software and its deploying environment. This position paper discusses the idea of deriving emergent properties based on system theory and then applying software verfication using formal methods.

1. ソフトウェア検証と創発的性質

コンピュータ及びソフトウェアの進展に伴って，複雑系システムやCPS（Cyber-Physical System）のソフトウェア開発ニーズが増えた．例えば，今日の組み込みシステムでは、ソフトウェアの機能が複雑で、複数ハードウェア及び外部の物理環境とのインタラクションが必要である．その上，車載システムのようなシステムでは，安全性の保証が重要で，きちんと開発過程で分析と検証を行うべきである．形式手法を導入することが有力なアプローチの一つであるが，如何にモデリングと検証技術の適用を行うかは課題となる．もう少し詳しくすると，対象システムをどうモデリングすれば良いか、及び対象システムのどの性質を検証すれば良いかのことである．ここでは，後者に注目することにする．

複雑系システムに対して，システム・エンジニアリングでは，創発的性質 (Emergent Properties) と創発的振る舞い (Emergent Behaviors) といった概念がある[1)]．これらの概念の中心は，システムをソフトウェアだけではなく，ソフトウェアとやり取りする環境及び人間までも含まれていて，ソフトウェア開発はこれら全体を１つのシステムとして考慮すべきことである．例えば，渋滞という状態では，各々の自動車では意味がなく，道路（道路網）の構造も一緒に考慮するときこそ意味がある．ソフトウェア検証に良く確認する安全性もソフトウェアとその環境を含むシステムの創発的性質である．

2. 創発的性質とシステム論

創発的性質について，学科によって説があり，定説のようなものがないが，（自然）科学の観点からは一般システム論（General System Theory）[2)] がある．一般システム論によれば，システムには，互いに作用している要素からなるものであり，独特の構造を持った複数の下位システムが存在する．つまり，システムは，単なる要素の集まりではなく，要素が相互作用の関係を持つことである．システム論のアプローチは，いくつかありますが，ソフトウェア工学的

†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

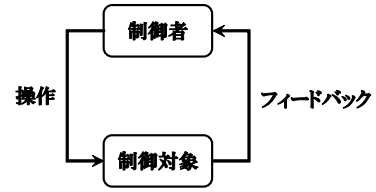

図 1 基本制御関係

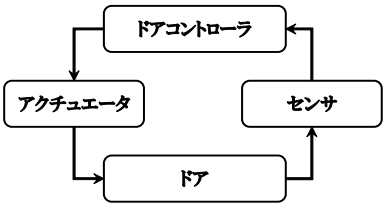

図 2 電車のドアの制御関係

には，STAMP/STPA(Systems-Theoretic Accident Modeling and Processes/Systems Theoretic Process Analysis)[3] のシステム理論と自動制御理論に基づくハザード分析手法がある．STAMP/STPA は，システム要素間の相互作用を図 1 の基本的制御関係で理解し，トップダウンのアプローチでシステムの分析を行う．例えば，電車のドアを制御するソフトウェアの分析には，図 2 で抽象的にシステムを理解し，分析する．分析結果の一部として,「電車が移動中に，ドアコントローラがドアを開く操作の指示をアクチュエータに出す」が危険だと分かる．

3. 創発的性質の記述と検証方式の提案

複雑系システムにおけるソフトウェア品質の保証について，我々は図 3 で示された作業と記述で創発的性質の導出及び形式検証技術の適用を考えている．創発的性質の導出について，第 2 節で述べた STAMP/STPA の分析手法が安全性に関する創発的性質の導出には有用だと考えられる．安全性以外の創発的性質を導出することについて，我々は STAMP/STPA をベースにして一般システム論に基づくアプローチを考えている．この作業には，STAMP/STPA の分析で VDM(Vienna Development Method) モデルを記述する試み[4] があった．しかし，STAMP/STPA の分析結果を直接に VDM モデルでの事前・事後条件及び不変条件によって表現することが難しい．創発的性質は，システムの全ての要素の相互的影響の総合現象なので，個別の操作や変数による表現では困難なためである．解決策として，VDM の記述のほかに，時相論理式も記述するアイデアを提案する．VDM の記述では，要素に関する変数と変数に関連する基本述語の定義を記述し，これらの

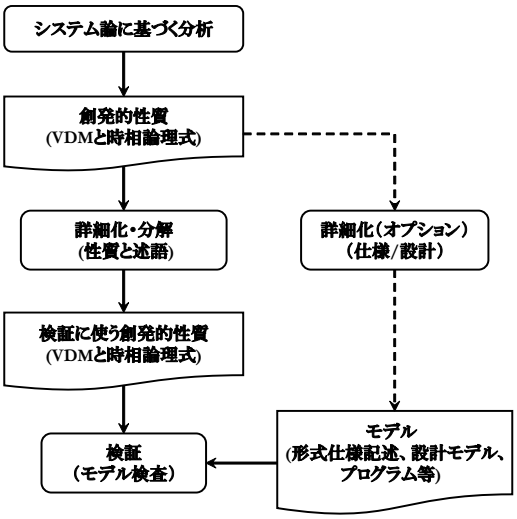

図 3 創発的性質の導出と検証に関する記述と作業

述語を用いて時相論理式で創発的性質を表現する．この記述方式では，モデルや仕様を定義することではなく，システムに関する述語を定義及び創発的性質を定義することが目的である．

性質の導出及び記述ができたら，次には検証の準備を行う．VDM で記述した変数及び述語の定義では，抽象度が高いため，詳細化または分解を行う必要がある．この詳細化・分解する作業では，検証のためのモデルを作成する作業でもある．例えば，VDM モデルを陽仕様まで拡張すれば，VDM モデルに対するモデル検査の適用が行える[5]．また，検証モデルを別のアプローチで用意する場合，例えば，UML による設計モデルまたはプログラムを検証することも考えられる．

現状では，図 3 で示した記述と作業のアプローチが提案の段階で，事例研究と関連技術の開発がこれからの課題である．

参 考 文 献

1) Hsu, J.: *Emergent Behavior of Systems-of-Systems*, INCOSE Mini-Conference, 2009.
2) Von Bertalanffy, L.: *General System Theory: Foundations, Development, Applications*, Braziller, 2003.
3) Leveson, N.: “An STPA Primer”, 2013, http://sunnyday.mit.edu/STPA-Primer-v0.pdf
4) 畑 彰拓ほか：**モデル指向形式仕様記述におけるハザード解析法** *STAMP/STPA* **の活用**, ソフトウェアエンジニアリングシンポジウム 2014 論文集，pp.33–38, 2014.
5) Lin, H-H., et al.: *Towards Model Checking VDM Specifications*, 信学技報, vol. 114, no. 23, SS2014-4, pp. 19–24, 2014 年 5 月.