2018年3月1日発行
第13巻 第4号(通巻55号)
ISSN 1349-8622

SEC journal 52

IPA 独立行政法人情報処理推進機構
Information-technology Promotion Agency, Japan

SEC journal No.52
Vol.13 No.4 Mar. 2018

巻頭言

IoT時代に変革を加速する組込みシステム技術

竹内 嘉一
一般社団法人組込みシステム技術協会 会長

政府より第４次産業革命の先にあるSociety5.0（超スマート社会）の実現に向けてConnected Industriesが打ち出され、様々なモノとサービスがつながるCPS/IoT（Cyber Physical System/Internet of Things）がますます必要不可欠なものとなっています。この時流の中で、エッジコンピューティングに代表される組込み技術の重要性が再認識され、センシング、ロボティクス、ビッグデータ、人工知能、セキュリティといったデジタル化の潮流が革新的サービスを生み出しつつあります。

そして、組込みシステム業界も、モノの提供からサービスの提供への変化を求められており、これまで自社のみで対応してきたクローズなビジネスから、競争領域と協調領域をうまくバランスさせたオープンイノベーションへの変革、正に、産業構造を変えるパラダイムシフトが起ころうとしています。

30年の節目を越えた新生JASA（一般社団法人組込みシステム技術協会）は、これまで歴史と共に培ってきた土壌の上に、時流に沿った変革の種を植えていきます。そして、この協調領域の旗振り役として『日本のお家芸である組込み技術』と『進化するデジタル化の潮流』をつかみ、組込みシステム業界の更なる発展に向けて、関係省庁との関係強化と産学官の連携を推進していく必要があります。

このような背景の中、IPA/SECによる多くの成果物の発表と取り組みは、KKD（感と経験と度胸）に依存した属人的な開発手法からの脱却を推進し、工学的なソフトウェア開発におけるエンジニアリング手法の普及啓発に大きく貢献され、『組込みソフトウェア』の発展を支える重要な活動になっています。

そして更に、IoT時代を迎え「モノづくり」から「コトづくり」に変遷しようとする中で、組込みシステムは、構成要素の複雑化・高度化が進み、組込みシステム同志がつながり、連動する大規模化が加速している状況であり、システム全体のライフサイクルをカバーした安全性・信頼性・セキュリティ対策などへの対応が課題となっています。この課題に対して、JASAとしても技術本部の安全性向上委員会の活動を中心に、IPA/SECが推進している、複雑なシステム設計において安全性を確保するためのシステム理論に基づく事故モデル（STAMP）及び、その安全性解析手法（STPA）に注目しつつ、独自の調査・研究、及びその普及を目的とした活動を進めており、IPA/SECには、今後更なる実務レベルの深いつながりを期待するものです。

また、技術本部では技術高度化委員会をIoT技術高度化委員会に改め、ワーキング・グループ（WG）を刷新して活動を開始しております。WGは、IoT時代の変革を見据え「IoTスキル検討WG」「組込みIoTモデリングWG」「エネルギーハーベストWG」「エモーションWG」「ドローンWG」の5つを設置し、IoT技術に関する調査・研究を進めていますが、このWG活動においてもIPA/SECの知見とのシナジーを生み出し、大きな成果が出せればと考えています。

更に、2015年の創刊に引き続き発刊された「組込みソフトウェア開発データ白書2017」は、各企業で門外不出であった組込みシステムの開発現場の現状を表すメトリクスとして、非常に有効かつ画期的な取り組みであり、JASAを含む組込み業界ひいては日本の産業界全体の組込みソフトウェアの高信頼化を促進する上で欠かせないものだと認識しています。今後、継続的な発刊への期待もさることながら、データ白書の具体的な活用の模索や別観点でのメトリクス作成など、JASAとしてIPA/SECとの協働を進めていければと考えています。

最後になりますが、これまでのIPA/SECの活動と成果に感謝と敬意を表し、JASAとIPA/SECとが更に一段上の連携強化を推進していければと考えています。

そして、JASAは組込みシステム業界の発展を加速させ、我が国の産業振興に寄与することを目的に、IoT時代の変革に対応して参ります。

システム思考の重要性について考える

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 教授
白坂 成功

IPA/SEC所長
松本 隆明

Connected Industriesに代表される“つながる世界（システムオブシステムズ）”の出現により、単体のモノが価値を生み出す時代から、モノ同士やモノと人とのつながりが価値を生み出す時代に変わりつつある。今後は、製品やサービスの設計・運用は、コンポーネントベースからシステム全体で考えるシステム思考へとシフトしていかなければならない。システムズエンジニアリングに関する我が国の第一人者である白坂成功教授に、システム思考の重要性や人材育成、日本の産業界に根付かせていくために必要なことなどについて伺った。

今、なぜシステム思考が重要なのか？

松本　最近システム的に考えるということが非常に注目されています。その要因をどのようにお考えですか。

白坂 成功（しらさか　せいこう）

東京大学大学院工学系研究科航空宇宙工学専攻 修士課程修了。その後、三菱電機株式会社にて宇宙開発に従事。技術試験衛星Ⅶ型（ETS-Ⅶ）、宇宙ステーション補給機（HTV）などの開発に参加。とくにHTVの開発では初期設計から初号機ミッション完了まで携わる。途中1年8カ月間、欧州の人工衛星開発メーカに駐在し、欧州宇宙機関（ESA）向けの開発に参加。「こうのとり」（HTV：H-II TransferVehicle）開発では多くの賞を受賞。2004年度より慶應義塾大学にてシステムエンジニアリングの教鞭をとり、2011年度より現職。

白坂　基本的には三つの要因があると思っています。一つ目は価値と要素の距離が広がってきていることです。今までは単一のモノに価値があったのですが、それが普及し誰でも持っているという状態になると、今あるモノではなく、それらを組み合わせて違う価値を生み出していかなければならなくなります。つまり、求められる価値が徐々に複雑化、高度化してきているということです。例えばボールペンで言うと、単に書ければ良いということではなくて、きれいに書けるとか、書き心地が良いとか、そういった新価値が求められているときに、単に書けるだけのペンを提供しているのでは駄目だということです。モノの要素と価値の距離がどんどん遠くなっています。

二つ目は外界の影響を受けやすくなった点です。システムを構成する要素が増えてくると、システムの外部要素との関係性が増えてきます。つまり、これまでは1台の車だけの関係性を考えれば良かったものが、車と船を組み合わせるとなったら、今度は船の外部要素との関係性も全体のシステムに関係してきます。船に対する変化が車に影響し、車に対する変化が船に影響するようになります。要素が増えるとそれだけ外界が増え、外界の影響を受けやすくなるわけです。この変化対応ということを考えるためにもシステム思考が重要となってきています。

三つ目は説明責任の必要性です。単純なものならば簡単に確認できたのですが、システムになってくると、システム全体として見なければ分からないシステム特性というものを説明しなければいけなくなります。例えば、安全性などはシステム特性の典型的な例です。このように、新しい価値の創造が求められ、関係性が増えて変化対応が上がり、これらをきちんと説明しなければいけなくなってきた。このような3つのことが、システムとして捉えなければいけないという考え方の背景にあると考えています。

松本　色々なモノがつながって動くような時代になってきました。それによって、新価値を生み出せるようになってきたというところが大きいわけですね。

白坂　両面あると思います。組み合わせたから新しいことがで

きるようになったという面もありますが、新しいものが必要だから組み合わせたという面もあります。先ほど申し上げたように、新たな価値を提供しようとすると、新たに組み合わせざるを得ない。新価値創造というのは、経営学では“知の探索”と言われているもので、新たな組み合わせで起こすものです。これに対して、改善活動は“知の深化”と言われます。いったん作ったもののコストを下げていくとか、性能を良くする・軽くする・小さくするというような活動は、今の組み合わせの中でできる改善です。一方、新価値創造では新しい価値を提供し始めようとすることは、今ない組み合わせをせざるを得ないということでもあります。いずれにしても、組み合わせたから新しいことができるようになったと同時に、新しいものが必要だから組み合わせた、という面があるわけです。

あいまい性が拡大するシステム

松本　加えて最近は、モノではなくてサービスで提供するという方向になってきました。とくにシェアリングエコノミーのような形で、自動車そのものを売るのではなく、みんなでシェアして、それを交通手段として活用する、というスタイルになっている。そういうこともシステム思考が求められる要因と考えて良いでしょうか。

白坂　その通りですね。単なるモノの組み合わせではなくなっています。例えばシェアリングエコノミーでは、法制度の問題が出てくる。システムとして提供しようとすると制度自体を変えていかなければいけません。自動運転だったら社会受容性のようなことまで含めてデザインする必要もあります。

松本　製品やサービスの提供側から見ると本当に大変な時代ですね。「VUCAの時代」と言われますが、あいまい性が増し、システムの範囲も増え、どこまでをシステムとして見れば良いのかということが定義できなくなっています。

白坂　もともとクローズドシステムで考えていたものの境界があいまいになってオープンで考えざるを得なくなってきているんですね。

松本　逆に言うと、それだけあいまい性が増え、なおかつその範囲が広域になると、システムとして捉えるのがますます難しくなってきます。むしろ、逆にコンポーネントで考えたほうが捉えやすいのではないかという発想に逆戻りしそうな気がしますね。

白坂　いや、そのアプローチでは価値につながらなくなります。上からと下からの両方が必要だと思います。何を考えているのか分からずに組み上げていくと、どこに到達するか分からないものが生まれます。更に難しいのは、コンポーネント側が変化するということです。テクノロジーが変化する。今これとこれを組み合わせました、というときに、これ自体も変化要素なので、ボトムアップで考えたらまた考え直しです。そうではなくて、上から考えておけば、手段としてのコンポーネントですから、これが変化しても違う手段で同じ価値を提供できるという話になる。

松本　なるほど。変化対応ということで考えても、全体で考えておけばコンポーネントを置き換えるだけで良いわけですね。

白坂　自動運転で言うと、前の車との距離を測るために今は双眼カメラかもしれませんが、次はライダーかもしれない。更にその次はレーダーかもしれない。しかし、前の車との距離を測らなければいけないということは変わりません。「前の車との距離を測るということを利用した安全設計」は同じなのです。手段であるセンサが変わるだけなら、そこだけを検証し直せば良い。下から積み上げていると全部を検証し直さなければなりませんが、上から考えていけば変化した一部だけを検証し直すことで済みます。そういう時代になり始めていると思います。

システム思考で考えるとは、どういうことなのか

松本　そもそも、システム思考とはどういうことだとお考えですか。

白坂　基本はすごく単純で、俯瞰的に捉える、全体として捉えるということだけなんです。ただ、俯瞰と言っても簡単ではありません。

　私は三つの軸で俯瞰するようにしています。一つ目は空間の俯瞰です。この範囲で見ましょうという空間俯瞰。二つ目は時間俯瞰。終わりまでのことを考えて見通して、今どうするかを考える。そして三つ目が意味俯瞰です。これは目的志向性ということになるのですが、ここをターゲットとして、このためにどういうふうにやっていくかという

松本　隆明（まつもと　たかあき）

1978年東京工業大学大学院修士課程修了。同年日本電信電話公社（現NTT）に入社、オペレーティング・システムの研究開発、大規模公共システムへの導入SE、キャリア共通調達仕様の開発・標準化、情報セキュリティ技術の研究開発に従事。2002年に株式会社NTTデータに移り、2003年より技術開発本部本部長。2007年NTTデータ先端技術株式会社常務取締役。2012年7月より独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア高信頼化センター（SEC）所長。博士（工学）。

俯瞰、手段に落とし込んでいく俯瞰です。空間の俯瞰、時間の俯瞰、意味の俯瞰、この三つの俯瞰を進めるのがシステム思考だと考えています。

松本　意味で俯瞰するというのは難しそうですね。目的を達成するためにこういうことをやれば良いという、その抽象度というか、レベルが難しいような気がします。人間はどうしても、何と何をモノとして組み合わせればできると考えてしまうので、その中間的な階層構造のようなものがなかなか思い付かない。

白坂　確かに思い付きにくいのですが、空間と時間だけを見ていても目的には到達できません。システム思考は基本的には全体俯瞰ですが、目的があってやることなので、必ずゴールオリエンテッドにならざるを得ないのです。難しいのは分かっているのですが、そこは慣れや訓練もあります。

私がよく言うのは、家を建てようとしたときに、建てる側からすると初めてなので、どう考えて良いか分からないけれど、家を提供する側からすると、意匠デザインはどうするか、排水や水回り、電気回りはどうするか、お金についてはどういうふうにやろうかという、どういった視点で考えると家が建つのかということが分かっている。家を提供する人たちは「どうしますか」と、家を建てる人に聞くわけです。考えなければいけない項目が分かっている。従って、それをすべての時間と空間においてやってあげればモノはできる。

松本　それはどちらかと言うと、要素単位で考えているということでしょうか。

白坂　いえ、全体を捉える視点です。全体を時間の視点で捉えるのか、空間の視点で捉えるのか、意味の視点で捉えるのか。俯瞰すると言っても、全体をそのままでは捉えられないので、抜けや漏れがないように全体を色々な視点から捉えることが必要です。一度全体を置いて、これをそのまま見ても分からないので分けて考えていく。全体を捉えて、分割して、どう統合していけば良いかを考えながら設計していくというのが、先に進むやり方です。

松本　そうすると、どういう視点で分けていくかというのは大きなポイントですね。

白坂　重要です。例えば、システム思考で有名なのが因果関係ループですが、あれは因果の関係でシステムを捉え直しているものです。システム思考は、基本的には全体を捉え全体で見ます、と言うだけでは先に進まないので、必ず何らかの形で分けなければいけません。

松本　確かにそうですね。

白坂　そのときに、どういう視点で分けていくか。分け方はケースバイケースで毎回変わってくるので難しくなるんです。よくある例が、前から壁が迫ってきたので、壁を押し返そうとする。壁が迫ってきたので押し返そうというのは、一見良さそうに見えるけれども、実際は前の壁を押すと後の壁が押し出される仕組みで、今度は後ろから壁が迫ってくる、という因果の関係になっていれば、何の解決にもなりません。つまり、全体を捉えないで目の前のことだけを見ているから間違った対応になる。物事はつながって関係しているのだから、全体がどういう関係性になっているのかを捉えて解決策を考える――これが今起きている事象をシステム的に捉えるというアプローチです。

松本　先ほどの家の例でも、例えば明るい家が欲しいといったときに、光が色々なところから入るような仕掛けにするのか、照明をたくさん付けるのか、様々な視点から落とし込んでいきますね。

白坂　最終的に実現していくしかないわけです。それをいきなり、明るい家だからいっぱいライトを持ってこよう、では本当は合わないかもしれません。ほかとの関係性もあります。値段の問題もあるかもしれないし、安全面で窓は駄目かもしれないし、強度の問題もあるかもしれない。それらを全部統合して成立させようというのが、システム的に捉えていくという話です。部分だけで見ると家中ガラスにしよう、というアイデアも出るかもしれないけれども、それは一つの観点でしか見ていないので、結局最後は統合していかなければいけません。更に、サービスのようなものが入ってくると、人の感じ方など、必ずしもこうあるはずだと言いづらいものが出てくる。こう考えたからこうやってみた、良かった、悪かった、じゃあここをどう考えれば良いか、というふうに変えていくわけです。物理的にモノを作るのであれば、物理の法則が成り立っているかもしれないのですが、そうではないものがどんどん出てきています。

SysMLは情報の関係性を定義するもの

松本　システム的に俯瞰して捉えるという意味で、例えばSysMLのようなツールを使ってモデル図を書いていくと、全体的な俯瞰の構造が分かるようになってくる、という利点があるでしょうか。

白坂　SysMLは考えたことを可視化するための表記法でしかないので、ただモデル図を書いたからと言って全く新しいものが見えてくることはないように思います。単純化して言うと、以前、3DCADがなかったときには、三面図というものを使っていました。3つの面でそれぞれ表現して、頭の中で3次元がどうなっているかを想像していました。これは単純でした。視点が90度ずつずれている、という状態です。

今、MBSEと呼ばれるモデルベースシステムズエンジニアリングで何をやろうとしているかと言うと、システムズエンジニアがやることをモデルを活用して支援しようとしているのです。その重要なアプローチの一つが、システム全体で考えライフサイクルを俯瞰することです。そうすると多数のステークホルダが見えてきます。使っているときはユーザがいて、メンテナンスのときにはメンテナンスをする人がいて、もしかしたら保険の人がいるかもしれない。廃棄のときには廃棄業者がいます。安全設計上だったら安全の人がいます。このように、様々な人たちが関連したときに、おのおのにとって重要な情報というのがあるはずです。メンテナンスのしやすさを考慮するためや、廃棄のしやすさを重視するための情報があったときに、それを先ほどの三面図のように、頭の中で一つの図にすることはなかなかできません。何らかの関係性がみんなある。それがどう関係しているかというのをどこかで決めてあげなければいけません。この部分を表現するためにSysMLがあるのです。システムモデルと言われているものは、それらを統合的に表した情報の構造体のことを指しますが、SysMLは関係性を定義するために使うものです。情報構造体を定義する人たち、どうなっているのかを見る人たちのための表記法です。

松本　どちらかと言うと、システムを設計すると言うよりも、実際にそれを実装するときの一つのモデルとして考えたほうが良いということでしょうか。

白坂　どちらかと言うと実装よりも設計です。設計するときの情報の関係性を定義するという感じです。つまり、何と何がどう関係するか、ということを決めていくための言語なので、初期の段階で決めてあげるものです。その機能を使ったりメンテナンスしたり、その機能を基に安全性を評価する人たちにとっては、どの機能がどの機能とどう関係しているのか、という情報が必須になります。これは良い、悪い、ここはこう変えたいというのがあれば、この人たちはそれをベースに変えていくことになります。

松本　一つのモデルを色々な観点から見たときに、それの相関関係を示しやすい表記法だということですね。

白坂　おっしゃる通りです。

モノとルールを同時に設計する

松本　俯瞰的に設計するということは、実際にはどのように進めて行けば良いでしょうか。何かツールのようなものがあって、とりあえずそれに従えば近いことができる、というようになっているととても助かるのですが。

白坂　残念ながらそのようなツールはありません。たぶんソフトウェア設計も、当初はそれほど簡単ではなかったと思います。それがソフトウェア分野だけでなく、複数の分野を超えてやっているわけですから、もう一段難しくなっている。つまり、機械設計や電気設計であれば、物理の制約でできることは限られています。それがロジックだけで好き勝手に何でもできるようになってきました。ソフトウェア単独で見てもそうだったのが、今度はハードウェアも含み、システム全体で考えて、ライフサイクルでメンテナンスの人たちも複雑度が増し、関係者がどんどん増えている中でみんなの要求を満たした設計をやろうとしているわけですから簡単ではないものになります。

松本　確かにそうですね。ソフトウェアも昔はそのままダイレクトコーディングでやっていたのが、フローチャートのようなものを使って論理的にやりましょうということになり、やがてフローチャートを書いてもデータ構造がどうなっているか分からないから、今度はデータ構造設計書のようなものができて、更に今は要素間の関係性が増えてきてインターフェース仕様書をきちんと決めないといけない、ということになってきました。

白坂　それが更に違う技術分野も統合せざるを得なくなり、人との関係性も出てきて、サービスやビジネスが関係する。メンテナンスをどうするのか、ということなどを考慮すると更に複雑です。先ほどお話ししたように、人がどう感じるのかとか、ルールなども同時に設計しなければいけない。自動運転でも、車の設計とルールの設計がトレードオフなんです。すごく単純化すると、自動運転車を運転する場合に運転免許を再度取得し直すというルールにしたとすると、ある部分の設計は楽になります。レベル3のような自動運転車をドライバーに戻します、と言われたときに、どんな人が運転しても問題が起こらないように設計するのは非常に大変なことです。しかし、免許を取り直すことを前提に設計すると、車の設計は非常に楽になるんです。

松本　その通りかもしれませんね。

白坂　車がどういう原理で動いているかを知らずに乗っている人はたくさんいます。でも、その人にポンと返すときに、その原理を知っているか知らないかで、当初の設計は全く変わる。今は自分が運転しているので、状況が分かっているわけです。なぜこういう操作をやっていて、その結果何が起きているかをすべて把握しながら運転している。すごく単純に言うと、我々の世代は、例えばハンドブレーキはレバーを引っ張り上げたり、ペダルを踏んだりするから物理的にブレーキをかけているというイメージがあります。でも、ボタン一つでハンドブレーキをかけることもできますよね。

松本　できますね。

白坂　そういう設計にすると、ブレーキという概念が分からない人が出てくるわけです。車の原理を知らないときに、自動運転側に何か起きて人に戻されたとき、今こういう状況ですということを、原理を知らない人に教えるのは大変なんです。原理を知っていれば、今ここのブレーキのこういう押し付けができなくなっていますとか、そういうのが分かれば、実は返し方が楽です。同じようなことはほかにもたくさんあって、例えば自動運転車を止めていて、バイクがぶつかり、そのためセンサのアライメントがずれたので直したとします。この修理で再び安全に走るという証明は誰がしますか、というのもその一つです。ルールとしてその車は必ずディーラーに戻し、ディーラーでしか安全は保証できません、ということにすればディーラーに安全を検証する装置や人を置いておけば良い。そうではなくて、普通に町工場で修理すれば良いだけとすると、町工場の整備士は多分微細なアライメントも含めた検証をすることはできないので、安全かどうかを評価する仕組みを、あらかじめ車に入れておかなければいけないことになります。これは車の中の設計と、ルールの設計との間でのトレードオフです。そういうのがたくさんあるんです。ルールは自分たちで決められないという前提で何でもやるとなると、大変なことをたくさんやらなければいけない。ルールとモノの設計の中身が完全にカップリングを起こしているのです。しかし、今までの開発はそこを同時に設計するということをやってきていない。分離してやっているので、それができるイメージをみんな持っていないわけです。MITが言っているのも、正に一緒にデザインしなければいけないということです。

どう考えて設計したか、情報を残す

松本　感じ方や能力を考慮した設計を行うとなると、人をモデル化しなければいけなくなってくるような気がします。

白坂　人の動きを忠実に再現するということはできないと思います。実際に今でも、自動運転車の設計で、人のモデル化はそこまで厳密にやっていません。人がどう動くかというのは人によって差が出る。決して画一的なモデル化はできません。ただ、それがなければ設計ができないというわけではないと思います。

松本　以前、ABSのような、ブレーキのアシストシステムの効き具合の感じ方が人によって全く違うので、ブレーキが効いていないんじゃないかと思って、それでおかしいと言って問い合わせをする人が増えたケースがあると聞きます。そういう意味で言うと、やはり人によって捉え方というのは様々なので、それを踏まえてシステムをどう設計するかはすごく難しいですね。

白坂　ポイントは、自分たちはどう考えて設計したのかということを残しておくということです。自分たちはこう考えて、こういう人たちはこう感じるだろうと思ってやりました、と。今日のお話の冒頭に説明責任と言ったのはそこなんです。なぜ自分たちはこうしたのかということがきちんと残っていて、正にこのバウンダリーのエッジの外にいる人たちが、危ないと感じるようなことが出てきたら、あなたはちょっと外側でしたね、ということが作った側で分からないといけない。どういう考え方で設計したか、誰がどう感じるように作ったかをちゃんと残して、ライフサイクルを通じて後のことを考えて設計すると同時に、後の人が当初の設計の情報を基に考えるという、行き来ができる仕組みが重要になってきます。

システム思考ができる人をどう育てるのか

松本　では、システム思考ができる人をどういうふうに育てていけば良いのか、ということになるのですが、そもそもシステム思考ができる人が持っているスキルというのは何だと考えたら良いですか。

白坂　我々もよく議論しますが、やはり必ず出てくるのは、抽象化力です。人間が一度に理解できる範囲に限りがあるとすると、幅広い範囲を見ようと思えば思うほど、大量の詳細な要素をそのまま扱うのではなく、抽象化して少ない要素にして全体を考えるようにしなければ全体をつかめないと思います。抽象化して、これは要するにこういうこととこういうこと、この組み合わせなのか、みたいなことがぱっと分かるようにならないと、全体の把握ができない。抽象化するのと具体化するのを、我々は抽象度のコントロールと言いますけれども、これができないと多分うまくいかない。

松本　抽象化力とは具体的にはどういうスキルですか。

白坂　抽象化力は目的志向性を持った帰納化能力と言えると思います。色々な色の車があったときに、色の観点で帰納的にまとめていくのか、車の観点で帰納的にまとめていくのかは、目的によって変わってきます。色々な視点から物事を見て、ぱっと考えることができて、その中のこのポイントが今の目的に対しては適合しているから、この視点で考えよう、ということが考えられないといけません。

松本　ある意味で言うと、論理的な思考ができるということでしょうか。

白坂　論理思考は前段階として必要です。ベースとして論理的な思考ができるというのは、おそらくあると思います。

松本　そういう人材はどのように育てていけば良いのでしょうか。

白坂　論理的な思考は訓練でできるようになります。抽象化力は、難しいのですが、訓練すれば身に付きます。ただ、100人に教えて100人全員が完璧にできるようになりますか、と言われると難しいでしょう。

松本　それは個人に依存するということでしょうか。

白坂　その人にとっての向き、不向きはあると思います。それが興味なのか、心理学的なものなのかまでは分析していないのですが、明らかに、ぱっとうまくできる人と、なかなかできない人がいます。

松本　システム思考ができる人間をどういう場で育てていくのが良いのか、大学できっちり理論的な研究、学習で教えていくのが良いのか、それとも現場で実践して身に付けていくのが良いのか、どうお考えですか。

白坂　両方必要だと思います。現場でやっていることには、システム思考以外のものがたくさんあります。何がシステム思考の範囲で、何がそうではないのか、現場だけではそこが分からなくなります。一方で、理論だけでは使えない。教育理論で言うと、行動主義と言われる知識を問うものがあります。これは知識なんです。使うかどうかは関係ない。やる過程が合っているかどうかを見るのは認知主義的なアプローチです。ワークショップなどで、ここはこうやったほうが良いですよ、ああやったほうが良いですよと教えていくようなものですね。しかし、これができても学んだことが自分の課題に使えるかと言うと、ダイレクトにはできないです。やったことをいったん自分のものにして、違うものに適用する能力というのは、もう一段上の能力で、これを教育するのは構造主義的アプローチです。実際に使うことを通じて、知識を一度昇華させる。学んだこれは、こういうふうにやることなんだ、こういう考え方をすることなんだ、と自分で受け取れると、違う分野に持っていけるようになります。実際に習ったことを使いながら自分のものにしていく作業をやらなければいけないので、プロジェクトベースラーニングと言われています。“やって、学んで、やる”――これがベストです。学んだ後は、やらないと駄目なんです。過去の経験を学んだことにマッピングするだけでは、なぜそれで良いかが分かりません。マッピングすれば、これはこういうことなのかと分かるのですが、新しいところに持っていったときに、結局自分のやったものでしか持っていけなくなります。違うものをやったときに分からなくなる。学ぶ場合は汎用的なものとなりますが、実際に使うためには、その分野に落とし込む必要があります。この落とし込みのときに、こう考えたから、これは実はやらなくても良いんじゃないかと、重要なポイントが違うのをちゃんと自分で理解して、今やろうとしているシステムはこういう特徴だから、こういうアプローチでやらなければいけないんだと、学んだ後にやるということことはすごく大きいのです。最も良いのは、システム開発をした経験がある人が、リカレント教育で学んで、更に実際に使うということです。ぜいたくですがそういうことなのです。

松本　いったん学び直して、きちんと体系化され、普遍化されたものにして、それを別の領域に適用していくということですね。

白坂　はい。その通りです。別の領域でやる必要がないのであったら、誰かが作ったものの通りにすれば良いだけです。しかし、そんなことはこれから先あまりないでしょう。新たな人材を育成するとなると、実際にものづくりを経験したことのある人たちが、一度きちんと学んで、自分たちの領域で使っていくという、ここをいかに早くやるかだと思います。日本では社会人が学ぶことはMBA以外でほどんどありません。この教育の弱さがシステム的な人材育成の弱さにつながっている可能性はあります。海外では、社会人がもう一度学ぶことは当たり前です。

松本　システム的に見ていくという能力がどこまで付いたのか、この評価については、どうお考えですか。

白坂　簡単ではないですね。例えばシステムズエンジニアリングで言うと、スキル標準的なものは世界中で作られようとしています。もともとイギリスが作ったものがあるのですが、それをブラッシュアップしようとしています。難しいのは、それで評価できるのが知識なのか、ということです。資格というのは大体において知識を問うものです。しかし、簡単に分かる通り「知識がある」=「できる」ではありません。前提になるかもしれないけれど、ここのギャップを埋めていくときにどう評価するかというのは、やっぱり単純ではありません。確かに、できる人とできない人というのは話していて分かります。ではそれを客観的に評価する仕組みは何だろうというのは、実は今悩んでいるところです。

学び直す機会を作る

松本　システムを設計するときに、いわゆる良いシステムと悪いシステムと言ったら良いのか、先ほどの色々なシステム、俯瞰的に見て考えられたシステムと、そうではなくて積み上げで局所最適の合計でできたようなシステムと、どのように評価すれば良いのでしょうか。

白坂　難しいのですが、システムの評価について今あるアプローチは、結局は目的適合性だと思います。何で評価するか自体もシステムごとに異なるというのは、基本的にシステムの分野で言われていることです。そのシステムにとっての重要なポイント、

低コストが重要なのであれば、低コスト性が主な評価軸でしょうし、何らかの品質が重要なシステムであれば、それがポイントになるでしょう。もちろん一つではなくて複数になるのですが、それによって変わるというのが今の世界的な流れだと思います。

松本　その優先度が分かるというのがシステム思考に優れた人間なのかもしれないですね。目的が幾つかあるときに、どの目的がメインなのかということが分かる。

白坂　確かにそこを間違えてしまうと、機能がたくさんあるから良いと考えてしまったりしますね。でも機能がたくさんあると結局、使いづらいこともあります。

松本　そうですね。コストにもつながってきますし、保守性の良し悪しにもつながってきます。やはり本来の目的、メインの目的は何かということをきちんと見定めることができる人間、そういう人をどうやって育てていくのかということですね。先ほどのリカレント教育によって、きちんと学び直す機会を増やしていくことの重要性を感じます。

白坂　同感です。

部分をシステムとして捉えて考える

松本　学び直すときの、学の受け入れ体制についてはどうお考えですか？

白坂　日本の教育には大きく2つ問題があります。一つは社会人が学べる環境です。これは内容を問わず、そもそも少ない。ビジネススクール以外ほとんどないのが現状で、働きながら学べる環境をもっと整備しなければいけないと思います。

もう一つは教える側として、システム的なものを教える人の確保が難しいことです。上述した通り、学んだ後に、それを実際の開発に適応して初めて分かるという意味では、ずっと学にいる人ではない人を学に連れてきたいわけです。アメリカでは普通に行われていることです。企業出身の人が、今度は学に行って教えて、そこで学んだ人が実務に適応して学んで、また学に来る人がいて、というようになっています。

松本　そもそも今の学は、どちらかと言うと要素技術偏重で、AIや量子コンピューターやデバイスなどには力を入れるけれども、システムなど目に見えないモデルのようなものを教えることへの取り組みが弱いですね。

白坂　その理由の一つが、論文になりにくいことだと思っています。システムの開発は、違う人が同じプロセスでやっても同じ結果にはなりません。再現性がないため、サイエンスではないと考えられてしまうんです。

松本　確かにそうかもしれないですね。システム思考でやらなければいけないということを、日本の産業界にもっと根付かせていきたいと思うのですが、日本の産業界というのは色々な問題を抱えていて、例えば産業構造の問題があります。ソフトウェアの開発も多重下請け構造で、末端になると自分が何のソフトを開発しているか分からないんです。本当に個別のモジュールでしかない。それをシステムで見ろと言われても見られないのではないでしょうか。しかも全体をシステマティックに見られる人間は発注側にもいません。

白坂　おっしゃる通りですが、とは言えきちんと作っている人たちがいます。システムは階層性を持つので、必ずしも全体を見る人だけがシステムを見ているかと言うと、決してそうではありません。私も「こうのとり」をやっているときは、「こうのとり」の全部を見ていたわけではありませんでした。その中の電気モジュールという一モジュールを見ていたんです。ここを見ている人が全体をシステム的に考えなければいけないのかと言うと、そうではなくて、部分をシステムとして捉えて考えるというのもあります。メンテナンスや試験の容易性を考えて、どうやってここの範囲を設計しようか、ということは考えられるはずです。自分のやる範囲の中でシステム的に捉えてやるという訓練はできるので、そこからでも良いと思うんです。

松本　最近は将来の機能拡張に容易に対応できるような仕組みにしておくとか、なるべくモジュールの構成にしておいて、取り替えられるようにしていく、そういうことを少しずつ考えるようになってきましたね。

白坂　例えば変化対応をどうしようか、何か変わったときに、これはどうやって変えやすくしておこうか、というような考えがあるのだと思います。システム・オブ・システムズの全体を設計しているわけではなくても、たとえセンサの中の組込みソフトだったとしても、セキュリティも考えなければいけない場面があるかもしれません。うちはここだけだから、と言い切るよりは、ここだけでもシステムとして考えておく、ということをやるべきでしょう。

その辺りが変わっていくとすごく良いと思います。日本の場合は技術力がある。システムズエンジニアリングは、結局はシステムなので、最後は技術になるわけです。技術というか、モノになります。ここがちゃんとできる人が日本はもともと多い。だから、システムズエンジニアリングが強くなれば鬼に金棒なんです。海外では、システムズエンジニアリングのようなシステムとして捉えるやり方を、工学部ではみんな学びます。とくにアメリカはそうですね。ヨーロッパも、ドイツはインダストリ4.0のもとで今どんどんその分野を強めていますが、結局そこを学んでいるかいないかの差がすごく大きい。日本にも学べ

ばできる人はたくさんいます。単に環境として学ぶチャンスが少ないから、できる人が少なくて、活用する人が少ないだけだと思います。

産業界で活躍している人からもっと学ぶ

松本　もう少し産と学が密に連携するようにしていかないといけないでしょうね。先ほどの学び直しのようなチャンスを増やしていくなど、学のほうも産と一緒にやって、製品化のところまで考えていくような、そういうプロジェクトを作っていくことも必要になってくるかもしれないですね。国レベルのテストベッドのようなものを立ち上げることも必要ではないかと個人的には思っています。昔アポロ計画でシステムズエンジニアリングが大躍進した、あのようなものです。

白坂　日本人は昔から人に思いをはせて作ることが得意なはずです。いわゆるシステムズエンジニアリング的な、要求を決めてがちっとやるというだけの話よりも、もう少し人の気持ちや感じ方に踏み込んでいって、例えば先ほどの新しいシステムの社会受容性を上げる設計のような体系ができるはずなんです。それを産業の幾つかのパターンでやって、日本としてどこを強めていけば良いのかということを考えたほうが良いと思います。申し上げた通り、やって学んで、学んでやって、使ってみればできるようになる人はたくさんいて、それを教えられるような人もたくさん出て、そこを起点にして次の世代を生んでいくというような、その最初の世代を生むのはプロジェクトベースでやらざるを得ないんです。そこは国が支援しても良いような気がします。

松本　とくに社会受容性のようなところの検証実験を考えていくべきなのかもしれないですね。単に技術の検証だけだったら、産業界だけでもできる話かもしれません。

白坂　そこを分離してやるんじゃなくて一緒にやる、一緒にどうデザインしていくのか、どう設計していくのかをやっていくようなパイロットにすると良いと思うんです。社会受容性だけを検討しますというグループと、技術をやりますというグループを分けて作ってしまうと、結局はばらばらで、全体を設計する人たちにならないので、全体を設計するような人たちのグループを作って試していく。

松本　正にシステム思考ですね。最後に、今後こういう考え方を進めていく上での課題について、どうお考えですか。

白坂　教えることができる人の絶対数の少なさです。最初のひと回し、最初の教えられる人たちを一度作り上げるというところをどこかで回さない限りは、本当に少しずつしか増えていかない気がしています。1年2年で数人鍛えても広がらないでしょう。

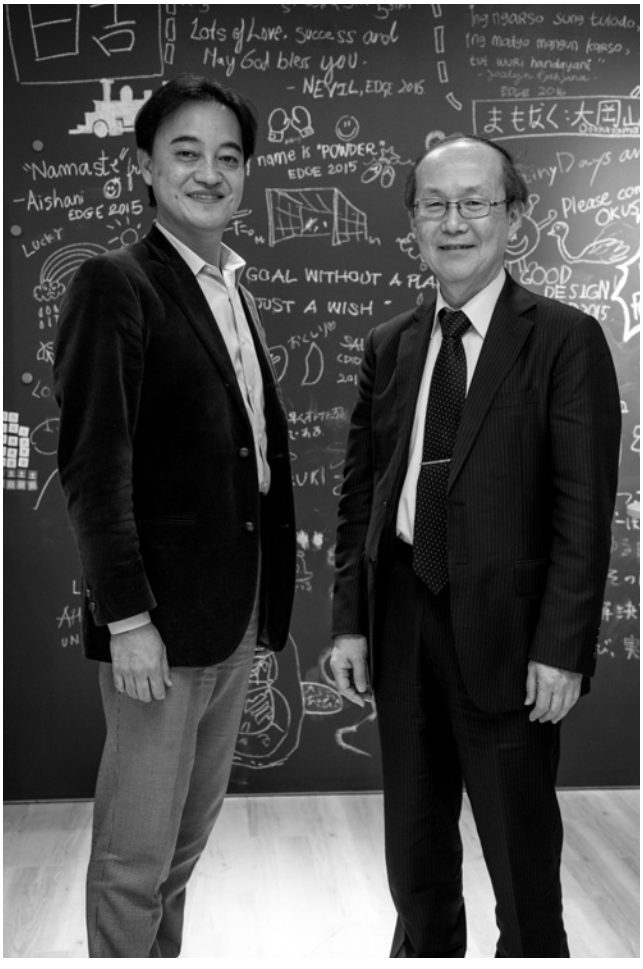
松本　そうですね。

白坂　どこかで数年かけて一回大きく回すということをやれば、その後はそこから生まれた人たちが回していくことができます。教え始め、学んでしまえば、できる人たちは絶対できるというのが今までやってきた私の感触です。

松本　その場合、本当にトップのいわゆるスーパーマン的な人材を育てていくのか、それとも全体をもう少し底上げしていくのか、どちらが良いとお考えですか。

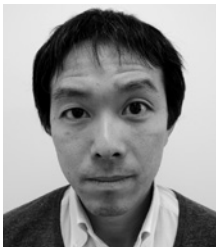
白坂　両方必要だと考えています。教えられる人が増えないと、結局絶対数が増えませんからね。その一方で、これだけではまだまだ足りない、広がらないということも感じています。

松本　そうですね。これだけ急激に変化する時代においては、システム思考をどんどん広めていかなければいけないと思います。ぜひ今後ともお力添えをお願いします。本日は貴重なお話をありがとうございました。



安全性評価に基づき演算量を削減するFail-operational E/Eアーキテクチャ評価手法


大塚 敏史※1


中西 健二※2


櫻井 康平※2

社会インフラにおける制御システムは，制御の高度化を目的とし大規模化を続けている．大規模システムのアーキテクチャを効率良く設計するためには，システムエンジニアリングプロセスに従い，要求を実現する論理アーキテクチャを構築後，複数の物理アーキテクチャ案のトレードオフを考慮して選定し，論理アーキテクチャを物理アーキテクチャに統合する．しかしアーキテクチャの統合には無数の代替案があるため，評価の演算量削減が必要である．本報告では統合時のアーキテクチャ評価効率化を目的とし，Fail-operationalの実現に必要な安全要件の充足判定のモデル化を行い，統合時に自動的に安全性の評価を行うアーキテクチャ評価手法を提案する．提案手法について自動運転システムを例題に評価を行い，アーキテクチャパターンを7,776通りから600通りへと92%を削減可能なことを確認した．

Safety-based E/E Architecture Evaluation Method for Fail-operational Systems to Reduce Calculation Complexity

Satoshi Otsuka※1, Kenji Nakanishi※2, Kohei Sakurai※2

Control systems for infrastructure continue to grow the scale to realize intelligent control. To design system architecture for large-scale control system efficiently, a system architect applies system engineering process to construct logical architecture which satisfies requirements of the system, to select physical architecture with considering tradeoffs, and to synthesize the logical architecture to physical architecture. However, synthesis of logical and physical architecture have numerous patterns. Therefore, the calculation complexity for evaluations of architecture needs to be reduced. In this paper, we propose architecture evaluation method which can evaluate safety automatically by modeling formula of safety requirements for fail-operational systems. Furthermore, the proposed method had been evaluated with an automated driving system and confirmed that the architecture pattern can be reduced by 92% from 7,776 to 600 patterns.

※1 株式会社日立製作所 研究開発グループ　※2 日立オートモティブシステムズ株式会社

1　はじめに

近年，社会インフラにおける制御システムは，制御の高度化や自動化を目的とし，CPS（Cyber Physical Systems）やIoT（Internet of Things）などのサイバー空間との連携や機器間制御による知能化が加速している．社会インフラで用いられる制御システムは，故障による不安全事象への影響が大きく，高度化や自動化が進むほど，故障時も制御を安全に継続する（Fail-operational）ことが重要となる．

制御システムの安全性については電気・電子・プログラマブル電子に関する機能安全規格IEC61508［IEC2010］及び自動車分野における機能安全規格ISO26262［ISO2011］がそれぞれ定められており，安全目標を達成するためのシステム，ハードウェア，ソフトウェアの設計プロセスの定義や，Fail-operationalを実現するハードウェア冗長系のパターンが示されている．

一方で，制御範囲が拡大し大規模化する制御システムの設計においては，システム全体を効率良く設計可能とするシステムエンジニアリングプロセス［INCOSE2015］の適用が必要となる．例えばIEEE1220［IEEE2005］で定義されているシステムエンジニアリングプロセスに従い，要件分析（Requirement Analysis），論理アーキテクチャ分析（Functional Analysis），統合（Synthesis）と，各プロセスでの比較検証を行い，システムの要件を満たす論理及び物理構成を決定する．本プロセスにおいて，前述するシステムの安全性についても評価及び検証が必要となる．

そこで本論文では，システムエンジニアリングプロセスにおけるアーキテクチャ評価時間を削減し，かつ自動運転システムなどFail-operationalが要求される高信頼システムでの安全要件を満たす安全性評価に基づくアーキテクチャ評価手法（Safety-Based Architecture Evaluation Method：SBAE）［Otsuka2016］の検討結果を報告する．本手法では，安全要件の充足判定のモデル化により安全要件の充足を自動的に判定しアーキテクチャを選定する．本手法について自動運転システムを例題としてアーキテクチャ評価の実証を行った．結果として，組み合わせ件数を7,776通りから600通りへと92％削減可能なことを確認した．

2節で本研究の背景，3節では関連研究，4節では提案手法の全体像とFail-operationalの要件を充足するための統合時の安全要件の充足判定方法，5節では本提案手法による自動運転システムの評価結果を述べ，6節にて考察を行い，7節でまとめを述べる．

2　背景

2.1　自動車E/Eシステム動向

自動車のE/E（Electrical and Electronic）システムは，安全性及び快適性の向上や環境負荷低減を目的とした高度な制御を実現するために，その制御範囲を拡大している．とくに，自動運転システムの実現に向け，E/Eシステムの複数の構成要素（センサ，アクチュエータ，コントロールを行うECU（Electronic Control Unit））の連携により車両全体の統合制御を行う必要がある．

自動運転レベル［SAE2016］が向上するにつれ，ドライバが行う運転操作がシステムにより代替され，より利便性が高まる一方で，運転に関する責任がドライバから自動運転システムへと委譲される．とくに自動運転中にドライバによる監視責任がない自動運転レベル3以上では，ハードウェアやソフトウェアに障害が発生した場合でも，不安全な状態を引き起こさないため，Dynamic driving task（動的運転タスク）について，Fallbackによる動作継続（Fail-operational）が要求されている［SAE2016］．

2.2　自動運転E/Eアーキテクチャ設計の課題

IEEE1220で定義されているシステムエンジニアリングプロセスの概要を図1に示す．ここでSynthesisプロセスにおいて，論理（Functional）アーキテクチャが物理（Physical）アーキテクチャに統合される．この際に代案との比較評価を行い，アーキテクチャを決定する．

機能連携が複雑化する自動車システムでは，システムを構成する物理エレメント（演算装置，ネットワークなど）及び論理エ

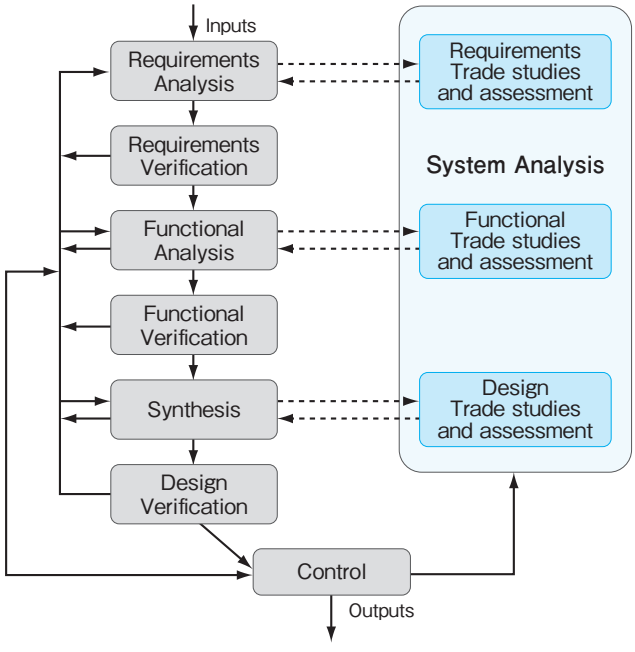


図1　IEEE1220システムエンジニアリングプロセス概要

レメント（論理機能，データリンクなど）の数，また論理エレメントを物理エレメントに対し配置するパターンの組み合わせが増加し，アーキテクチャ決定に必要な評価の組み合わせが増大する．

例えば論理エレメント数がm，物理エレメント数がnの場合，配置の組み合わせはnのm乗となり，それぞれ論理及び物理のエレメント数が10で，評価時間1秒の場合，すべてのアーキテクチャパターンを評価する時間は10の10乗秒となる．

システム設計及び評価は，システム要件や設計の変更など，システムの構成要素が変化するごとに繰り返し行う必要があるため，上記時間の削減を目的とした評価手法が重要となる．

3　アーキテクチャ安全判定の関連研究

アーキテクチャを評価する手法としてATAM（Architecture Tradeoff Analysis Method）[Kazman2000]があり，トレードオフポイントを定めステークホルダと議論を行い，安全性を含めてアーキテクチャを評価する手法を提案している．

自動車E/Eシステムの安全設計・評価手法は，欧州プロジェクトのSAFE（Safe Automotive soFtware architEcture）[Voget2012]があり，アーキテクチャ記述言語のEAST-ADL [Blom2013] を用いた，モデルベース開発によるシステム全体の安全設計・検証方式を提案している．また航空・自動車分野で用いられるアーキテクチャ記述言語のAADL（Architecture Analysis and Design Language）[Feiler2006] を用い，故障の伝播モデルを構築して安全分析を行う手法が提案されている [Delange 2014]．

アーキテクチャの自動最適化を実現するための手法としてHiP-HOPS [Adachi2011] があり，アーキテクチャ及び故障伝播のモデル化を行い，安全要件を検証し，要求コストと信頼度を満たすパレート最適解の導出を行う．

アーキテクチャの最適解を設計空間から探索するため，システムモデルをCSP（Constraint Satisfaction Problems）として定式化し，最適解を導出する研究がある [Li2014]，この中では各種要件（コスト，スケジューラビリティ，リソース）を制約条件として定式化しソルバにより最適解を導出している．

これら研究においては，システムの安全性評価及び検証を行う手法を構築しているが，本研究の課題としているアーキテクチャ統合時の組み合わせ量の削減手法，及び安全要件の充足判定の定式化については述べられていない．そこで本研究ではアーキテクチャ統合において，安全要件の充足判定の定式化により演算量削減を行うことを目的とする．

4　提案手法

4.1　提案手法の全体像

Synthesisプロセスで実施する，安全性評価に基づくアーキテクチャ評価手法の全体構成を図2に示す．提案手法であるSBAEでは高信頼システムで必須となる安全要件の充足を最初に評価して要件を充足しないアーキテクチャを排除し，アーキテクチャ評価全体での演算量の削減を行う．自動運転のように高信頼が要求されるシステムでは，安全要件を満たすことが必須である．そこで，安全要件を満たさないアーキテクチャをトレードオフの評価から排除し，演算量を削減する．

本プロセスでは，最初に物理アーキテクチャのエレメント（演算装置）に対して論理アーキテクチャのエレメント（論理機能）を割り当てる統合を行い，統合を行ったアーキテクチャに対して，安全性評価として後述する安全要件の充足判定を行う．

安全性評価では，機能安全規格ISO26262に倣い安全要件の充足判定を実施する．機能安全規格では，安全機能を実行する物理エレメントに対して信頼性（安全度水準：Safety Integrity Level）が要求される．これはハザード分析（Hazard Analysis）及びリスクアセスメントにより導出される．導出された安全度水準に対し，各エレメントが対応しているか，要求から実装までのトレーサビリティが要求されている．

ISO26262ではASILのレベルごとに異常検出率の要求値が定義されている．論理機能で要求される信頼度を，配置先の物理エレメントが実現可能か否かを判定することにより，前記論理機能を実現する物理エレメントで故障が発生した場合に，十分な信頼度で故障を検出することが可能か否かを評価することが可能となる．

また一方で，単一故障発生時にFail-operationalを実現するためには，機能安全規格に記載のある，共通原因故障（Common Cause Failure）やカスケード故障（Cascading Failure）などの従属故障（Dependent Failure）について，機能及びその情報伝達経路の独立性を確保することが必要となる．

これらより，下記がFail-operationalを実現するアーキテクチャの安全要件の充足判定における必要条件となる．

1. 配置時信頼性
2. 機能独立性
3. 経路独立性

図3を例にアーキテクチャ配置（統合）の例を示す．L_i及びDL_mは論理エレメントであり，それぞれ論理機能と論理機能間をつなぐデータリンクを示しており，P_j及びNW_nは物理エレメ

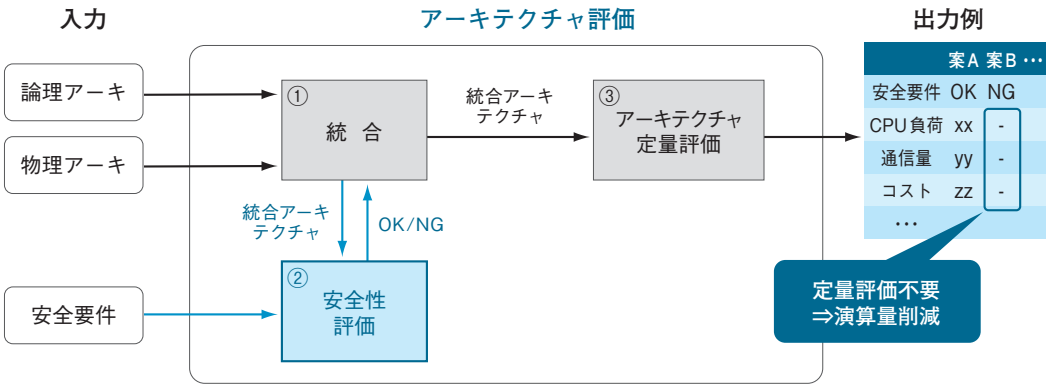


図2　安全性評価に基づくアーキテクチャ評価手法

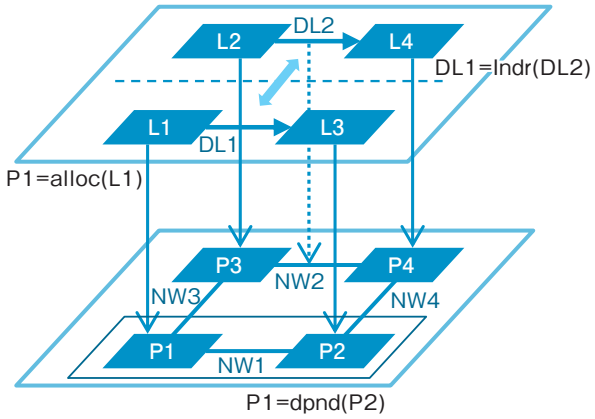


図3　論理アーキテクチャの物理アーキテクチャへの配置（統合）

ントの演算装置とネットワークを示している．

ここではL_1とL_3の連携により実現する機能と，L_2とL_4の連携により実現する機能が冗長系となっており，いずれか一方の機能連携の実行が保証されることにより，Fail-operationalが実現可能となる例を示している．

ここでアーキテクチャ統合における論理アーキテクチャの物理アーキテクチャへの配置について以下のように定義する．ここでは，論理エレメントL_iが物理エレメントP_jに配置されることを下記のように定義する．

$P_j = \mathrm{alloc}(L_i)$・・・(1)

また，論理エレメント（論理機能及びデータリンク）の独立性要求について下記の通り定義する．ここでは，論理エレメントL_iとL_jについて独立性が要求されている場合である．

$L_j = \mathrm{Indr}(L_i)$・・・(2)

また同様に，物理エレメントの依存関係について下記の通り定義する．ここでは物理エレメントP_iとP_jについて依存関係がある場合である．

$P_j = \mathrm{dpnd}(P_i)$・・・(3)

4.2　安全要件の充足判定方法

以下それぞれの安全要件の充足判定方法について説明する．

配置時信頼性

論理機能に割り当てられる要求信頼度に対し，統合後のアーキテクチャが要件を満たしているかを判定する．例えば物理エレメントで実現可能な信頼度に対し，配置された論理エレメントの要求信頼度がより高い場合にはNGとして判定する（図4）．

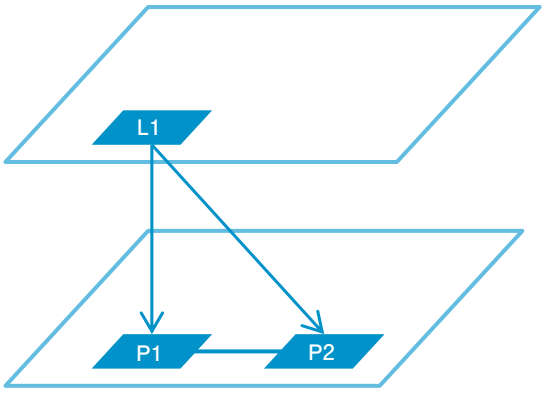


図4　配置時信頼性の判定

判定式はそれぞれ$L_i.\mathrm{asil}$をL_iに要求される信頼度，$P_j.\mathrm{asil}$をP_jが実現可能な信頼度とした場合に，

$(P_j = \mathrm{alloc}(L_i)) \wedge (P_j.\mathrm{asil} \geqq L_i.\mathrm{asil})$・・・(4)

となる．

機能独立性

冗長化を行っている機能が同時に故障することを防ぐため，論理機能間の独立性要求の充足可否を統合アーキテクチャ上で評価する．

図5の例では，論理機能L_1とL_2が独立に動作継続を要求される例を示している．ここで物理エレメントP_1とP_2が例えば同じリソース（電源など）を共有している場合，L_1とL_2を同じリソースを共有している物理エレメントに配置していないことを判定する．

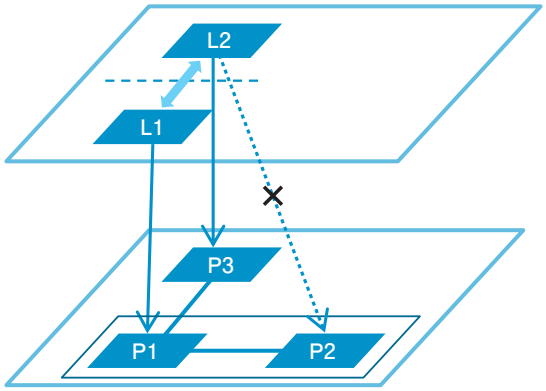


図5　機能独立性の判定

判定式は以下の通りとなる．

$(L_j = indr(L_i)) \wedge (alloc(L_j) \neq dpnd(alloc(L_i)))$・・・(5)

経路独立性

故障発生時にも連携して動作すべき論理機能について，単一故障の発生時でもそれぞれが連携され，動作継続可能か否かを経路の独立性の観点で判定する．

ここではL_1とL_3の連携で動作する機能と，L_2とL_4の連携で動作する機能がそれぞれ独立して動作することを要求される機能である．例えばL_1とL_3の連携で動作する機能が主機能，L_2とL_4の連携で動作する機能が縮退機能である．

この場合に，L_1とL_3の機能連携で使用している物理エレメント（演算装置またはネットワーク）と同じ物理エレメントをL_2とL_4の機能連携で使用した場合，共通部分が単一故障点となる．そのためそれぞれの機能連携で使用する物理エレメントで共通点がないことを判定する（図6）．

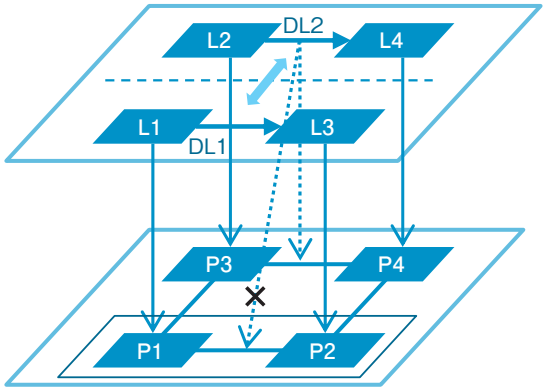


図6　経路独立性の判定

判定式は以下の通りとなる．

$(DL_j = indr(DL_i)) \wedge (alloc(DL_j) \neq dpnd(alloc(DL_i)))$・・・(6)

5　評価

5.1　実験

提案手法について，自動運転システムのアーキテクチャを例題に評価を行った．

評価に用いた論理アーキテクチャについて図7に示す．ここでは主機能を自動運転システムにおける認知・判断・操作とし，安全機能は主機能の診断及び縮退制御の実行，及び通常時制御から縮退制御への切替えを行うとした．

本実験ではFail-operationalを実現する安全コンセプトとして，IEC61508に記載のある1oo2D（1 out of 2 with Diagnostics）を用い，主機能と安全機能の系を独立で有し，それぞれの動作を診断して出力を切替える構成としている．

論理アーキテクチャの各エレメントには，演算量，ROM/RAM使用量など定量評価に必要なパラメータ，機能の要求ASIL，独立性要求の対象（機能独立性が必要な論理エレメントの関係）を付与した．

また評価に用いた物理アーキテクチャについて図8に示す．ここでは各四角は物理エレメントであるコントローラ，コントローラ内部の角丸四角は演算装置を示しており，内部に複数の演算装置があるコントローラも存在する．また青色で示すエレメントは，入力となるセンサまたは出力となるアクチュエータを示しており，それらに論理エレメントは配置せず，本評価の対象となる演算装置は含まないこととした．

各物理エレメントには，パラメータとして，実行可能な演算量や，有しているROM/RAM量など定量評価に必要なパラメータ，対応可能なASIL，物理依存対象（従属故障を引き起こす物理エレメントの関係）を付与した．またネットワークについてはここではピアツーピア型の構成とした．

これら評価モデルの実験諸元について表1に示す．論理アーキテクチャにおける論理エレメントである論理機能数は18，データリンク数は23であり．また物理エレメントである演算装置数は12，ネットワークの数は12である．ここで示す論理機能の数は，図7において，切替えを除く検出から制御までのブロックである．

本評価に適用したアーキテクチャでは，すべての統合パターンを網羅的に作成した場合，演算装置と論理機能のすべての組み合わせは，12の18乗通りである．

しかし評価においては，例えば特定の論理機能を特定の演算

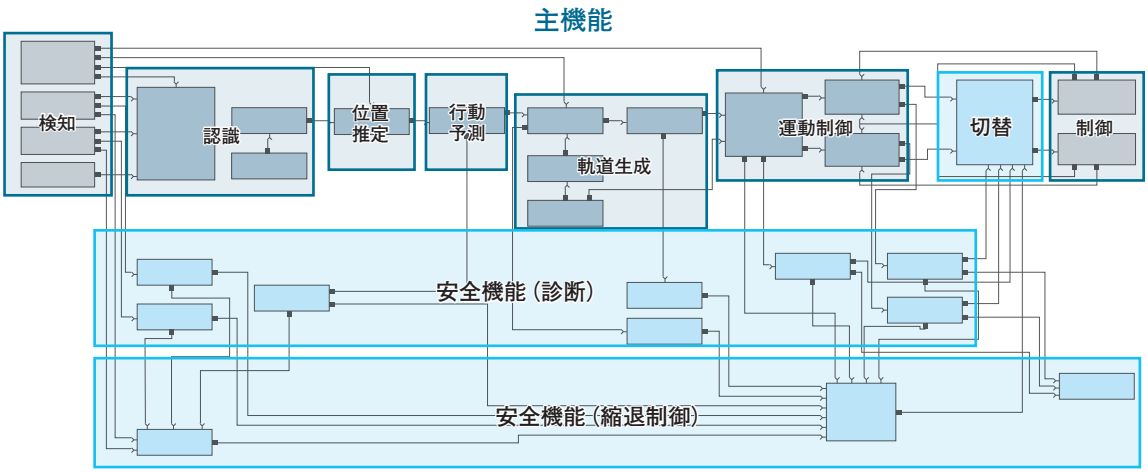


図７　論理アーキテクチャ

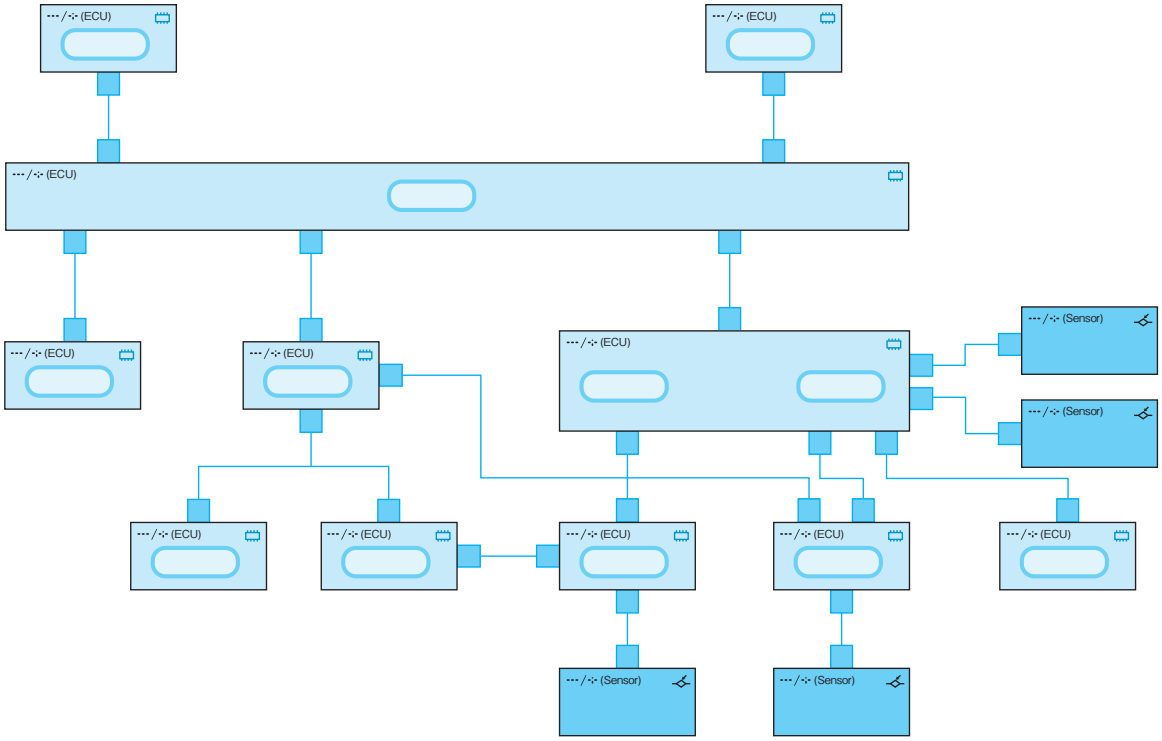


図８　物理アーキテクチャ

装置に固定し，また特定の論理機能を同じ演算装置に配置するようにグループ化を行った．

これは例えばアクチュエータを制御する論理機能について，アクチュエータと演算装置間でネットワークを経由すると要求レイテンシを満たさない場合や，大容量のデータを送受信するため，同じくネットワークを経由するとスループットの要求を満たさない複数の論理機能など，レイテンシやスループットなどあらかじめ与えられている制約について反映して配置の固定化を行い，組み合わせをあらかじめ7,776通りまで削減した．

安全要件については，ASILを付与された主機能の論理エレメント（例えばASILが付与される論理機能），安全機能の論理エレメント（図７の水色枠で示す安全機能），機能連携の論理エレメント数（例えば縮退に用いられる安全機能の論理エレメント）が安全要件の充足判定に使用されるため，その合計数を記載した．

表1　実験諸元

論理エレメント数	論理機能	18
	データリンク	23
物理エレメント数	演算装置	12
	ネットワーク	12
安全要件数		47

5.2　結果

以上のアーキテクチャについて，ツールを用いて，提案手法によりアーキテクチャの評価を行った結果について表2に示す．

ここでは前記記載したグループ化及び配置の固定化により削減した組み合わせの総数が7,776通りに対し，提案手法による安全性評価で排除した組み合わせ数がそれぞれ表2に記載の排除数となり，安全性評価の結果，組み合わせを600通りに削減した．

表2　評価結果

総組み合わせ数	7,776
配置時信頼性判定による排除数	5,184
機能独立性判定による排除数	1,692
経路独立性判定による排除数	300
評価組み合わせ数	600

6　考察

6.1　実験結果に対する分析と考察

提案手法により，アーキテクチャとして評価すべき組み合わせ数について，ツールの自動実行により7,776通りから600通りに削減することが可能になった．

この安全性評価でOKと判定されたアーキテクチャのパターンについては，高度な自動運転システムなど，安全面でのシステム責任が重要となり，かつ故障時も安全性確保のために動作を続ける必要があるシステムについて，機能安全の観点からFail-operationalの必要要件（配置時信頼性，機能独立性，経路独立性）を満たす組み合わせである．これら必須の要件を満たすアーキテクチャの中から，更に定量的なトレードオフ評価を行い，最適なアーキテクチャを選定する．

とくに制御システムでは，アーキテクチャの定量評価においては，システム全体でのレイテンシの要件充足性や，各演算装置及びネットワークにおけるスケジューラビリティなど，一つ一つのアーキテクチャで多くの評価項目が必要となる．

本手法により，定量的評価を実施するアーキテクチャのパターンを削減することは，定量評価での演算量が大きくなるほど，アーキテクチャ評価時間削減の効果は大きくなると考える．

6.2　限界と妥当性への脅威

本提案手法については，独立性の判定について多重故障は想定せず，独立性の判定のみを実施している．これはIEC61508などの機能安全規格でも，N重の同時故障は（独立性を有していれば）同時に発生することは極めて確率が低いと考え，同時故障は考慮外としている．ただし，潜在故障（Latent fault）は検出する必要があるため，潜在故障の検出手段は必要となる．本手法でも潜在故障の検出手段について安全要件の充足判定を実施することにより．安全性確保が可能になる．

また故障に起因しない不具合については本手法の対象外である．ただし大規模システムの場合にはコンポーネントの相互作用による不具合も発生し得ることから，システム理論に基づく事故モデル及び安全分析手法（STAMP/STPA［Leveson2004］）を適用し，その分析結果に基づく安全機能について，本手法を用いて判定することが重要である．

6.3　内的妥当性への脅威

今回の実験では，物理アーキテクチャは安全設計を実施し，冗長系や安全機構を設計したため，安全要件の充足判定によりパターンの排除を行えた可能性がある．本実験は，モデルの作成は自動車システムの知見を有した開発者のレビューを通じて構築したものであり，実開発に合わせた適用という点で信頼性があると考える．

一方で，手法としてはすべてのパターンが安全要件の充足判定によりOKと判定され，排除ができない可能性もある．しかし，パターンが排除できないということは，すべての論理機能の配置が許容されることから，物理設計の信頼度が過剰であることや，冗長性が高いなど，過度に安全な設計がなされていることが想定される．冗長系は信頼性が高まる一方でコストが高くなるため［Armoush2010］，適切な冗長系と信頼度の設計を行い，本手法により可能な配置を探索することが必要であると考える．

6.4　外的妥当性への脅威

提案手法は自動運転システムとして自動車システムへの適用を評価したが，機能安全規格であるIEC61508の観点は各製品分野で同様であるため，他分野への適用も同様に実施可能であると考える．

しかし，System of Systemsなどの事前に全体システムの構成

が決定しないシステムでは，事前に安全要件のすべてが決定されない可能性がある．そのような場合にはConditional Safety Certification［Schneider2013］のように，それぞれのシステムが事前に契約に基づく設計により安全制約を導出し，安全制約を安全要件として判定することにより適用が可能であると考える．

7　おわりに

7.1　まとめ

本報告では安全性評価に基づくアーキテクチャ評価手法を提案し，とくにFail-operationalを実現するための安全要件について，ツールでの自動実行が可能となるように安全要件の充足判定を定式化し，アーキテクチャを評価するための安全性評価手法を構築した．

本提案手法について自動運転システムの例題を用いて評価を行い，アーキテクチャの評価パターン数を，安全性を満たす必要な組み合わせのみを残すことをツールで自動実行可能にすることにより，アーキテクチャ評価の演算量を削減可能な見通しを得た．

7.2　今後の課題

安全性の評価については，機能安全規格にて定められているFTTI（Fault Tolerant Time Interval）など時間制約も制御システムの評価においては重要である．物理アーキテクチャにおける時間制約の判定も含めて，アーキテクチャの安全性を評価可能とすることが今後の課題である．

また安全性評価により600件に削減した組み合わせについて，定量的な評価を用いて更に組み合わせを削減することが必要となる．そのためには，例えば物理アーキテクチャのコストの比較評価や，システムとしてのEnd to Endでの要求レイテンシと実行時間による判定，製品展開を考慮した場合の拡張性の評価により，更にアーキテクチャ候補を絞り込むことについても今後の課題である．

【参考文献】

［Adachi2011］Adachi, Masakazu, et al. “An approach to optimization of fault tolerant architectures using HiP-HOPS.” Software :Practice and Experience 41.11 (2011):1303-1327.

［Armoush2010］Armoush, Ashraf. Design patterns for safety-critical embedded systems. Diss. RWTH Aachen University, 2010.

［Delange 2014］Delange, Julien, et al. AADL fault modeling and analysis within an ARP4761 safety assessment. No. CMU/SEI-2014-TR-020. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2014.

［Feiler2006］Feiler, Peter H., David P. Gluch, and John J. Hudak. The architecture analysis & design language (AADL):An introduction. No. CMU/SEI-2006-TN-011. Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, 2006.

［Blom2013］Blom, Hans, Henrik Lönn, Frank Hagl, et al. :EAST-ADL :An Architecture Description Language for Automotive Software-Intensive Systems. EAST-ADL WhitePaper, Volume 1, 2013.

［IEC2010］IEC 61508 ed2.0, Functional safety of electrical/electronic/programmable electronic safety-related systems, 2010.

［IEEE2005］IEEE Std. 1220-2005 Systems engineering — Application and management of the systems engineering process, 2005.

［INCOSE2015］INCOSE Systems Engineering Handbook :A Guide for System Life Cycle Processes and Activities, fourth edition, 2015.

［ISO2011］ISO International Standard, Road vehicles – Functional safety, ISO Standard 26262, Rev. Nov. 2011.

［Kazman2000］Kazman, Rick. Mark Klein, and Paul Clements. ATAM :Method for architecture evaluation. No. CMU/SEI-2000-TR-004. Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, 2000.

［Leveson2004］Leveson, Nancy. “A new accident model for engineering safer systems.” Safety science 42.4 (2004):237-270.

［Li2014］Li, Shuo. Ahmed Hemani. “Three-dimensional design space exploration for system level synthesis,” in Digital System Design(DSD), 2014 17th Euromicro Conference on, Aug 2014, pp. 419–426.

［Otsuka2016］大塚 敏史, 中西健二, 櫻井康平, 安全性評価に基づくE/Eアーキテクチャ評価手法, 第14回クリティカルソフトウェアワークショップ, 2016.

［SAE2016］SAE International J3016A, Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles, 2016.

［Schneider2013］Schneider, Daniel, and Mario Trapp. “Conditional safety certification of open adaptive systems.” ACM Transactions on Autonomous and Adaptive Systems (TAAS) 8.2 (2013):8.

［Voget2012］Voget, Stefan ;SAFE project presentation ;ARTEMIS Technology Conference on interoperability ;Nuremberg/Germany ;March, 2012.

IoT時代を迎え、従来個別のシステム／サービスだったものがネットワークを通じて連携し、利用者にとって便利で高価値な新しいサービスとして提供される事例が増えている。一方、それに伴ってシステム相互間での複合的な原因による障害も増加しているが、従来型の安全性分析手法では限界があり、新たな安全性分析手法としてSTAMPが提唱され、活用が期待されている。

特集にあたって

SEC調査役　三原 幸博

現在の複雑なシステムは、故障しないソフトウェアや動作に不確定性のある人間を含むサブシステムコンポーネントで構成されており、故障は構成要素間の相互作用で発生するものが中心となる。従って、事前の安全性解析や事故の原因分析には、従来型の「ハードウェア中心」「動作が確定的」「事故は構成要素の故障に帰着できる」ことを前提とした分析手法では限界がある。

こうした背景から、本特集では、新たな事故発生メカニズムに基づくハザード要因解析手法として米国で提案され、欧州を含めて数々の実績を上げている「STAMP（System Theoretic Accident Model and Process：システム理論に基づく事故モデル）」が提唱している安全性解析手法ついて紹介する。この手法は、新たに登場しつつある複雑なシステムに対して、安全性の事前解析・事故原因の分析が可能である。IPA/SECにおいて設置したWGを核に推進してきた普及のための継続的な取り組み内容と、産業界及び学界における先進的な取り組み事例を紹介する。

本特集では、次の記事を掲載している。

会津大学の兼本による「これからの複雑システムの安全分析STAMP/STPA」では、人と高度なソフトウェアが連携・協調して高度な機能を提供する、これからの複雑な工学製品で期待されている安全分析法STAMP/STPAの応用可能性について解説すると共に、故障をなくすのではなく、安全を制御するというパラダイムシフトの重要さを指摘し、複雑システムの安全論証のあり方について解説している。

仙台高等専門学校の岡本、株式会社チェンジビジョンの平鍋による「安全性モデリングとSTAMP/STPA、その最新ツール紹介」では、モデルベース手法一般に必要な機能・性能とSTAMP支援ツールで実現したいコンセプトと機能について、既存の支援ツールとIPA/SECが実現しようとしている支援ツールを対比して解説している。

東日本旅客鉄道株式会社の北村による「JR東日本におけるSTAMP活用の取り組み」では、鉄道における各種信号保安システムが、情報通信技術の採用による技術革新に伴って発生し始めた、装置単体レベルでは説明ができないような複雑な事象に対応するために、新たな安全解析手法STAMPを活用した取り組みについて紹介している。

日本電気株式会社の向山による「エンタープライズ系システムを対象としたSTAMP/STPA分析試行」では、仕様をデータ処理の観点で表現することが一般的であり、制御が明示的ではない業務系システムでも、ハザードにかかわる処理に着目することによりSTAMP/STPAの適用が可能であり、更には業務仕様に欠けていた安全要件の導出まで得られた知見を解説している。

仙台高等専門学校の岡本、信州大学の岡野による「STAMP海外事例の紹介：STPA-SafeSec」では、米国における広域送電網とローカル送電網の接続に関するセキュリティ面のリスク分析をSTPA（STAMP based Process Analysis）の拡張手法である“STPA-SAFETY-SEC”を使って行った事例について解説している。

有人宇宙システム株式会社の野本による「STAMP初心者を卒業する」では、与えられた制御構造をSTAMPのお作法に従って分析するだけでは、安全が実現できない事例として航空機事故を取り上げ、分析から一歩踏み込んで真に安全な制御構造を設計できる能力（システムズエンジニアリング）について解説している。

IPA/SECによる「STAMPワークショップに関する活動報告」では、2017年9月にアイスランドで開催されたEuropean STAMP Workshop2017と同じく11月に東京で開催された第2回STAMPワークショップの模様を中心に最近のイベントを紹介している。

読者の方々には、本特集で紹介した事例などを参考に、各社・組織で開発あるいは運用されているシステムの特性を勘案し、安全性の向上・確保にSTAMPの活用を進めていただくことを期待する。また、IPAで開発してきたSTAMP支援ツール“STAMP Workbench”の公開・無償提供の開始を2018年4月に予定している。併せて活用いただきたい。

これからの複雑システムの安全分析STAMP/STPA

会津大学 名誉教授 **兼本 茂**

これからの複雑な工学製品では、人と高度なソフトウェアが連携・協調して高度な機能を提供することになるが、そこで期待されている新しい安全分析法STAMP/STPAの応用可能性を論じた。故障をなくすのではなく、安全を制御するというパラダイムシフトの重要さを指摘すると共に、複雑システムの安全論証のあり方を議論した。

1 STAMPと安全分析

MITのナンシー・レブソン教授は、その著書「Engineering a Safer World」の中で次世代の安全分析法の必要性を図1のような形で主張している[1]。今の製品開発、システム開発で用いられているFTAやFMEA、HAZOPといった安全分析法は、いずれも50年以上も前に開発された方法論であり、今のコンピューター制御が導入された複雑システムの安全を担保するには不十分であるということである。従来の安全分析法はコンポーネント故障が事故を引き起こすという仮定のもとで、その故障を最小化する信頼性工学に基づいた方法論であるが、複雑システムにおいては、コンポーネント間のコミュニケーション・ミスマッチが事故を引き起こしているという現実を考慮した安全分析が必要になる。

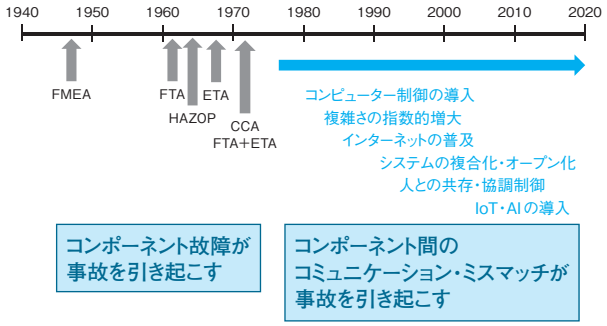


図1　工学製品開発の進展と安全分析時代変化

ナンシー・レブソン教授による講演[2]では、この製品開発の技術変化を、コンピューター制御の導入、複雑さの指数的増大、新技術の導入という3点で象徴しているが、これをもう少し具体的に記述したのが図1である。ここで言う複雑さの増大は、本質的には、そこで使われるソフトウェアの高度化、知能化に起因している。例えば、自動車やロボットのような製品のソフトウェアによる柔軟で知的な制御は、人との協調制御を可能にし、より安全なシステムを作ることができる。しかしながら、その一方で、人とソフトウェア制御のコンフリクトや相互作用のミスマッチが起きて、想定しなかったような事故が起こる可能性もある。

このような背景から、ナンシー・レブソン教授は、次世代の安全分析に用いる基本モデルとして、図2に示すようなSTAMP（System Theoretic Accident Model and Process）を提唱している。安全を制御するために、人やコントローラは被制御対象の挙動を示すプロセスモデルに基づいて、被制御対象に対して能動的な制御行動（Control Action、CA）を提供することで安全を確保する。また、CAの結果はフィードバック情報を通して認識し、その妥当性を評価する。この安全をCAによって確保する、更にはCAの乱れが事故を引き起こすという考え方は、従来のコンポーネント故障が事故を引き起こすという信頼性工学の考え方と本質的に異なるパラダイムシフトであり、福島原発の事故後に注目されているレジリエンス工学とも相通じるものである。人と高度ソフトウェアを内包する複雑システムにおけるエラーやコミュニケーション・ミスマッチは完全には避けることができないものであり、その影響を最小限にとどめたり、更には、最悪の事態を防いだりするためのCAが重要であるというメッセージでもある。

既にソフトウェアを用いた安全の制御は広く用いられるようになっているが、これらの多くは従来のフェイルセーフなどの機械安全技術の置き換えであって、IEC61508などの国際規格

によりその開発プロセスを保証することで信頼性を確保している[3]。一方で、通信ネットワーク、センサ技術とソフトウェアの組み合わせで、大規模化、複雑化、高度化（人工知能など）が始まり、規格の範囲を越えて人間と機械の協調制御によって安全を守る時代にもなっている。中村教授は、それを協調安全と称して、Safety2.0という次世代の安全の考え方を提唱している[4]。「協調」という概念は二つの独立した主体の協力関係を示しており、どちらが主（制御主体）でどちらが従（被制御主体）かという視点でのあいまいさは残るが、ソフトウェアの高度化で状況に応じて主従を切り替えるような柔軟な安全制御機構が可能になると考えられる。しかしながら、このような協調安全によって複雑システムの安全をどのように論証するかはそう簡単ではない。

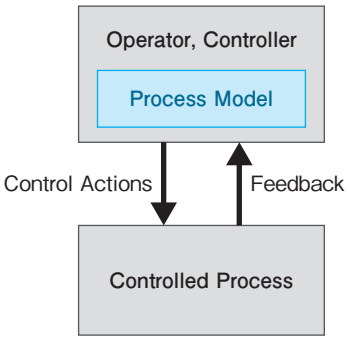


図2　STAMPの基本モデル

現状の安全クリティカルシステムにおけるソフトウェアの安全規格IEC61508やISO26262では、人と機械の協調制御や人工知能のような高度なソフトウェアによる安全の確保は言及されていないが、上述のように、人と高度なソフトウェアを含んだ今後の複雑システムの安全設計の問題にも正面から取り組む必要が出てくると思われる。このような状況の中で、STAMPをどう活用していくかは喫緊の課題と言える。ナンシー・レブソン教授によると、複雑システムの安全は創発特性を持っており、構成要素であるコンポーネントの安全制約を守っていれば、システム全体の安全を守れるものではないと言われている。システムの振る舞いを、個々の要素に分解して説明する信頼性工学的手法、還元主義的手法では複雑システムの安全は確保できないという主張であるが、創発特性をそのままにしておいては、複雑システムの安全な設計はできない。もちろん、工学システムに「絶対安全はない」というのも事実ではあるが、エンジニアとしての知見をより広く集めて、より安全なシステムを作るための方法論を探ることが本稿の目的になる。

2　安全論証としてのSTAMP/STPA

上述のSTAMPに基づく安全設計や事故分析の手順として、ハザード分析法STPA（System Theoretic Process Analysis）や事故分析法CAST（Causal Analysis based on STAMP）といった方法論が提案されている。また、米国、欧州、日本で、STAMPワークショップが定期的に開催され、多くの産業分野での応用事例が報告されている。しかしながら、これからの複雑システムの安全設計プロセスにこのSTPAやCASTをどのように取り込んでいくかはまだ手探りの段階であると言うべきであろう。

これまでの大規模な工学システムの安全設計は、FTA、FMEA、HAZOPのような信頼性工学方法論でなされているが、設計時に想定していない環境変化や運用のされ方、更には仕様書そのものの欠陥によって事故が起こっているのも現実である。事故の後には、事故調査委員会等で原因の究明がなされ、更には、水平展開といった形で対策が取られているが、このような安全管理・対策は、事後に行うという意味でリアクティブな安全論証と呼べる。想定外の大きな地震と津波により引き起こされた福島の原発事故は、その後の安全規制のあり方まで大きく変えたが、これがリアクティブな安全論証の典型と言える。しかしながら、そこには、事故の後に初めて気づいた知識である「後知恵」が必ず含まれる。一方で、もし、先に述べた創発特性を持ったハザードを事前に予測して設計に反映できるとすれば、これは、プロアクティブな安全論証と言うことができよう。そのための具体的な安全論証法として、STAMP/STPAはどのように役立てることができるのであろうか？

IEC61508やISO26262などの国際規格では、安全クリティカルシステムの適用範囲を総合的に定義した後、そこにあるハザードとリスクの評価（Hazard Analysis and Risk Assessment, HARA）を行い、更には、安全整合性水準（SIL/ASIL）を割り当てた上で安全要求事項の仕様を作成して詳細設計に入る。SIL/ASILの評価では、ハザードの深刻度、曝露確率、回避可能性を見積もるといった手順が必要である。これらの一連の評価手順が安全論証と呼ばれるが、人や人工知能のような高度なソフトウェアが入った複雑システムは対象外とされている。ここで期待されているのが、新しい安全分析法としてのSTAMP/STPAである。

図3は、複雑システムの創発的な安全特性を、ナンシー・レブソン教授の考え方に著者の解釈を加えて象徴的に示したものである。個々のコンポーネントの安全制約だけではシステム全体の安全制約に漏れがあり得ること、更に、その周囲に想定外の

環境変化による危険が潜んでいるという二つの重要な論点を示している。STAMPの最大の特徴は、図2に示したような制御主体と被制御主体の相互作用の抽象モデルである。この図は簡単化し過ぎてはいるが、これらを階層的に積み重ねていくことで、複雑システムの安全制御機構を第三者にも理解可能な形で可視化することができる。抽象化と可視化が複雑システムを理解する唯一の方法であるというのがナンシー・レブソン教授の格言でもある。このようなモデルに基づいた分析により、上述の最初の論点である複雑システム全体に網をかけるようなシステム安全制約を導出することができる。この安全制約を網羅的に導出するために、STPAでは、図4に示すような4つの非安全制御行動（UCA）を用いる。これは、Step-1の手順と呼ばれ、UCAごとに、図2のSTAMPモデルを用いて、それがどのようなハザードにつながるかのシナリオを導出していく。これは、従来法との対比で言えば、FMEAに似た推論法になる。ここで安全論証として大事なことは、安全制御行動（CA）とその結果として観測されるフィードバック情報を明示化しておくことである。また、CAの決定に必要なプロセスモデルや外部環境要因も第三者にも分かるような形で表現しておく必要がある。これらの情報があいまいであると評価者によってハザードに至るシナリオが異なってくるためである。このUCAが決まれば、Step-2の手順の中で、UCAを引き起こすハザード誘発要因・シナリオを、STAMPモデルの因果関係を逆向きにたどりながら導出し、更には、それらの要因を防止するためのコンポーネント安全制約を導出していく。

もう一つの論点である想定外の環境変化によるハザードについても、制御構造図のプロセスモデルの中で外部の環境変化も明示化してUCA分析をすることで、複数のレビュアによる相互の議論を行うことができる。想定外というのは、設計に携わるエンジニアの知識や過去の経験に大きく影響されるわけであるが「ある人の想定外は、ほかの人にとっては想定内であり得る」という現実に目を向けた安全設計の相互レビューにより、この想定外をより少なくすることが期待できる。先に述べた東日本大震災の際にも、悲惨な事故にあった福島第一原発と同時に、同じ地震と津波の影響を受けながら、これを乗り越えて安全を

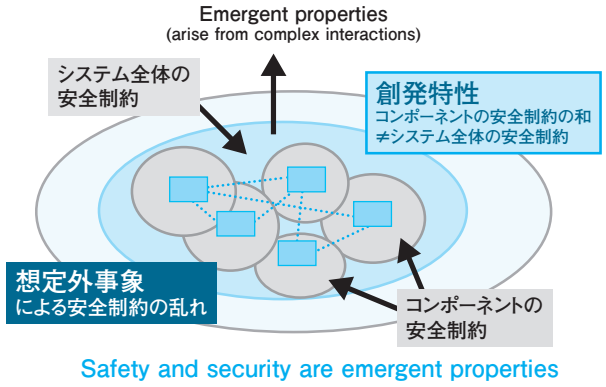


図3　複雑システムの安全と創発特性

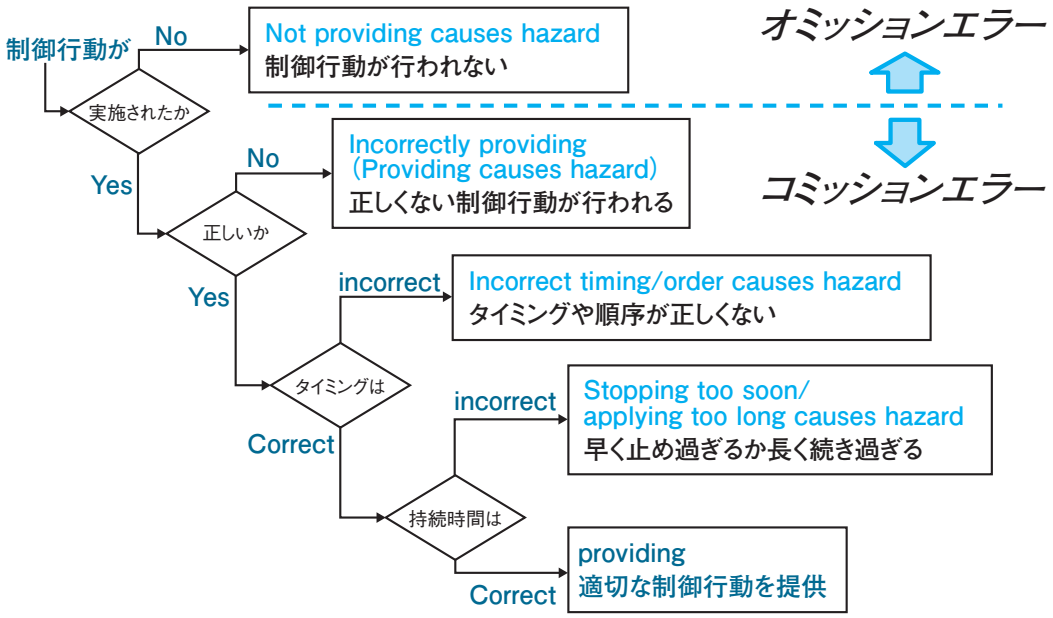


図4　非安全制御行動（UCA）の分類

表1　従来の安全分析法とSTPAによる安全分析法の特徴の比較

手法名	分析方法	特 徴
従来手法 （FTA、FMEA）	●FTAは望ましくない事象をゴールとして、その要因をツリー状に展開・分析して故障要因を分析する ●FMEAは、構成要素に起こり得る故障モードを予測し、考えられる原因やシステム全体への影響を解析・評価する方法 ●システムの構成要素と故障モードが決まるアーキテクチャ設計の段階から適用できる	●構成要素の故障が事故を引き起こすと考え、その故障を最小化する信頼性工学的手法 ●ドミノモデルやスイスチーズモデルといった故障の一方向への伝搬モデルに基づいて、故障の因果関係を分析する ●故障要因に故障確率を割り当てることで定量的な故障分析ができる ●人や複雑なソフトウェアの相互作用を伴う複雑システムの故障分析では、すべての故障モードを拾い出すことが難しい
STAMP/STPA	●アクシデント、ハザード、安全制約に基づき、安全制御構造図を明示化（Step-0） ●４つの非安全制御行動（UCA）に分類してハザードへの影響を分析し、安全制約を詳細化（Step-1） ●非安全制御行動を引き起こすハザードシナリオを、制御構造図と経験に基づくヒントワードを用いて分析（Step-2）	●故障を低減するという信頼性工学的手法ではなく、事故を回避するための制御行動の乱れを分析する安全制御工学的手法 ●安全を確保するための制御行動とそのフィードバックという動的システムの中で事故が起こるというモデルに基づく ●抽象化・階層化したモデルで複雑さを理解するトップダウンのシステム工学的手法で、人・組織や複雑なソフトウェアの相互作用の不具合に伴うハザードを分析する ●安全制御構造が可視化でき、第三者を含む複数の専門家によるレビューがしやすい。抽象化・階層化したモデルなのでドメイン知識の少ない人でもレビューに参加できる ●解析する人の能力への依存度が大きい

保った女川原発や福島第二原発は、想定外へ対処できた良好事例として参考にすべきものであろう。

上述の一連のSTAMP/STPAのハザード分析手順は、従来のHARAと比べると、リスクの定量評価をせずに、ワーストケースで見積もるという違いがあることに気づくが、本質的な違いはないと言えよう。手法としての特徴は、網羅性を持ったUCAという行動に着目して、それがシステム安全に与える影響と、UCAが引き起こされる要因を分析するということである。

以上の議論をもとに、従来の安全分析法とSTAMP/STPAの特徴を比較して表1にまとめておいた。定性的な比較ではあるが、STAMP/STPAの事例を見る際の参考にしていただきたい。

3　まとめ

本稿では、抽象的な表現ではあるが、これからの複雑システムの安全分析のためにSTAMP/STPAをどう活用するかを論じた。STPAは下記の観点から今後の活用が期待されるが、本シリーズの別稿のような具体的な事例で理解を深めていく必要がある。

(1) 抽象化・階層化したSTAMPという安全制御構造モデルでシステムを明示化して、複雑さに惑わされることなく、システムとコンポーネント安全制約を考える。

(2) トップレベル安全制約からトレーサビリティを持ってコンポーネント安全制約を導くことで、運用後の更新まで含めた安全論証に役立てる。

(3) 多様な環境・コンテクストのもとで、第三者レビューも含めて想定外のハザードも発想して安全制約を立てる。また、本稿では論じていないが、複数の制御主体と複数の被制御主体の中で、CA相互のコンフリクトが起こったり、制御主体と被制御主体の逆転が起こったりする際の安全評価にも、可視化された安全制御構造図は役立てることができる。

引用文献

[1] Nancy G. Leveson :Engineering a Safer World :Systems Thinking Applied to Safety (Engineering Systems), The MIT Press, 2012.
[2] SEC特別セミナー: システムベースのエンジニアリング最新動向、2015年6月18日、https://sec.ipa.go.jp/seminar/20150618.html
[3] JASA安全性向上委員会編・著：組込み技術者のための安全設計入門、電波新聞社、2010年
[4] 日経BP「Safety2.0」http://www.nikkeibp.co.jp/atcl/column/16/safety2/

安全性モデリングとSTAMP/STPA、その最新ツール紹介

仙台高等専門学校／IoTシステム安全性向上技術WG委員 **岡本 圭史**　株式会社チェンジビジョン **平鍋 健児**

システム開発は、近年急速に高機能化、複雑化している。更に、IoT、つながる社会の到来によって「相互作用」にも目を向けなければ、ますます安全性の確保が難しくなってきている状況にある。MBSE（Model Based Systems Engineering）が注目されてるのも、そういった俯瞰的な視点が必要となっていることの表れである。本稿では、初めにこの中でのモデルの意味を考察する。次に、現在ハザード分析手法として脚光を浴びているSTAMP/STPA[1]と支援ツールについて紹介する。

1 モデルとは何か

モデルを定義することは難しいが、筆者なりに定義をすると、

現実世界の対象物を、ある目的で捨象し、
その目的下で扱いやすくした抽象物。

ということになる。現実世界は多くの場合複雑である。人間が複雑な問題を扱う場合、着目する「目的」に応じて、不要な情報を意図的に捨て去って（捨象）、本質情報を単純化して表出させる（抽象）することで、その目的に人間の知的活動の焦点を絞ることができるようになる。ここで「抽象」と「捨象」はちょうど作用と反作用のような関係になっている。モデル化に使う言語（表現形式）としては、数式、プログラミング言語、ブロック図、図面、UML/SysMLのようなある程度フォーマルなモデリング言語がある。あるいは、物理的な整形物や木型（モックアップ）などもモデルに含まれるだろう。

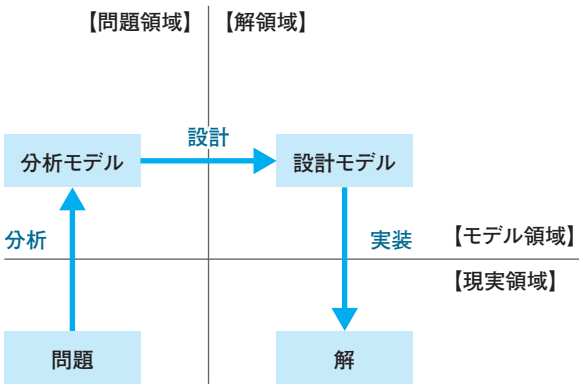


図1　モデルの役割

図1はモデルの役割を描いたものである。問題領域を解領域に実装することを最終成果とする場合、現実の複雑なものを、いったんモデル空間に「抽象化」（そのためにほかを捨象）する。この行為を分析と呼ぶ。抽象・捨象（何を拾って何を捨てるか）は目的によって異なるが、モデルを使うことによって注目する問題がより鮮明になり、設計を導くための導線になる。そして、それをモデル領域でいったん解決したものを設計モデルと言い、それを実装したものが実際の解、すなわちシステムとなる。

1.1 様々なモデリング言語

どのような目的で何を抽象・捨象の対象にするのか、という観点から、モデリング言語は多岐にわたる。とくに専門分野の特化から、現在のシステム開発ではモデルの役割は非常に広くなってきている。以下は、システムを分析・設計する上でよく使われるモデルをカテゴリ化したものである。

自然現象の記述に数学を利用し、解析的・代数的に現象を記述したものが最初のモデリングであろう。ここでの目的は物理量とその予測である。更に、自然現象を人間の役に立つように応用した「エンジニアリング」分野で、とくにコンピューターを利用したモデリングが始まった。機械や電気的な「モノ」をコンピューター上に実現しようとしたCADの歴史は古く、実物でなくコンピューター上で構造物を表現して破壊実験をしたり、シミュレーションをしたり、意匠の確認ができるようになった。その後、ソフトウェア自身もプログラミング言語よりも高い抽象度で（例えばUMLを使って）モデリングされるようになった。ソフトウェア自身のモデリングは、組込みシステムだけでなく企業の情報システム、データベースやワークフローとしても多く利用されている。

制御システムが各産業で重要になり、制御対象となるモノ（プラント）、制御ソフトウェア自身もコンピューター上でモデリングできるようになってきた。そして、エンジニアリング領域を横断的に、システム全体をモデリングしようという流れが生

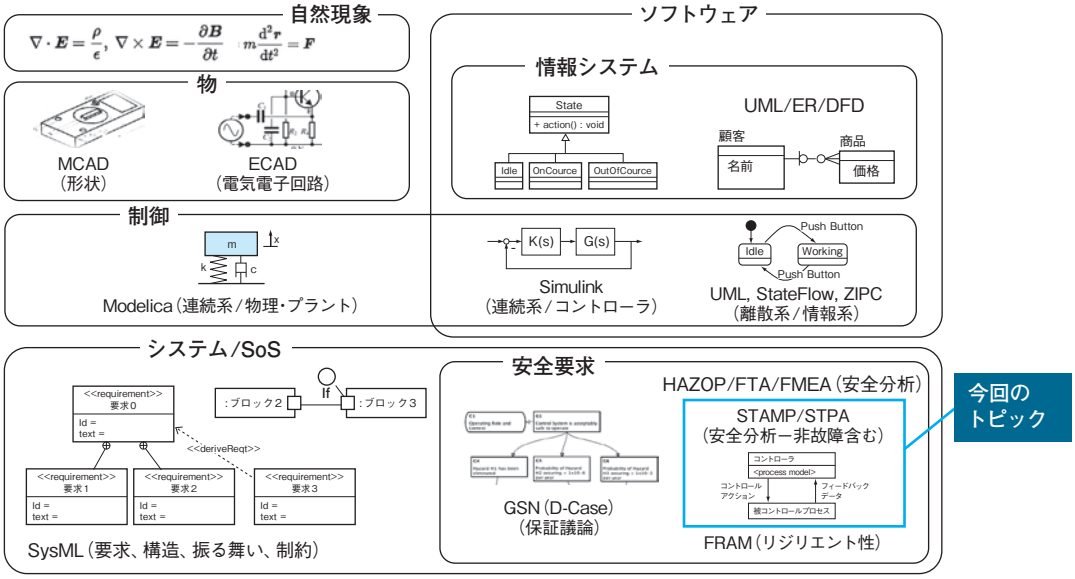


図2　システムのモデリング言語

まれ、MBSE（Model Based Systems Engineering）が注目を集めている。例えば、そのモデリング言語の一つであるSysMLでは、要求、振る舞い、構造、制約の視点から、システムを俯瞰的にモデリングできる。ここまで来ると、モデリングの目的はそのシステムのステークホルダごとに様々になる。構造的な視点、実現におけるコストの視点、要求の充足度の視点、などなど多岐にわたる。それぞれの視点ごとに最適な視点でモデルを提供する必要が出てきた。

その中でも、最近注目されているのが、安全分析の視点であり、中でもこれから開発が加速する、人とソフトウェアを内包した複雑工学システムの安全分析のためのモデリングツールSTAMP/STPAである。

1.2　安全分析のモデリング

従来から、安全分析手法としてFTA、FMEAなどが広く利用されている。FTAは「製品」（トップレベル）の起こり得る故障を想定し、「要素」にブレークダウンすることで原因を分析する。対してFMEAでは、個々の「要素」（ボトムレベル）の故障モードから「製品」の故障を洗い出す。

これらの安全分析では、トップダウン、ボトムアップのアプローチの違いはあれど、最終的には故障が起こる潜在的な状況と個々の要素に注目し、故障の原因を分析してきた。しかし今回紹介するSTAMP/STPAでは、個々の要素が「非故障」であっても安全に影響を及ぼす「要素間」の関係性に注目して分析を行う。

ここでのモデリングの目的は、満たすべき「安全制約」に関連する「コンポーネント」（要素）と、それらをつなぐ「相互作用」を取り出してそれらが形作る「構造」をあぶり出すことである。

このように、現実を抽象・捨象して、システムの構成要素とその相互作用をモデリングし、安全分析を行っていくモデリング手法の一つとして、STAMP/STPAが注目されている。以下では更に詳しく解説をしていく。

2　STAMP/STPAと支援ツールの紹介

本章では、安全分析用モデルとして、従来のアクシデントモデルを拡張した新しいアクシデントモデルであるSTAMP（Systems-Theoretic Accident Model and Processes）[1]とSTAMPに基づくハザード分析手法STPA（System Theoretic Process. Analysis）[1]に注目する。とくに、STAMP/STPAを適用する際のツール支援に着目し、STAMP/STPA適用時の分析結果の様式や課題について述べ、その後、STAMP/STPA支援ツールを紹介する。

STPAはシステマチックなハザード分析手法である。しかし、STPAは幾つかの単純であるが手間のかかる作業を必要とする。例えばコントロールストラクチャー図（Control Structure Diagram、以下CSD）の記述では、コンポーネント間の接続にコントロールアクション（Control Action, 以下CA）を対応させたり、コンポーネント内部にプロセスモデルを記述したりといったSTAMP特有のデータ構造を記述する必要がある。

また、CSD中のCAとstep1で分析するCAの対応付けも必要である。図表の解釈やデータ連携を人間が適切に補うことで、汎用的な図表作成ツールを用いてSTPAを実施できる。しかし、専用ツールを用いて単純な作業を支援することで、分析者は本質的な分析に専念できるようになる。そこで、幾つかの専用ツールが公開されている。

2.1節でSTAMP/STPAを概説し、2.2節で既存のSTAMP/STPA支援ツールを紹介する。更に、2.3節でIPA/SECで開発中のSTAMP/STPA支援ツールを紹介する。

2.1　STAMP/STPA概説

本節では、文献[2]を基にSTAMP/STPAの手順を簡単に解説する。とくに、各ステップでの出力（分析成果物）の典型的なデータ形式を文献[2][3]の例を基に紹介し、各ステップにおける課題

について述べる。これにより、STAMP/STPA支援ツールに求められる機能を洗い出す。

ドローイングツールを用いたCSDの記述、表計算ツールを用いた分析結果表の記述を念頭に各ステップの課題を洗い出す。どちらのツールも大変普及しているという利点があるが、反面STPA支援専用ツールではないためデータ連携が取れず、CSDや分析結果の修正時に生じる派生的な修正は人手により行う必要があるといった課題がある。

2.1.1 Step0準備1

Step0準備1では、アクシデント、ハザード、安全制約を識別する。Step0準備1の入力は要求仕様書やドメイン専門家（の知識・知見）であり、出力はアクシデント、ハザード、安全制約の一覧表（表1）である。

出力の一覧表では、アクシデント、ハザード安全制約間に関係があることを、同じ行に記載することで表している。一般にアクシデント、ハザード、安全制約間の関係は多対多であるため、複数個所に（例えば）同じアクシデントが現れることになる（表1[2]）。そのため人手により表を記述・管理する場合には、修正漏れに注意する必要がある。またSTPAの後工程で、この一覧表中に記載したアクシデント、ハザードと安全制約を参照する場面が多々あるため、これらに番号を付けることは有効である。

2.1.2 Step0準備2

Step0準備2では、コントロールストラクチャーを構築する。Step0準備2の入力は要求仕様などであり、出力はCSDや表形式のコントロールストラクチャーである。図3[2]は、責務とプロセスモデルを加えたCSDである。責務はコンポーネントに対する高抽象度の要求であり、プロセスモデルはCA出力の判断に必要な変数とその値の組である。例えば、列車ドアコントローラは、変数“ドア位置”が値“閉”で、変数“列車位置”が値“プラットフォーム停車中”のとき、ドア開命令を出す。

ドローイングツールでCSDを記述する場合、コンポーネントは単なる四角形である。従って、例えばコントローラには責務とプロセスモデルを追加できるといった、コンポーネント種別による違いは人手により付ける必要がある。また、CSD中のCAとプロセスモデルは次のstep1の出力であるUCA表中で参照される。このような連携は大変手間のかかる作業であり、人手で

表1　アクシデント、ハザード、安全制約の一覧表[2]

アクシデントID	アクシデント（Loss）	ハザードID	ハザード（Hazard）	安全制約ID	安全制約（Safety Constraints）
A1	列車と人・車が踏切内で衝突する	H1	列車が在線中に踏切が閉まらない（警報が鳴らない）	SC1	列車が在線中は踏切が閉まらなければならない
A1	列車と人・車が踏切内で衝突する	H2	踏切遮断後、列車が在線中に踏切が開く（警報が鳴りやむ）	SC2	列車が在線中は踏切が開いてはならない
A2	踏切が開かず、交通が渋滞する	H3	列車が不在なのに踏切が閉まる（警報が鳴りだす）	SC3	列車が不在ならば踏切を閉じない
A2	踏切が開かず、交通が渋滞する	H4	列車が通過したのに踏切が開かない（警報が鳴りやまない）	SC4	列車が通過したら踏切を開ける

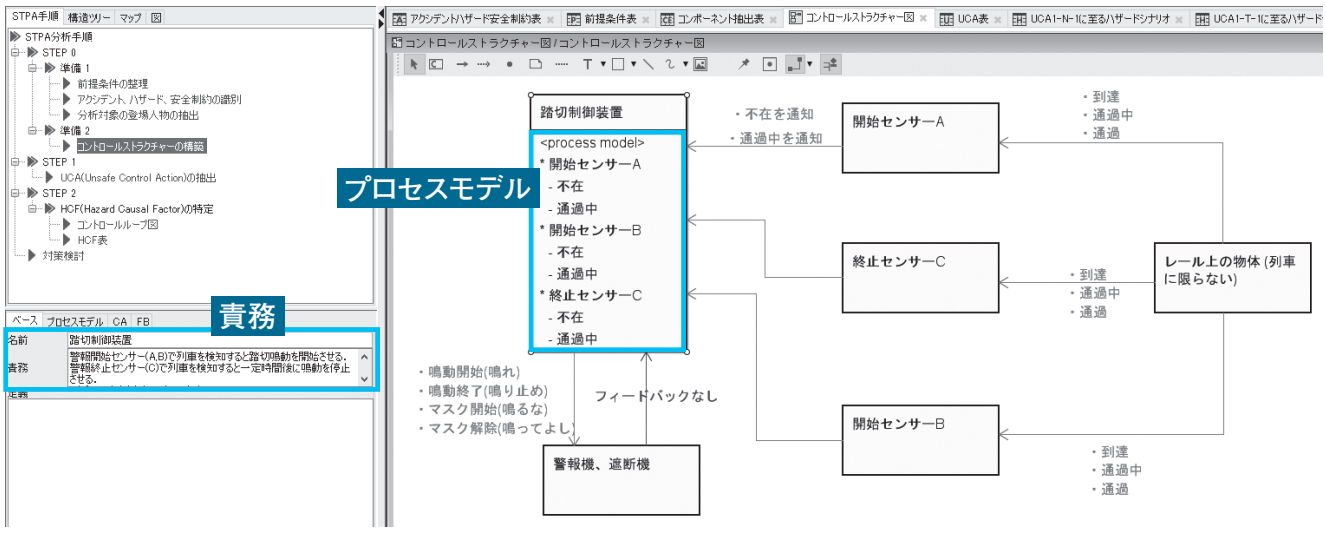


図3　コントロールストラクチャー図[2]

表2　UCA一覧表[2]

No	コントロールアクション	Not Providing	Providing causes hazard	Too early / Too late	Stop too soon / Applying too long
1	鳴動開始	(UCA1-N-1) 警報が鳴らずに列車が踏切を通過する(踏切が閉まらない) [SC1]	列車が来ないのに警報が鳴る	(UCA1-T-1) 警報鳴動する前に列車が踏切に到達する(閉まるのが遅く間に合わない)	開始指示が継続するので、列車通過後に鳴動停止指示が出ても鳴動し続ける
2	鳴動終了	列車が通過後も警報が鳴りやまない	(UCA2-P-1) 列車が通過中に鳴動停止する [SC2]	(UCA2-T-1) 列車が通過完了する前に鳴動停止する(閉めた後、開くのが早すぎる) [SC2]	(UCA2-D-1) 列車通過後も鳴動停止指示が続き、次の列車が来ても鳴動しない(開始指示と競合) [SC1]
3	マスク開始	A, Cを通過した列車がBに到達したときに再鳴動する	(UCA3-P-1) 列車が来ないのにマスク指示し、警報鳴動しない [SC1] (UCA3-P-2) 反対側の開始センサにマスク指示し、警報鳴動しない [SC1]	(UCA3-T-1) 終止センサへのマスク指示が遅れ、列車の当該センサ通過に間に合わないと、マスク指示が残り、対向列車が2本続いたときに警報鳴動しない [SC1]	(UCA3-D-1) 列車が反対側の開始センサ通過後までマスク指示し続けると、対向列車が来ても鳴動しない [SC1]
4	マスク解除	(UCA4-N-1) 反対側の開始センサにマスク解除指示が出ず、対向列車が来ても鳴動しない(マスク指示後に列車が引き返す場合を含む) [SC1]	警報が再鳴動する	列車がBを通過完了前に出ると再鳴動する	解除が後続列車によるマスク開始指示と競合するとマスクされずに再鳴動する可能性がある

連携する場合には参照関係が壊れがちである。

分析対象は一般に大変複雑なので、CSDを抽象化することや、逆にCSDの一部を詳細化することは、分析対象のコントロールを理解するのに役立つ。他方、CSD構築は適切な抽象度にするためのコンポーネント粒度統一作業を含むため、変更と修正を繰り返すことが多い。この修正作業により、ドローイングツールにより記述する場合には、高抽象度のCSDとその一部を詳細化したCSDの整合性を保つことは大変手間のかかる、かつ間違いやすい作業となる。

2.1.3　Step1

Step1では、UCA(Unsafe Control Action、非安全制御動作)を抽出する。Step1の入力はUCAを導き出すための(STPAが提供する)4つのガイドワード、step0準備1出力のアクシデント、ハザード、安全制約の一覧表、step0準備2出力のCSDであり、出力はUCA一覧表である(表2)。

UCA一覧表の各行はそれぞれCSD中のCAに対応し、各列は4つのガイドワードに対応し、各セルには対応するCAとガイドワードを組み合わせた際に(もし至るならば)UCAへ至る条件と対応するハザードや安全制約が記載される。

表計算ツールを用いてUCA一覧表を作成する場合には、CSD中からCAを抜き出す際の漏れや、CSD変更に伴うUCA一覧表への影響反映忘れに注意する必要がある。また、UCA一覧表中のUCAはstep2で参照するため、番号を付与すると参照が容易になる。

2.1.4　Step2

Step2では、HCF(Hazard Causal Factor、ハザード誘発要因)を特定する。Step2の入力は、HCF特定のためのヒント(抽象化されたハザード誘発要因の例)、step0準備2で構築したCSDとstep1で作成したUCA一覧表である。またstep2の出力は、ハザード要因の一覧表(表3)とハザードシナリオである。表3のハザード要因の一覧表は、step1で識別したUCAごとに作成される。このハザード要因の一覧表の各行は一つのHCFに対応し、対応するHCFヒントとハザードシナリオを記載する。更に各行には複数のハザードシナリオを記載できる。

[2]では、表4の形式のハザード要因の一覧表を紹介している。表4の各行は一つのUCFであり、各列はHCFヒントワードで、対応するセルにHCFまたはシナリオを記載している。複数のUCAに対する表3のデータをまとめ、適切な形式に変換することで、表4が得られる。このような一覧表を用いることで、HCFの抜け、すなわち、その網羅性の確認に役立てることができる。

表3、4では、ハザード要因をハザード要因の一覧表で、ハザードシナリオを文章で記述している。これ以外にも、ハザード要因はリスト形式で、ハザードシナリオはアノテーション付きコントロールループ図(Control Loop Diagram、以下CLD)で、それぞれ記述されることもある。

表計算ツールでハザード要因の一覧表を作成する場合には、UCA一覧表中のUCAとハザード要因の一覧表のUCAの対応付けは人手で行う必要があり、漏れや修正による影響の反映といった点に注意する必要がある。また、ハザード要因の一覧表とシナリオを独立して記述する場合には、それらの対応付けも必要となる。

HCFは、コントローラがソフトウェアか人間かといった条件や分析対象のドメインにより異なる。従って、HCFのヒントを適切に変更・提供することで、HCF発想が容易になる。

STAMP/STPA支援ツールSafetyHAT[4]ではドメインに特化したヒントを提供しており、ユーザによるヒントの編集も可能である。

2.2　STAMP/STPA支援ツール

本節では、既存のSTAMP/STPA支援ツールを紹介する。またSTAMP/STPAの専用支援ツールを紹介する前に、他ツールによるSTAMP/STPA支援について紹介する。文献[5]や[6]ではSTAMPベースのツールとして、モデル検査連携を含む多彩な拡張機能を持つXSTAMPP[7]、データベース連携やガイドワード編集可能なSafetyHAT[4]、トーマス博士が提案した拡張STPA[8]をサポートするan STPA tool[9]、Enterprise Architectの拡張であるSHARA[9]などが紹介されている。

表3　ハザード要因の一覧表

ID	HCF	ヒントワード	シナリオ
HCF1-N-1-1	踏切通過後に引き返す列車向け制御が不適切	(8) 不適切、有効でない欠けたコントロールアクション	Aから来た列車がCを通過した後、連結を切り離して、後部車両がA方向に引き返す Aから来た列車がCを通過した後、A方向に引き返す
HCF1-N-1-2	鳴動停止継続により次の鳴動指示と競合	(8) 不適切、有効でない欠けたコントロールアクション	Aから来た列車がCを通過してBをマスクした後、BとCの中間で停止。救援列車が反対方向から侵入してセンサBを通過。A方向に進行する
HCF1-N-1-3	センサAが故障してAから踏切制御装置への通知が欠落	(9) プロセスへの入力が欠けているか間違っている	センサAが故障してAから踏切制御装置への通知が全く届かない センサAが不電導物（葉っぱなど）に覆われて、車輪経由の検知電流が流れず、列車到達を検知できない

表4　ハザード要因の一覧表 [2]

	❶上位からの指示や外部情報の誤り・欠落	❷Control actionが不適切・無効・欠落	❸動作の遅れ	❹プロセスへの入力の誤り・欠落	❺意図しない、または範囲外の外乱	❻不十分な制御・アルゴリズム
(UCA1) 警報が鳴らずに列車が踏切を通過（踏切が閉まらない）		• 踏切通過後に引き返す列車向け制御が不適切 • 鳴動停止継続により次の鳴動指示と競合		• センサAが故障してAから踏切制御装置への通知が欠落		
(UCA2) 鳴動前に列車が踏切に到達（閉まるのが遅い）			• センサAが故障してAから踏切制御装置への通知が欠落			• センサAが故障してAから踏切制御装置への通知が欠落
(UCA3) 列車が踏切を通過する前に鳴動停止（開くのが早い）					• 列車がAを通過後、踏切に到達する前に、Cが外乱により短絡する	
(UCA4) 不正なマスク開始指示が出て、列車が来ても警報鳴動しない		• 踏切制御装置の状態管理が不適切				• 踏切制御装置の状態管理が不適切
(UCA4) 不正なマスク開始指示が出て、列車が来ても警報鳴動しない			• 超高速列車に対応できずにマスク解除の指示遅れ	• レール上の物体による外乱		• 制御装置の処理に問題があり、マスク解除の指示遅れ
(UCA6) マスク開始指示漏れ	• 誤った外部入力（外乱）でマスク解除漏れ	• 制御装置の処理遅れでマスク解除漏れ	• 状態制御誤りでマスク解除漏れ	• 不正な外部入力によりマスク解除漏れ		• 非定常運行への対策漏れでマスク解除漏れ

2.2.1　XSTAMPP

eXtensible STAMP Platform（XSTAMPP）は、A-STPAのスタンドアロン版の後継機以外にも、多くのプラグインを含み、基本的なSTPA以外にも多くのSTMPベース分析や開発連携を可能にしている。具体的には、XSTAMPPは、A-STPA以外に、A-CAST（CAST用）、XSTPA（トーマス博士の拡張STPA用）、STPASec（STPA for Security用）、STPAPriv（STPA for Privacy用）、STPA Verifier（モデル検査とSTPAの連携）、STPA Safety-based Test Cases Generatorといったプラグインを含む。

XSTAMPPでは、STAMP/STPAの構成要素（コンポーネント、CAなど）が用意されており、各ステップ出力中に記述されるこれらの構成要素は関連付けられている。具体的には、step0準備2においてCSDを構築する際には、CSDの構成要素（コントローラ、アクチュエータなど）を選択肢から選び、CSD中に追加できる。また、コンポーネント間の接続や接続に付随するCAやフィードバックも選択肢から選び、CSD中に追加できる。追加されたCAは、step1のUCA一覧表中に記載されるCAと連動する。しかし、CSDのコンポーネント中にサブコンポーネントを記述するといった階層的記述は対応していない。

2.2.2　SafetyHAT

A Transportation System Safety Hazard Analysis Tool（SafetyHAT）[4]は、その名が示す通り交通機関の安全分析に重点を置いたツールであり、交通機関に特化した4つのstep1ガイドワードとstep2のHCFのヒントを提供している。更に、コンポーネントタイプやガイドワードをユーザがカスタマイズ可能になっている。また、コントロールストラクチャーの記述は図形式ではなく、表形式を採用している。ただし、別途ユーザが作成した図形式のコントロールストラクチャーとのリンクは可能である。他方、データベースとの連携やエクセル形式で分析結果を出力することも可能である。

2.3　STAMP Workbench

本節では、IPA/SECが開発しているSTAMP/STPA支援ツールSTAMP Workbenchについて紹介する。なお、STAMP Workbenchは本稿執筆時点では評価版であるため、公開版との差がある可能性があることには留意いただきたい。

STAMP Workbenchの目的は、基本的な清書機能、分析手順ガイド機能に加え、分析者が分析に注力できるよう支援する機能を提供することである。例えばSTAMP Workbenchは、反復しながら分析を進める際に発生する手間（図表変更とその変更が引き

起こす修正作業など）から分析者を解放することを目指している。

STAMP Workbenchはモデルベース開発で使用されるツールが持つ機能に加え、以下の5つの機能（図4）を持つよう設計されている。1）STAMP図生成機能（STAMPで使用する図形及びSTPA基本手順支援機能で使用する図形を生成する機能）、2）STAMP表生成機能（STAMPで一般的に使用する表及びSTPA基本手順支援機能で使用する表を生成する機能）、3）汎用形式データ出力機能（STAMP Workbenchが保持する分析データを、汎用的な形式で出力する機能）、4）外部ツール連携I/F（STAMP Workbenchと他ツールの間で片方向あるいは双方向に入出力データを受け渡すためのインターフェース）、5）STPA基本手順支援機能。

STAMP Workbenchの特徴的機能を解説する。STAMP Workbenchは、STPAの各stepでの標準的出力をサポートしている。実際2.1節においてSTPAの各stepの出力を紹介したが、アクシデント、ハザード、安全制約の一覧表（表1）、コントロールストラクチャー図（図3）、UCA一覧表（表2）、ハザード要因の一覧表（表3）は、STAMP Workbenchを用いて作成している。これらの基本的機能に加え、表形式で整理したデータからCSDを生成する機能、step2のHCFヒントを選択・編集する機能、UCAやHCFなどへ自動採番する機能を持つ。

CSD生成機能について解説する。Step0準備2では、コントロールストラクチャーを要求仕様などの利用可能な資料から構築する、しかし一般的なモデル構築と同様に、コントロールストラクチャー構築は一般に困難であり、試行錯誤が要求されることが多い。そこでSTAMP Workbenchは、SafetyHATと同様に、表により情報を整理し、整理された情報からCSDを生成する機能を持つ（図5）。また、直接CSDを記述できるようにも設計されている。

Step2のHCFヒントの選択・編集機能について解説する。Step2では、ハザード要因の一覧表とハザードシナリオを作成する。STAMP Workbenchでは、HCFのヒントはユーザが既存のヒントワードセットから選択可能であり、更に各ヒントワードセットはユーザが編集可能である（図6）。この機能により、分析対象に適したHCFのヒントを与えられ、HCFを特定しやすくなる。

自動採番機能について解説する。STPA各stepの表中の分析結果（UCA、HCFなど）に番号を付けることで、参照が容易になる。しかし、分析を行う中で過去の分析結果に追加・修正することもあり、それに伴い番号の変更を余儀なくされることがある。STAMP Workbenchは、分析結果中のデータに自動的に番号付け（番号変更）を行う機能を持つ。

2.4　2章のまとめ

本章では、STAMP/STPA支援ツールを紹介した。2.1節では、STAMP/STPAの手順、各stepでの出力形式や課題について解説し、STAMP/STPA支援ツールに求められる機能を挙げた。2.2節では、STAMP/STPA支援ツールの紹介として、XSTAMPPとSafetyHATを紹介した。2.3節では、IPA/SECで開発中のSTAMP/STPA支援ツールSTAMP Workbenchについて、その目標と執筆時点での機能を紹介した。

IPA/SECで開発中のツールSTAMP Workbenchは、IPA/SECのワーキンググループでの事例検討の実績をもとに開発されている。例えば、データの修正による番号振り替えなどの作業を自動化して思考を妨げないこと、内部データの論理構造の一貫性を保ち、トレーサビリティを容易に確保できること、データのエクスポート機能を持たせユーザニーズに応じたカスタマイズが容易にできること、といった特徴を持たせている。今後、STAMP Workbenchの現場での活用が期待される。

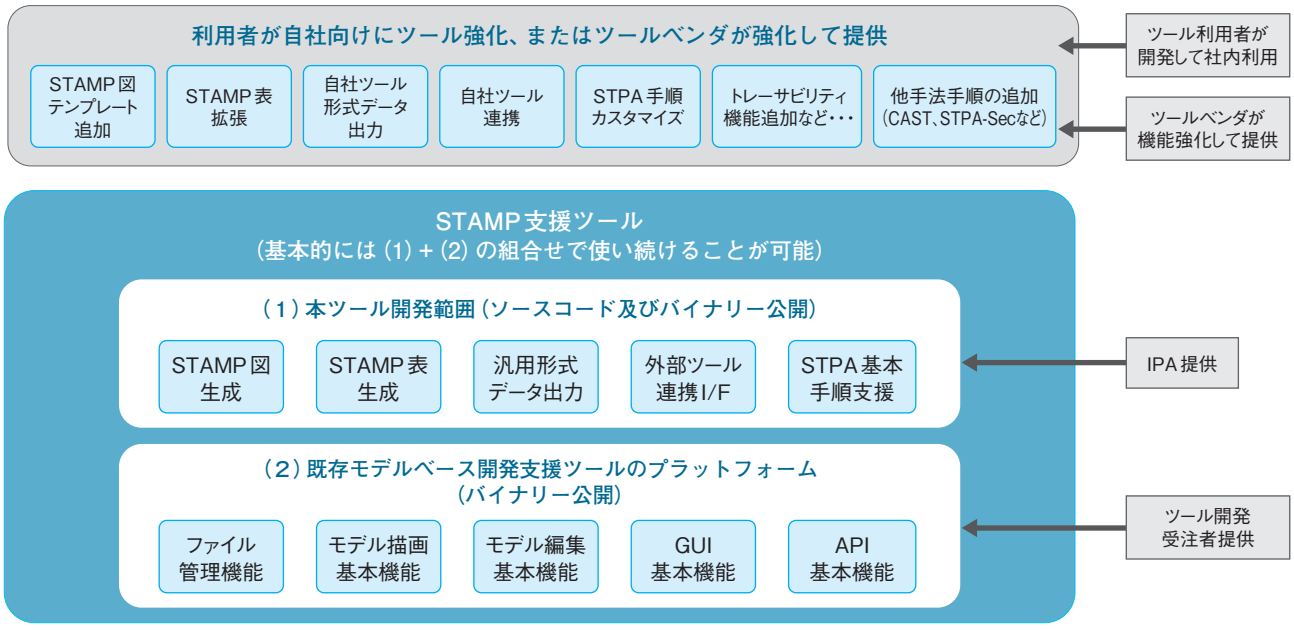


図4　STAMP Workbenchの構造

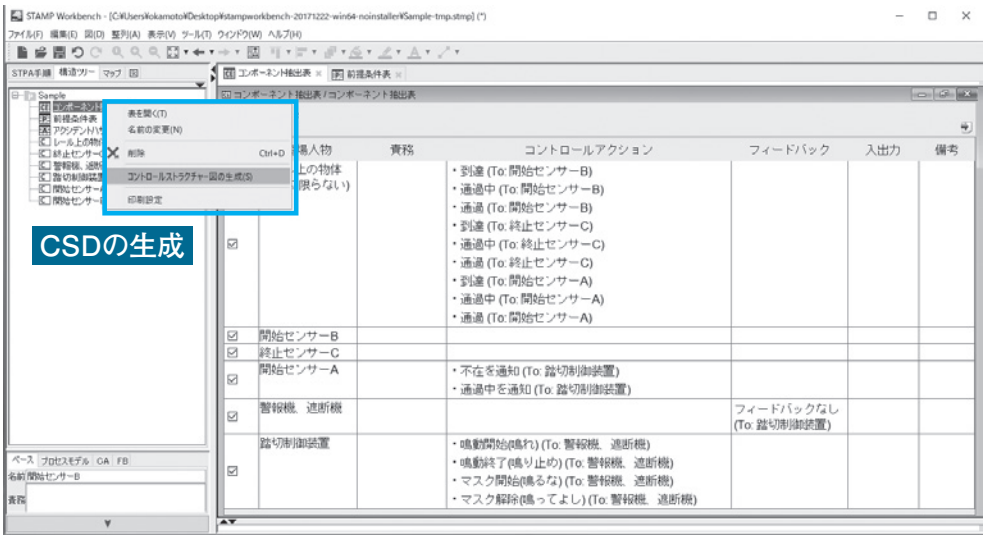


図5　コントロールストラクチャー図生成機能

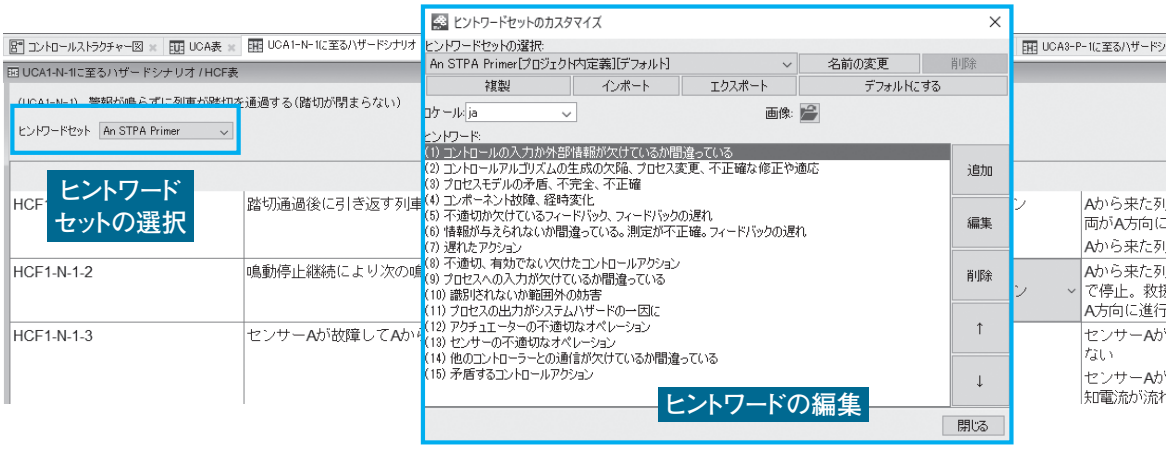


図6　HCFヒントの選択・編集機能

【参照文献】

[1] Nancy G. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety, MIT Press, 2011.

[2] IPA, はじめてのSTAMP/STPA入門編, 2016.

[3] Nancy G.Leveson, John Thomas, “A STPA Primer,” 2013.

[4] “SafetyHAT：A Transportation System Safety Hazard Analysis Tool,” https://www.volpe.dot.gov/infrastructure-systems-and-technology/advanced-vehicle-technology/safetyhat-transportation-system

[5] “Partnership for a Systems Approach to Safety (PSAS) web pages,” https://psas.scripts.mit.edu/home/

[6] Sven Stefan Krauss, Martin Rejzek, Christian Hilbes, “Tool qualification considerations for tools supporting STPA”, Procedia Engineering, Volume 128, 2015, Pages 15-24.

[7] Asim Abdulkhaleq, “XSTAMPP For Safety Engineering of Software Intensive Systems”, http://www.xstampp.de/

[8] John Thomas, “EXTENDING AND AUTOMATING A SYSTEMS-THEORETIC HAZARD ANALYSIS FOR REQUIREMENTS GENERATION AND ANALYSIS,” Ph.D. Thesis, 2013.

[9] John Thomas, Dajiang Suo, “A Tool-Based STPA Process”, 2015.

JR東日本におけるSTAMP活用の取り組み

IPA/SEC IoTシステム安全性向上技術WG委員／東日本旅客鉄道株式会社　北村 知

JR東日本では、信号保安システムのソフトウェア化が進み、従来の手法による安全性分析が難しくなっていることから、宇宙開発分野で実績のあるSTAMP活用の取り組みを始めている。
信号保安システムへのSTAMPの活用事例として、STAMP/STPAを用いたシステム設計段階での安全性分析と、STAMP/CASTを用いた過去の障害事例からの安全要求抽出に関する事例を紹介する。

1　信号保安システムの現状

鉄道では、連動装置※1や踏切保安装置※2といった各種信号保安システムにより安全が確保されている。これらの装置は、従来はリレーを用いたシーケンス回路の論理により安全性を確保してきたが、近年では情報通信技術の進展と共に、これらの装置についてもソフトウェアを用いた電子制御装置に置き換わっている。

このように、近年では信号保安システムにおけるソフトウェアの割合が拡大している。このため、システム設計段階での安全性分析として、これまでのFTAやFMEAといった手法の適用が難しくなっている。なぜなら、ソフトウェアはハードウェアのように故障するわけではなく、開発時の仕様や製作プロセスでの不備がシステム不具合につながるため、従来の安全性分析手法だけでは限界があるからである。

また、発生する不具合も、装置単体レベルでは説明ができないような複雑な事象が多くなっている。ソフトウェアで構成される装置特有の、新たな不具合の発生を考慮しなければならない。

この状況を踏まえ、JR東日本では、2013年度から、宇宙開発分野で先行実績のあるSTAMPの信号保安システムへの応用について検討を始めた。

2　信号保安システム開発とSTAMP/STPA

信号保安システムには、連動装置や踏切保安装置など様々な装置があるが、ここでは踏切保安装置を題材にして、STAMP/STPA手法の信号保安システムへの適用例について述べる。

2.1　踏切保安装置の構成

踏切保安装置は、現地の状況や導入する鉄道事業者によって異なるものの、当社においてはおおむね図2-1のような装置で構成されている。

踏切保安装置の動作ロジックは、おおむね次の通りである。

- 列車が「始動点制御子」と呼ばれるレールに設置したセンサに進入すると、踏切が列車の接近を検知する。
- 列車の接近を検知した踏切は、警報機を鳴らし、しゃ断機を閉める。
- 踏切を通過した列車が「終止点制御子」と呼ばれるセンサを通過し終わると、警報機が鳴動を停止し、しゃ断機を開放する。

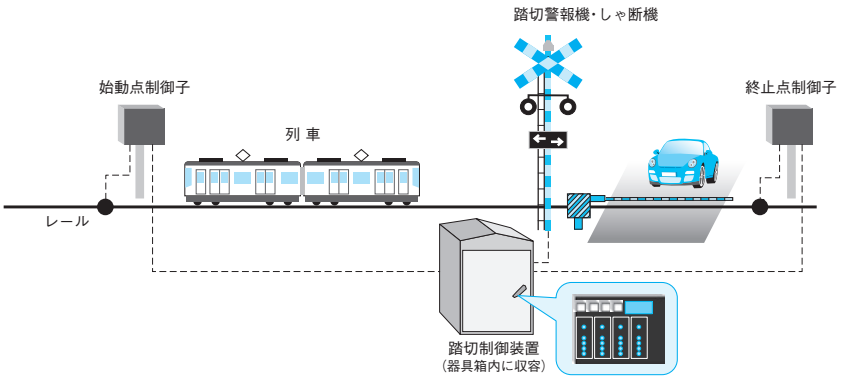


図2-1　踏切保安装置の概要

※1　駅構内の転てつ機などの設備と信号機を相互に関連付けて、列車運行の安全を担保する装置
※2　列車が踏切に接近したことを検知し、警報機やしゃ断機を作動させることで事故防止を図る装置

なお、駅構内※3の踏切の場合は、ポイントによる進路の分岐方向に応じて踏切制御の要否を識別しなければならないことから、「始動点制御子」ではなく、駅の一定区間ごとに設けた軌道回路※4と呼ばれるセンサと構成された進路の条件により、列車の接近を検知する場合が多い。

このように、駅間の踏切は比較的シンプルに構成されている一方で、駅構内の踏切については、走行する列車の進路が多岐にわたることから、その踏切内部の処理も複雑になる。

また、装置自体が故障した場合は、安全側すなわち踏切を警報するように動作しなければならない。万が一、列車が通過しても踏切が動作しない事象を「無しゃ断」と呼び、これは非常に危険な事象であるため発生させてはならない。

2.2 電子踏切保安装置の開発における STAMP/STPA分析の活用

踏切保安装置の電子化は1986年頃から行われているが、一部を除き、論理がシンプルな駅間の踏切に限られている。駅構内の条件を電子化した踏切（以下、構内電子踏切）は、多様な列車の動きに対応させるために制御論理が複雑となり、現状では本格的な導入には至っていない。

本節では、現在開発を進めている新たな構内電子踏切の制御機能にSTAMP/STPA手法を適用し、開発段階での制御論理における安全要件の抽出を行った事例を紹介する。

2.2.1 ハザード制御にかかわるControl Structureの作成（Step 0）

構内電子踏切へのSTAMP/STPA手法の適用にあたり、ハザードを以下の通りに定義した。

■安全上問題となるハザード

H1：列車が在線で踏切がしゃ断しない

H2：踏切がしゃ断後に列車が在線にもかかわらず開く

H3：警報時間の不足（法令への抵触）

■運用性の低下につながるハザード

H4：警報時間の過剰

H5：列車が非在線で踏切がしゃ断する

ただし、今回は「安全上問題となるハザード」を優先的に分析するために、ハザード定義を限定し、H1からH3のみを対象とすることとした

次に、検討中の仕様案に基づき、図2-2に示す構内踏切全体のControl Structure Diagramを作成した。

ここで、本分析の対象としている構内電子踏切は、踏切本体を制御する論理装置「構内LC」を中心に、列車位置を検知する軌道回路や、列車の進路を構成する信号機・転てつ機などの条件を入力していることに留意していただきたい。

図2-2のControl Structure上の、Control Actionの実行条件と実行内容の整理を行った。

構内LCから踏切本体への制御電文のうち、踏切警報開始／終止の命令を意味する「SR落下／扛上」※5を分析対象のControl Actionとして、次のStep 1の対象とすることとした。

2.2.2 非安全なControl Actionの識別による ハザードシナリオの分析（Step 1）

Control Actionごとに、4つのガイドワードを適用して、表2-1に示すようにハザードにつながる非安全なControl Actionを分析した。

非安全なControl Actionによるハザードシナリオを表2-2に示す通り整理した。H1からH3に関係するハザードシナリオに絞り込み、次のStep 2の対象とすることとした。

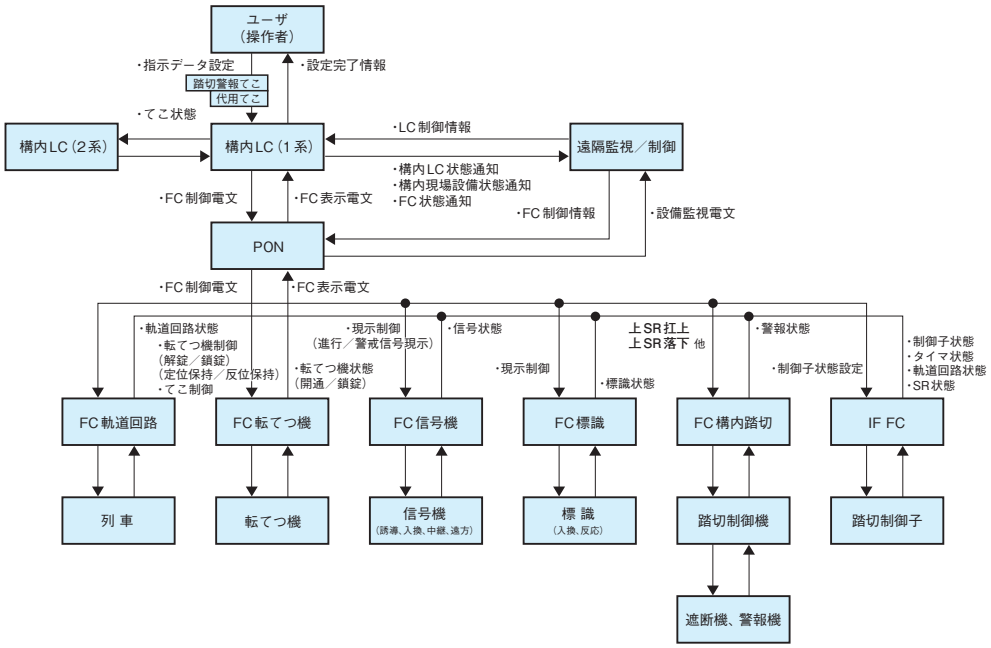


図2-2　構内踏切全体のControl Structure Diagram

※3　ここで言う「駅」とは、ポイント（分岐器）を持つ駅を指す。以下同様

※4　レールを介して電気回路を構成し、その電圧変化で列車の有無を検知する装置

※5　落下/扛上とは、従来から信号保安装置に用いられてきたリレーシーケンス回路において、リレーの電圧をしゃ断してOFF/電圧をかけてONにする制御ロジックに由来する。SRは当社内で一般的に踏切警報条件に用いるリレー名称であり、フェールセーフの観点から「SR落下」で踏切が鳴動するようになっている

表2-1　ハザードにつながる非安全なControl Action（抜粋）

制御アクション	内容	提供されない Not Providing	誤って提供される Providing causes hazard	早過ぎる／遅過ぎる／順序が違う Too early / Too late	途中で停止する／（過剰に長引く） Stop too soon / Applying too long
上SR落下	●警報区分進路の警報状態が「警報中」または「在線監視」のとき、SR落下。 (1)「警報中」となる条件（「警報準備」経由）（以下の条件のAND） ●「非警報（非在線）」→「警報準備」〈詳細は省略〉 ●「警報準備」→「警報中」遷移〈詳細は省略〉 (2)「警報中」となる条件（「非警報（在線）」経由）（以下の条件のAND） ●「非警報（非在線）」→「非警報（在線）」〈詳細は省略〉 ●「非警報（在線）」→「警報中」〈詳細は省略〉 (3)「在線監視」となる条件（以下の条件のOR） ●「警報準備」→「在線監視」〈詳細は省略〉 ●「警報中」→「在線監視」〈詳細は省略〉 ●「非警報（在線）」→「在線監視」〈詳細は省略〉	列車が警報区分進路に実際に在線しており、左記条件（1）～（3）が合致する状況において、上SR落下が提供されないと、ハザードH1「列車が在線で踏切がしゃ断しない」に至る。【UCA 1-1-A】	列車が警報区分進路に実際に在線しておらず、左記条件（1）～（3）が合致しない状況において、上SR落下が提供されると、ハザードH5「列車が非在線で踏切がしゃ断する」に至る。【UCA 1-1-B】	列車が警報区分進路に実際に在線しているが、左記条件（1）～（3）が合致しない状況において、早まって上SR落下が提供されると、ハザードH4「警報時間の過剰」に至る。【UCA1-1-C】 列車が警報区分進路に実際に在線しており、左記条件（1）～（3）が合致する状況において、遅れて上SR落下が提供されると、ハザードH3「警報時間の不足（法令への抵触）」に至る。【UCA 1-1-D】	列車が警報区分進路に実際に在線しているときに、上SR落下の周期出力が途中で停止しても、直ちに上SR扛上にはならないので、ハザードH2「踏切がしゃ断後に列車が在線にもかかわらず開く」に至らない。 列車が警報区分進路に実際に在線しておらず、左記条件（1）～（3）が合致しない状況において、上SR落下が過剰に長引くと、ハザードH4「警報時間の過剰」に至る。【UCA 1-1-E】

表2-2　絞り込んだハザードシナリオ

No.	ハザードシナリオ
1	列車が警報区分進路に実際に在線しており、上SR落下が合致する状況において、上SR落下が提供されないと、ハザードH1「列車が在線で踏切がしゃ断しない」に至る。
2	列車が警報区分進路に実際に在線しており、上SR落下が合致する状況において、遅れて上SR落下が提供されると、ハザードH3「警報時間の不足（法令への抵触）」に至る。
3	列車が警報区分進路に実際に在線しており、上SR扛上のいずれも合致しない状況において、上SR扛上が提供されると、ハザードH2「踏切がしゃ断後に列車が在線にもかかわらず開く」に至る。
4	列車が警報区分進路に実際に在線しておらず、上SR扛上のいずれも合致しない状況において、早まって上SR扛上が提供されると、ハザードH3「警報時間の不足（法令への抵触）」に至る。

2.2.3　Control Loopの作成によるハザード要因の分析（Step 2）

Step 1で識別したハザードシナリオごとに、関係するControllerとControlled Processを識別してControl Loop Diagramを作成し、ガイドワードを適用して詳細なハザード要因を分析した。ハザードシナリオNo.1：（UCA1-1-A）のControl Loop Diagramを図2-3に示す。

図2-3中の外部状況の認識誤り要因の抜粋を以下に示す。

(a) 列車の在線状況を正しく判断しない

(b) 信号機（進路）の設定状態を正しく判断しない

(c) 転てつ機の開通方向を、正しく判断しない

ハザードシナリオNo.1：（UCA1-1-A）について、ハザード要因を整理したものを表2-3に示す。

なお、これまでのStepを通じて、検討中の仕様案にて定義されていた警報状態の遷移条件の抜け・誤りを5件識別した。そのうち2件は、H1のハザードに至る条件であった。

これらは安全上問題となるため、状態遷移図を再定義して、Step 0～2までを再度分析した。

図2-4に仕様案における変更前及び変更後の状態遷移図を示す。

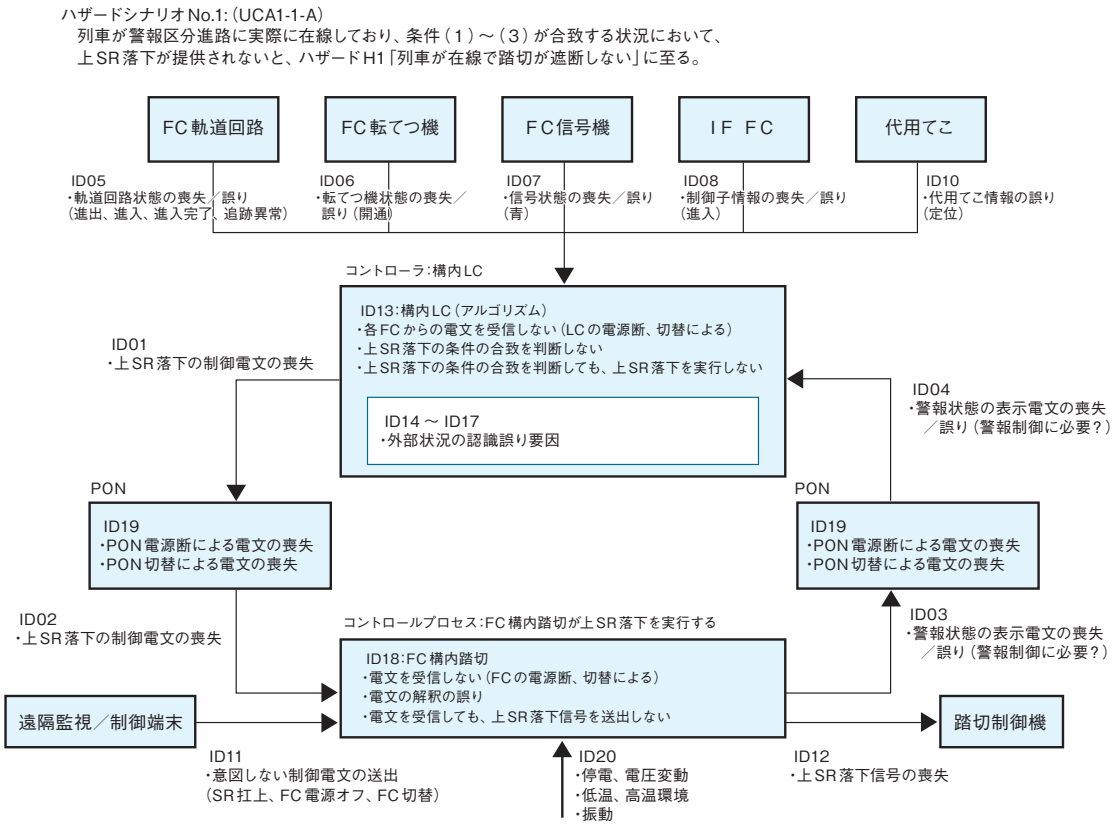


図2-3　Control Loop Diagram

表2-3　ハザード要因（抜粋）

分析対象			要因のカテゴリ				ハザード要因
カテゴリ	ID	対象	喪失、不実行	誤り、保持	遅延／早出	停止／継続	
データ伝送	ID01	構内LC → PON	×	−	−	−	・上SR落下の制御電文の喪失
	ID02	PON → FC構内論切	×	−	−	−	同上
	ID03	FC構内論切 → PON	−	−	−	−	・警報状態の表示電文の喪失／誤り（警報制御に必要？）
	ID04	PON → 構内LC	−	−	−	−	同上
	ID05～ID10	その他FC → 構内LC	×	×	−	−	・ID05:軌道回路状態の喪失／誤り ・ID06:転轍機状態の喪失／誤り ・ID07:信号状態の喪失／誤り ・ID08:制御子情報の喪失／誤り ・ID10:代用てこ情報の誤り
		遠隔監視／制御端末 → FC構内論切	−	×	−	−	・意図しない制御電文の送出（SR扛上、FC電源オフ、FC切替）
		FC構内論切 → 論切制御機	×	−	−	−	・上SR落下信号の喪失
内部（機器、人）	ID13	構内LC（アルゴリズム）	×	−	−	−	・各FCからの電文を受信しない（LCの電源断、切替による） ・上SR落下の条件の合致を判断しない ・上SR落下の条件の合致を判断しても、上SR落下を実行しない
	ID14	構内LC（外部状況「軌道回路状態」の認識）	×	−	−	−	(a) 軌道回路状態（警報区分進路）を、「進出」と判断しない ・「非警報（非在線）」に遷移する前（「在線監視」、「非警報（在線）」）に、既に「進出」と判断されていたので、あらためて判断しない ・FC軌道回路の出力が異常であると判断しているため (d) 軌道回路状態（警報区分進路）を、「進入」or「進入完了」と判断しない ・軌道回路状態と制御子情報が整合していないため（制御子がある軌道の場合） (i) 非在線中において、軌道回路状態（警報区分進路）が、「追跡異常」と判断しない ・「警報準備」に遷移する前（「非警報（非在線）」）に、既に異常と判断されていたので、あらためて判断しない (k) 在線中において、軌道回路状態（警報区分進路）が、「追跡異常」、かつ、進路予約状態が、「なし」と判断しない ・「非警報（在線）」に遷移する前（「非警報（非在線）」）に、既に異常と判断されていたので、あらためて判断しない

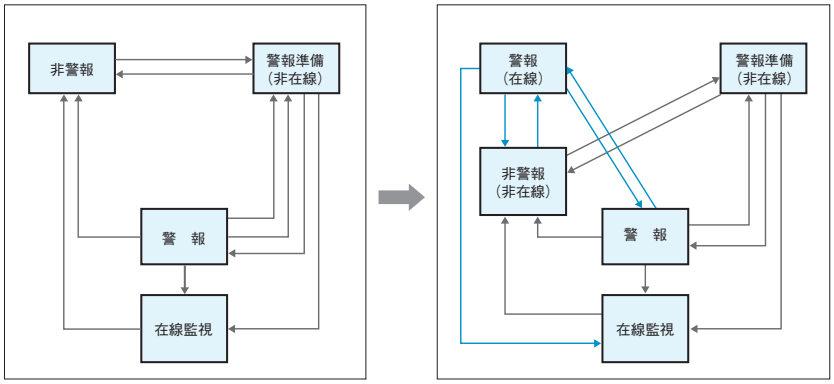


図2-4　警報状態遷移図（変更前及び変更後）※遷移条件の記載は省略

2.2.4　安全制約の識別（Step 3）

Step 2で識別したハザード要因ごとに、ハザード要因を制御／除去するための安全制約を識別した。ハザードシナリオ1～4の合計で、111件の安全制約（安全要求）を識別できた。ハザードシナリオ1におけるハザード要因／安全制約（安全要求）の抜粋を表2-4に示す。

2.2.5　設計時の安全性検証における STAMP/STPAの位置付け

これまでのStepにて、STAMP/STPA分析で安全要件を抽出したが、安全性分析はここで終わるわけではない。実際には、その結果により絞られた検査対象に対し、モデル検査などのツールを用いた安全性の検証が必要となる。逆に、検査対象を絞り込

表2-4　ハザードシナリオ1のハザード要因／安全制約（安全要求）抜粋

ID	ハザード要因	安全制約（要求）
ID01	●上SR落下の制御電文の喪失	●制御電文を喪失しない。あるいは、喪失を検知する。
ID02	同上	同上
ID03	●警報状態の表示電文の喪失／誤り	同上
ID04	同上	同上
ID05～ID10	●ID05：軌道回路状態の喪失／誤り ●ID06：転てつ機状態の喪失／誤り ●ID07：信号状態の喪失／誤り ●ID08：制御子情報の喪失／誤り ●ID09：ユーザ（操作者）の誤り ●ID10：代用てこ情報の誤り	●各FCからの電文を喪失しない。あるいは喪失を検知する。 ●各FCからの電文を誤らない。あるいは誤りを訂正する。
ID11	●意図しない制御電文の送出（SR扛上、FC電源オフ、FC切替）	●構内LC運用中において、遠隔監視／制御端末の利用制限を設ける。

まずにモデル検査を行おうとすると、容易に状態爆発を引き起こしてしまい、実用的な時間内での検査は不可能となる。

すなわち、STAMP/STPA手法は、設計時の安全性検証における「検査対象を安全上重要な個所に絞り込む」役割を担っていると言える。

また、今回紹介した分析事例から、とくに、イベントの前後関係が動作上クリティカルになることの多い信号保安システムにおいては、状態遷移図の抜け・漏れの分析としてSTAMP/STPA手法の適用が有効であることも分かった。

3 STAMP/CASTと用いた事故事例の分析と安全要件の抽出

前節では、STAMP/STPA手法を用いた信号保安システムの設計段階での安全要件抽出の事例を示した。一方、STAMP/CAST（Causal Analysis using System Theory）手法は、過去に発生した障害事例から安全要求を抽出するのに有効な手法である。

本手法においても、コントロールループを用いて、安全制約を実現する制御機能のコントロールストラクチャ（Safety Control Structure）を作成する。しかし、解析事象の網羅性を重視するSTAMP/STPAとは異なり、実際のシナリオと各機能の流れを一覧表にて整理することで、関連した要因を結び付けながら背後にある要因の分析を容易にする点が特徴である。

3.1 分析対象

分析対象として、踏切制御における過去の不具合事例をもとに、12件のケースを想定した。想定にあたっては、STAMPモデルの特性から、以下の2点を満たすものとした。

①2つ以上のコンポーネントが関係する不具合事象であること

②事象によって引き起こされたハザードが万遍なく（2-2-1節のH1～H5）選択されること

ここではその一つの例として以下の題材を取り上げる。

「▲▲本線○○駅の■■踏切にてしゃ断不良※6が発生した」というケースを考える。図3-1に示すように、下り列車1レによって警報開始した踏切制御に対し、同じ構内にて上り方向〈3LT〉へ出発する列車2レによって誤って終止条件が成立した事象を想定する。

3.2 STAMP/CAST分析

3.2.1 Step 1：不具合シナリオの分析

このStepでは、不具合に至るシナリオを時系列（event chain）で分析する（表3-1）。すなわち、本事象にて登場する列車や制御機能と発生した事象を列挙する。

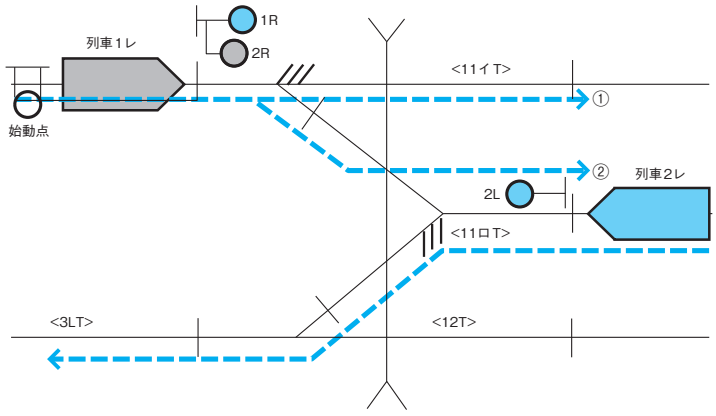


図3-1　想定事象のイメージ

表3-1　本事象のevent chain

Step.1 不具合シナリオの分析

ID	列車1レ	列車2レ	連動制御	踏切制御
①-1			転てつ機11イ定位（信号機1Rが青、信号機2Rが赤）てつ器11ロ定位（信号機2Lが青）	
①-2	下り始動点に進入			
①-3				踏切鳴動開始し、警報時間の計測開始
②-1		11ロTに進入		
②-2				列車1レが11ロTへ進入したと誤判断し、警報時間の計測開始から15秒以内であったため、最少警報時間異常を検知し、警報持続

※6　ここでは、警報中の踏切が、不正なタイミングで警報を停止し、しゃ断機を上げてしまう事象

3.2.2 Step 2：不具合事象と安全要求の定義

このStepでは、不具合事象とその発生を防ぐための安全要求を定義する。本Stepにおける安全要求は、不具合事象の裏返しである抽象度の低い安全要求と、ハザードの裏返しである抽象度の高い安全要求とのバランスを考慮して識別する。その理由は以下の通りである。

- 抽象度が低い安全要求は、具体的なシステムの用語を用いて定義されるので、不具合に気づきやすい。
- 抽象度が高い安全要求は、ほかの類似した不具合事例の分析時に、再利用しやすい。

本事象において識別した安全要求を表3-2に示す。なお、本事象においては、これは不具合事象の裏返しと同等であった。

3.2.3 Step 3：Safety Control Structure作成

安全要求の実行にかかわるSafety Control Structureを作成する（図3-2）。

Safety Control Structureは、安全要求に関係するコントロールアクションとフィードバックデータの流れとして作成する。各コンポーネントの安全要求を満たす役割として、安全制約を定義し、各コンポーネントの内部に記載する。

3.2.4 Step 4：Physical system levelの要因分析

ガイドワードを使用して、末端の制御対象である物理的要素（＝Physical system level）で不具合要因を分析する（表3-3）。ここでガイドワードとは、図3-3を基本として定めた、不具合の発生要因を導き出すための考え方である。

なお、本事象では関係する装置がすべて正しく動作しているので、物理的要素の不具合はない。

3.2.5 Step 5：Controller levelの要因分析

ガイドワードを使用して、制御を実行するController levelで不具合要因を分析する（表3-4）。すなわち、制御論理要素の不具合を列挙する。この中で、「踏切制御」のControllerが果たす安全制約を実行するための役割の一つである「警報の開始／停止指示を行う」点において不具合が発生している。

3.2.6 安全要求の導出

以上のようなSTAMP/CAST分析を、12件のケースすべてで実施した。その結果を整理し、踏切制御機能における一般的なSafety Control Structure及びその安全要求を作成した。

本研究で作成したSafety Control Structureを図3-4に、踏切制御機能に関する安全要求を表3-5に示す。

表3-2　抽象度を考慮した安全要求の識別

不具合事象の裏返し（抽象度：低）	識別した安全要求	ハザードの裏返し（抽象度：高）
連動制御による進路の独占権を踏切制御に反映し、警報開始させた列車が警報終止条件を成立できるようにする。	連動制御による進路の独占権を踏切制御に反映し、警報開始させた列車が警報終止条件を成立できるようにする。	警報時間を過剰に取らない。

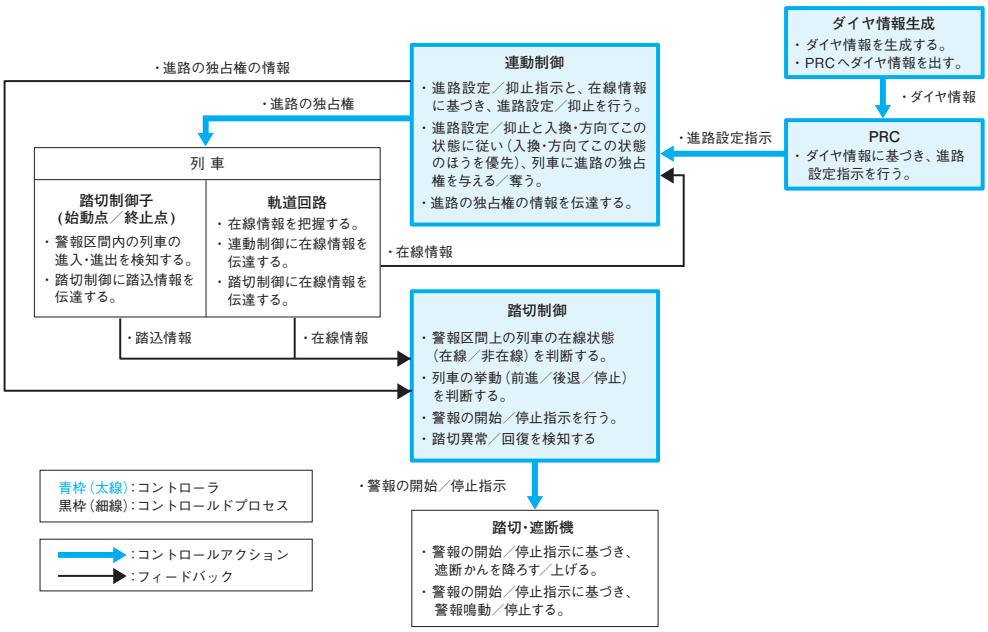


図3-2　本事象のSafety Control Structure

表3-3　Physical system levelの要因分析

No.	Physical system	(a) Safety Responsibilities アクターが果たす安全制約を実行するための役割／責任	(b) Safety Equipment (controls) 安全制約の実行に係るアクター内の機器とその動作	(c) Failures and Inadequate Control Actions (a) (b) を果たせない機器の故障、及び、不適切なコントロールアクション	(d) Physical Contextual Factors (c) のアクションがなぜ起こったか
1	軌道回路	在線情報を把握する	レールと車輪で電気回路を形成する	正しく動作したためN/A	正しく動作したためN/A
2		連動制御に在線情報を伝達する	通信手段を用いて在線情報を伝達する	正しく動作したためN/A	正しく動作したためN/A
3		踏切制御に在線情報を伝達する	通信手段を用いて在線情報を伝達する	正しく動作したためN/A	正しく動作したためN/A
4	踏切制御子 (始動点／終止点)	警報区間内の列車の進入・進出を検知する	始動点／終止点において、レールと車輪により電気回路を形成し、踏込を検出する	正しく動作したためN/A	正しく動作したためN/A
5		踏切制御に踏込情報を伝達する	通信手段を用いて踏込情報を伝達する	正しく動作したためN/A	正しく動作したためN/A
6	踏切・しゃ断機	警報の開始／停止指示に基づき、しゃ断かんを降ろす／上げる	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A
7		警報の開始／停止指示に基づき、警報鳴動／停止する	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A

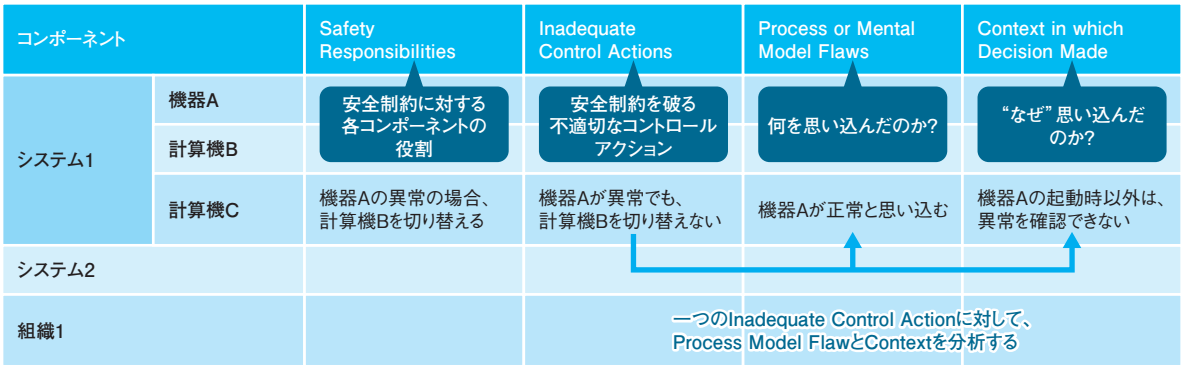


図3-3　Step 4,5で行う要因分析（ガイドワード）

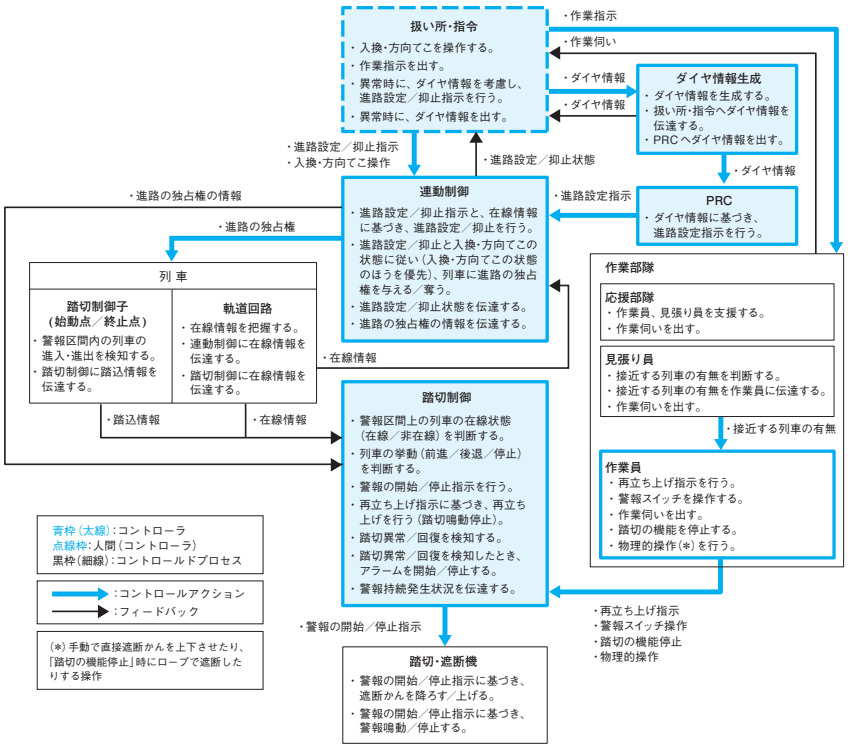


図3-4　今回のSTAMP/CAST分析において作成された踏切制御に関するSafety Control Structure

表3-4　Controller levelの要因分析

No.	Controller	(a) Safety Responsibilities アクターが果たす安全制約を実行するための役割／責任	(b) Inadequate Control Actions (a) を果たせない不適切なコントロールアクション	(c) Process or Mental Model Flaws (b) の振る舞いとなる動作条件について、アクターが信じている事実	(d) Context in which Decision Made (c) の動作条件や思い込みがなぜ生じたのか
6	踏切制御	警報区間上の列車の在線状態（在線／非在線）を判断する	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A
7		列車の挙動（前進／後退／停止）を判断する	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A
8		警報の開始／停止指示を行う	【要因】 ● 警報始動条件を成立させた列車（進路の独占権を持つ列車）以外の列車により、警報終止条件が成立してしまい、異常検知してしまった。その結果、非在線となっても踏切鳴動が継続してしまった	【要因】 ● 連動制御が独占権を与えていない進路（警報経路②）も警報経路としてしまった	【要因】 ● 独占権を与えている進路の情報（進路設定）を連動制御からもらっていなかった ● 警報進路を推定して作っていた 【提言】 ● 独占権を与えている進路の情報（進路設定）を連動制御からもらうようにする
9		踏切異常／回復を検知する	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A	【要因】 正しく動作したためN/A

表3-5　踏切制御機能に関する安全要求

Control Action	踏切制御に関する安全要求
警報開始	以下の①～④のORで警報を開始する。 ① 少なくとも1つの、独占権が確保され挙動が前進である列車の進路に警報区間が含まれるとき ② 先行列車の通過後、進出確定までの設定時素以内に、続行列車が警報区間に進入したとき ③ 警報スイッチが落下状態のとき ④ 踏切異常状態のとき
警報停止	以下の⑤～⑧のANDで警報を停止する。 ⑤ すべての、独占権が確保され挙動が前進である列車の進路に警報区間が含まれていないとき ⑥ 先行列車と続行列車の間隔が、進出確定までの設定時素より大きいとき ⑦ 警報スイッチが扛上状態のとき ⑧ 踏切正常状態のとき
進路設定／抑止指示 入換・方向てこ操作 作業指示 （扱い所・指令）	常に、踏切異常／回復を誤検知しない。 常に、踏切異常／回復を検知したとき、アラームを開始／停止する。 常に、警報持続発生状況を伝達する。

4　信号保安システムの開発プロセスとSTAMPの役割

これまで述べてきたように、STAMP/STPA手法は信号保安システムの開発段階における安全性分析への活用が期待できる。また、STAMP/CAST手法により従来システムの不具合の分析から抽出された安全要求は、システムの更新・展開及び新たな信号保安システム（例えば、次世代の踏切システム）の開発において活用できる。

これを一般的な信号保安システムの開発プロセス（Vモデル）に表すと図4-1の通りとなる。

当社においては、STAMPの導入はまだ試行段階であるが、今後は更に導入実績を積み、手順のブラッシュアップを図ることで、信号保安システムのソフトウェア開発プロセスの一部としてSTAMPの活用を図っていきたい。

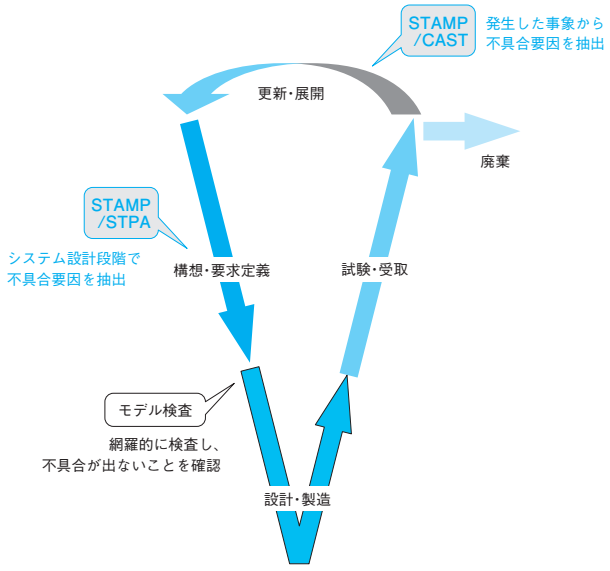


図4-1　信号保安システムの開発プロセス

【参照文献】
[1] Nancy G. Leveson, Engineering a Safer World -Systems Thinking Applied to Safety, MIT Press, 2011
[2] エリック・ホルナゲル著 小松原明哲監訳, 社会技術システムの安全分析～FRAMガイドブック～, 海文堂出版, 2013
[3] 重田ほか, 駅構内論理装置の開発, JR EAST Technical Review No.36, pp.19-26, 2011
[4] 信号システムの進歩と発展 ＝近年20年の展開と将来展望＝, 日本鉄道電気技術協会, 2009
[5] 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～, 情報処理推進機構, 2016

エンタープライズ系システムを対象としたSTAMP/STPA分析試行

IPA/SEC IoTシステム安全性向上技術WG委員／日本電気株式会社　向山 輝

1 はじめに

STAMP/STPAが分析対象とするアクシデントは「損失につながるような意図せざる事象」とされている。「損失」の定義には、人命やけがにかかわる損失だけでなく、ミッションの未達や経済的な損失、所有物の毀損なども含まれ、広範囲のシステムが分析対象となる［Leveson2015］。金融、流通、運輸、情報通信などの分野で利用されるエンタープライズ系システムは、社会を支えるインフラの一部となっており、サービスが停止した場合には日常生活や経済活動に大きな損失が生じる。上述のアクシデントの定義に従えば、これらのシステムのサービス停止を「アクシデント」としてSTAMP/STPA分析を行うことは可能であり、分析によって危険要因を認識し、対策を打つことができれば有益である。

しかしながら、これまでに公開されているSTAMP/STPA分析の事例は制御系システムを対象としたものが多く、エンタープライズ系に適用する場合に参考にできる事例がほとんどない。そこで、IoTシステム安全性向上技術WGでは、エンタープライズ系システムにSTAMP/STPA分析を適用する場合のコツや注意点を明らかにすることを目的とした試行を実施した。本編では、実施した試行の概要と、得られた知見について解説する。

2 分析対象とするシステム

分析対象には、多くの人に利用経験があり、仕様を理解しやすいという理由で、ネット通販システムを取り上げる。試行に用いる例題として、表 1に示すような典型的な機能を持つ架空のネット通販を想定する。

表1　分析対象とするネット通販システムの仕様

機 能	内 容
販売管理機能	・利用者からの注文を受けると、在庫管理に引当て指示を出す ・引当て可であれば受注ステータスを「確定」とし、出荷機能に対して出荷指示を出す ・引当て不可であれば受注ステータスを「不可」として利用者に通知する
在庫管理機能	・商品の在庫データ（総在庫数、引当て済み数）を管理する ・引当て指示を受けると、在庫データを確認し、引当て可否を応答する
出荷機能	・出荷指示を受けると、商品をピックアップし、配送機能に引き渡す（出荷する） ・出荷後、在庫データを更新する
配送機能	・出荷機能から商品を受け取り、指示された宛先に配達する

図 1は、表 1に示す仕様を、エンタープライズ系システムの仕様記述でよく用いられるUMLアクティビティ図で表したもので

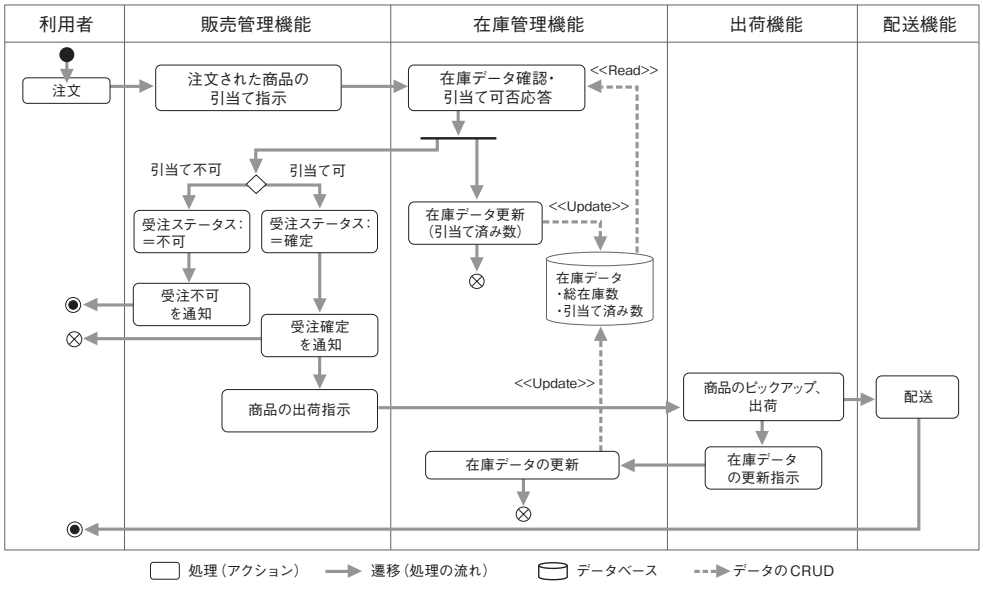


図1　ネット通販システムの仕様を表すUMLアクティビティ図

ある。この図には、データのCRUD※1に関する情報も付記している。本来のUMLアクティビティ図には含まれない情報であるが、各機能間の相互関係を分かりやすくすることを意図したものである。

表1及び図1に示す4つの機能のうち、販売管理機能と在庫管理機能は電子化され自動処理されるものとし、出荷機能と配送機能は人手によって処理されるものとする。

3 STAMP/STPA分析

IPA/SECより公開されているSTAMP/STPAの解説書[IPA 2016]に示される手順に沿い、ネット通販システムの分析の様子を示す。

3.1 Step0 準備1 アクシデント、ハザード、安全制約の識別

アクシデントとしては、期待されるサービスが遂行されず損失を生じる状況が相当する。ネット通販システムの場合は、「注文した商品が利用者に配送されない」ことがアクシデントの一つとして考えられる。

ハザードは、アクシデントにつながる可能性の高いシステムの状態のことであり、今回の分析では「受注が確定した注文に対して商品が出荷されていない状態」とする。出荷が遅れているだけでいずれ出荷されるならアクシデントにはならないが、そのまま忘れられたり、商品が足りなくなったりした場合は、定義したアクシデントにつながる。安全制約は、ハザード状態を起こさないことであり、「受注が確定した注文に対して、速やかに商品が出荷されなければならない」となる。以上を表2にまとめる。

表2 アクシデント、ハザード、安全制約

アクシデント	注文した商品が利用者に配送されない
ハザード	受注ステータスが「確定」である注文に対して、商品が出荷されていない状態
安全制約	受注ステータスが「確定」である注文に対して、速やかに商品が出荷されなければならない

3.2 Step0 準備2 コントロールストラクチャーの構築

コントロールストラクチャーはSTAMP/STPA分析で用いるモデルであり、ここで表現する情報が分析のすべてに反映されるため、どのようにモデル化するかは重要である。しかしながら、モデリングのスキルは属人性が高く、ノウハウを明文化することが困難である。コントロールストラクチャーの構築は、STAMP/STPA分析における最も難易度の高い作業と言える。

今回の試行では、分析対象とするシステムの仕様がアクティビティ図で表現されることに着目し、コントロールストラクチャーの構成要素であるコンポーネント、コントロールアクション、フィードバックデータの3つをアクティビティ図から抽出することを試みる。具体的には次に示す手順である。

(1) アクティビティ図から、安全制約に関係する「処理」を抽出する。

(2) 抽出された処理のうち、その処理から発する矢印(遷移)が機能間をまたぐものについて「コントロールアクションを発行する処理」と「フィードバックデータを発行する処理」のいずれかに該当するかどうかを判定する。

(3) コントロールアクションまたはフィードバックデータに該当する矢印(遷移)の矢元と矢先の機能を「コンポーネント」と識別する。

この手順を図1のUMLアクティビティ図に適用した結果を図2に示す。青く色付けした処理は、表2に示した安全制約が依存

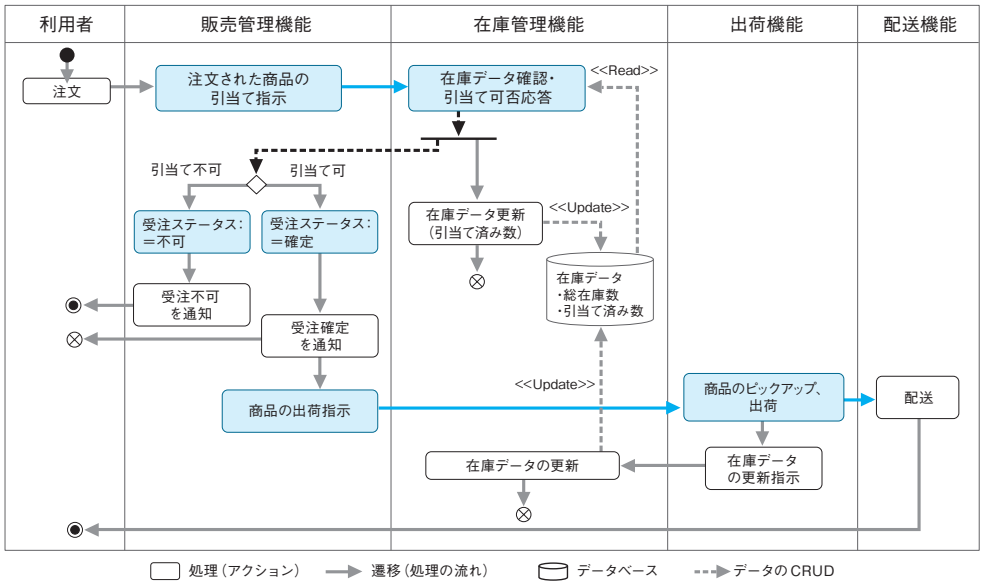


図2 コンポーネント、コントロールアクション、フィードバックデータの識別

※1 データのCreate(生成)、Read(参照)、Update(更新)、Delete(削除)を指す略語

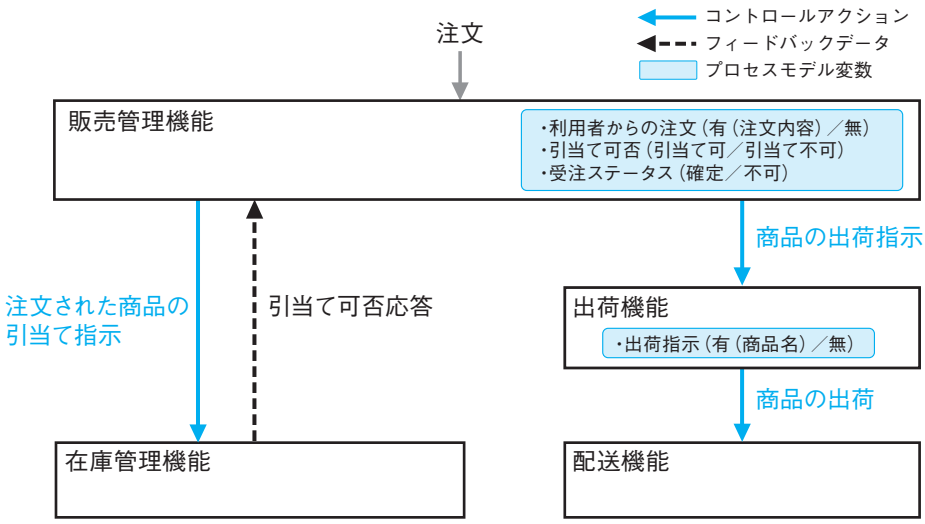


図3　コントロールストラクチャー

する「受注ステータス」と「商品の出荷」にかかわると判断した処理である。これらの処理から発する矢印のうち、機能間をまたぐものは、機能間の相互関係を表すものと考えられる。青で示す矢印はコントロールアクションに相当すると判断されるものであり、黒の破線で示す矢印はフィードバックデータに相当すると判断されるものである。例えば、「注文された商品の引当て指示」は、販売管理機能から在庫管理機能に対するコントロールアクションに相当する処理と解釈でき、「在庫データ確認・引当て可否応答」は、引当て指示のコントロールアクションに対する結果をフィードバックする処理と解釈できる。コントロールアクションとフィードバックデータの矢元と矢先に該当する「販売管理機能」「在庫管理機能」「出荷機能」「配送機能」の4つの機能がコンポーネントに相当する。

このようにして識別したコンポーネント、コントロールアクション、フィードバックデータを用いると、図3に示すようなコントロールストラクチャーを構築できる。プロセスモデル変数は、コンポーネントが認識するシステム内外の状態を表す変数であり、コントロールアクションを実行するかどうかの判断にかかわるものを列挙する。

3.3　Step1　非安全なコントロールアクションの抽出

3つのコントロールアクションのそれぞれに対して、STAMP/STPAが定める4つのガイドワードを利用して、安全制約に反する非安全なコントロールアクション（Unsafe Control Action, 以下UCAと略記）を洗い出す。IPA/SECのWGで実施した試行では7つのUCAを識別したが、本編では紙面の制約上「商品の出荷指示」というコントロールアクションに関するUCAのみを示す（表3）。実際の試行で抽出した7つのUCAは［IPA2017］を参照されたい。

UCAの抽出は、コントロールアクションにガイドワードを当てはめ、それがどのような状況でハザードにつながるかを考えることによって行う。「状況」は、プロセスモデル変数を手がかりとして考える。プロセスモデル変数の値、すなわち、コンポーネントが認識する状況が、現実と食い違っている場合にハザードが起きやすい。例えば、現実には「引当て不可」の状況で、販売管理機能が「引当て可」と誤認識すると、受注ステータスが確定となり「商品の出荷指示」が与えられてしまうが、実際には在庫が足りないため、「受注ステータスが確定にもかかわらず出荷されない」というハザード状態となる。表3に示したUCAのうち、「与えられてハザード」に該当するUCAはこのような分析から導かれる。

表3　抽出されるUCAの一部

コントロールアクション	ガイドワード			
	与えられずにハザード	与えられてハザード	早過ぎ／遅過ぎ／誤順序でハザード	長過ぎる／短過ぎる適用でハザード
商品の出荷指示	受注ステータスが確定となった注文に対して出荷指示が出ない	引当て不可の状況で受注ステータスが確定となり、出荷指示が発行される	受注ステータスが確定となった後、出荷指示が遅れ、速やかに出荷されない	該当なし

3.4　Step2　UCAの原因の特定

Step1で抽出したUCAのそれぞれに対して、その原因を分析する。本編では、紙面の制約上、表3に示したUCAのうち、「引当て不可の状況で受注ステータスが確定となり、出荷指示が発行される」のUCAの原因分析を解説する。

このUCAは、販売管理機能が、現実には引当て不可であるにもかかわらず引当て可と誤認識すると発生する。誤認識の要因の一つとして、在庫管理機能から販売管理機能に入力される「引当て可否応答」の情報が誤っていることが考えられる。

その原因の分析には、図2のアクティビティ図を参照することが役立つ。アクティビティ図の中で、在庫管理機能が引当て可否応答を行う個所に注目する（図4）。引当て可否応答は、在庫データを参照（Read）して行われる。一方、在庫データの一部である「引当て済み数」は、引当てが行われると更新（Update）される。ここで、もしも、在庫データが最新の状態に更新される前に参照されることがあれば、不正確な引当て可否応答が行わ

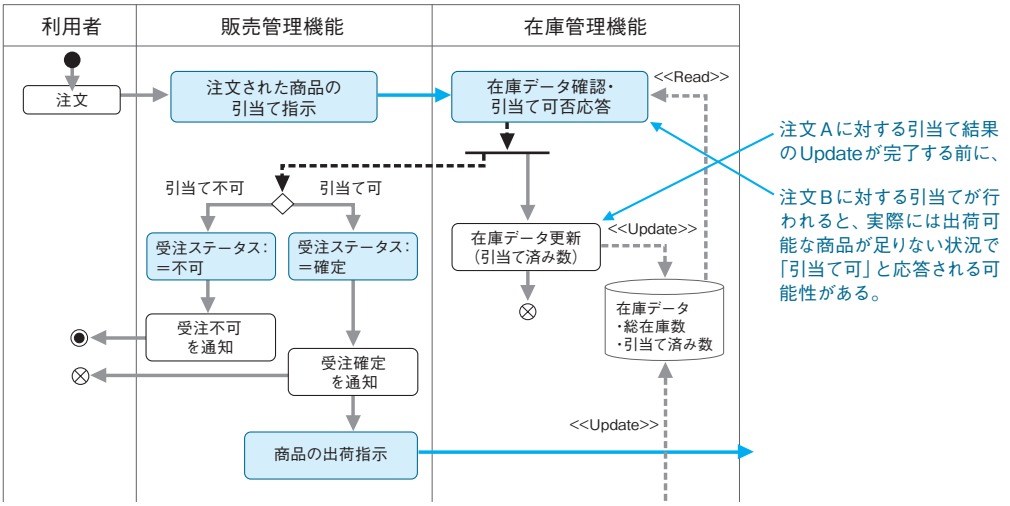


図4　引当て可否を誤認する要因の分析

れる可能性があることに気づく。例えば、注文Aと注文Bの2つを並行して処理するとき、注文Aに対する引当て済み数で在庫データを更新する前に、注文Bに対する引当て指示を受けて在庫データを参照すると、実際には在庫が足りない（引当て不可である）状況にもかかわらず「引当て可」と応答する可能性がある。「引当て不可の状況で受注ステータスが確定となり、出荷指示が発行される」のUCAの原因の一つとして、このようなシナリオが考えられる。

4　試行によって得られた知見

冒頭に述べた通り、本試行は、エンタープライズ系システムにSTAMP/STPA分析を適用する場合のコツや注意点を明らかにすることを目的として行った。この経験から、次のような知見が得られた。

- アクティビティ図で表される情報は、コントロールストラクチャーの構成要素であるコンポーネント、コントロールアクション、フィードバックデータを識別する手がかりとなる。
- Step2のUCAの原因特定には、アクティビティ図の情報が役立つ。とくに、データの更新、参照に関する依存関係に着目することが有効であった。従って、アクティビティ図にはデータのCRUD情報も含まれると良い。
- 分析対象のシステムを構成する機能が、自動処理されるか人手によって処理されるかは、分析に影響するため、明確化されていることが望ましい。

以上の知見は、今回実施した試行の経験から得たものであるが、いずれもエンタープライズ系システムに特徴的な要素に関するものであるため、汎用的に通用するものと予想する。

なお、上記の3番目は、本編で取り上げなかった別のUCAの分析で経験した結果によるものである。架空のシステムを対象としたため、当初、各機能が自動処理か手動処理かが未定であった。その状態で分析を行うと、UCAが現実に起こり得るか否かの判断が難しいことに気づき、第2節に示したような自動処理/手動処理の前提を加えた。自動処理/手動処理の区別が暗黙的には決まりにくいエンタープライズ系に特徴的な注意点と考えられる。

5　まとめ

例題としたネット通販システムとそれにかかわるアクシデントは、冒頭に述べたような「社会インフラのサービス停止」のイメージとは距離があるかもしれない。しかし、エンタープライズ系システムを対象としたSTAMP/STPA分析の特徴を理解し、知見を得る目的としては、今回実施した試行は有意義であった。

また、STAMP/STPA分析の価値について、次のような理解も得られた。本編で示した分析は、従来から行われているアクティビティ図で書かれた仕様のレビューと変わらないようにも見える。しかし、アクティビティ図のどの部分に着目し、どのような問題意識を持ってレビューするかを、レビュアの主観ではなく、システムレベルのアクシデントからトップダウンに導出したUCAの観点で、理由付けと網羅性を持って説明できる点にSTAMP/STPAを用いる価値があるのだと考えられる。

なお、WGで行った分析では、Step2で識別したハザードシナリオに基づく対策の検討や、業務効率を改善することを目的としたSTAMP/STPA分析の応用の検討も行った。詳細は［IPA 2017］を参照いただきたい。

【参照文献】

［Leveson2015］ Nancy Leveson, An STPA Primer v1, 2015.
［IPA2016］ IPA, はじめてのSTAMP/STPA, 2016.
参照先：https://www.ipa.go.jp/files/000055009.pdf
［IPA2017］ IPA, はじめてのSTAMP/STPA（実践編）, 2017.
参照先：https://www.ipa.go.jp/files/000058231.pdf

STAMP海外事例の紹介：STPA-SafeSec

仙台高等専門学校／IoTシステム安全性向上技術WG委員　岡本 圭史
国立大学法人信州大学／IoTシステム安全性向上技術WG委員　岡野 浩三

本稿では、STAMP海外事例としてSTPA-SafeSecを紹介する。STPA-SafeSecは、安全性と脆弱性を統合して分析するためのSTPA拡張である。また、とくに脆弱性分析をSTPAベースで実施する際に有用となる関連事項を併せて紹介する。

1　はじめに

STAMP（System Theoretic Accident Model and Processes）はシステム理論に基づく事故モデルであり、STPA（System Theoretic Process. Analysis）はSTAMPに基づくハザード分析（安全分析）手法である［1］［2］。STPAは既存の安全分析手法で分析が困難であった複雑な対象に対し有効であると言われている［1］。

走行中の自動車のエンターテインメント系から走行系への乗っ取り、コンピューター・ワームによる遠心分離機の破壊といった、セキュリティ侵害が安全性を脅かす事例がある。これらの事例は、STAMP/STPAの用語を用いれば、セキュリティにかかわるハザード誘発要因（Hazard Causal Factor、HCF）が最終的に安全制約を破るという事例である。このような事例を分析するためには、安全分析とセキュリティ分析を統合したSTPAの拡張が必要となる。

前述の事例は、STPAのstep0準備1において安全性にかかわるアクシデント、ハザード、安全制約を識別し、step2のHCF特定のヒントとしてセキュリティにかかわるヒントを導入するだけで、分析できそうに思える。しかし、セキュリティ分析にはシステムの詳細情報が必要なことが多く、STPAで使用するコントロールストラクチャー図（Control Structure Diagram、以下CSD）がセキュリティ分析に十分であるかといった検討は必要であろう。このような背景の下に、STPAの拡張として、STPA-Sec［3］やSTPA-SafeSec［4］が提唱されてきた。

本稿では、具体的な事例（マイクログリッドにおける広域電力網と局所電力網の接続（併入）をSTPA-SafeSecで分析）を用いてSTPA-SafeSecの手順が説明されている論文を紹介する。はじめに、STPA-SafeSecの特徴と手順を紹介し、次にSTPA-SafeSecの適用事例を紹介する。更に、とくに脆弱性分析をSTPAの拡張で分析する際に有用な事項を紹介する。

なお本稿では、STPA-SafeSecで用いられている用語を、標準的STAMP/STPAの用語に著者の解釈で置き換えている。

2　STPA-SafeSecの概説

本節では、文献［4］で提案されているSTPA-SafeSecを紹介する。STPAは安全性分析を目的としているが、STPA-SafeSecは、安全制約と脆弱性を統合して分析できるSTPA-SafeSecの拡張であり、文献［4］では、STPA-SafeSecの詳細な手順が提案されている。本節ではスペースの関係から、STPA-SafeSecの手順詳細を割愛し、標準的STPAの手順［2］に合わせて解説する。

2.1　STPA-SafeSecの特徴

STPA-SafeSecでは以下の上2つが貢献として挙げられている。また本稿では、1つ目の貢献からの派生効果であるが3つ目も貢献として挙げる：

1. 機能CSDと物理CSDを持つ
2. Step2で使用するHCFヒントのセキュリティ拡張
3. 安全性・セキュリティ対策の統合

これらの特徴について、それぞれ述べる。

機能レイヤ（Control Layer）のCSD（以下、機能CSD）内のコントロールループごとに構築する物理レイヤ（Component Layer）のCSD（以下、物理CSD）を用いることで、step2でセキュリティ侵害の経路が特定しやすくなる。例えば、上位の機能CSDの時刻同期機能は、下位の物理CSDではGPSに詳細化されたとする。この詳細化により、GPSの既存の脆弱性をHCFとして利用できる。また機能CSDと物理CSD間のコンポーネントを対応付けることで、機能CSDで特定したUCAから、物理CSDにおいて識別するそのUCAへ至るHCFとハザードシナリオとの対応が容易になる。更に、物理CSDを考えることで、既存の脆弱性分析を活用するには、抽象化した機能レベルの分析では限界があり、この事例のような物理CSDの導入が必須となる。

標準的STPAは安全性を分析するために、アクシデント、ハザード、安全制約を識別し、最終的にHCFとハザードシナリオを特定する。HCFを特定する際には、コントロールループ中で安全性にかかわるHCFヒントとして、コンポーネント故障、ヒューマンエラー、コミュニケーションエラー、ソフトウェア不具合、要求仕様不具合などを用いることが一般的である。STPA-SafeSecでは、これら従来のヒントにセキュリティにかかわるHCFヒントを追加し、HCFとしてなりすましなどのセキュリティにかかわる誘発要因を特定できるようにしている。

なお、STPA-SafeSecで採用されているセキュリティにかかわるHCFヒント以外では、例えば、セキュリティにかかわるヒントとしてSTRIDE［5］の利用が考えられる。

STPA-SafeSecのstep2では、抽象的ハザードシナリオから具体的ハザードシナリオを導出し、安全制約やセキュリティ制約を特定している。この導出方法により、ハザードシナリオ（安全・セキュリティ制約）たちは木構造となる。この木構造を分析することで、安全制約とセキュリティ制約間の関係が明らかになり、これらを統合できる。例えば、機能CSDであるフィードバックが間違っていることがUCAの指示につながり、ひいてはハザードを引き起こすこというシナリオ1が策定できたとする。更に、この機能CSDを詳細化した物理CSDにより、サイバー攻撃によりそのデータが改ざんされ、同じUCAへ至るというシナリオ1.1を特定できたとする。このとき、安全の観点から導入されたデータチェック機構は改ざん検出にも利用できるため、シナリオ1の安全対策がシナリオ1.1のセキュリティ対策を兼ねることになる。

2.2 STPA-SafeSecの手順

STPA-SafeSecは詳細な手順に分割されている（［4］図3）。本稿では、標準的STPAのプロセスである［2］で解説されている手順にまとめSTPA-SafeSecを解説する。

Step 0準備1（STPA-SafeSec II～IV）：対象とするシステムのロス（アクシデント）、ハザードを定義し、各ハザードに対する安全制約とセキュリティ制約を識別する。セキュリティ制約といったセキュリティにかかわる事項を対象とすること以外は、標準的step0準備1と同じである。

Step0準備2（STPA-SafeSec V）：上記制約の実現に必要な、機能コンポーネントとそれらの相互作用（コントロールアクションとフィードバック）を分析して機能CSDを構築する。機能CSDは標準的step0準備2で構築するCSDに対応する。

Step1（STPA-SafeSec VI～IX）：機能CSD内の各コントロールループに対し、トーマス博士が提案する拡張step1［6］により、UCA（原文Hazardous Control Action）を抽出する。拡張step1は、コントローラの入力の組み合わせに対し網羅的にUCAか否かを判定するため、自動化に適しているといった特徴を持つ。なおSTPA-SafeSecの他手順との関連から、STPA-SafeSec step1は標準的step1でも良いと考えられる。他方、STPA-SafeSec step1で用いるガイドワードは標準的STPAの4つのガイドワードと同じである。

Step2はSTPA-SafeSecの特徴的な概念・手順を多く含むため、［4］に従い、以下の3つの手順に分割して解説する。

Step2a（STPA-SafeSec X、XI）：機能CSDの各コントロールループに対し、物理CSDを構築する。物理CSDは機能CSDをアーキテクチャレベルへ詳細化した記述である。このとき、これらの構成要素間を対応付ける。また、抽象的ハザードシナリオ（原文Safety Related Flaws、System Flaws）を策定する。この抽象的ハザードシナリオは、標準的STPA［1］の安全にかかわるHCFヒントを参考に特定される。

抽象レベルでのシナリオとして、ハザードシナリオのみを扱うのは、標準的STPAが扱うシナリオに加え、セキュリティ侵害が安全性を脅かすシナリオを扱うことを目的としているためと考えられる。

Step2b（STPA-SafeSec XII）：step0準備1で識別済みのハザード及びリスト1、2のセキュリティ制約を基に機能CSDのコンポーネントへ抽象的安全・セキュリティ制約を課し、機能CSDと物理CSDの対応に基づき、抽象的安全・セキュリティ制約を物理CSDの要素に割り振る。

物理CSDのコンポーネントに課される安全制約は、step1準備1で識別した安全制約であり、標準的STPAの安全制約である。他方、セキュリティ制約はSTPA-SafeSec step2bで登場する。後の事例において、物理CSDのコンポーネントに既知の脆弱性としてスプーフィング（spoof）とジャミング（jam）が知られている場合に、このコンポーネントへセキュリティ制約（CSTR-A-1、CSTR-A-2）を課すという利用法からは、リスト1、2の内容はセキュリティにかかわるHCFヒントであるとも言える。

リスト1：完全性に対する汎用的脅威
- CSTR-I1 コマンド・インジェクション（Command injection）
- CSTR-I2 コマンド欠落（Command drop）
- CSTR-I3 コマンド操作（Command manipulation）
- CSTR-I4 コマンド遅延（Command delay）
- CSTR-I5 観測値インジェクション（Measurement injection）
- CSTR-I6 観測値欠落（Measurement drop）
- CSTR-I7 観測値操作（Measurement manipulation）
- CSTR-I8 観測値遅延（Measurement delay）

リスト2：可用性に対する汎用的脅威
- CSTR-A1 通信遅延（Communication delay）
- CSTR-A2 通信欠落（Communication dropped）
- CSTR-A3 ノード過負荷（遅延）（Node overloaded（delay））
- CSTR-A4 ノード過負荷（欠落）（Node overloaded（drop））

Step2c（STPA-SafeSec XIII）：step2aで識別した抽象的ハザードシナリオを機能CSDに対するトップレベルのハザードシナリオとし、それを物理CSDへ詳細化していく。このとき、詳細化関係があるため、ハザードシナリオたちは木構造となる。

このように、抽象的ハザードシナリオを具体的ハザードシナリオへ詳細化するアプローチは、［6］3.4 Identifying causal factor scenariosでも紹介されている。しかし［6］では、機能CSDのみを用いて詳細化しているのに対し、STPA-SafeSecでは2つのCSDを用いて詳細化している点が異なる。

3 STPA-SafeSecによる分析事例

本節では、[4]の4、5節にある事例を解説する。文献[4]は事例としてマイクログリッドを用いており、とくに広域電力網と局所電力網の接続(併入)におけるハザード分析を実施している。事例対象の簡単な解説は、本稿のstep0準備2とstep1にある。また詳細な解説は、[4]と[7]を参照いただきたい。

3.1 Step0準備1

STPA-SafeSec Step0準備1では、安全に関する事柄に加えセキュリティに関する事柄を考える以外は、標準的STPAと同じである。この事例では、次のロスを識別している：

- L1：人間への危害
- L2：電力機器の損傷
- L3：ユーザの電器機器の損傷
- L4：停電

続いて、次のハザードを識別している(カッコ内は関連するロスを表す)：

- H1：非同期での系統併入(L1、L2、L3、L4)
- H2：電力機器の運転制限外での運用(L1、L2、L3、L4)
- H3：電力品質指標の逸脱
  - ▶ H3.1 電圧(L1、L3)
  - ▶ H3.2 周波数(L3、L4)
- H4：同期制御の不調(L4)
- H5：地域の電力需要への対応不可(L4)

更に、システムに対する高抽象度の安全制約を、ハザードの否定形を取ることで識別している。このとき、制約は安全制約(CSTR-Sn)、可用性制約(CSTR-An)、完全性制約(CSTR-In)のようにどのような属性に対する制約かを分けて番号付けしている。なおこの事例では、安全制約CSTR-S1からCSTR-S5(H1からH5の否定形)しか登場しないが、一般には可用性制約と完全性制約も扱う。

3.2 Step0準備2

STPA-SafeSecのstep0準備2では機能CSDを構築する。この機能CSDが標準的STPAのCSDに相当する。機能CSDにおけるコンポーネント(原文Node)はNnで、接続はCnで番号付けされる。なおstep2aで、この機能CSDを詳細化した物理CSDを構築し、2つのCSDのコンポーネントを対応付ける。従って、対応が分かりやすい番号付けが望ましい。

この事例では、とくに速度制御器が制御するコントロールループ図に着目し、機能CSD(図1)を構築している。

図1について解説する。速度制御器(N1)、ローカルPMU(N4、ローカルマイクログリッドにある電圧位相計測装置(Phasor Measurement Unit))、ホストPMU(N5)、速度制御器とローカル・ホストPMU間の接続(C4とC5)により接続されている。各PMUからは、電圧(Xm)、周波数(ω)、位相(φ)が周期的に送られてくる。速度制御器は、同期が取れているかを確認し、サーキットブレーカ(N6)へ再開が安全か否かを送信する(この事例では、サーキットブレーカは自動ではなく、操作員が相当する操作を実行すると仮定しているため、開閉命令ではなく、安全か非安全かの情報が送信されている)。また速度制御器は、原動機制御器(N2)を経由して、ジェネレータ(N3)へ運転設定値(set point、C1)を設定する。

3.3 Step1

STPA-SafeSecのstep1では、機能CSDからトーマス博士が提案した拡張step1[6]を用いて、UCAを識別する。ここで、コントローラである速度制御器(N1)が参照する変数は、ΔXm(t)(電

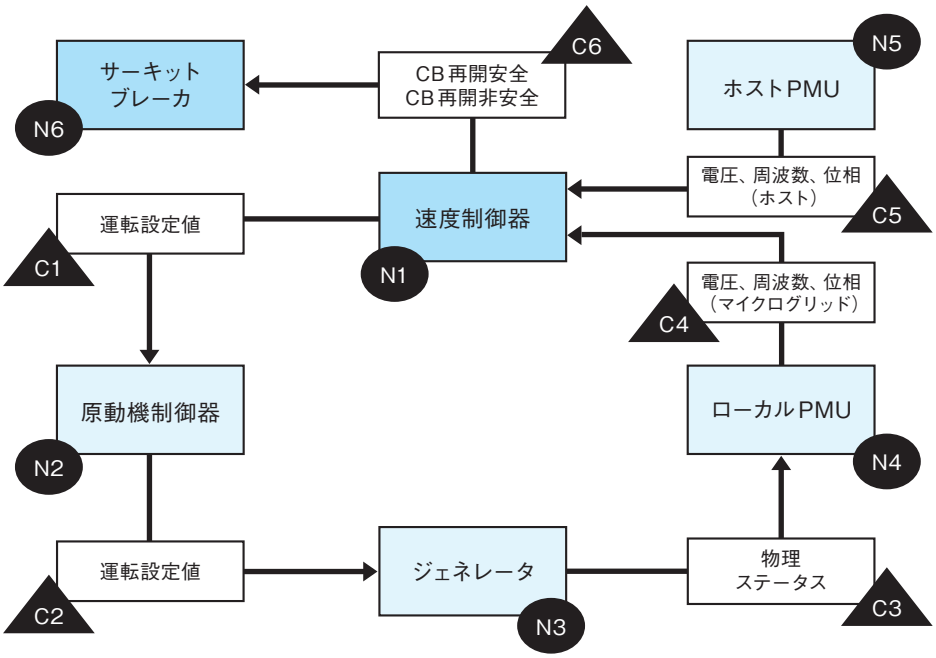


図1　機能コントロールストラクチャー図

圧差：ホストとローカルの電圧差、値は制限内、制限外）、Δω（t）（周波数差：ホストとローカルの周波数差、値は制限内、制限外）、Δφ（t）（位相差：ホストとローカルの位相差、値は制限内、制限外）、St_{cb}（サーキットブレーカの状態、値は開、閉）の４変数であり、速度制御器は、C_{sp}（原動機制御器（N2）への指示、値は運転範囲内、運転範囲外）、C_{cb}（サーキットブレーカ（N6）への連絡、値はCB再閉安全、CB再閉非安全）の２つのコントールアクションを指示する。

STPA-SafeSecは拡張step1を採用しているため、step1分析結果の記述形式が、標準的な記述形式［1］［2］と異なる。この事例のUCA1は、速度制御器のコントールアクションC_{cb}＝CB再開安全とハザードへ至る条件ΔXm（t）＝制限外の組み合わせに対し、ガイドワードProviding（Anytime）、Too early、Too lateのときにハザード（H1、H3）へ至ると記述されている（すなわちUCA1には、Too earlyとToo lateが一つのガイドワードであるとすれば、２つのUCAがまとめられている）。

速度制御器からのコントールアクションに対するstep1の結果は以下の通りである：

- UCA1：ブレーカが解放状態のとき、電圧差が制限外であるにもかかわらず、サーキットブレーカへCB再開安全をProviding,Too early,Too lateで指示（H1、H3）
- UCA2：ブレーカが解放状態のとき、周波数差が制限外であるにもかかわらず、サーキットブレーカへCB再開安全をProviding,Too early,Too lateで指示（H1、H3）
- UCA3：ブレーカが解放状態のとき、位相差が制限外であるにもかかわらず、サーキットブレーカへCB再開安全をProviding,Too early,Too lateで指示（H1、H3）
- UCA4：運転範囲外の設定値を原動機制御器に指示（H2）
- UCA5：ブレーカが解放状態のとき、運転範囲内の設定値を原動機制御器にToo late、Notで指示（つまり、設定値の更新が行われない）（H3、H4、H5）

3.4 Step2a：物理CSDの構築

STPA-SafeSec step2aでははじめに、機能CSDを物理CSDへ詳細化する。物理CSDは機能CSDをアーキテクチャレベルで実現した記述である。図２は図１の機能CSDを基に作成した物理CSDである。

元の事例では、物理CSDの要素（node）はNnの形で、接続はCnの形で表現される。また両CSDの要素間には対応が付けられる。本稿では、機能CSDと物理CSD間の対応の理解性向上のために、機能CSD内のNmと対応する物理CSD内のコンポーネントはNm-nと表記する。なお、機能CSDと物理CSDの要素は多対多対応のため、Nm-nとNm'-n'が同じ要素を表すことがある点には注意が必要である。

機能CSDと物理CSDの対応の一部を示す。N1（速度制御器）は、N1-1（速度制御器CPU）、N1-2（アナログ・デジタル変換器）とN1-3（N1-1とN1-2間のUSB接続）により構成される。またC5（ホスト電圧）は、C5-3（ホスト電圧）、C5-2（ファイアウォール）、C5-1（スイッチ）、C5-4（ホスト電圧、ローカル電圧）により構成される。

機能CSDと物理CSDの対応付け後に、続いて、安全にかかわる抽象的ハザードシナリオ（この事例ではSystem Flaw）を特定している（STPA-SafeSec X）。抽象的ハザードシナリオは、標準的STPA［1］の安全にかかわるHCFヒントを参考に、特定される。この事例では、以下の６つの抽象的ハザードシナリオを特定している。F1：速度制御器は電圧が制限内と誤認識、F2：速度制御器は周波数が制限内と誤認識、F3：速度制御器は位相角が制限内と誤認識、F4：原動機制御器は運転範囲外の設定値を受け

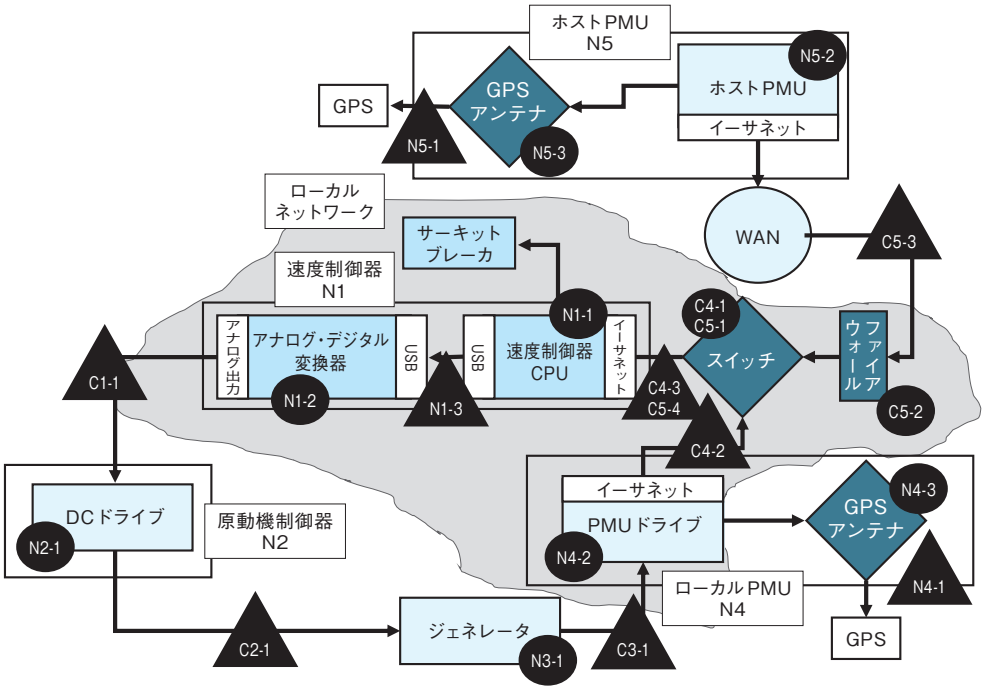


図２　物理コントロールストラクチャー図

取る、F5：速度制御器は設定値変更が要求されていないと誤認識、F6：遮断機制御器は「CB再開安全」という誤情報を受信。

3.5　Step2b：制約の詳細化

標準的STPAでは、この後にハザードシナリオを導出する。他方STPA-SafeSecでは、ハザードシナリオ導出の前に、機能CSDの要素へ安全・セキュリティ制約を課し、機能CSDと物理CSDの対応に基づき、安全・セキュリティ制約を物理CSDの要素に割り振っている。この安全・セキュリティ制約を破る要因がHCFとなる。

物理CSDの要素に対し、安全・セキュリティ制約を課す利点としては、例えば以下の利点が挙げられている。物理CSDのコンポーネントとしてGPSが使用されていることが決まれば、GPSの既知の脆弱性としてスプーフィング（spoof）とジャミング（jam）が知られているため、これらに対するセキュリティ制約（CSTR-A-1、CSTR-A-2）を要素N4-3、N4-1に課す必要があることが分かる。しかし、機能CSDではGPSが使用されるか否かは決定されていないため、これらの制約を課すべきか否かは決定できない。

STPA-SafeSec step2bでは、はじめに、安全・セキュリティ制約と機能CSDの要素を対応付ける。このとき、安全制約として識別済ハザードを用い、セキュリティ制約としてリスト1、2の制約を用いる。正確にはハザードやリスト中記述の否定形が制約である。

次にSTPA-SafeSec step2bでは、step2cで策定するハザードシナリオの理解容易性を高めるため、機能CSDの制約を物理CSDへ詳細化する。この事例では、幾つかのデバイスに対して制約の詳細化が示されているが、本稿では速度制御器（N1）に対する制約の詳細化のみを紹介する。

はじめに、ハザードより安全制約を導出し、リスト1、2よりセキュリティ制約を導出する。速度制御器に対しては、H1（非同期での系統併入）、H2（電力機器の運転制限外での運用）、H3（電力品質指標の逸脱）とH5（地域の電力需要への対応不可）が課される。次に、これらの安全制約とセキュリティ制約を、速度制御器の構成要素である、速度制御器CPU N1-1、アナログ・デジタル変換器N1-2とUSB接続N1-3に割り当てる。ここでは、H1、H3とH5は速度制御器CPUに、H2はアナログ・デジタル変換機に割り当てられる。

3.6　Step2c ハザードシナリオ策定

はじめに、トップレベルのハザードシナリオ（シナリオ1）を策定する。トップレベルのハザードシナリオは、step2aで識別した抽象的ハザードシナリオであり、それに関連するUCA（Hazardous Control Action）と機能CSDのコンポーネント及び物理CSDのコンポーネントから構成される。次に、トップレベルのハザードシナリオに含まれる各層の要素に着目し、ハザードシナリオを詳細化していく。このとき、詳細化されたハザードシナリオのHCFに対応する制約を合わせて記述する。下位シナリオ（シナリオ1.1以降）は、機能CSDのコンポーネント、物理CSDのコンポーネントに加え、安全制約、セキュリティ制約から構成される。

トップレベルのハザードシナリオから詳細化して策定されたハザードシナリオたちは、木構造を成す。木構造の上位ノードはより抽象的ハザードシナリオが対応する。すなわち、あるノードの子ノードには当該ノードに割り当てられたシナリオのサブシナリオが付される。このような木構造にすることで、あるハザードシナリオへの対応策は、木構造におけるそのノードの下位ノードの対応策になる。

本稿では、一部のハザードシナリオのみ紹介する。

シナリオ1：速度制御器は、（ローカル・ホスト間の）電圧差が制限値内と誤認識する。「ハザード：H1（非同期での系統併入）、H3（電力品質指標の逸脱）、抽象的ハザードシナリオ：F1（電圧差が制限内と誤認識）、UCA：UCA1（ブレーカが解放状態のとき、電圧差が制限外であるにもかかわらず、サーキットブレーカへCB再開安全をProviding,Too early,Too lateで指示（H1、H3））、機能CSD関連コンポーネント：N1、N4、C4、N5、C5、物理CSD関連コンポーネント：N1-1、N4-2、C4-2、C5-1、C5-2、N5-2、C5-3、C5-4）」

シナリオ1.1：速度制御器（N1）は、正しいフィードバックを間違って認識する。「機能CSD関連コンポーネント：N1（速度制御器）、物理CSD関連コンポーネント：N1-1（速度制御器CPU）、安全制約：デバイス（速度制御器CPU）とアルゴリズムの信頼性、アルゴリズムの正しさ、セキュリティ制約：CSTR-I5 Measurement injection（フィードバック（以下FB）信号へのインジェクション攻撃）、CSTR-I7 Measurement manipulation（FB信号の操作）」

シナリオ1.2：速度制御器は、ホストPMUからの間違った信号を受け取るが、それを正しいと認識する。「機能CSD関連コンポーネント：N1、N5、C5、物理CSD関連コンポーネント：N1-1、C5-1、C5-2、N5-2、C5-3、C5-4（注意：C4-3と同じ対象を指す）、安全制約：N5の信頼性、セキュリティ制約：CSTR-I5（FB信号へのインジェクション攻撃）、CSTR-I7（FB信号の不正操作）」

シナリオ1.2.1：ホストPMUが間違ったFB信号を送る。「機能CSD関連コンポーネント：N5、物理CSD関連コンポーネント：N5-2、安全制約：N5-2の信頼性、セキュリティ制約：CSTR-I5（FB信号へのインジェクション攻撃）、CSTR-I7（FB信号の不正操作）、N5-2への脆弱性攻撃成功（Successful exploit）」

シナリオ1.2.2：リモートPMUからの正しいFB信号が、ホスト電圧（C5）で改ざんされる、またはインジェクション攻撃される。N1-3の通信は正常であるとする。「機能CSD関連コンポーネント：C5、物理CSD関連コンポーネント：C5-3、N5-1、N5-2、安全制約：なし、セキュリティ制約：CSTR-I5（FB信号へのインジェクション攻撃）、CSTR-I7（FB信号の不正操作）」

シナリオ1.2.3：リモートPMUからの正しいFB信号が、ホスト電圧（C5）で改ざんされる、またはインジェクション攻撃される。N1-3の通信は異常だが受け入れられるとする。「機能CSD関連コンポーネント：N1、C5、物理CSD関連コンポーネント：N1-1、C5-3、N5-1、N5-2、安全制約：なし、セキュリティ制約：CSTR-I5（FB信号へのインジェクション攻撃）、CSTR-I7（FB信号の不正操作）」

4　今後の課題

本節では、STPAをベースに脆弱性分析を行う際の課題として、

step2で用いるHCF導出のヒントに関する課題を述べる。またSTPAでは分析時に妥当な仮定を置かずに分析を実施すると、分析対象が肥大化したり、分析者により分析結果が大きく異なったりといった状況に陥りがちである。そこでSTPA一般の課題として、分析時の仮定について述べる。

4.1 セキュリティにかかわるHCFヒントに関する課題

STPA-SafeSecでは、標準的STPA step2で利用されるHCFのヒントに加え、セキュリティにかかわるHCFを導出するために、リスト1、2にあるヒントを利用する。他方、STAMP WorkbenchやSafetyHATといったSTAMP/STPA支援ツールでは、HCFヒントを分析対象領域に依存して適切に変更でき、更に分析者が独自に編集できる。例えば、[8]にあるHCFヒントは機械のコントローラを想定しており、機械のコントローラに対しては適切なヒントであるが、人間のコントローラに対しては異なるヒントのほうがHCFを導出しやすいであろう。従って、セキュリティにかかわるHCFヒントも、適宜修正・変更することで、HCFを導出しやすくなると考えられる。

STPA-SafeSecで採用されているセキュリティにかかわるHCFヒント以外にも、例えば、セキュリティにかかわるヒントとしてSTRIDE[5]の利用が考えられる。STRIDEは、Spoofing(スプーフィング)、Tampering(改ざん)、Repudiation(否認)、Information Disclosure(情報漏えい)、Denial of Service(サービス拒否)、Elevation of Privilege(特権の昇格)の頭文字から成り、それぞれは代表的な脅威、すなわちセキュリティにかかわるHCFヒントを表す。

4.2 分析時の仮定に関する課題

STAMP/STPAで解析を行うときに一般に難しい点は、どの抽象度とどの仮定のもとでコントールストラクチャやHCFの設定を行うかであろう。モデル化を行う際にはある程度のドメイン知識を暗黙裏に仮定する。この仮定の妥当性は解析とモデル化を繰り返すことにより補強するのが現在の標準的な手順である。この際にknown-unknownsやunknown-knownsなどの仮定の境界上の事項[9]を意識することが強く望まれる。

STPA-SafeSecでは物理CSDの導入によりunknown-knownsの気づきに貢献している。例えば、「物理CSDのコンポーネントとしてGPSが使用されていることが決まれば、GPSの既知の脆弱性としてスプーフィング(spoof)とジャミング(jam)が知られている」というのは「GPSの既知の脆弱性としてスプーフィング(spoof)とジャミング(jam)が知られている」というドメイン知識を解析者のunknown-knownsからknown-knowsへの変換に寄与しているとみなすことができる。

一方、known-unknownsについては次のような対策が取れる。known-unknownsについては典型的には定性的要因が分かっているが、定量的な値が不明であるという特徴を持つことが多い。その場合は値に関する変数を不定値とみなしたり、あるいは統計的量として捉えることによりモデル化できることがある。その場合はそれぞれに適した数理モデルや解析手法の活用が可能となる。

5 まとめ

本稿では、STAMP海外事例として、STPA-SafeSecを紹介した。STPA-SafeSecは、安全性とセキュリティを統合して分析するためのSTPA拡張であり、機能CSDと物理CSDを持ち、STPA step2で使用するHCFヒントをセキュリティ拡張したという特徴を持つ。併せて紹介したSTPA-SafeSecの適用事例はSTPA-SafeSecの有用性を示している。しかし、セキュリティにかかわるHCFヒントはリスト1、2のヒント以外にもSTRIDEを活用するといったことも考えられる。また、既存のセキュリティ分析手法とSTPA-SafeSecを統合し、分析結果を充実させるといった点にも改善余地はあると考えられる。

【参考文献】

[1] LevesonG.Nancy. (2011). Engineering a Safer World: Systems Thinking Applied to Safety. MIT Press.

[2] システム安全性解析WG. (2016). はじめてのSTAMP/STPA. 情報処理推進機構.

[3] YoungWilliam,LevesonNancy. (2013). Systems Thinking for Safety and Security. In Proceedings of the 29th Annual Computer.

[4] Ivo FriedbergMcLaughlin,Paul Smith,David Laverty,Sakir SezerKieran. (2017). STPA-SafeSec: Safety and security analysis for cyber-physical systems. Journal of Information Security and Applications.

[5] マイクロソフト. (日付不明). モノのインターネットのセキュリティ アーキテクチャ.
参照先 :https://docs.microsoft.com/ja-jp/azure/iot-hub/iot-hub-security-architecture

[6] ThomasJohn. (2013). EXTENDING AND AUTOMATING A SYSTEMS-THEORETIC HAZARD ANALYSIS FOR REQUIREMENTS GENERATION AND ANALYSIS. Ph.D Thesis,MASSACHUSETTS INSTITUTE OF TECHNOLOGY.

[7] IvoFriedberg,DavidLaverty,KieranMacLaughlin,PaulSmith. (2015). A Cyber-Physical Security Analysis of Synchronous-Islanded Microgrid Operation. Proceedings of 3rd International Symposium for ICS & SCADA Cyber Security Research.

[8] LevesonG.Nancy, ThomasJohn. (2013). An STPA Primer.

[9] Sebastian ElbaumS. RosenblumDavid. (2014). Known unknowns: testing in the presence of uncertainty. Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering.

STAMP初心者を卒業する

有人宇宙システム株式会社　野本 秀樹

1 はじめに

ナンシー・レブソンによる“A STPA Primer”[1]は、STAMPの入門書として非常に優れたものである。ここに記された様々な事例や深い洞察の助けを借りてSTAMPのアナリストは多くの安全分析を行ってきた。

しかし、“Primer”は、文字通り「初心者のための手引き」である。プロフェッショナルなSTAMPアナリストになるためには、ここを卒業し、その先に向かって道を切り開いていかねばならない。

本稿では、STPA Primerからどちらの方向へ向かうべきかを、Primerを参照しながら論じてみたい。

2 STAMPの神話

STAMPを始めたばかりのアナリストが陥りやすい間違いが、このSTPA Primerに書かれている。

“FAQ: Does the control structure always have to be hierarchically linear, like a ladder?
No. The control structures in this primer are only examples from various applications, and yours may be different. Although some control structures look like a ladder, like the generic operations and development structure shown in Figure 1.5 or the simple Aircraft -> Pilots -> ATC structure for the NextGen ITP example in Figure 2.2, this is not always the case. For example, the high-level control structure for the Gantry 2 radiation therapy machine shown in Figure 2.9 shows how control and feedback paths can skip across levels in the control structure.” (“Primer” p.41)

よくある質問：制御構造図は常にきれいな階層構造である必要がありますか？
答え：いいえ。

STPA Primerをはじめ、多くの論文に現れる制御構造図は制御構造が階層化されており、常にControlled Processは単一のControllerを持っている。しかし、現実は必ずしもその通りではない。Primerにも書かれているように、安全を制御している構造というものは、もっとバラエティに富んでおり、レブソンの言う創発的な問題を生む可能性のあるシステムにこそ、通常とは異なる複雑な制御関係が存在していると考えるべきである。本稿では、Primerに一個所だけ書かれている制御構造図のないセクションに書かれた事例を対象に分析を行う。この事故事例について、詳細の制御構造図があえて書かれておらず、読者の今後の課題のように紹介されているのは、この事例が正に単純ではない制御構造の事例であり、初心者であることを卒業するための、読者への「宿題」だからである。

3 Multiple Controllers

“Primer” p.20にある事例には唯一制御構造図が書かれていない。そこで、Primer卒業のため、ここで、どんな制御構造図が書けるのか試してみる。

事故の概要は、以下のようなものである[2]。

2002年、ドイツユーバーリンゲンで起こった航空機の衝突事故である。2機（2937便と611便）はそれぞれ2937便が真西に向かって、611便は真北に向かって飛行中であった。

それぞれには2種類の安全コントローラが存在している。

TCASとATCである。そのうちの一つTCAS（Traffic Collision Avoidance System）は、他機の接近を検知して、パイロットに対

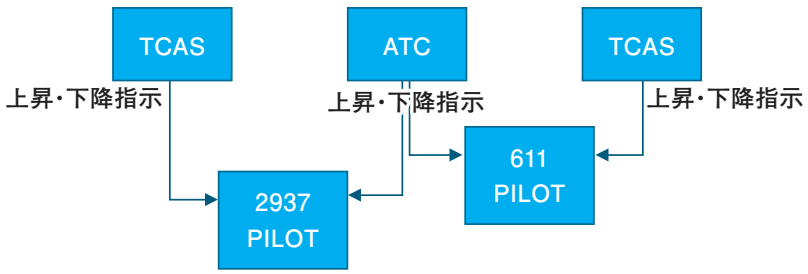


図1　TCAS/ATC/PILOT間の制御構造

して警告を発し、上昇・降下・方向・速度調整を指示する自動システムである。

もう一つのATC（Air Traffic Control）は、航空機の運航を取り仕切る管制業務の一般名称であり、こちらは、航空管制官が状況によってパイロットに指示を出す。

2つの安全システムは互いに独立であり、どちらかがどちらかに従属していないため、システムエラーや管制官のミスなどに互いに影響されない。つまり、完全な独立二重系による安全性の向上を目指した制御構造である。

2機は、それぞれがTCASとATCによって安全を制御される図1のような関係にあった。

まず、2機に搭載されたTCASが双方の機影を検知した。7秒後に地上のATC管制官が危険を察知し、それぞれに「2397は上昇」「611は下降」の指示を出した。

一方、機体搭載のTCASは、2397に対しては「下降」、611に対しては「上昇」を指示していた。

結果、2397は管制官の指示に従い「上昇」、611もTCASの指示に従い「上昇」し、両機が空中で衝突した。

この事故を分析するとき、制御構造図に書かれた「上昇指示」や「下降指示」を個別に分析しても、得るものは少ない。「上昇指示が来ない場合」も「上昇指示が遅れる場合」も、「間違った上昇指示が来る場合」も、すべてハザードの発生要因である。しかし、ここでの最大の問題は、そのような個々の問題ではない。2つのcontrolled processに対する2つのcontrol actionが、2つのcontrollerから同時に発せられるときの問題である。もし、2つのcontrollerからの指示が逆になってしまうと、2機は衝突してしまうという問題に言い換えることができる。

正常状態の組み合わせは表1の4通り、更に、TCAS故障時の組み合わせは表2の通り8通りである：

表1　正常状態におけるATC指示値とTCAS指示値の影響

	ATC指示値		TCAS指示値		衝突?
	2397向け	611向け	2397向け	611向け	
(1)	↑	↓	↑	↓	
(2)	↑	↓	↓	↑	衝突
(3)	↓	↑	↑	↓	衝突
(4)	↓	↑	↓	↑	

表2　TCAS不調時におけるATC指示値とTCAS指示値の影響

	ATC指示値		TCAS指示値		衝突?
	2397向け	611向け	2397向け	611向け	
(5)	↑	↓	—	↓	
(6)	↑	↓	—	↑	衝突
(7)	↓	↑	—	↓	衝突
(8)	↓	↑	--	↑	
(9)	↑	↓	↑	—	
(10)	↑	↓	↓	—	衝突
(11)	↓	↑	↑	—	衝突
(12)	↓	↑	↓	—	

表1、2に示したように、2つのcontrollerによる指示を出すシステムは、正しい指示が行われている場合に限定したとしても、確率50％でしか保証されていない（表1）。もともとTCAS不能時のために維持されているATCであるが、TCAS故障状態を安全化できているとは言えない。確率は相変わらず50％である（表2）。もし確率50％であれば、この安全装置があることで得られる安全上の利得はゼロということになってしまう。どのようにTCASの信頼性を上げても、この確率は改善できない。表1＋表2の50％が表1単独の50％に戻るだけである。

現在では、この事故をきっかけにルールが明確になり、「TCASと地上管制官の指示に食い違いがある場合はTCASに従う」ことになっている。すなわち、制御構造は以前のままの3頭立てのcontrollerであるが、PILOTのプロセスモデルに、「両者が食い違った場合は、TCASを優先する」というロジックが追加された。

しかし、実は、肝心のTCAS故障時に、このルールは安全性を高めない。

TCASが故障している場合は、表2のように、パイロットは地上管制に従わざるを得ないが、それでは、この事故が全く同じ理由で再発してしまうからである。現在のルールのように、TCASの信頼性を多とし、地上管制をヒューマンエラーの発生源のごとく扱うルールでは、上記の表1の問題が解決されるにすぎないのである。

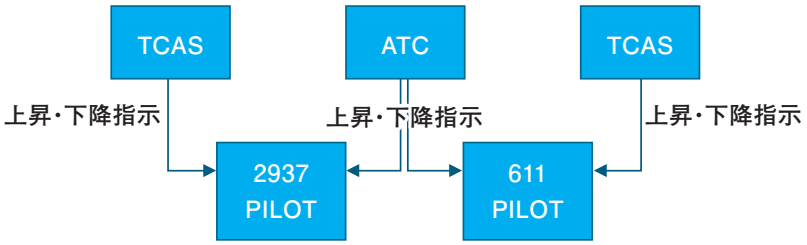


図2　ルール制定後の制御構造は同じ
パイロットのプロセスモデルが複雑化した

図1の制御構造が変更を受けないまま、図2のように、人間の持つべきプロセスモデルが複雑化される例は多い。ヒューマンエラー防止のための自動化が行われているが、完全自動化ではなく、システム故障時には人間に責任が帰ってくる自動運転などがその典型例である。通常のドライバーであれば難なく安全化可能と思われるシチュエーションでも、自動運転モードが突然手動運転モードへ遷移すると、ドライバーの多くは安全な回避行動を取ることができなくなることが、自動車技術会（JARI）の実験で明らかになっている[3]。

4　問題の解決

このような問題点を解決するためには何が必要になるであろうか。

STAMPが提示した新しさこそが、この問題を解決する決め手となる。STAMPの神髄は「安全制御構造」のモデル化である。従って、我々は安全制御構造そのものを見直すことによって安全化を目指す。図2のような独立した3頭立て馬車による制御ではなく、図3のように2つの情報源であるTCASとATCが協調関係を築く制御構造が必要となる（“Primer” p.20 Figure 1.9）。

しかし、この「調停」による協調をTCAS側に担わせることは困難である。そもそもTCASが故障した場合には、そのような仕組みは役に立たない。

従って、これは人間が担うべき役割となる。管制官は2機のTCASの指示を入手する、若しくは、2機のTCASと全く同一の答えを出すTCASシミュレータを地上に持ち、その指示と同じ指示をパイロットに指示するという仕組みが必要となる。図3の青太矢印の「調停」がこのために追加となる。これは、制御アクションの追加という小さな変更ではなく、TCAS中心の制御構造からController間の協調による制御構造への変革を意味する。この制御構造の変更によって初めて、TCAS不能時にはATCが、ATC不能時にはTCASが安全化を担保できる真の二重系となり、正常時にも異常時にも高い信頼性を発揮できるようになるのである。ちなみに、このような協調安全はSafety2.0と呼ばれている[4]。Safety 2.0の目指す安全は、これまでのような「ヒューマンエラー防止のための自動化」という、図2のようなものではなく、図3で実現される「人と機械の協調的安全」である。機械が人のエラーを監視するという対立構造ではなく、機械と人が互いに助け合う関係が、新たな時代の安全化の主流となると期待されている。

5　まとめ

STAMPの初心者を卒業し、プロフェッショナルなアナリストになるためには、制御構造の分析眼と真に安全な制御構造を設計できる能力、すなわちシステムズエンジニアリング能力が必要になる。本論で示したように、現在安全と信じられている仕組みも、制御構造を分析してみれば、実は問題を抱えている可能性がある。事故後に修正された航空管制のルールはTCASという自動警報システムを管制官の判断に優先させよ、というものであるが、この修正では真の安全は達成できない。真の安全を達成するためには、3頭立て馬車のいずれかを優先させるのではなく、3頭の馬の間の協調を「構造の中に創り込む」ことが必要である。これこそ、レブソンがかねてより主張している「システム理論に基づく安全化」であり、新しい安全、Safety 2.0への道である。

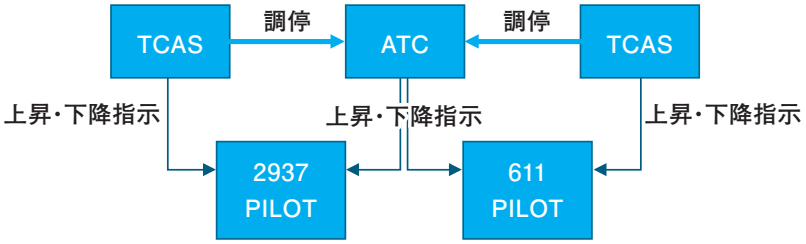


図3　新たな制御構造

【参考文献】

[1] Leveson (2013), “A STPA Primer v.0” http://sunnyday.mit.edu/STPA-Primer-v0.pdf

[2] Gallagher Paul (2002) “Jet pilot’s 14 seconds dilemma before fatal crash”, The Scotsman (Edinburgh, UK) (http://news.scotsman.com/germanplanecrash/Jet-pilots-14-seconds-dilemma.2341758.jp)

[3] 本間, 若林, 小高, (2017)「高度自動運転における権限移譲方法の基礎的検討（第4報）」公益社団法人自動車技術会2017年秋季大会学術講演会

[4] 中村英夫（2017）IoT時代の新しい安全「Safety 2.0」の全貌 ET2017 独立行政法人 情報処理推進機構 SEC先端技術入門ゼミ

STAMPワークショップに関する活動報告

SEC調査役 十山 圭介　SEC調査役 三原 幸博

■European STAMP Workshop 2017参加報告

2017年9月13日～15日にレイキャビク大学（レイキャビク、アイスランド）で開催されたEuropean STAMP※1 Workshop（ESW）2017において、IPA/SECのIoTシステム安全性向上技術WGでの検討成果を発表すると共に、ほかの参加者の講演・発表を聴講・調査して、欧州を中心とするSTAMPの状況や動向について情報収集を行った。また、日本でのSTAMPワークショップに参加いただくことなど、ESW主催者団体と今後の連携樹立について打ち合わせを実施した。

1 ESWとは

MITでの最初のSTAMP Workshopの1年後、2013年から欧州においてもSTAMP Workshopが開催されており、ドイツやオランダ、スイスなど欧州内持ち回りで毎年実施されている。また、ワークショップの開催を含め、欧州でのSTAMPに関する産学の活動を支援するステアリングボード（ESSB※2）が構成されている。

2 ESW2017の内容

今回のワークショップは第5回であり、16カ国から参加者97名を得て、講演22件、ポスター3件の発表があった。日本からはIPAの2名を含めて4名の参加であった。プログラムは、初日がSTAMPとSTPA※3に関するチュートリアルで、2日目がキーノートと講演、3日目が講演という構成である。

2.1 キーノート

Nancy Leveson教授のキーノートは「STAMP：Where To From Here？」と題してSTAMP/STPAのこれからの方向を示そうとするものであった。Leveson教授のこれまでの講演でも触れられているが、

- リスク分析に基づく意思決定手法
- STPA/CAST※4の標準化
- STPA/CAST利用のハンドブックの作成
- 産業界での利用
- STPA/CAST教育のオンラインコースの作成

の必要性が挙げられた。また、システムズエンジニアリングへのSTPAの統合として、概念開発段階で始めることの必要性が述べられた。

2.2 講演

講演内容を大まかに分類すると、①STAMPの考えの適用や比較を行うもの、②STPAの適用報告、③STPAの拡張、④CASTの適用の4つである。今回のワークショップでは、STPAやCASTで解析した結果を提示する発表より、STPA手法をどのように適用していくべきかといった方針に関する発表が多かった。IPA/SECからの発表は、STPAに対するヒューマンファクタを考慮した拡張提案であり、日本からのもう一件の発表は長崎県立大学の日下部教授（IoTシステム安全性向上技術WG委員）によるセキュリティ侵害に対するSTAMPアプローチに関するものであった。

特記すべき講演は下記のものである。

(1) **Maritime Spatial Planning as a tool for ecosystem-based adaptive safety management of maritime transportation system in the Gulf of Finland (Baltic Sea)**
（タルトゥ大学／エストニア ほか）

海洋空間計画プロセスにSTPAを適用し、Safety guided Designを行うもの。不確実な状況（要求仕様）で安全性の評価ができるか（安全性がどこまで達成できるか、想定外がどのくらい残るか）を考える。

(2) **Extending the Human-controller methodology in Railway System Accident Analysis based on STAMP**
（北京交通大学／中国）

MITでThomas博士らが提案している手法をベースに、human controllerの状態の解析を拡張してHCFを導出する手順を提案するもの。

(3) **Missing no Interaction – Using STPA for Identifying Hazardous Interactions of Automated Driving Systems**
（シュトゥットガルト大学、コンチネンタル社／ドイツ）

コンチネンタル社の高速道路における高度自動運転技術に対し、ヒューマンエラーや合理的でない機能によるハザードがな

いことを解析するためにSTPAを適用したもの。

⑷ STPA applied to a radiotherapy process - first steps and lessons learned（アムステルダム自由大学／オランダ）

発表者はアムステルダム自由大学でSTAMPの講義を実施している。講演内容は、HFMEA※5とSTPAとの比較作業を含む実習教材とレクチャーをradiotherapy（放射線治療）の例で紹介。human controllerを含んだ分析となっている。

3 欧州STAMPソサエティでの人脈構築

ESSBボードメンバを日本での第2回STAMPワークショップに招聘するべく、Nectarios Karanikas博士（ESSBに日本での講演を依頼して紹介された方）と打ち合わせを行い、欧州での標準化関連を含めてSTAMP適用や普及の状況、人や組織に関する適用について講演いただくことでご確認いただいた。

IPAと欧州STAMPグループ両者の協力関係や、とくに日本のワークショップで海外からの講演をどのように扱うかなどについては、Karanikas博士来日時も含め、今後意見交換していくことにした。

4 感想と今後の取り組み

STPAやCASTを適用した結果を報告する発表より、リスク分析に基づく意思決定手法に関連して、様々な分野に適用する方針の発表が多いと感じた。2017年3月のMITのSTAMP WSではWorkplace Safetyがそれに対応する主な項目であったが、今回は、海難事故や地熱発電でのリスク管理などに関して、現状がどうであり、課題はどこかといった点を挙げて議論しようとするものであった。

今回、IPA/SECからはヒューマンファクタに関する発表を行った。発表時間外にも質問やコメントがあり、今後もこのような場でIPA/SECの活動を印象付けると共に、情報収集することが大事であると感じた。

脚注

※1 Systems-Theoretic Accident Model and Processes
※2 European STAMP Steering Board
※3 Systems-Theoretic Process Analysis
※4 Causal Analysis based on STAMP
※5 Healthcare Failure Mode and Effect Analysis（保険医療におけるFMEA）

キーノート

レイキャビク大学

■第2回 STAMPワークショップ開催報告

IPA/SECは、2017年11月27日～29日の3日間、慶應義塾大学 三田キャンパス北館ホールにおいて、慶應義塾大学、有人宇宙システム株式会社（JAMSS）、一般社団法人情報処理学会の共催、一般社団法人組込みシステム技術協会（JASA）の後援で第2回STAMPワークショップを開催した。

1 ワークショップの概要

IPA/SECでは2015年よりシステム理論による事故モデルSTAMPとそれに基づいた解析手法STPAに注目し、産業界に有効なシステム安全性向上手法の調査・検討と普及に関する活動を委員会（WG）によって進めており、2016年には第1回STAMPワークショップを九州大学と共催している（参加者：130名、招

待及び一般講演：22件)。第2回となる今回のワークショップは、4カ国から181名の参加で、基調講演と招待講演、海外からの1件を含む講演24件とポスター2件の発表を得ている。また、IPAで開発中のSTAMP支援ツール「STAMP Workbench」の紹介デモも実施した。プログラムは表1に示す通りである。

2 内容

2.1 基調講演と招待講演

MITのThomas博士による基調講演は、STAMP及びSTPAのイントロダクションと踏切での制止用バリアを題材とした演習を行い、最新トピックスとして自動運転への適用などが紹介された。欧州STAMPステアリングボード(ESSB)のアムステルダム応用科学大学Karanikas博士による招待講演は、欧州におけるSTAMP Workshopの開催状況やSTAMPに関するコミュニティや推進委員会、教育、研究などの実情が紹介された。

2.2 講演内容と結果

講演は産業界から13件、学術界から11件あり、内訳はSTPAの適用事例とSTPAの拡張や活用法に関するものが8割以上で、ツールなどが4件、CAST関連が1件であった。対象分野では、自動車、鉄道が多くを占めている。IPA/SECのWGメンバーからは、鉄道踏切システム、電動アシスト自転車を対象とした解析事例や要求仕様記述への適用、プロジェクト管理への適用、国際安全規格への適用の考察、IPAで開発中のSTAMP支援ツールの紹介を行った。

一般講演も含めて、「STAMP/STPAがどう役立つか」「安全設計手法との組み合わせ」「人間と機械との制御行動の検討」「ツールに関するテーマ」などが発表され、活発な議論が行われた。

実施したアンケートによると、およそ8割の参加者がSTAMPを業務に有効と感じており、STAMPを導入済みもしくは検討中である組織が6割近くとなっている。本ワークショップへの参加理由でも半数が「自部門の業務と関係が深い」であり、STAMPの着実な浸透を感じさせている。

3 今後に向けて

参加者や発表数が増加し、STAMPに対する関心の高まりを感じさせるワークショップであった。一般からの発表は今回がほぼ初めてであり、今後、各所での実践や検討が深まるにつれ、発表される産業領域の拡大やベストプラクティス事例などによって、得られる事項の共有や比較なども可能になると期待できる。

発表テーマの編成や開催期間、開催場所やワークショップの形態に関して、また、米・欧のSTAMP Workshopとの連携など検討を深めて今後につなげていきたい。

なお、本ワークショップの基調講演や招待講演、発表の詳細な内容がIPAのWebサイト※6で公開されているので、ご参照いただければと思う。

脚注

※6 https://www.ipa.go.jp/sec/events/20171127.html

表1 プログラム

11月27日		
9:45～14:15	基調講演	
	STAMP and STPA introduction	John Thomas博士 (MIT)
	STPA Exercise	
	Advanced STPA topics	
14:30～15:30	招待講演	
	Situations of STAMP in Europe	Nectarios Karanikas博士 (欧州STAMPステアリングボード)
15:40～17:00	講演3件	
17:20～18:00	STAMPツールデモ	
11月28日		
9:00～17:40	講演16件	
11月29日		
9:00～11:30	講演5件	

会場の様子

Embedded Technology 2017/ IoT Technology 2017出展報告

SECプロモーショングループ 主任 荒川 明夫

IPA/SECは、2017年11月15日～17日の3日間、パシフィコ横浜展示ホールで開催された「Embedded Technology 2017（ET2017）」及び同時開催の「IoT Technology 2017」に出展した。また、パシフィコ横浜会議センターでは、IPAセミナーを3日間8部構成で開催した。

1 展示会概要

Embedded Technology（ET）とは、一般社団法人組込みシステム技術協会（JASA）が主催する最新テクノロジーの専門技術展であり、組込みシステム開発にかかわる技術者や開発者向けに最新技術などの情報を発信している。

また、IoT技術の最新動向を発信する展示会であるIoT Technologyが同時開催された。

2 出展概要

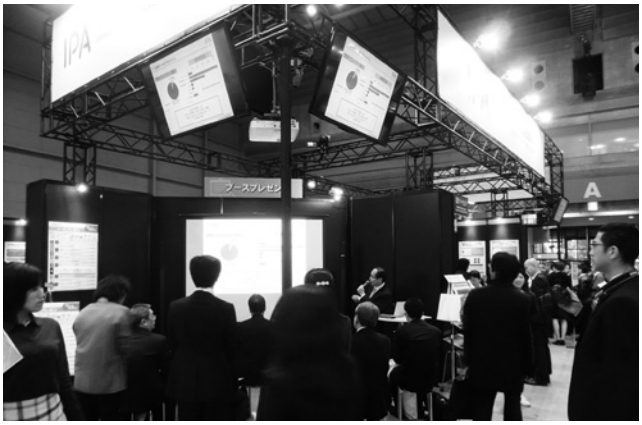
IPA/SECは、事業成果の普及・啓発を目的として、本展示会に毎年出展している。

本年は、IPA展示ブースにてブースプレゼンを3日間38回、SEC先端技術入門ゼミを3日間16回、隣接会場で行ったIPAセミナーを3日間8回、計62セッションを実施した。

また、SECの事業を中心にIPAで取り組んでいる組込みやIoTに関連する事業のパネル展示や資料配布、開発中のSTAMPツールのデモンストレーションを実施し、事業成果の普及啓発を図った。

3 IPA展示ブース

会期中は、展示ブースでブースプレゼン・SEC先端技術入門ゼミ・パネル展示・資料配布・STAMPツールのデモンストレーションを実施した。



展示コーナーでは、新刊である「組込みソフトウェア開発データ白書2017」をはじめ、IoT時代のソフトウェア開発におけるポイントをまとめた事業成果の展示や障害情報の共有促進、システム再構築時の上流工程強化策、コーディング作法ガイドなどのパネルを配置した。また、SECのみならず、情報処理技術者試験センターで行っている国家試験（情報処理技術者試験・情報処理安全確保支援士試験・エンベデッドシステムスペシャリスト試験）の紹介、2017年4月に設立した産業サイバーセキュリティセンターの紹介、セキュリティセンターから制御システムのセキュリティリスク分析ガイド、未踏グループからスーパークリエイタの事業を紹介するなど、各担当職員より来場者に説明を行った。

とくに「STAMP」や「システムズエンジニアリング」への関心が高く、関連したセッションを聴講する来場者の姿が多く見受けられた。

4 ブース内ステージ

今回も展示ブース内に２ステージを設け、ブースプレゼンとSEC先端技術入門ゼミを並行して実施した。ブースプレゼンは、IPA職員や関係者による事業成果の説明を中心に構成し、SEC先端技術入門ゼミは、外部有識者による最新技術動向や事例などの紹介を行った。

多くの方が両ステージの聴講に見え、３日間で延べ2,177名が参加した。

参加者からは「指標となるものを持っていなかったため、非常に勉強になった」「ユースケースが分かりやすかった」（ブースプレゼン）、「大変有用で参考になるゼミだった」「講演資料をウェブに公開してもらえるので、大変役立つ」（SEC先端技術入門ゼミ）などの意見が寄せられた。

5 IPAセミナー

展示ブースの運営と並行して、パシフィコ横浜の会議センターでは、会期中の３日間IPAセミナーを８部構成で開催し、延べ1,005名の参加があった。

今回のIPAセミナーは、各日異なるテーマにて実施した。

- 11月15日（水）テーマ：
「IoT/AIで世界がどう変わっていくのか？
どう変えていくのか？」

- 11月16日（木）テーマ：
「つながる世界の安全・安心をデザインするには？」

- 11月17日（金）テーマ：
「ソフトウェアの開発現場から見えたデータを
いかに活用するか？」

初日は、理事長の富田より「頼れるIT社会の実現を目指して」と題して、IPA全体の事業や社会環境について講演した。次に昨年に引き続き、2017年度 SEC journal論文賞表彰式を行った。表彰式の後、受賞者はIPA展示ブースにて受賞者講演を実施し、論文の内容を紹介した。更に、新進気鋭の若手起業家・技術者をお招きし、トークセッションを実施した。この模様は、IPA/SEC Webサイトに公開したので、是非ご覧いただきたい。

２日目は、IoT社会でいかに安全・安心を実現するかについて、IPAの取り組みを理事の川浦より紹介し、国立大学法人名古屋大学の高田教授並びに情報セキュリティ大学院大学の後藤学長にご登壇いただいた。

３日目は、メトリクスの切り口から、IoT時代のソフトウェア開発について、IPA/SEC所長の松本より紹介すると共に、沖電気工業株式会社エバンジェリスト五味氏、早稲田大学の鷲崎教授にご登壇いただき、「組込みソフトウェア開発データ白書2017」についてなど、SEC事業成果の普及を図った。

6 出展を振り返って

今回の出展は、展示ブース位置の変更、小間数縮小、IPAセミナー会場変更と、これまでの出展に比べ、コンパクトかつ効率的な来場者とのコミュニケーションが求められた。展示コーナーのスペースを拡張し、IPAセミナーの開催日数を増やしたことにより、昨年同様、多くの方にIPA展示ブースに足をお運びいただいた。来場者の方から直接多くの貴重なご意見をいただいており、これらを次回以降の出展や今後の事業の参考としたい。

7 謝辞

SEC先端技術入門ゼミ、ブースプレゼン、IPAセミナーにご登壇いただいた外部講師の皆様、並びに関係団体の皆様には、本展示会出展に際し、多大なるご支援を賜りました。ここに深謝いたします。

▶Embedded Technology 2017/IoT Technology 2017
ーIPA/SEC Webサイトー
https://www.ipa.go.jp/sec/events/20171115.html
- IPAセミナー・ブースプレゼン・SEC先端技術入門ゼミの講演資料がダウンロードいただけます（一部）
- 講演の様子を動画でご覧いただけます（一部）

日本のソフトウェア産業と技術者の現状を国際的に評価する：
ソフトウェア技術者の5カ国調査結果の分析

同志社大学 総合政策科学研究科　中田 喜文

1 日本のソフトウェア産業の現状

1.1 ソフトウェア関連財・サービスの貿易収支

我が国のソフトウェア産業の現状評価を行う場合、2つの視点が存在する。一つは、マクロ経済の視点である。そこでの注目点は、マクロ経済レベルでの世界の競合国との相対的な評価となる。そのような評価にふさわしい指標は、ソフトウェア及び関連情報サービスの国際的な収支状況である。そこで、日米間の貿易収支の状況を見てみよう。表1がその収支数値である。ここでは、直近の統計データが存在する2016年の数値と約10年前の数値を比べることで、現状とその変化が見えてくる。これを見ると、日本のソフトウェア産業がいかにアメリカとの比較において競争力が劣っているか、一目瞭然である。この表が示す日本の大幅な輸入超過とその近年の拡大傾向は、対欧州においても同様に確認できる。マクロの指標による評価では、日本のソフトウェア産業の国際競争力は、欧米諸国と比べ劣位にあると言える。

表1　ソフトウェア及び関連情報サービス日米貿易収支　（百万ドル）

	2007年	2016年	2016年/2007年 増加率
輸入	856	1,459	70%
輸出	620	447	-28%
収支	-236	-1,012	329%

出典：Bureau of Economic Analysis, U.S. Department of Commerce

1.2 ソフトウェアと企業の市場価値

では、視点を企業というミクロな経済単位に移すと、日本企業の国際競争力はどのように評価できるだろう？産業全体で見て国際競争力が低くとも、個別の日本企業の中には、独創的なソフトウェアを開発したり、あるいは既存のソフトウェアを用いて、国際的にも大きな経済価値を創出している企業が存在するのだろうか？ 表2には、世界で最も大きな企業価値を持つ企業上位5社を示した。アップルをはじめとするこれら最上位企業は、すべて米国企業であり、かつソフトウェアから大きな企業価値を創出している企業が並んでいる。他方日本企業の中で、最大の企業価値を有するのは、42位のトヨタである。今日乗用車の上位車種では、その製品開発コストの大半が各種走行機能を管理するソフトウェアの製造費用と言われている[1]。その意味で、正にトヨタは日本を代表する、ソフトウェアから大きな企業価値を創出する企業と言えよう。とは言え、世界的に見た企業の価値創出において、日本企業の存在感が希薄である実態は否めない。

表2　世界時価総額上位企業　（2017年12月末時点）

	社名	企業価値（米十億ドル）	国籍
1	アップル	868.90	米国
2	アルファベット（Google）	729.50	米国
3	マイクロソフト	659.90	米国
4	アマゾン・ドット・コム	563.50	米国
5	フェイスブック	512.80	米国
42	トヨタ自動車	189.10	日本

出典：180.co.jp

2 日本のソフトウェア技術者の現状（1）：彼らの生産性は高いのか?

以上、マクロ・ミクロの経済指標で見る限り、日本のソフトウェア産業の国際競争力の高さを示唆するよりも、むしろ低さが示唆されたが、視点を変えて、ソフトウェアを制作し、活用するソフトウェア技術者の目線で日本のソフトウェア産業の国際競争力を見てみよう。

ここでは、ソフトウェア技術者による自身の生産性に関する評価から、日本のソフトウェア産業の国際競争力、生産性を評価することとする。用いるデータは、IPA/SECの委託研究で2016年に実施した日中米独仏5カ国のソフトウェア技術者に対するアンケートによって収集したものである[2]。

2.1 ソフトウェア技術者の生産性自己評価比較

このアンケートでは、5カ国のソフトウェア技術者に対して、職務での能力発揮、職務成果、及び職務の重要度についての自己評価を行ってもらった。これら3つの評価情報を主成分分析し、その第一主成分得点を技術者レベルの生産性の自己評価指標とした。その各国の平均値を比較した結果が図1である。ここでは、職務内容をより同質化するために、ソフトウェア技術者が専ら扱うソフトウェアのタイプにより、ERP技術者と組込みソフトウェア技術者に分けて比較した。

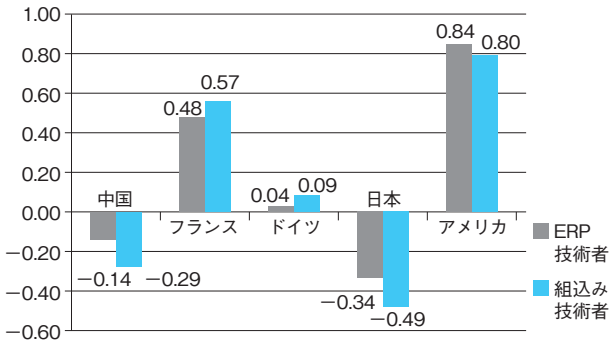


図1　ソフトウェア技術者の生産性の自己評価比較（2016年）

この指標でも日本の水準は、ERP、組込み技術者共に、5カ国中最も低い水準である。日本のソフトウェア技術者は、自分たちの生産性を高く評価しておらず、ミクロな技術者の視点からの評価でも、日本のソフトウェア生産性は比較対象4カ国と較べ、最も低い水準である。

2.2　ソフトウェア技術者の能力自己評価指標での比較

上記の生産性の自己評価には、様々な要因が影響を与えていると思われる。そこで、それらの要因の影響を除去した指標比較を行うために、次にソフトウェア技術者の能力に着目して同様な国際比較を行うことにする。生産性の自己評価は、各人に任された仕事の内容に大きく依存する。例えば、高い能力を持ちながらも、その能力を十分に発揮できない仕事を担当した場合、生産性の自己評価は低くなる。しかし、技術者の能力そのものの評価であれば、能力と職務のマッチングの程度に影響を受けない安定的な指標となる。その意味では、能力評価指標での生産性比較は、生産性のポテンシャル比較とも言える。質問票の中では、様々な能力項目について自己評価を求めた。その自己評価データに対し主成分分析を行うことで、3つの主成分を抽出した。各主成分を構成する質問項目を検討したところ、3主成分は、専門職力、組織人力、及びマネジメント力と定義できた。これら3能力の5カ国の数値を比較すると、結果は図2のようになる。ここでも日本の低さは一目瞭然である。

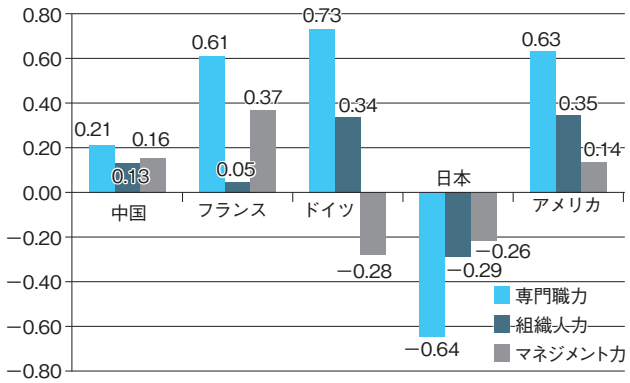


図2　3能力の5カ国比較（2016年）

これら3つの能力指標のうち、専門分野の知識や論理的思考力を表す専門職力とコミュニケーション力、責任感や向上心などを含む組織人力では、日本の水準が5カ国中で最も低い。また、リーダーシップや管理能力を表すマネジメント力においても、日本のソフトウェア技術者の水準は、最も低い中国とほぼ同水準の低さである。以上からこれらの指標でも、日本のソフトウェア技術者の潜在的な生産性は、ほかの4カ国と比べ、どの国よりも低い水準であることが確認できた。

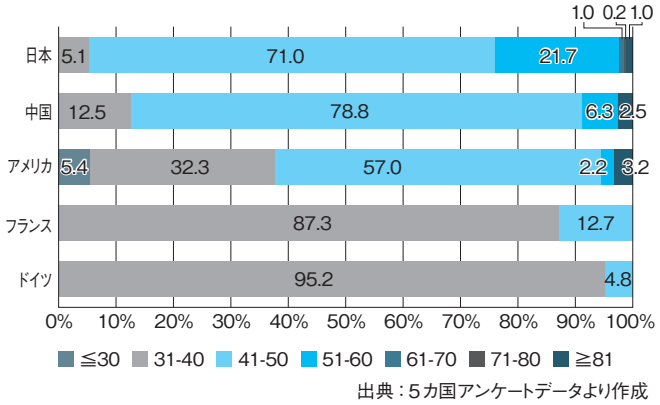
以上を総括すると、マクロ、ミクロどちらの視点から評価しても、また日本のソフトウェア技術者の生産性で見た国際比較においてでも、日本の水準は、主要国と比べ大きく見劣りすると結論できる。

3　日本のソフトウェア技術者の現状（2）：日本の労働条件に国際競争力はあるのか?

前節の結論である日本のソフトウェア産業の生産性の低さは、そもそもどこから来ているのだろうか。本節と次節では、ソフトウェアの制作が資本設備を用いず労働集約的に行われることから、ソフトウェア技術者の労働条件や働き方、及び彼らの仕事と職場のあり様に、この低い生産性の原因を探ってみよう。

3.1　労働時間で見た日本のソフトウェア技術者の労働条件

日本のソフトウェア技術者の労働時間は、国際的に見て長いのだろうか？2人の技術者が同じ仕事を行った場合、より生産性の高い技術者のほうが短い時間で作業を終えることができる。つまり生産性と労働時間は、ある条件の下では逆相関の関係となる。以下に示す図は、通常週における総労働時間の各国分布を、組込みソフトウェア技術者ついて比較したものである。同様の図をERP技術者についても作成したが、組込みソフトウェア技術者の場合と極めて類似性が高く、ここには掲載しない。



出典：5カ国アンケートデータより作成

図3　組込みソフトウェア技術者の週労働時間

各国のソフトウェア技術者がどのように働いているか、その特徴がこの図から読み取れる。この図では、左から順に、労働時間が短い者の割合が示されている。日本の法定週労働時間が40時間であることから、図では灰色部分より右側が、週労働時

間が40時間を超える残業労働を行っている者の割合となる。日本は、5カ国中残業を行っている者の割合が最も高く約95%である。また、週残業時間が10時間を超える長時間残業者の割合も、日本が最も高く約24%である。他方、フランス、ドイツではそのような長時間残業を行うソフトウェア技術者は皆無である。以上から、日本のソフトウェア技術者は、国際的に見て極めて長時間の労働を行っていることが確認できた。

3.2 2つの給与指標での比較

次に給与水準を見てみよう。給与の国際比較には注意が必要だ。各国では異なる通貨で給与が支払われており、その通貨価値の共通化が必要となる。その場合、標準的な手法は為替レートを用いた共通化である。もう一つは物価水準の差異を考慮した購買力平価を用いる共通化である。このどちらを用いるのが適切か判断が必要となる。第二の注意点は、通常給与額は労働量(労働時間)に大きな影響を受ける。既に見たように、各国でソフトウェア技術者の労働時間には大きな差異が存在する。その差異の影響を調整するためには、単位時間当たりの給与で見なければならない。そこで以下では、単純に為替レートを用い統一通貨で表した年収額比較結果と、物価の差異と労働時間の差異を考慮した購買力平価で評価した、時間当たり給与での比較結果の両者を示す。

為替レートを用いた年収比較

図4Aに各国のERP及び組込みソフトウェア技術者を含むソフトウェア技術者全体に関する年齢グループ別平均年収のプロフィールを示した。この図を見ると、日本のソフトウェア技術者の年収は、どの年齢グループについても5カ国中3位と中位に位置することが分かる。また、年齢グループが上がるにつれて日本の位置は、4位のフランスの平均値からのかい離が増大し、2位に位置するドイツの平均値に近づいていく。その限りで、日本のソフトウェア技術者の平均年収は、年齢と共に世界の中での相対的な位置は向上し、50歳グループでは、5カ国中で2番目に位置するドイツと同水準となることが分かる。

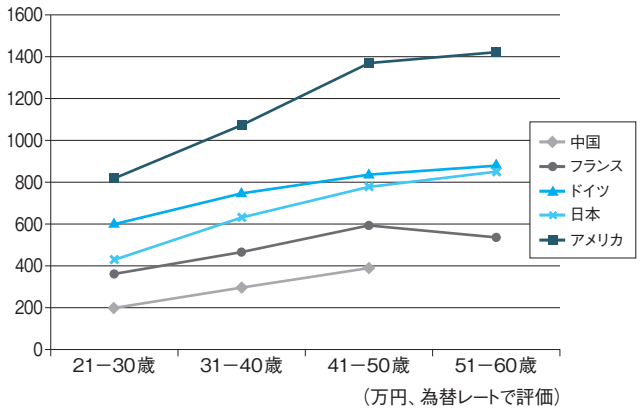


図4A　年収の年齢プロフィール：ソフトウェア技術者全体

では、各国の物価の差異と労働時間の差異を調整すると日本の相対位置は変化するのだろうか。

購買力平価を用いた時給比較

図4Bがその比較結果である。観察単位を時間当たり収入とし、物価の5カ国間の差異を調整すると、日本の時給水準の相対位置は低く、フランスの水準を下回り、5カ国中4番目となる。そして、最も低い中国の時給水準との差が大きく縮小すると共に、最上位のドイツとの格差が広がり、20歳代の若年技術者では、ドイツの時給水準の50%にも届かない低さとなる。

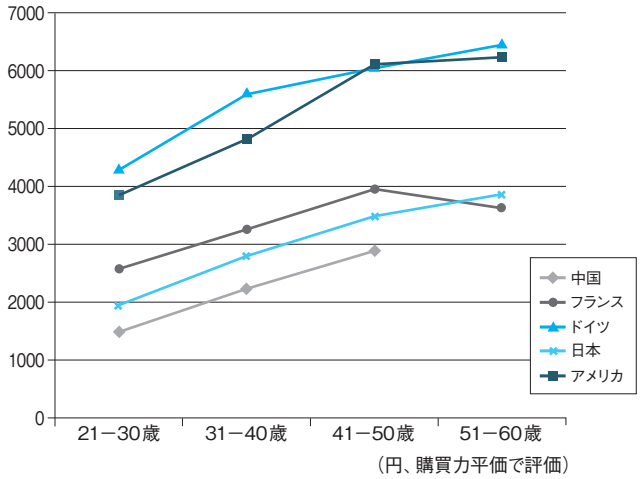


図4B　時給の年齢プロフィール：ソフトウェア技術者全体

以上、労働時間と給与の2つの労働条件の現状を主要5カ国の中で評価したが、結果を総括すると、日本のソフトウェア技術者の労働条件は、欧米3カ国から大きく劣り、国際的にはほぼ中国の労働条件と同水準の低さであると評価できる。

4 日本のソフトウェア技術者の現状(3):彼らの職場はどうなっているのか?

前節では、日本のソフトウェア技術者の労働条件が低位にあることを確認したが、この結果は、彼らの生産性についての低い評価結果とも整合的である。なぜなら、労働条件と生産性は、相互に規定し合う関係にあるからである。良い労働条件は、生産性の高い技術者を集めるための誘因であると共に、高い生産性は、より良い労働条件を実現するのに必要なより多くの原資を提供するからである。では次に、労働条件とは独立に生産性を規定すると思われるソフトウェア制作のマネジメントについて見てみよう。

4.1 人的資源マネジメントの比較

ここで注目するマネジメントは、ソフトウェア技術者の人的資源マネジメントである。具体的には、ソフトウェア技術者の能力や仕事を適切に評価し、またその評価に納得できない技術者の苦情へは的確に対応しているか。また、そのような評価と対応の結果、技術者たちは、能力、業績評価に納得しているのか。これら人的資源管理に対する技術者による評価を主成分分析し、そこから得た第一主成分得点を「良好な評価処遇制度の運用」指標として示したのが図5である。

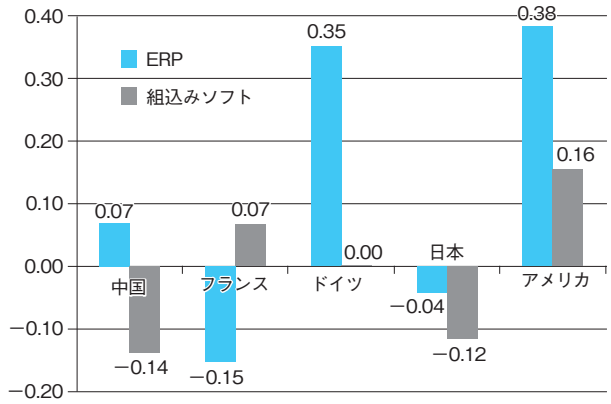


図5　評価処遇制度の運用の良好度比較

日本企業の評価処遇制度運用の状況は、ERP、組込みソフトウェア両者において決して良いとは言えないことが分かる。ERP、組込みソフトウェア技術者共に、日本の数値は、5カ国平均値を下回り、マイナス値となっている。ERP、組込み両者の評価が5カ国平均を下回るのは、5カ国の中では日本のみである。ソフトウェア技術者たちの能力や貢献が適切に評価されないなら、技術者たちは仕事に対するモチベーションを維持することが難しくなる。そのようなモチベーションの低下は、彼らの働きぶりの変化を経由して、結果的に生産性に対してマイナスの効果を持つだろう。生産性の向上には、日本のソフトウェア企業において人的資源マネジメントを改善する余地があることを示唆する結果である。

4.2　職場マネジメントの比較

次に職場のマネジメントを見てみよう。ソフトウェアの制作やそれらを経済・経営活動に効果的に活用する作業は、高度な専門知識と高いレベルの抽象的思考力を必要とし、技術者の創造的なマインドが最大限に発揮されなければならない。そこで注目したのは、職場がそのような創造的な活動ができる環境であるかどうかである。そこで「新しいことに挑戦することが歓迎され」「仕事や研究に関して議論」でき、「仕事で困ったときなど、気軽に相談できる」職場かどうかを見てみた。これらの創造的職場を構成する要因に関する技術者評価を主成分分析することで、「オープンで進取の職場」指標を作成した。それを図示したのが下記の図6である。

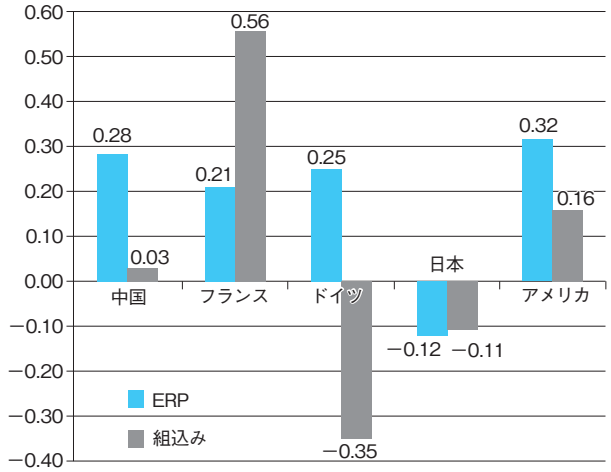


図6　「オープンで進取の職場」マネジメント

この指標でも、日本のソフトウェア職場は世界の中では低位であることが分かる。5カ国中、日本のみが、ERP、組込みソフトウェア両者で、「オープンで進取の職場」指標が、5カ国平均を下回っている。扱う製品がERPか組込みソフトウェアかにかかわらず、日本企業のソフトウェア職場には、自由で闊達な創造的空気が不足しているようだ。

5　将来に向けてのまとめ

以上、本稿では、日本のソフトウェア産業とそこで働くソフトウェア技術者の現状を、2016年にIPA/SECの委託調査で収集した、日米中独仏5カ国のソフトウェア技術者データに基づき分析を行った。具体的には、ソフトウェア技術者の生産性、労働条件、そして人と職場のマネジメントの現状を5カ国比較の形で評価した。その結果、日本のソフトウェア産業の国際競争力、そしてそこで働くソフトウェア技術者の生産性は米、中、独、仏4カ国と比べ低位にあることが確認できた。また、彼らの労働条件を構成する給与と労働時間では、日本は5カ国中、中国と共に最も低い水準であることも確認された。更に、これらの低い評価の背景には、多くのソフトウェア企業において、人と職場のマネジメントに問題を持っていることも明らかになった。

日本のソフトウェア産業は、国際競争力を高め、その競争力に見合った労働条件を技術者に提供できなければ有能な若者の参入は期待できない。日本のソフトウェア職場が、オープンで進取の空気が満ちた創造的職場となり、技術者の能力と成果が適切に評価され、処遇されれば、より創造的に働くインセンティブが彼らの中で高まるだろう。その創造的意欲を職場の成果に効率良く結び付ける職場マネジメントを行えば、閉塞的な現状は、おのずと改善されよう。

【参考文献】
[1]　刀川眞：我が国の社会的特性に着目した組込みシステム開発の方向性, 科学技術動向, 2011年9・10月号, pp12-22.
[2]　同志社大学：「日本のソフトウェア技術者の生産性及び処遇の向上効果研究：アジア, 欧米諸国との国際比較分析のフレームワークを用いて」に関する成果報告書, 2016.

システムズエンジニアリングを活用したITSのセキュリティ機能設計の取り組み

三菱重工業株式会社 ICTソリューション本部 制御技術部 ソフトウェア設計課 原 健太

近年の技術進歩に伴い増加するセキュリティリスクに対応するため、三菱重工機械システム株式会社ではシステムズエンジニアリングの考え方を活用し、システム開発のプロセス面からセキュリティ機能向上とその設計根拠の明確化に取り組んでいる。その一部を紹介し、おのおのの取り組み効果について述べる。

1 概要

MHI-MS（三菱重工機械システム株式会社）は、国内外の高速道路などの有料道路料金収受システム及びITS（高度道路情報システム）において、その黎明期から携わる長い歴史を持つ。この事業は、道路利用者から徴収する料金や、各種センサから収集する位置情報や画像を含んだ個人情報などを扱うといった性質上、セキュリティ機能に対する要求が非常に高い。

従来、料金収受システムは、クローズド環境というセキュリティ性の高いシステムであった。その一方で、近年は様々な業界で技術のオープン化・汎用製品の利用・クラウドサービスの利用などが進んでいる。これらは高度な技術を安価かつ容易に導入できるという利点があるが、使い方によってはセキュリティ性が低下する可能性がある。このような動向を踏まえて当社のITS事業においても、これまで以上にセキュリティ対策に注力した取り組みが必要となる。

ただし、一口にセキュリティと言っても「どの程度まで対策すれば十分なのか」を明らかにすることは難しい。あらゆるリスクを考慮すると必要なセキュリティ対策には際限がないが、すべてを実施することは不可能である。将来的に変わる可能性のある要素を考慮しながら、予算や期間を含めた限りあるリソースの範囲内で実現可能であり、ビジネスにとって適切な対策範囲を明確にしてステークホルダへと提示・実現することが求められる。

我々は、システムの特性に合わせた、かつ妥当性の高いセキュリティ対策の実現に向けた取り組みの一つとして、システムズエンジニアリングの考え方を取り入れた検討活動を実施している。前述した近年の技術動向を踏まえ、あらためてITSのセキュリティ対策を見直すにあたり、システムの立ち上げに至る企画段階からライフサイクル全体を通した観点で検討・開発を進めるシステムズエンジニアリングの考え方が有用であると考えたためである。

本記事では、プロトタイプシステムの開発を通してビジネスレベルの目的展開やプロジェクト分析といった超上流工程での活動、及びその結果を基に実施したセキュリティ要件定義と、開発の各工程で実施する妥当性確認と課題管理といった活動を実施した例を紹介し、各取り組みの目的や効果について解説する。

2 取り組み対象・項目

2.1 取り組み対象

ITSのモデルの一つとして、主に野外などに設置した各種センサ・機器と車両が無線通信するなどして課金や違反取り締まりを行う「路側システム」と、その路側システムや様々な機器とネットワーク経由で通信して情報を収集し、総合的な管理を行う「上位系システム」の構成がある。

本記事では、路側システムからネットワーク経由で情報を収集・管理してサービスを提供する上位系システムのプロトタイプ開発とセキュリティ機能検討を対象として紹介する。

2.2 取り組み項目

本記事では、超上流工程における検討項目として実施した「プロジェクト要素分析」、セキュリティ機能の検討結果を記録した「セキュリティ要件定義」、各開発工程で実施した「妥当性確認」と「課題管理」について紹介する。

プロジェクト要素分析は、設計プロセスと言うよりは経営としての企画・営業活動の段階という印象が強いが、設計プロセスの最上流工程であるということを意識して実施することによる効果について述べる。

セキュリティ要件定義は、システムのセキュリティ方針を定義する重要な工程である。プロジェクト要素分析の結果を加味し、ステークホルダにとってリーズナブルな提案を実現するための検討手法について述べる。

妥当性確認と課題管理は、全工程を通して実施する活動である。定義した開発範囲と目的に基づいた開発活動推進の監視や、次回以降の開発サイクルに向けた申し送り事項の管理について述べる。

なお、従来の開発においてもこれらと同等の作業は実施していたが、企画段階での検討結果を設計情報として捉え、かつシステム開発プロセス全体で参照し設計内容に反映する取り組みは、経験あるエンジニアの知見に依存した作業になっていることが多かっ

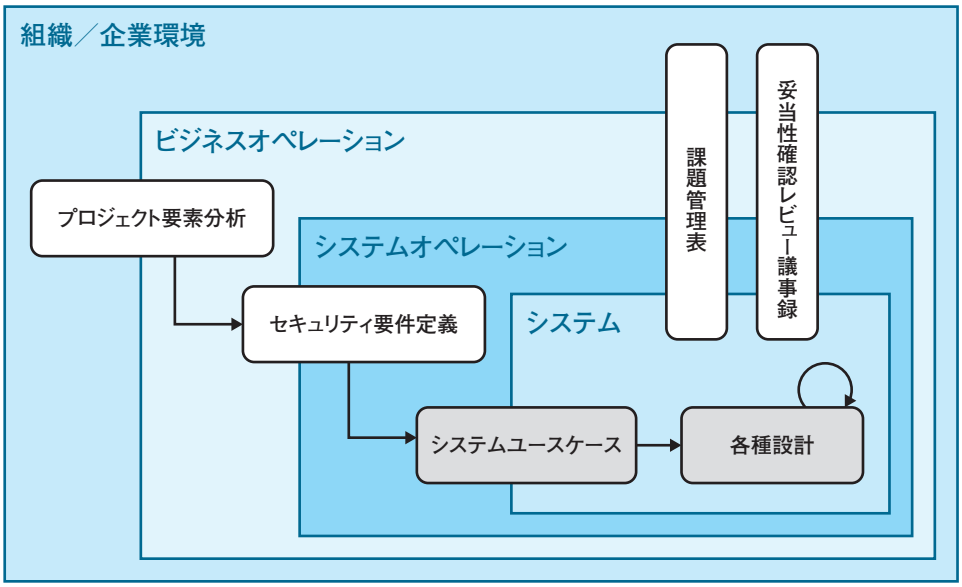


図1　プロトタイプシステム開発における各種活動

た。システムズエンジニアリングの考え方を取り入れることで、これらを誰もが設計プロセスとして明確に意識してシステム開発に取り組むことができる。

3 取り組み内容

3.1 プロジェクト要素分析

本取り組みの目的は、具体的なシステム要件検討を開始する前に、そのシステムによって解決するビジネスの課題を明らかにすることである。

前述の通り、企画・営業活動としてこれまで実施している作業であるが、設計開発部門がその内容を設計プロセスの一部として捉え、意識することで、本来の要求に即した設計を進めていく上での指針となる。

プロトタイプシステム開発に先立ち、ビジネス上の課題や、それに紐付くシステム開発の上位要求を分析し、プロジェクトそのものに求められる要件を整理した上で、プロトタイプシステム開発方針や範囲の整理をするために要素分析シートを作成した。表1に、分析内容の抜粋を示す。

導入の効果としては、開発プロジェクトに求められる要件の深掘りを実施でき、かつ設計活動の一環として取り組むことにより、その内容を設計メンバが十分に理解できるところにある。なお、本プロトタイプシステム開発では、この作業を設計メンバのみで行ったが、企画・営業・設計などを含む複数部門で実施することで、より適切な分析結果を得ることができると考える。

また、幾つかのステップを踏んで最終的に到達したいビジネスレベルの目標と、そのための最初のステップとしてまず取り組む範囲や、次回以降のステップで取り組む範囲などを明らかにすることもできる。セキュリティ機能を検討する場合は、最終的に目指すセキュリティレベルや、そこに到達するまでの段階的な開発活動と個々の達成目標などを明確にしておけば、システムが提供するセキュリティ機能の範囲に関する妥当性を主張する根拠にもなる。

3.2 セキュリティ要件定義

このプロセスではセキュリティ要件定義書を作成した。これは、開発活動の対象となるシステムに対するリスク抽出・分析・対策それぞれの検討方針、及びNIST SP800-53／82やIEC62443-3-3といった制御システムセキュリティに関する国際規格の分析内容を踏まえたリスク対策検討結果について記載したものである。

この文書では、プロジェクト要素分析で明確化したビジネスレベルの目的と、本プロトタイピングの目的を考慮して具体的な要求事項を抽出し、ニーズとの関連性が分かるようまとめている。

セキュリティ要件定義書の作成は、開発活動に対するビジネスレベルの要求の具体化につながり、結果として以降の設計作業の指針とすることができる。

図2に検討の手順を示す。リスク抽出として、システムの情報資産及びリスク発生ポイントを特定し、その組み合わせから考えられるリスクの洗い出しを行った。次に、リスク分析として、抽出したリスクの被害規模と攻撃容易性の数値化に際し、前述の「要素分析シート」で検討した内容に従ってSQuaRE品質モデルで定義される各品質特性との関連についても評価した。

上記の考え方に基づいてセキュリティ要件定義書を作成することの効果は以下の通りである。

- 上位概念であるビジネスレベルの目的を確実に反映して、かつ要素分析時に定義した範囲のセキュリティ機能に関する要件定

表1　要件分析シート

ビジネスレベルの課題	具体的な機能と比べて設計根拠が希薄であったセキュリティについて、ITSとして必要な機能を向上させたシステム構築を進めたい。
原因	• 従来は「固定された」「閉じた」環境の中にシステムを構築するという前提があり、オープンな情報システムと比較してセキュリティリスクが低かった。ただし、近年急速に進む技術のオープン化やクラウド利用に伴い、ITS事業もこれまで以上に高いセキュリティレベルが求められるようになる。 • セキュリティリスクは技術の発展と共に常に形を変え、それに合わせてセキュリティ対策も常に変化し続けることが必要な状況の中で、システムにとって必要なセキュリティ機能の根拠を明確にしてステークホルダへ提供することが難しい。
対策	• これまでは経験を積んだ技術者が独自に実施していたセキュリティ機能に対する設計範囲・根拠の明確化プロセスを設計活動として明示的に取り入れる。 • 既存の設計プロセスを定めたセキュリティ標準や設計標準と照らし合わせ、今後ITSとして必要になるセキュリティ機能を満たすシステム設計／開発手法の確立を目指す。 • 同様の手法を今後のITS開発に展開し、課題解決につなげる。
対策を決定・実施するための手段	• ITSのプロトタイプ開発を通し、検討範囲を限定して実施する。 • セキュリティ機能については、SQuaRE品質（＊1）のうち検討対象範囲に影響すると考えられる「機密性」「完全性」「真正性」「責任追跡性」「否認防止性」の観点から分析し、必要となる機能検討を進める。

＊1：Systems and software Quality Requirements and Evaluation；システム及びソフトウェア製品の品質要求及び評価に関する国際規格ISO/IEC 25000シリーズ、国内規格JIS X 25000シリーズの総称。

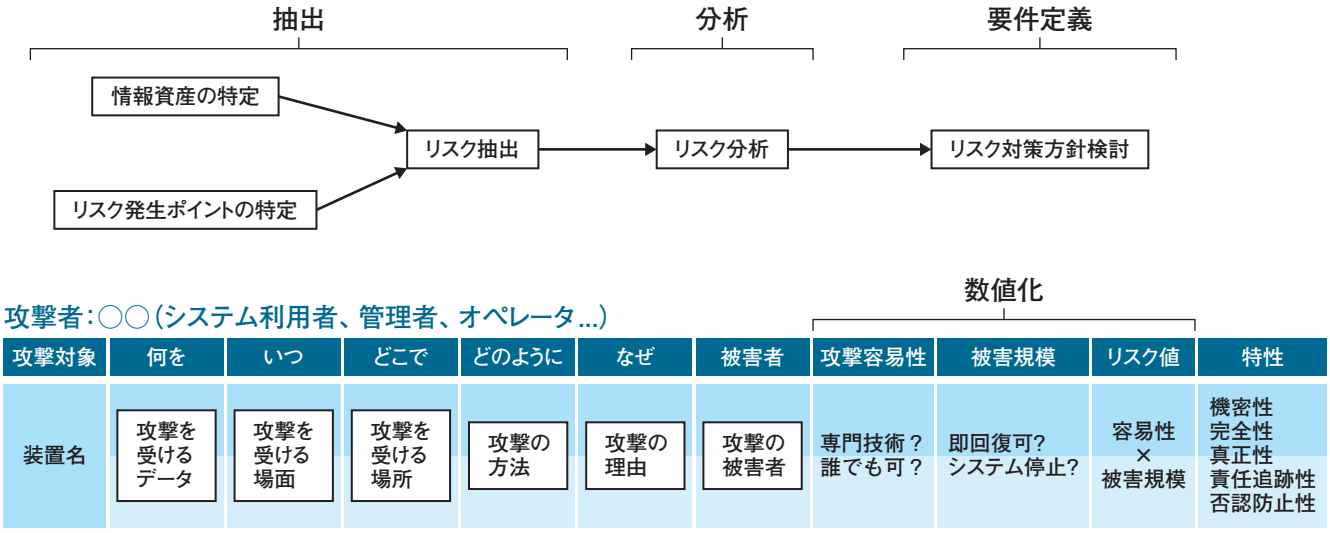


図2　セキュリティ要件定義書作成手順

義を行うことで、開発活動で必要とするセキュリティ機能の要件を漏れなく検討することができる。

● 検討結果は設計の妥当性を示すエビデンスとして利用することができる。

3.3　妥当性確認と課題管理

このプロセスは、開発活動全体を通して実施する取り組みである。超上流工程において、ビジネスレベルの背景とで達成すべき「セキュリティ機能の向上」と今回の開発活動で実現すべき範囲を明確化し、その内容について具体的なシステムとして定義したセキュリティ要件の設計・実装・テストの各設計プロセスで実施するレビューにおいて、「本来の目的に対する達成度」「次回以降の開発まで保留すること」を確実に管理するための取り組みを実施した。

① 妥当性確認

設計プロセスで実施するレビューにおいて、上位図書の内容が確実に反映されていることを確認する（検証）だけでなく、超上流

工程で検討した今回の開発活動で達成すべき範囲とその内容、セキュリティ要件定義書で明確化した具体的なシステム要件からの乖離がないかを確認する（妥当性確認）という観点を強く意識した。

長いライフサイクルを持つシステムの場合、システムの複雑化や、機能追加・改善により開発プロセスが何度も繰り返されるが、軽微な機能追加や修正が前工程との間では整合性が取れていたとしても、本来そのシステムを導入することでステークホルダが実現したい目的の達成や、そのシステムで提供すべき機能の範囲から乖離してしまうことがあり得る。開発プロセスの最後に実施するシステムテストや、客先への導入後にその乖離による問題が発生する前に、各設計プロセスで常に上流で定義した要件に対して確実に妥当性確認を実施することで、本来の目的に即した開発活動を進めることができ、その証拠とすることもできる。

プロトタイプ開発においては、本来の目的と乖離していないかどうかをチェックする妥当性確認の観点からレビューを実施する意識付けのため、上位図書の内容を漏れなく反映しているかをチェックする検証とレビューシートを分けて妥当性確認を行った。

これにより、例えば「セキュリティ監査目的として、システムオペレータの操作履歴を取得する」というセキュリティ要件に紐付く機能の設計書に対して妥当性確認を行う場合、記録する内容は十分か、記録しない場合はどのような理由があるのか、記録した結果をどのように保存してどう活用するのか、といった内容を関係者間で調整し、考えを共有することができた。

抽象的な要件に対する機能は人によって考えに違いが生まれるため、都度本来の要件や目的と照らし合わせて妥当性確認を行うことは、セキュリティ検討の面から見ても設計根拠やシステムに適切なセキュリティ機能の提供に有力な手段であると考えている。

② 課題管理表

妥当性確認により見つかった本来目的との乖離が課題として残った場合や、プロトタイプシステムの開発時は実現する機能を制限し、次回以降の開発で追加する機能など、開発活動全体としての課題を整理するための課題管理表を運用した。

課題管理表には、課題の内容と原因、対策方針、暫定対策の実施内容、次回以降の開発に向けた申し送り事項などを記載する。プロジェクト全体を通しての課題を一覧として管理し、引き継いでいくことで、現時点の暫定対策を次回の開発で修正し、機能を追加することでプロジェクトとしての本来目的を達成することができる。

本プロトタイプシステムの開発においては、製品化時点では必要だが、プロトタイプ時点では運用でカバーできるような機能など、セキュリティ要件定義書の内容から一部簡略化して設計・実装したセキュリティ関連機能の内容（具体的機能、簡略化方針、その妥当性など）と、次回以降の開発に向けた申し送り事項などを記録した。これにより、プロトタイプシステムが提供する機能の妥当性を示す資料になると共に、次回の改修で追加設計や実装が必要な範囲を漏れなく引き継ぐことができた。

4 取り組みの成果

今回は、プロトタイプシステム構築を通したITSに必要なセキュリティ機能検討の活動例として、超上流工程でのプロジェクト分析・その結果を踏まえたセキュリティ要件定義・全工程で継続的に実施した妥当性確認と課題管理について紹介した。個々の取り組みに対する導入効果は３節で述べた通りだが、開発活動全体を通して実現できることは、下記の内容を全て明文化することでシステムの目的を意識しつつ、関係者間で考えを共有しながら開発を進められるところにあると考える。

- ビジネスレベルで最終的に目指す目的
- そのために現段階で必要なシステムとその機能範囲
- 各工程のアウトプットと現段階で必要な機能との対応
- 最終的に目指す目的に対する残課題と対策方針

セキュリティ機能の検討にシステムズエンジニアリングを活用したプロセスを適用することで、冒頭で述べた「“セキュリティ”という将来的に変わる可能性のある要素を考慮しながら、予算や期間を含めた限りあるリソースの範囲内で実現可能であり、ビジネス・システムにとって必要な対策範囲とその設計根拠を明確にしてステークホルダへと提示・実現する」という課題の改善につながっている。

今後は、今回のプロトタイプ開発で抽出した課題の解決を製品化時に実施すると共に、更に幅広い案件に対してセキュリティ機能設計を含む各種開発に、システムズエンジニアリングの考え方を導入した仕組みを検討していく。

No.	状況	課題記述文書	課題	原因	対策案	現時点の方針	対応済み文書	申し送り事項
	クローズ 対策検討中 文書反映中 …	課題提起した文書	課題内容	課題の根本的な原因	課題に対する最終的な対策	対策案に従い現時点の開発で実施する内容	現時点の対応内容を明記した設計書	次回以降の開発に向けた申し送り事項

図３　課題管理表の作成

情報システムの事故データ

情報システムの障害状況 2017年後半データ

IPA顧問 松田 晃一
SEC研究員 目黒 達生

2017年7月から12月の間に情報システムの障害は25件報道されており、相変わらず障害の発生は高い水準にある。特徴的な事故としてはJアラートに関するものが3件、システム障害が原因で個人情報が漏えいするセキュリティ問題を引き起こした事例が2件発生している。また、システムの更改のために事前に実施したテストが本番環境に悪影響を与えた結果、通常運用に入って障害となった事例も1件報告されている。

1. 2017年後半の概況

2017年の後半に報道された障害25件は表1に示す通りである。今期の発生件数は月平均4.2件と、かなり高い水準である（図1）。2017年前半の件数も加え年間を通して見ると、月平均4.0件となる。

本誌50号でも取り上げたが、システム障害が原因で個人情報が流出した可能性のある事例が今期2件報道されている（事例1725、別表 事例15）。更に、国民の安全確保にとって重要な役割を果たすべきJアラートに、相変わらず障害が複数報告されている（事例1731、1734、1735）。また、ソフトウェア保守の一環として実施した作業が本番環境に影響を与えたために障害が発生した事例（事例1739）が報告されている。これらについてはそれぞれ節をあらためて取り上げる。

2017年8月には、インターネットに大規模な障害が発生し、大きく報道された（事例1733）。この通信障害によって、ネットサービスを提供する多数のWebサイトにおいて、サービスが利用できない、あるいは利用し難いといった影響が発生した。この事故の原因は、米国グーグルが大量の誤ったネットの経路情報を誤配送したため、それを受けたOCNなどの大手サービスプロバイダの通信機器が過負荷に陥り通信ができなくなったため、と発表されている。米国グーグルで発生したわずか8分間の誤設定の影響が、日本に伝搬して大きく拡大し、多数のWebサイトに長時間の影響を及ぼす事故となった。グローバルにつながっているネットワークの信頼性、安定性の確保にしっかりした対策が望まれる。

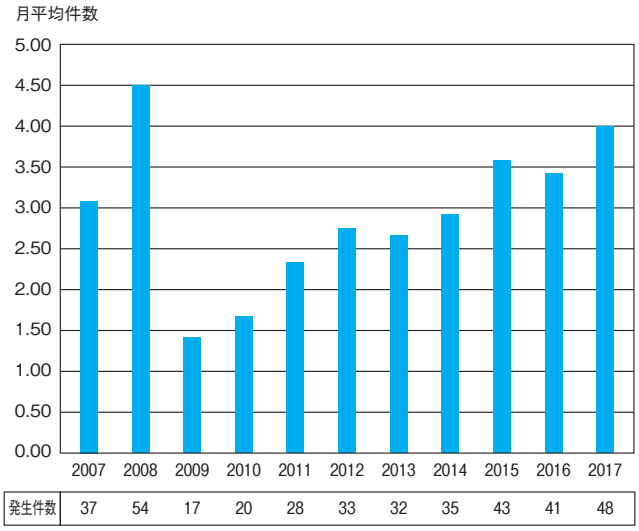


図1　報道された情報システムの障害件数の推移

また、プログラムの不具合や設定誤りによって、請求金額の計算誤りが発生した事例が7件報道されている（事例1727、1729、1748、別表の事例2, 4, 7, 14）。50号でも取り上げたが、システムは正常に動作するが処理に誤りがあるこの種の障害は、誤りの検知が遅れるため影響は大きい

表1　2017年後半の情報システム障害データ（報道に基づきSECが整理）

No.	システム名	発生日時（上段） 回復日時（下段）				影響	現象と原因	直接原因	情報源
		年	月	日	時				
1724	ジュピターテレコム インターネット 接続サービス	2017	7	3	11時50分	7月3日11時50分から4日11時20分にかけ、関西エリア、熊本エリア、東日本エリアにおいて、インターネットにつながりにくくなった。	3つのエリアを受け持つ中継通信機器で不具合が発生。DNSサーバーが高負荷となり、インターネット接続サービスで不具合が生じた。	ハードウェア障害	• 日経コンピュータ（2017.7.4） • ジュピターテレコムニュースリリース（2017.7.4）
		2017	7	4	11時20分				
1725	メルカリ 個人間取引 アプリ	2017	7	6		個人情報の流出が発覚。対象となった顧客54,180人は、名前や住所、メールアドレスのほかに、銀行口座やクレジットカード番号下4桁と有効期限、購入・出品履歴、売上金、ポイントなどが第三者に閲覧される状態になっていた可能性があった。住所や氏名、メールアドレスなど個人を特定できる情報が閲覧された可能性があるのは29,396人。	6月22日、サーバー切替えで、その時間帯にアクセスした人の個人情報が誤って表示された。1時間以内に同じURLへのアクセスがあり、偶然同じサーバーに接続された場合に誤表示が起きた。対策として①外部からの定期的アクセス②アクセスログをリアルタイムに監視③意図しないキャッシュの早期発見、通知するツール導入④キャッシュの設定を自動検証する機能の構築、とした。	設定ミス	• 通販新聞（2017.7.6） ※障害発生は、報道された日とした。
1726	JAL 国際線システム	2017	7	9	18時30分	国際線航空券（特典航空券含む）において空席照会、予約、購入、各種手続きができない。	9日、18時30分頃より、国際線システムが利用できない状態が発生。10日1時30分頃復旧した。	不明	• 日本航空お知らせ（2017.7.9）（2017.7.10）
		2017	7	10	1時30分				
1727	西日本高速道路 自動料金収受 システム（ETC）	2017	7	29	23時56分	大阪府八尾市の近畿自動車道八尾インターチェンジ（IC）の自動料金収受システム（ETC）で、7月29日～8月1日の利用者の一部に料金を多く請求。対象は約11,000件。過請求額は計200万～300万円となる見込み。普通車で20円から480円多く請求。	近畿道は6月3日の料金改定で、定額料金から走行距離に応じた変動料金に変更した。その際ETCのプログラムミスにより7月29日23時55分以降八尾ICを利用した時刻が実際より1時間早く記録される障害が発生。このためICの出入りの順番に矛盾が生じ、料金が誤請求された。1日1時30分以降は正しい料金に修正して請求し直した。	プログラム不具合	• デジタル毎日新聞（2017.8.9） • NHKニュース（2017.8.9） • 朝日新聞（2017.8.11）
		2017	8	1	1時30分				
1728	原子力発電所 放射能測定 プログラム	2017	8	7		中国電力（株）島根原子力発電所、四国電力（株）伊方発電所、北陸電力（株）志賀原子力発電所、日本原子力発電（株）敦賀発電所では、放射性廃棄物の最大放射能濃度の再計算をする必要が発生。	電子力発電所放射性廃棄物放射能濃度計算プログラムに不具合があり、一部の廃棄体の放射能量が少なめに評価された。各電力会社は、保有するデータを調査し、問題なく適切に放射能量が測定されていることを確認した。	プログラム不具合	• 原子力規制委員会（2017.8.7） • 中国電力お知らせ（2017.8.7） • LogisticsToday（2017.8.7） • 日本原燃HP（2017.8.7） ※障害発生は、報道された日とした。
1729	中部電力 送配電料金 システム	2017	8	8		新電力の小売り電気事業者に対する送配電料金で約1億9,000万円の誤請求があった。延べ18社に影響した。	新電力の事業者へ送配電設備の使用料金を請求するとき、割引制度のプログラムミスで、2016年4月から2017年5月まで料金を過大割り引きした。事業者の問い合わせで今年7月に発覚。	プログラム不具合	• 毎日新聞（2017.8.9） • 産経新聞（2017.8.9） • 中部経済新聞（2017.8.9） ※障害発生は、報道された日とした。
1730	愛媛県交通管 制システム	2017	8	13	6時30分	通行止めや渋滞情報などを示す国道の交通情報板（全25カ所）の表示と、カーナビへの県内の一般道の渋滞情報の提供を停止。	13日6時30分頃、2カ所の情報板に誤った通行止め情報が表示されたため、12時18分からカーナビへの情報提供を停止し、同45分に情報板の消灯を各署に指示した。17時10分頃に復旧した。	不明	• 愛媛新聞（2017.8.14）
		2017	8	13	17時10分				
1731	総務省消防庁 Jアラート	2017	8	18		中四国の9県で18日、全国瞬時警報システム（Jアラート）の送受信訓練があったが、4県3市町で警報が発信されなかった。	18日、訓練を実施。プログラムの設定ミスにより防災メールが文字化けで読めず、防災行政無線の自動起動装置のソフトウェア不具合から音声が流れず、整備した端末が作動せず、など、各地で不具合が発生。各自治体は不具合の機器の再テストを行い、復旧を確認した。	プログラム不具合	• NHK NEWSWEB（2017.8.18） • デジタル毎日新聞（2017.8.19） • 時事ドットコム（2017.8.23） • 総務省報道資料（2017.8.23） • 産経ニュース（2017.8.31）

No.	システム名	発生日時（上段） 回復日時（下段）				影響	現象と原因	直接原因	情報源
		年	月	日	時				
1732	東京商品取引所（東商取）売買システム「J-GATE」	2017	8	22	17時40分	原油や金などの貴金属の取引ができなくなった。17時40分からすべての商品について取引を停止する措置を取った。	22日17時頃に売買システムで障害発生。17時40分には全ユーザの接続を切断し、全商品の取引が停止。19時に注文受付を開始し、20時頃に取引を再開した。原因は、ハードウェア障害。また、別システムへの自動切替えができなかった。	ハードウェア障害	• 日経コンピュータ（2017.8.22） • 日本経済新聞電子版（2017.8.22）（2017.8.25） • 毎日新聞（2017.8.22）（2017.9.15） • J-CAST会社ウオッチ（2017.8.23）
		2017	8	22	20時00分				
1733	NTTコミュニケーション、KDDI、米グーグル	2017	8	25	12時22分	多くの企業で8月25日12時24分から17時頃まで、一部のネットサービスが利用しにくくなるシステム障害が発生。スマートフォン向け電子サービスや電子マネーのチャージなど多くの機能が不安定になった。そのほか、金融機関、証券会社などのネットサービスなども一時使えなかった模様。	米グーグルから大量の経路変動があり、NTTコムを介してインターネットに接続していた企業のルータが大量の経路情報を受け取り高負荷となり、通信障害につながった。原因は、米グーグルが8月25日12時22分頃に、OCNとピアリング（対等な関係でネットワークの経路情報をやり取りすること）していたとき、誤った経路情報を大量に送ったことによる。	設定ミス	• 日経コンピュータ（2017.8.25） • ITpro（2017.8.25）（2017.8.26）（2017.8.28） • 総務省電気通信事故検証会議（2017.12）
		2017	8	25					
1734	総務省消防庁Jアラート	2017	8	29		16の市町村で住民に伝える防災行政無線の放送が流れないなどの不具合が発生。	29日、北朝鮮のミサイル情報を配信。発射から約4分後の6時2分に「発射情報」を、約16分後の同14分に日本上空を飛んだとの「通過情報」を伝えた。その際、防災行政無線の放送が流れないなどの不具合が発生。	不明	• 時事ドットコム（2017.8.29） • 日経コンピュータ（2017.8.29）
1735	NTTドコモJアラート	2017	8	29		浦幌町のNTTドコモの利用者220人に情報が届かなかった。	8月29日に北朝鮮のミサイルが北海道上空を通過した際、浦幌町の同社の利用者に情報が届かなかった。電波を届ける基地局のシステム障害が原因で、調査の結果約16,000人に届かない可能性があった。13日、システム改修を行った。	ソフトウェア障害	• 日テレ NNNニュース（2017.9.12） • 日本経済新聞電子版（2017.9.13） • 毎日新聞（2017.9.13）
		2017	9	13					
1736	JR九州運行管理システム	2017	9	12	10時05分	吉都線は一時都城ー吉松間で運転を見合わせた。普通列車上下4本が運休し、約100人に影響。	12日10時5分頃、列車の運行状況を表示する装置に異常が発生。原因は、除草作業をしていた業者が誤って通信ケーブルを切断したため。	作業ミス	• 西日本新聞（2017.9.12） • NHK NEWSWEB（2017.9.12） • 南日本新聞（2017.9.13） • 宮崎日日新聞（2017.9.13）
		2017	9	12	13時00分				
1737	日本気象協会ネットワーク	2017	9	13	0時58分	気象情報、台風情報、地震・津波情報、注意警報などの情報を、契約している企業・団体へ配信することができず。	9月13日0時58分頃、データセンターにあるネットワークスイッチ1台が故障。機器を復旧し、3時10分順次復旧を確認。障害の発生したスイッチを使用している他のネットワークについても予防対策を計画した。	ハードウェア障害	• 日本気象協会お知らせ（2017.9.13） • 伊勢新聞（2017.9.14）
		2017	9	13	3時10分				
1738	北洋銀行ネットワークシステム	2017	9	27	9時55分	道内などの全171店舗で窓口業務に用いる端末が使えなくなった。これに伴い、口座の新規開設や現金による税金納付など窓口のみで受け付けている業務が滞った。	27日9時55分頃、勘定系と窓口端末をつなぐ社内ネットワーク内の機器障害が発生。2系統ある社内ネットワークの稼働系統で障害が発生したため、手動で予備系統に切り替えて11時35分に復旧。	ハードウェア障害	• 朝日新聞デジタル（2017.9.27） • 日経コンピュータ（2017.9.27）
		2017	9	27	11時35分				

No.	システム名	発生日時（上段） 回復日時（下段）				影響	現象と原因	直接原因	情報源
		年	月	日	時				
1739	東日本銀行	2017	10	16	早朝	ATM利用（提携ATMの利用含む）、インターネットバンキングの利用など、銀行取引のすべてが利用できなくなる。	10月14日の夜間に新ATM稼働に向けたリハーサルを実施したが、誤ってリハーサルプログラムが勘定系システムの本番環境で適用された。そのため14日、15日の取引反映処理に異常が発生。復旧作業が、16日早朝までに間に合わなかった。同日10時45分に店頭窓口業務、11時40分にインターネットバンキング、18時45分にすべて復旧した。	テスト作業ミス	• 東日本銀行お知らせ（2017.10.16）（2017.11.8）
		2017	10	16	18時45分				
1740	福岡空港管制システム	2017	10	22	21時00分	羽田や那覇に向かう予定だった6便が欠航、到着予定だった14便のうち9便が出発地の羽田や宮崎などに引き返し、5便が目的地を関西空港や長崎に変更。少なくとも20便に欠航や出発地に引き返すなどの影響が出た。	22日21時頃、およそ4分間の停電が発生。約10秒後に非常用発電設備で復旧したが、一部管制塔の電気システムに障害が発生した。管制塔のレーダー管制が34分間使えず、滑走路の誘導灯が消えるなどした。停電は、台風による飛来物が高圧線に触れ断線したことによる。システムに障害が出た原因については不明。	不明	• NHK（2017.10.23） • 朝日新聞（2017.10.23） • 毎日新聞（2017.10.23） • 西日本新聞（2017.10.23）
		2017	10	22	21時04分				
1741	大阪モノレール	2017	11	10	11時50分	全18駅で5時間にわたって券売機や改札機が使えなくなったため、ICカードや切符で料金を支払わなくても利用できるようにし、運賃収受を行わなかった。	10日11時50分頃、券売機や改札機を制御する情報システムで障害が発生。モノレールは通常通り運行。およそ5時間後に2系統あるシステムのうち、障害が起きた系統を切り離して再稼働したが、1系統のみで運用しているため券売機や改札機の一部が使えない状態が発生した。原因は、コンピューター機器の不具合。	不明	• 朝日新聞デジタル（2017.11.10） • 読売新聞（2017.11.11） • 毎日新聞（2017.11.11） • 日経コンピュータ（2017.11.10）
		2017	11	10	16時45分				
1742	警視庁運転免許システム	2017	11	19	8時30分	都内府中と鮫洲、江東の各運転免許試験場で、2時間にわたり約1,280人の免許証の交付や更新の手続きができなくなる。	19日8時30分に同庁の運転免許管理システムが起動しなかった。都内3カ所の運転免許試験場で免許証の交付や更新手続きができなかった。原因は、18日の定期メンテナンス時に、新規追加パソコンの設定を誤ったため。	設定ミス	• 中日新聞（2017.11.19） • 時事通信社（2017.11.19） • 時事ドットコムニュース（2017.11.20）
		2017	11	19	10時20分				
1743	JR西日本運行管理システム	2017	11	22		北陸線と越美北線、小浜線、七尾線では始発から運転を見合わせ、普通列車計13本が運休。普通列車21本に最大78分、特急列車5本に最大52分の遅れ、約5,400人に影響した。	22日2時30分頃JR西日本金沢支社の運行管理システムにトラブルが発生。5時10分頃に復旧し、同45分頃から全線で運転を再開した。システム関連の社内工事が原因と見ている。	作業ミス	• 朝日新聞（2017.11.23） • 北国新聞（2017.11.23） • 福井新聞（2017.11.23）
1744	札幌航空交通管制部無線システム	2017	11	24	19時45分	設備が復旧するまでのおよそ1時間半にわたり、北海道と青森県、秋田県、岩手県などの管轄空域で航空機の飛行が制限され、合わせて20便以上が欠航した。	24日19時45分頃、航空機との交信ができず。別の管制部がほかの周波数を使って交信し航空機を誘導。原因は、無線設備の電力供給装置の故障。バックアップも正常に作動しなかった模様。	ハードウェア障害	• NHK（2017.11.25） • 日本経済新聞（2017.11.25）
1745	伊予銀行ATM	2017	11	24	20時29分	当行ATMでのカード取引、他行ATMでの当行カード取引、コンビニATMでの当行カード取引、WEB口座振替受け付け、デビット取引などの取引できず。	11月24日20時29分～21時32分の間、システム障害が発生。出金・キャッシングサービスが利用できない状況になった。	不明	• 伊予銀行HP（2017.11.27）
		2017	11	24	21時32分				

No.	システム名	発生日時（上段） 回復日時（下段）				影響	現象と原因	直接原因	情報源
		年	月	日	時				
1746	大分県警 運転免許 センター	2017	12	3	8時10分	約150人の運転免許の手続きができず。有効期限の近い25人は手作業で対応した。	3日8時10分頃、運転免許申請機械5台と免許証作成機器2台でエラーが発生。原因は、県警本部の運転免許管理システムを毎日業務終了後に終了させる運用だったが、1日（前営業日）に終了させていなかったことによる。	操作ミス	• 大分合同新聞 (2017.12.4) • 朝日新聞（2017.12.4） • 西部読売新聞 (2017.12.4)
		2017	12	3	9時30分				
1747	日本年金機構 支給システム	2017	12	20		2010年1月から2017年3月までに発覚した事務処理ミスで、10件以上のミスが33種類で発生している可能性があることが判明。うち今後も支給ミスの発生可能性があるものは18種類に上った。	元公務員の配偶者の基礎年金に一定額加算する「振替加算」の事務処理ミスで支給漏れがあった問題で、総点検を実施。結果、更に問題があることが判明。問題の対象者を特定するプログラムを作成して2018年4月から1年間、対象者に通知し年金の未払いと過払いに対応する。	プログラム 不具合	• 産経新聞（2017.12.21） • 毎日新聞（2017.12.21） • 朝日新聞（2017.12.21） • 読売新聞（2017.12.21） ※障害発生は、報道された日とした。
1748	関西エアポート (伊丹空港) 駐車料金 システム	2017	12	23	0時00分	出庫した車1,238台から駐車料金を取り過ぎるミスがあった。1台当たり50～900円多く徴収し、合計は約29万円に上った。	20日から駐車場料金を改定したが、料金システムの設定を間違え、23日は平常料金でなく繁忙期の料金を請求した。利用者からの問い合わせで判明。今後は設定結果をダブルチェックする。	設定ミス	• 読売新聞（2017.12.25） • 日本経済新聞 (2017.12.25) • Aviation Wire (2017.12.25)
		2017	12	23	17時05分				

別表　2017年後半の情報システム障害データ（報道に基づきSECが整理）

No.	システム名	発生日時（上段） 回復日時（下段）				影響	現象と原因	直接原因	情報源
		年	月	日	時				
1	大阪市 斎場予約受付 システム	2017	7	11	19時00分	斎場予約の受け付けができない状況が発生。	7月11日19時過ぎに斎場予約受付システムに障害が発生し、22時30分頃復旧。12日未明頃、再度障害が発生、10時45分頃復旧した。障害原因は不明。	不明	• 大阪市報道発表 (2017.7.12)
		2017	7	12	10時45分				
2	水俣市 固定資産税	2017	7	21		283件の課税ミスが発覚。本来徴収するべき金額よりも少なかったのは計約3万円、多かったのは計約130万円だった。	市民からの問い合わせにより、課税ミスが判明。誤った修正用プログラムを使用した土地評価額の算定ミス、職員のデータ入力ミス、09年度の土地の固定資産評価基準に関する改正内容の反映ミスなど。チェック体制を強化する。	プログラム 不具合	• 熊本日日新聞 (2017.7.22)
3	彦根市 保険料督促	2017	7	25		国民健康保険料と介護保険料の督促状1,973通で、合計額を「0」として発送していた。	保険料の督促状で、納付額と督促手数料の合計額が全通「0」になっていた。25日に金融機関からの問い合わせで発覚。8日にプログラムを更新した際、誤った。	プログラム 不具合	• 朝日新聞地方版 (2017.7.27) • 中日新聞地方版 (2017.7.27)
4	高知市 税務情報システム／後期高齢者医療システム	2017	7	27		市内の後期高齢者ら93人に対し、2017年度分の後期高齢者医療保険料を計200万4,900円過大請求した。	5月のプログラム改修で、被保険者の所得データを税務情報システムから後期高齢者医療システムへ取り込んだ際、株式に関する所得を誤って二重計上した。今後は委託業者と市で相互チェックを行う。	プログラム 不具合	• 高知新聞（2017.7.28）

No.	システム名	発生日時（上段） 回復日時（下段）				影響	現象と原因	直接原因	情報源
		年	月	日	時				
5	高岡市 総合行政情報 システム	2017	7	31	8時30分	市役所や各支所などの端末から接続できなかったため、住民票や税証明といった各種証明書の発行など190件に対応できなかった。	31日8時30分～11時15分、「総合行政情報システム」が立ち上がらず、窓口業務の一部を一時停止した。同日市役所や各支所などの端末から接続できなかった。	不明	• 毎日新聞地方版（2017.8.1） • 朝日新聞地方版（2017.8.1）
		2017	7	31	11時15分				
6	相模原市 住民サービス	2017	8	7	13時00分	市内274店舗のコンビニエンスストアで戸籍証明書が受け取れるサービスについて、6人が取得できなかった。	市の委託業者が4日夜に実施したメンテナンス後、サーバーの再起動を行わなかったため、7日13時頃から住民サービスで障害が発生。	作業ミス	• 朝日新聞（2017.8.9） • 神奈川新聞（2017.8.10）
		2017	8	9					
7	岐阜市 市営駐車場料 金システム	2017	8	17		岐阜市営の駅西駐車場と岐阜シティ・タワー43地下駐車場で、約16年にわたって障害者利用時の駐車料金を過徴収。昨年度の利用実績から過徴収は約2万件、約20万円に上る見込み。	障害者利用の駐車料金は通常の半額で10円未満は切り捨てだったが、精算機のプログラムは切上げの設定だった。そのため、2001年12月1日から1件当たり10円ずつ過徴収が発生していた。職員の問い合わせで発覚。	プログラム 不具合	• 岐阜新聞（2017.8.18） ※障害発生は、報道された日とした。
8	埼玉県HP	2017	9	1	18時00分	埼玉県HP「問合せ」フォームが利用できず。	9月1日18時から23時30分まで、システム障害発生。	不明	• 埼玉県お知らせ（2017.9.4）
		2017	9	1	23時30分				
9	秋田大仙市 選挙管理 システム	2017	10	16	8時30分	16日午前、大仙市内8カ所に設けられた衆院選の期日前投票所で約230人が投票できない事態となった。	8カ所の期日前投票所で16日8時30分からシステムに接続できない状態になった。同10時頃には復旧した。	ネットワーク 障害	• 読売新聞（2017.10.17） • 河北新報（2017.10.17）
		2017	10	16	10時00分				
10	一関市 総合行政情報 システム	2017	10	17	15時00分	130人の期日前投票受付や窓口業務ができなくなった。	市のイントラネットのファイアウォールに不具合が発生した。同不具合は、再発防止の措置を行い、一両日中に完了した。	ソフトウェア 障害	• 一関市市政情報HP（2017.10.18） • 岩手日報（2017.10.18） • 毎日新聞地方版（2017.10.19）
		2017	10	17	16時40分				
11	三重県、 県内21市町 メールシステム	2017	10	20	未明	県庁や市役所などでメールが使えなくなった。県のほか、21市町、紀北広域連合、三重県後期高齢者医療広域連合、紀南介護保険広域連合においても同様の障害が発生。	21日未明、県と県内21市町の共用メールサーバー（自治体情報セキュリティクラウド内）でシステム障害が発生。サーバー内の設定誤りで、メールの送受信の記録データを保存する際に異常が発生。サーバーの設定を変更して復旧。	設定ミス	• 中日新聞（2017.10.21） • 三重県（2017.10.20）
		2017	10	20	18時00分				
12	神戸市	2017	11	13	12時00分	神戸市西区、北区、東灘区の各区役所や支所など17カ所で市民ら85人前後の証明書が発行できなくなった。約10人が年金や保険などの資格手続きができなくなった。	13日正午過ぎ、通信障害が発生し、住民票や市民税などの証明書を発行する機能が停止した。原因は、ネットワーク通信機器の入れ替え作業中、サーバーが停止した。原因調査中。	不明 （作業ミス?）	• 産経新聞（2017.11.14）
		2017	11	13	13時00分				
13	鈴鹿市 ネットワーク	2017	11	20	8時30分	地区市民センター窓口端末から住民票や印鑑証明の発行業務ができず。計12人が延べ18通の発行を本庁で受けたが、5人が未手続きで帰宅。	20日午前8時半から25分間、情報ネットワークシステムのトラブルが発生。原因は18、19日のネットワーク制御装置更新時の設定ミス。設定変更で復旧した。	設定ミス	• 伊勢新聞（2017.11.21）
		2017	11	20	11時55分				
14	市原市下水道 料金システム	2017	12	13		平成15年度以降約5,300件で過大請求し、計約700万円を過徴収	下水道料金算定プログラムで、引っ越しなどで定期検針後に使用した人に、誤った計算式で料金を算定し、誤請求となった。	プログラム 不具合	• 産経新聞（2017.12.14） ※障害発生は、報道された日とした。
15	鳥取県HP	2017	12	21		鳥取県のフォトコンテストのWebサイトに、受賞者14人の住所や電話番号などが数日間にわたり誤って公開された。	21日、受賞者の個人情報を管理システムに入力したところ、プログラムミスでWebサイトにも表示。受賞者の1人から、不具合を伝えるメールを受け取っていたが25日まで気づかず。	プログラム 不具合	• 産経ニュース（2017.12.27）
		2017	12	25					

[松田 2017]。

なお、障害の影響は特定の地域に限られてはいるが、地域の住民にとって大切なサービスが影響を受けたシステムの障害が15件報道されている。その安定的な運用への注意を促すために別表に別枠として整理した。

2. Jアラート関連の障害

2017年に入って北朝鮮のミサイルが日本上空を通過したり、日本海の排他的経済水域へ落下するなど、我々の安全が脅かされる事態が繰り返し発生している。このような事態に対し安全を確保するためには、速やかに正確な情報を確実に伝達することが欠かせない。Jアラートはこのような目的のために設けられているにもかかわらず、今期にも3件の不具合が報告されている。過去にもJアラートの障害は報告されており（事例1635、1636）[松田2 2016]、送受の訓練が行われているようであるが残念ながら不具合が収束していない。

またJアラートとは異なったシステムではあるが、同様に緊急時の情報を伝達するための自治体の防災情報システムのトラブルも過去に数件報道されている（事例1430、1432、1506、1531、1532、1541、1601、1608）[松田 2 2014] [松田2015] [松田 2 2015] [松田2016] [松田 2 2016]。

いざと言うときに頼るべきシステムの信頼性が低くては、その役割を果たすことはできない。日常には使われず、緊急事態が起こったときにのみ機能するシステムの保守・運用についてはとくに注意を払い、定期的な訓練に合わせてシステムの動作状況を確認、点検するなどの運用ルールを定着させ確実に実施することが必要である。

3. システム障害に起因するセキュリティ問題

事例1725では、ネットを通して個人間の売買を仲介するシステムにおいて、サーバーの切替えに不具合が発生し、5万4000人を超える顧客の名前、住所、メールアドレス、銀行口座やクレジットカード番号などの個人情報が第三者に閲覧される状態になり、顧客からの問い合わせによって事故が発覚した。更に、別表に示す事例15では、プログラムの不具合によって氏名、住所、電話番号などの個人情報が4日間もWebサイト上に掲載されたままになるという事故が発生している。これらは、いずれもシステム障害が原因で個人情報が流出した事例である。

セキュリティ問題は、システム障害と同様に利用者の安全、安心を脅かす重大な脅威であるが、それを引き起こす要因は異なり、前者は悪意を持った攻撃者による意図的なものであるのに対し、後者は意図しない何らかの理由によってシステムに内在していた欠陥が顕在化する偶発的なものである。しかし、偶発的なシステムの障害であっても、本事例のようにその影響が個人情報の流出につながり、セキュリティ問題を生じさせる危険性があることに注意が必要である。

なお、このような事例は2017年前半にも2件報告されており、更にさかのぼってみると、過去に6件の事例が報告されている（事例1127、1216、1329、1423、1530、1621）。また、漏えいには至らないがその可能性のある事故も発生している（事例1533、1534）[松田2 2015]。

それぞれの詳細については50号で取り上げた[松田2017]のでここでは繰り返さないが、セキュリティ事故は、システム外部の悪意ある攻撃者によって引き起こされるだけではなく、意図しないシステムの障害によっても発生していることに留意が必要である。

4. テスト作業の本番環境への悪影響による障害

事例1739は、銀行のオンラインシステムにおいて障害が発生し、ATMやネットバンキングなどを含む銀行の全業務が終日停止した事例である。この障害の原因は、ATMの新機種への更改に向けて深夜に実施したテスト用プログラムが、誤って勘定系の本番環境に適用されてしまい取引データに不整合が発生、その復旧に長時間を要したことであった。

システムの保守のために、本番環境を用いてテストを実施することは通常よく行われることであり、本件のような

事故はどのシステムでも起こり得る。事実、過去にさかのぼって同様の事例を見てみると、事例1011、1013［松田2010］、事例1405［松田 2014］などでは、テストに用いたデータの削除を忘れて本番稼働に入ったために事故を起こしている。ケアレスミスではあるが、発生したときの事故の影響は大きいため、テスト終了後のテスト環境から本番環境への切り戻しが確実に行われるよう、作業手順の整備と正確な実施、作業完了の検証などが重要である。

5. むすび

2017年後半の情報システムの障害について、報道などをもとに整理し報告した。

これまでの連載記事のバックナンバーについては、SEC Webサイトにまとめて掲載されているので、「SEC journal連載」などで検索するか、下記のURLからアクセスして参考にしていただきたい。

■SEC journal連載：情報システムの障害状況

URL：https://www.ipa.go.jp/sec/system/system_fault.html

IPA/SECでは、ITシステムの事故の経験を共通の財産として共有し、安心・安全なIT社会を目標に、これらの障害事例を分析し、参考にすべき教訓をくみ取る活動を進めている。教訓がまとまるごとに下記SEC Webサイトで公開している。

■情報処理システム高信頼化教訓のリンク集（ITサービス編）

URL：https://www.ipa.go.jp/sec/system/lesson.html

更に、これらの教訓やSEC journalの報道事例を検索しやすいように、ポイントや全体像をつかめるよう「注意すべき観点」に基づいて分類した障害事例の一覧を用意した。

■「注意すべき観点」に基づいた障害事例の分類

URL：https://www.ipa.go.jp/sec/system/index.html#shougaijirei

また、教訓集活用メールマガジンの配信も行っているので、興味のある方は上記SEC Webサイト「情報処理システム高信頼化教訓のリンク集」のページからメール配信の登録をしていただきたい。

■「情報処理システム高信頼化教訓集（ITサービス編）」をより有効にご活用いただくためのメールマガジンの登録について

URL：https://www.ipa.go.jp/cgi-bin/enquete/registEnquete.cgi?EID=55387577eb35c55e7ca118cb3c043e85

更に、教訓をまとめた教訓集がSEC Webサイト上に公開されているので併せて参考にしていただきたい。

■「情報処理システム高信頼化教訓集（ITサービス編）」2016年度版公開

URL：https://www.ipa.go.jp/sec/reports/20170327.html

【参考文献】

［松田 2010］松田晃一・金沢成恭：情報システムの障害状況 2010年データ、SEC journal No26、Vol. 7, No3, pp.102-pp.104, Oct.2011
［松田 2014］松田晃一・八嶋俊介他：情報システムの障害状況 2014年前半データ、SEC journal No38、Vol. 10, No3, pp.42-pp.47, Sep.2014
［松田2 2014］松田晃一・八嶋俊介：情報システムの障害状況 2014年後半データ、SEC journal No40、Vol. 10, No6, pp.44-pp.47, Mar.2015
［松田 2015］松田晃一・八嶋俊介：情報システムの障害状況 2015年前半データ、SEC journal No.42、Vol. 11, No2, pp.32-pp.37, Sep.2015
［松田2 2015］松田晃一・八嶋俊介：情報システムの障害状況 2015年後半データ、SEC journal No.44、Vol. 11, No4, pp.48-pp.53, Mar.2016
［松田 2016］松田晃一・八嶋俊介：情報システムの障害状況 2016年前半データ、SEC journal No.46、Vol. 12, No2, pp.43-pp.49, Sep.2016
［松田2 2016］松田晃一・八嶋俊介：情報システムの障害状況 2016年後半データ、SEC journal No.48、Vol. 12, No4, pp.62-pp.67, Mar.2017
［松田 2017］松田晃一・目黒達生：情報システムの障害状況 2017年前半データ、SEC journal No.50、Vol.13 , No.2, pp.52 -pp.58, Sep.2017

SECjournal 論文賞　受賞論文発表

SECは、我が国ソフトウェア産業発展のための様々な取り組みを実施しておりますが、その一つとして、ソフトウェア工学に関する論文を募集し、優秀な論文に対し、表彰を行っております。
今年度のSECjournal 論文賞は、2016年8月から2017年7月までに査読者による審査を経て採録が決まった9編の論文を候補とし、更に選考委員会と表彰委員会による厳正な審査の結果、3編を選出いたしました。
各賞の発表と表彰式は2017年11月15日にEmbedded Technology 2017内で実施いたしました。本年は最優秀賞1編、所長賞2編が選出されました。

【最優秀賞】

要求仕様の一貫性検証支援ツールの提案と適用評価

位野木 万里　　近藤 公久
（SEC journal 49号掲載）

【所長賞】

CMMI成熟度レベル別に見たソフトウェア品質の良否にかかわる要因の複合的分析

柳田 礼子　　野中 誠　　誉田 直美
（SEC journal 49号掲載）

【所長賞】

提案依頼書に含まれる無理難題の分類

門田 暁人　　住吉 倫明　　神谷 芳樹
（SEC journal 51号掲載）

SECjournal 論文賞　表彰委員会審査報告

SECjournal論文賞
表彰委員会委員長

北陸先端科学技術大学院大学
東京工業大学
名誉教授

片山 卓也

今回は、2016年8月からの1年間に採録となった論文を対象に、論文賞選考委員会、表彰委員会で審査を行い、以下の論文を優秀論文として表彰することを決定した。優れた内容のものであると同時に、実際の開発現場における有効性などを評価の主な観点とした。授賞に値する具体的評価は次の通りである。

「要求仕様の一貫性検証支援ツールの提案と適用評価」

ソフトウェア開発の入口であり、重要な位置付けにあるのが要求仕様である。しかしながら、要求仕様の記述は必ずしも一貫しておらず、直接設計に関連する用語にぶれがあると、その後の開発に大きな影響を与える。本論文は、要求仕様の品質特性である「一貫性」に着目し、ベテラン技術者が経験的に得た検証知識、例えば、「アクター」「データ」「画面」「振る舞い」の設計要素が、要求仕様書で一貫した定義で記述されていることを確認するといった検証ノウハウを、ルールと辞書として形式知化し、それら知識に基づき、要求仕様の一貫性検証支援ツールを実現したものである。要求仕様の一貫性検証という、開発現場にとって重要課題である非常に魅力的なテーマに取り組んだこと、提案の検証支援ツールにより、ベテラン技術者のみが従事していた要求仕様検証を、これまで検証の経験がない組織でも取り組むことが期待できることなどの点が評価された。検証支援ツールの完成度も高く、開発現場での適用検証によりその効果も確認されている点、多くの組織で利用できるよう辞書やルールのカスタマイズが考慮されている点など、実フィールドで供し得る実用性も高く評価された。

「CMMI成熟度レベル別に見たソフトウェア品質の良否にかかわる要因の複合的分析」

本論文は、ソフトウェア品質を効率的に向上させるための研究である。ソフトウェア品質の良否に影響する要因の一つとして、開発組織の能力がある。業界では、CMMI（Capability Maturity Model Integration）で示される開発組織の能力が、インデクスとして認識されている。本研究では、商用ソフトウェア開発プロジェクト522件のデータを対象に、成熟度レベル別に分類木を構築し、有意差検定と相関分析を組み合わせて、ソフトウェア品質の良否に影響する要因を複合的に分析し主たる要因を導いている。多くの社内の実データによって評価し、信頼性が高いことに加え、社内の実データや分析結果を公開し、産、学で広く利用可能にした点も高く評価された。分析過程の解説も記述されているため、読者の自社データを用いた分析も可能となり、利用性も高まっている。CMMIの記述や、過去の知見や経験則が、自社の実データで実証されており、現場を動かすモチベーションにもなる研究であると判断した。

「提案依頼書に含まれる無理難題の分類」

本論文は、提案依頼書RFPにどのような無理難題が含まれているかを明らかにし、ベンダの損失発生のリスクを軽減させる研究である。RFPに注目しプロジェクト失敗の原因を探るというアプローチはこれまでにも存在するが、RFPの記述の漏れや正確さといった観点ではなく、記述内容の無理難題への着目は新たな観点であり、今後の発展が期待できる。分析結果から得られた「無理難題事例集」は発注者、受注者共に参考にできるものであり、利用性の高さも評価された。研究結果を活用できる場面も多いと同時に、RFPにあえて無理難題を含めることがあるという点も、広く認識されている。現実に存在するこのような商慣習に関して、問題提起につながるという点も評価された。

Column

変革期のリーダーシップ

IPA顧問、学校法人・専門学校HAL 東京 校長 **鶴保 征城**

新年が明けて早々の1月4日、星野仙一氏が亡くなった。生涯勝利146勝で投手としても一流だが、何と言っても星野氏の手腕は中日・阪神・楽天監督として発揮された。優勝に導いたチームはいずれも下位に低迷していたチームであり、組織を立て直し好成績に結び付けた手腕は高く評価されている。阪神ファンの筆者にとっては、18年間も優勝に縁がなかった「ダメ虎」を2003年に勝たせてくれたことが印象深い。道頓堀川に飛び込む若者まで出てきて、大阪は大いに盛り上がった。

管理職を対象としたものだったが、理想とする上司イメージに関するアンケートで、星野氏は常に上位に挙げられている。叱咤激励の中にも選手への信頼があり、ここぞという場面では選手を守ろうとする姿勢が、理想の上司イメージとして支持されたようだ。

経営学の分野では多くのリーダーシップ論が交わされている。経営資源として最も重要なものが人材であることに異論はないが、優秀な人をたくさん雇って、より多くの仕事で成果を出し、更に多くの仕事をこなすために人を雇うというサイクルをうまく回すためにはどのような管理やリーダーシップが良いのか。これまでの研究では2つのタイプが議論されている。一つは、部下の自己意思を重んじながらも、正に取引のように部下とやり取りするリーダー。部下に対して「アメとムチ」をうまく使えるタイプで、管理型のリーダーと言える。もう一つは、大きな方向性は示すが実行の細部は部下に任せるタイプ。ビジョナリーなリーダーであり、トップダウンよりもチーム構成員のコミュニケーションを重視する。これまでの研究では、どちらかが優れているというよりも、両方のマネジメントスタイル共に大事で、一つの会社の中でも使い分けることが重要というのが結論のようだ。ただ、変化が激しいこれからの時代には、変革を志向する後者の重要性が増してくると言われている。

ITサービス業界を席巻しているGAFAの一角を占めるGoogleは人事問題にも熱心で、常に「優秀なボスとはどんな人だろう？」と考えていた。そのためにOxygen（オキシジェン）というプロジェクトを2009年にスタートした。オキシジェン・チームは、社員1万人以上に100項目以上から成る詳細な調査を1年近い時間を費やして実施した。その結果を8つのポイントにまとめている。それは、

1. **良いコーチであること**
2. **ある程度部下に任せ、細かい管理をしないこと**
3. **部下の成功と幸せを気にかけていることを態度で示すこと**
4. **生産的で成果志向であること**
5. **コミュニケーションを良く取り、チームの意見に耳を傾けること**
6. **部下のキャリア開発を支援すること**
7. **チームのために明確なビジョンと戦略を持っていること**
8. **部下にアドバイスできる重要な技術的スキルを持っていること**

であった。

誰もが知っている当たり前なことばかりで、今や世界中から秀才・天才が集結する企業Googleのリーダーシップ論としては物足りないぐらいだ。だからと言ってGoogleの研究に価値がないのではない。むしろ逆で、誰もが知っているし口にしているけれど、誰もきちんと調べていなかったことを、初めて科学的な手法を使って確かめたことは大きな前進だと思う。

あらためて星野氏の監督ぶりを振り返ってみると、選手や家族の誕生日にはお祝いの言葉やプレゼントを欠かさなかったなど、Googleの8項目に一致するものが多い。会社やプロジェクトのマネジメントで大いに参考になるのではないだろうか。

書籍紹介

目で見る機能安全

神余 浩夫　著

ISBN：978-4-542-30703-2
一般財団法人 日本規格協会
A5判・204ページ
定価2,000円（税抜）
2017年4月6日刊

社会の重要インフラを担う制御システムでは、ライフサイクル全般にわたる安全性を重視したものづくりが行われており、「機能安全」という“コンピュータ技術による安全制御で安全性を確保する方法”で実現されている。

しかし、「機能安全」と聞くと、国際安全規格の「IEC 61508」や「ISO 26262」を思い浮かべ、専門用語が多く難しい、規格は分かりにくいというイメージが先行しているのも事実である。

本書は、機能安全規格の解説書ではなく、家電、自動車、鉄道、エレベーター、ロボットなど、各分野の製品事例を取り上げ、多くの写真や図による解説を通じて、直感で機能安全を理解することができる、正に「目で見る機能安全の実例」であり、入門書として最初に読んで欲しい内容に仕上がっている。

著者は、三菱電機株式会社にて、長年実際の製品開発に携わり、機能安全分野、及び最近注目されている制御システムセキュリティ分野の国際エキスパートである。本書のプロローグで著者は、機能安全技術は、製品価値・企業価値を向上させることができ、その結果「機能安全で儲かる」とビジネス視点でのメッセージを読者に伝えようとしている。

「機能安全を知ることは、安全なシステムを作る第一歩」だが、昨今、制御システムはセキュリティ対応も必須であり、セキュリティを考える上で、機能安全の基本を理解しておくためにもぜひお薦めしたい一冊である。

（細目 紀子）

プロジェクトをうまく進めるための17の鍵
～ImprovAbilityによるプロジェクトリーダのための プロジェクト健全化技法～

清 雄一、菊島 靖弘、
石谷 靖　他著

ISBN：978-4-274-70000-2
株式会社オーム社
四六版・158ページ
定価1,800円（税抜）
2017年8月1日刊

本書は2013年に発行されたImproVability, Success with process improvementと、それをベースにしたISO/IEC TR 33014 Guide for process improvementの、日本における実践からの学びをまとめたものである。前述2つの文献は「プロセス改善」を題に付けているが、日本においては、実践を通してむしろプロジェクトを成功裏に進める上で効果的であったことから、本書の題は「プロジェクトをうまく進めるための17の鍵」となっている。この、プロジェクト実行中に活用できるゆえんが「17の鍵」にある。

「測れないものは管理できない」ということが、ともすれば多くの中間チェック、レビュー、メトリクス測定・評価を要求することに陥りがちであり、それに関連する労力は無視できない程度になり、下手をすれば形式的に運用されることになる。17個のパラメータであれば、そんなに労力を投入せずとも、現状把握、分析、対応策作成・実施が可能であろう。実際のプロジェクト導入経験から、著者らは「ImproVabilityは、慣れさえすれば短期間で実施可能です（ケースによっては1日で実施できます）」と言及している。前述17個のパラメータがいかに導出されたかについて関心ある方は、「2.3 17パラメータの導出経緯」を参考にされると良いであろう。ImproVabilityのエッセンスを説明するものである。

本書の特色は、ベストプラクティスとそれをベースにした国際規格を説明するのではなく、日本における実践を通した経験を読者に伝え、ImproVability認定者でなくてもこの方法を活用できるように、ステップを踏んで評価するテーブルなどが具体的に説明されていることである。

（新谷 勝利）

編集後記

今号はシステム安全性分析手法STAMPを特集しています。高度なソフトウェアで制御されるシステムを、普段の生活の中で当たり前に使っているというこの状況は、ますます加速するものと思います。そのため、システムの安全確保のための方法論に産業界の興味が集まってきています。欧米を中心に注目されているシステム理論に基づく新しい安全性分析方法論「STAMP」を今号で紹介することとしました。今号の特集をご覧になり、システム安全性に関する各界の最新の取り組みに触れていただくことにより、自社への活用のための新たなヒントが得られればうれしく思います。

年度が変わる時期、新たな組織や担務に移る方もいらっしゃると思います。新年度を迎えましても、変わらぬお付き合いの程、よろしくお願い申し上げます。

（編集長）

編集部より

次世代のソフトウェア・エンジニアリングに関してなど、忌憚のないご意見をお待ちしております。下記の FAX またはメールにてお気軽にお寄せください。

SEC journal 編集部　FAX：03-5978-7517
e-mail：sec-journal_customer@ipa.go.jp

春の匂い　撮影：K.Hasegwa

SEC journal 第13巻 第４号（通巻55号）　2018年３月１日発行

編集兼発行人　独立行政法人情報処理推進機構
技術本部　ソフトウェア高信頼化センター
所長　松本 隆明
〒113-6591　東京都文京区本駒込2-28-8　文京グリーンコート センターオフィス16階
Tel：03-5978-7543　Fax：03-5978-7517
URL：https://www.ipa.go.jp/sec/　e-mail：sec-journal_customer@ipa.go.jp

SEC journal 論文募集

独立行政法人情報処理推進機構（IPA） 技術本部 ソフトウェア高信頼化センターでは、下記の内容で論文を募集しています。

論文テーマ

・ソフトウェア開発現場のソフトウェア・エンジニアリングをメインテーマとした実証論文または先導的な論文
・ソフトウェアが経済社会にもたらす革新的効果に関する実証論文

論文分野

品質向上・高品質化技術、レビュー・インスペクション手法、コーディング手法、テスト／検証技術、要求獲得・分析技術、ユーザビリティ技術、プロジェクト・マネジメント技術、設計手法・設計言語、支援ツール・開発環境、技術者スキル標準、キャリア開発、技術者教育、人材育成、組織経営、イノベーション

応募要項

締切り　：1 月・4 月・7 月・11 月　各月末日
査読結果：締切り後、約 1 カ月で通知。「採録」と判定された論文は SEC journal に掲載されます。
応募方法：投稿は随時受付けております。応募様式など詳しくは HP をご覧ください。
https://www.ipa.go.jp/sec/secjournal/papers.html

SEC journal 論文賞

毎年「採録」された論文を対象に審査し、優秀論文には SECJournal 論文賞として最優秀賞、優秀賞、所長賞を副賞と併せて贈呈します。

IoT 時代に活躍する**【組込みシステムの腕利きエンジニア】**を目指す！

国家試験 エンベデッドシステムスペシャリスト試験

高度な実践能力の証明に！

▶ 身近な場面を想定した出題を通して、最適な組込みシステム実現のために必要となる高度な実践能力（レベル 4）を問います。
（レベル 4 の定義）：専門分野において、自らのスキルの活用によって、独力で業務上の課題の発見と解決をリードするレベル。

技術要素	開発技術	管理技術
プロセッサ、メモリ、バス、計測・制御、リアルタイム OS、プラットフォーム、電気・電子回路、ネットワーク、セキュリティ	・要求分析の実行とレビュー ・設計の実行とレビュー ・テストの実行とレビュー	・開発環境マネジメント ・知財マネジメント ・構成管理、変更管理

▶ 近年の試験では、「無線通信ネットワークを使用した安全運転支援システム」、「3 次元複写機」、「通信機能をもつ電子血圧計を用いた健康管理システム」、「非接触型 IC カードを使用した入退場ゲートシステム」などのテーマを出題しました。

▶ 自動車、家電、モバイル機器などに搭載する組込みシステムや重要インフラの制御システムを、ハードウェアとソフトウェアを適切に組み合わせて構築し、求められる機能・性能・品質・セキュリティなどを実現できる組込みエンジニアを目指す方に最適です。

試験概要

【試験区分】エンベデッドシステムスペシャリスト試験（情報処理技術者試験 高度試験の 1 区分として実施）
【日　　時】年 1 回の実施（毎年 4 月第 3 日曜日）
【申込受付】毎年 1 月中旬から 2 月下旬（予定）まで WEB・郵送で申込み受付

詳しくは、Web ページをご覧ください。https://www.jitec.ipa.go.jp/index.html
試験概要の最新情報、過去問題、活用事例などをご紹介しています。

IPA Better Life with IT

SEC journal No.52
第13巻第４号（通巻55号）
2018年３月１日発行

ISSN 1349-8622