2017年3月1日発行
第12巻 第4号(通巻51号)
ISSN 1349-8622

SEC journal 48

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

SEC journal No.48
Vol.12 No.4 Mar. 2017

巻頭言

IoT時代のシステムズエンジニアリング

狼 嘉彰
慶應義塾大学システムデザインマネジメント研究所顧問

システムズエンジニアリング国際協議会（INCOSE）について

システムズエンジニアリングをシステム工学と和訳してみると、1960年代のアポロ計画が華やかな頃に一世を風靡したことを思い起こさせる。「アポロの成功はシステムエンジニアリングの勝利である」（フォン・ブラウン）とまで言われ、大規模なプロジェクトを成功に導く手法として、多くの人々の注目を集めた。それにもかかわらず、1990年代中頃から再び注目を集め、会員数１万人を超える「システムズエンジニアリング国際協議会（INCOSE：International Council of Systems Engineering）」が活発に活動するに至っているのは、なぜであろうか。理由は、領域の拡大と利害関係者の多様化にある。確かにアポロは、大規模複雑かつ先進的なシステムであったが、すべての主要技術は、米国航空宇宙局（NASA）の手のうちにあり、NASAが自在に操ることができた。しかし、時代の変遷と共に、自動車・コンピュータ・家電などの一般消費財の技術革新が宇宙分野の技術を追い越し、ついには、インターネットが中心的なプレーヤーとなるIoT時代に突入し、事態は一変した。

1990年代初頭、米国における宇宙・防衛関連の巨大プロジェクトについて、コスト・納期・性能（QCD）すべてを満たすべしという厳しい観点からは、ほとんどすべてのプロジェクトが失敗と評価せざるを得ない事態が生じた。その反省から、アポロ時代の遺産であるシステムエンジニアリングを見直し、新たなシステムズエンジニアリングを強力に推し進める協議会を設立し、現在に至っている。両者の相違は、「ズ（英語ではs）」の有無だけである。現在は、南北アメリカ、欧州・アフリカ、アジア・オセアニアの３拠点（Sector）に分かれ、日本はINCOSEの日本チャプターをスタートさせJCOSEを設立して、オーストラリア・シンガポール・中国・インドなどと共にセクター IIIの一員として活動している。システムズエンジニアリングの目的が、大規模システムの実現成功を目指すものであるから、INCOSEは航空宇宙関連企業や政府系機関が賛助会員組織（CABと呼ばれる）であるが、工学分野で顕著な活動をしている米国MITやその付属研究所、及び多くの大学が主要メンバである。

システムズエンジニアリングはシステム開発成功の鍵を握る

このINCOSEの主要な出版物の一つである「INCOSEハンドブック」によれば、システムズエンジニアリングとは、「システムの実現を成功させることができる複数の専門分野にまたがるアプローチ及び手段」と定義される。従来のシステム工学と類似のようで実は大きな相違がある。最も重要なことは、具体的なモノ・コトを実現するエンジニアリングであり、成功が最も重要なキーワードである。複数の専門分野にまたがることは今の時代では常識であろうが、アプローチは意味が深く、日本語には適訳が見当たらない。これには、システム思考やアーキテクティングあるいはプロセス重視などの意味が含まれる。また、手段には、ソフトウェアに支援された多くの数学的な手法が含まれる。究極の目的は、「成功」である。

JCOSE（日本におけるINCOSE支部）の活動

アジア・オセアニア領域では、アジア太平洋システムズエンジニアリング国際会議（APCOSEC）が中心である。2007年から2016年まで、オーストラリア・シンガポール・台湾・日本・韓国・中国・インドにおいて毎年開催され、本年2017年はINCOSEシンポジウムと共催でオーストラリアにおいて開催される。JCOSEは、2008年と2013年に横浜市での開催を主催した。しかし、日本では、主体となるべき企業や事業体のINCOSEに対する関心が極めて低い。具体的には、INCOSE会員やCSEP資格（INCOSEが認定するSEのプロの資格）を持つ人の数がほかのAPCOSE関係国に比べて圧倒的に少ない。学会・国際会議などの活動は、大学の役目であり企業のミッションではないとする傾向が最近は顕著になってきたが、このほかにも専門分野に特化すること、あるいは職人芸を重視する文化的な特質もあろう。これに関して、2013年のAPCOSEC会議におけるエピソードを紹介したい。この年は、伊勢神宮の遷宮の年であることから、神宮禰宜（ねぎ）の方に「精神文化継承システム」と題してキーノートをお願いした。遷宮は、神社本体のみならず1500点を超える神宝を前のモデルと寸分たがわぬ形で作り変えるため、2000人を超える当代一流の匠が参画すると説明された。まさに職人芸の極致であり、日本人の誰もが称賛する成果である。しかし、海外からの出席者から、「イノベーションをどのように反映するのか」という質問が繰り返し出されたが、「イノベーションは文化伝承になじまないので、全く不必要」と明快に回答された。この文化的風土を物語っていると思われる。

日本におけるシステムズエンジニアリングの活動を活発化するために

このような文化を背負いつつ、グローバル化の時代を生き残っていくには、日本に相応しいシステム・アプローチが不可欠である。IoTあるいはICT時代のシステムズエンジニアリング（SE）は、ますますソフトウェア・システム・エンジニアリング（SWE）が協調・補完していく傾向が顕著である。第一の理由は、ビッグデータと通信手段の飛躍的な発展により、多用なステークホルダ（利害関係者）と複雑な関係性を瞬時に持つことが可能になり、データの収集・分析・結論の導出など、両者（SEとSWE）が混然一体となって取り組まざるを得ない事態が日常的に生じている。第二の理由として、いわゆる組込みソフトウェアの一般化・普遍化である。従来は、扱いが単純であった計測・制御機器にもほとんど例外なくコンピュータが組み込まれ、使いこなすには高いレベルのハード・ソフトウェア技術が要求される。更に、GPSに代表されるように、ナノ秒という極めて微小な時間管理が様々な機器に要求されるようになった。このような機器を多数含むシステム全体の完全な検証は、SEとSWEとの協調作業によってのみ実現可能となる。

その意味において、この度SECジャーナルでシステムズエンジニアリング特集を組まれた意義は大きく、広い分野の読者からのフィードバックを期待する次第である。

IoT時代における システムズエンジニアリングの重要性

ドイツ フラウンホーファー研究機構
実験的ソフトウェア工学研究所（IESE）
イェンス・ハイドリッヒ博士
マーティン・ベッカー博士

SEC所長
松本 隆明

IoT時代を迎え、多様・複雑に連動するシステムの設計に、システムズエンジニアリングが有効であると言われるようになってきた。IPA/SECでは、フラウンホーファー研究機構／IESEと協業して、欧州企業におけるシステムズエンジニアリングの先進適用事例・課題克服のベストプラクティスの調査・分析を実施した。本対談では、その調査・分析から得られた成功事例・教訓事例などを踏まえて両博士からシステムズエンジニアリングの有用性・有効性、実践に向けたアドバイス、更にはインダストリ4.0への取り組みなど幅広く話を伺った。

イェンス・ハイドリッヒ博士

フラウンホーファー IESE、プロセス工学部門長。2005年以来、日本のIPAとのコラボレーション・プロジェクトに関与。大学で教鞭をとりつつ産業界の分野で研究に従事。複数の国際会議のプログラム委員会のメンバーでありGI（ドイツコンピュータ科学協会）のソフトウェア測定グループのステアリング委員会のメンバー。「GQM+Management Strategy： 残念なシステムの無くし方」の共同著者の一人。

マーティン・ベッカー博士

フラウンホーファー IESE、組込みシステム開発部長。大学で教鞭をとりつつ産業界の分野で研究に従事。複数の国際会議のプログラム委員をつとめ、ACMのメンバー。システムズエンジニアリング分野の国際的な出版物も複数共同執筆。主な専門知識は、ソフトウェアプロダクトライン開発と様々な管理手法。

スマートエコシステム化に向けた取り組み

松本　IoT時代におけるシステムズエンジニアリングの重要性について、お話をしていきたいと思います。まず最近のIESEの主な取り組みについて、簡単に紹介していただけますか。

ハイドリッヒ　最近は、あらゆるドメインにおいてスマートエコシステムと呼ばれる統合システムへ進む傾向があります。これを達成するためにモノリシックな単一システムから、オープンで相互接続や拡張可能なサービス指向型のソフトウェアエコシステムへというパラダイムの変化が起ころうとしていて、IESEでもこれを見据えたシステム・インテグレーションの研究に取り組んでいます。

企業は、新たなビジネスチャンス、ビジネスモデルを求めています。その例が、スポーツ用品のアディダス社です。同社がランタスティックという企業を買収しました。ランタスティック社はスマートフォン向けのフィットネス用アプリケーションの会社です。

このように、以前はハードウェアを中心に事業を行ってきた企業が、全く新しい製品を出し、全く新しいビジネスモデルを使う形になっており、

その手段となっているのが、デジタル化でありIoTなのです。デジタル化、IoTによって、大きなチャンスが生まれてきています。

その中で、どういう状況が出てきているかと言えば、これまでは閉鎖型、一枚岩のシステムであったものが、複数のシステムから構成されるシステム、すなわちsystem of systemsという統合型のものに移行してきているということです。それをスマートエコシステムと呼んでいます。どういうものかと言えば、業務プロセスをコントロールする情報システム、それから技術的なプロセスをコントロールする組込みシステム、これを統合するというものです。

このスマートエコシステムが適用されるアプリケーションとして、いくつかの異なる業界、領域というものがあります。その一つの大きなものとして、いわゆるインダストリ4.0が挙げられます。そして、もう一つ重要なアプリケーションとして、とくに最近顕著に出てきているのが、スマートルーラルエリアです。“スマート田舎”とでも言えば良いでしょうか。ドイツでは、都市だけではなくカントリーサイドに住む人たちが多く存在します。例えばラインランドペレッテネイトという州がスポンサーとなった一つのプロジェクトで、デジタルビレッジというプロジェクトがあります。

このプロジェクトでは、二つのモデル地域を設け、田舎の中でのIoTまたはデジタル化の活用にどういう可能性があるのかを模索しています。

松本　スマートルーラルエリアの場合、メインはやはりスマートファーミング、つまり農業のスマート化が中心ですか。

ベッカー　スマートファーミングも、その一つではありますが、具体的なプロジェクトとして出ているのは、スマートモビリティというものです。例えば、田舎に住む人が商品の配送、いわゆるクラウド・デリバリーを行えるように参加していくというものです。

ハイドリッヒ　スマートヘルスもあります。とくに、田舎では遠隔医療というのが重要になってきますからね。

松本　日本でも、とくに地方では高齢化が進んでおり、非常に大きな問題になっています。

ベッカー　数年前にヨーロッパ全体でスマートシティの取り組みというのがありましたが、その中でも都市だけではなく、ルーラルエリアに対して何ができるのかを考えなければならない、ということが明確になってきました。と言うのも、ドイツにおいてもヨーロッパにおいても、大勢の人々が田舎に居住をしており、また、彼らはなるべく長く自宅で過ごしたいと思っています。スマートシティで使われてきた機能の、どれぐらいを、どのように田舎に適用できるのか、というのを考えていこうという流れになったわけです。

松本　先程のスマートファーミングに関して言えば、日本でも農業従事者の平均年齢が65歳を超えていて、しかもどんどん高くなりつつある。農作業をやることができなくなってきた。人がいなくなってきたということがあり、それが今、大変大きな問題になっています。それをいかにスマート化し、省力化して、あまり人手をかけずに済むようにするのか、ということを、日本でも議論するようになってきました。ドイツでは何か具体的な取り組みがありますか。

ハイドリッヒ　ドイツでは、農業全体が、かなり機械を重視した形で行われています。とくに、大きな農地がある場合には、ハイテクの機器を駆使しています。実は私どもは、大変戦略的な協力関係をジョンディア社と持っています。ジョンディア社は、米国の農機の会社ですが、ヨーロッパにリサーチセンターを設けており、スマートファーミングに関しての様々な研究を行っています。それによって農作業を、容易に行えるようにするという目的が一つ。もう一つは、若い人たちにとって農業をやっていくことの魅力を増していくということです。

松本　それは良い取り組みですね。

インダストリ4.0に必要な標準を提供していく

松本　先程のシステム・インテグレーションの話ですが、確かに今はIoTの時代になって様々なサービスがつながり、システムが融合化する時代になっています。そのときに、今まで比較的はっきりしていたハードウェアとソフトウェアの境界がだんだんあいまいになってきたような気がしています。

ハイドリッヒ　同感です。

松本　最近の傾向として、ソフトウェアを生業としていた企業がハードウェアに進出してくる。極端に言えば、グーグルが自動車を作る、というような時代になっているのではないかと思います。

ハイドリッヒ　その流れは、ハードウェアからソフトウェア、ソフトウェアからハードウェアへと、両方向で起きていると思います。例えば、元々はハードウェアを生業にしていた企業が、IoTやデ

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）。

ジタル化の機会を探るためにソフトウェアのほうに入ってくる、というようにです。しかし、IT企業が、既に確立されている市場に入っていこうとするほうがよりリスクが高いとも言えるでしょう。

ベッカー　最近のトレンドとしては、やはり様々なアプリケーションの領域において製品の革新化を図っていく際に、そのイノベーションがソフトウェアで行われていくという状況が増えてきていると思います。従って、これまでの典型的なハードウェアの企業も、そういった製品イノベーションをどう提供していくのか、それをソフトウェアでどう行っていくのか、ということに関しての投資をしていかなければならなくなっていますし、またソフトウェアの企業も、ハードウェアの能力を身に付けていくことが必要になっています。

松本　システム・インテグレーション以外に、今IESEで取り組まれているものはありますか。

ハイドリッヒ　色々あります。一つがBaSys4.0（Basic System Industry 4.0）というプロジェクトです。これはいわゆるインダストリ4.0の基本プラットホームになる部分のアーキテクチャにかかわるものです。また、プロオプトというプロジェクトがあります。これは、スマートプロダクション、つまりスマート化された生産の中で、ビッグデータをどう活用していくのか、という取り組みです。更にIUNOというプロジェクトがあります。これはインダストリ4.0向けのITセキュリティに関するものです。

インダストリ4.0に関しては様々なアプリケーション領域があり、そこに向けて様々な取り組みを行っています。

松本　インダストリ4.0に関しては、実際にプラットホームができて、色々なプロバイダがその上で製品を開発できるような状況になってきたのでしょうか。

ハイドリッヒ　そうであれば非常に良いと思いますが、プラットホームのプロジェクトは、今年半ばに始まったばかりです。プロジェクト自体の期間は３年間です。

松本　共通プラットホームということは、インダストリ4.0に準拠した製品を作る人は、共通のAPIのようなものを介してプラットホームと通信し合う、という形になるのですか。

ハイドリッヒ　大体はそういうことですが、BaSys4.0というプロジェクトでは、インダストリ4.0に必要なスペック、標準を提供していくというような形になります。例えば、物理的な機械を抽象化しデジタルに表現していくということが必要になります。それを、デジタル・ツインとかデジタル・シェルと呼んでいます。

ベッカー　その最初のステップとして、インダストリ4.0のプラットホームに対しての参照アーキテクチャを開発する。その参照アーキテクチャが、RAMI4.0（The Reference Architectural Model Industrie 4.0）と呼ばれています。BaSysのプロジェクトでは、この参照アーキテクチャに基づいて、インダストリ4.0向けの具体的なOSを開発し、そのOSを様々なアプリケーションの領域で共有して使っていく、という形になります。

松本　そのOS自身は、インダストリ4.0に準拠するプロダクトに埋め込まれる形になるのですか。

ベッカー　おっしゃる通りです。このOSは、インダストリ4.0準拠の製品の中に組み込まれる、と言いますか、このOS上に製品が構築される形になります。また、インダストリ4.0に準拠したアプリケーションや製品は、これまでの製品にない、新たなプロパティ、特徴が必要だということも認識されています。

例えばインダストリ4.0の製品やアプリケーションに関しては、いわゆるコンテキスト、周辺の環境というものを認識する能力が必要だ、と言われています。また、自身で系統立てる、整理していく、適応していくという能力も必要だと考えられていますし、きちんとセキュリティと安全を確保したものにしていくことも必要だと言われています。こうした、インダストリ4.0に特有なものとして必要とされる特徴、それをきちんと組み込んでいくということです。

松本　その場合、既存の製品を大幅に作り変えなければいけないという意味で、かなりコストがかかるような気がしますが。

ハイドリッヒ　今のものをすべて捨てるという考え方ではありません。既存のマシーンに関しても、スマート化を図っていきます。そして、既存のレガシーと言われるようなハードウェアも、システムの中に統合していく方法を見つけていかなければならないと考えています。それができなければ、インダストリ4.0のビジョンは実現できないでしょう。

ベッカー　既にインダストリ4.0向けのプラットホームやOSも、一般の企業から出ているものがあります。例えばシーメンスは、インダストリ4.0向けの製品をもう既に出しています。それらは他の企業が使うことも可能になっています。考え方は、まだインダストリ4.0に参入していない企業が参入しやすくするというもので、例えば、中小企業もインダストリ4.0に入っていきやすくなるということです。彼らの製品を適用して、インダストリ4.0対応にすることができるような技術を提供するということです。

まずアーキテクチャを きちんと決めることが重要に

松本　そのあたりは非常に重要なパラダイム・シフトになるのではないかと思います。というのは、今のIoTというのは、いわば無原則に勝手にものがつながり始めてしまっている。

きちんとしたアーキテクチャを決めていかないと全体としての安全性やセキュリティが担保されなくなってしまう、ということが危惧されています。アーキテクチャをき

ちんと決め、しかも既存のものが参入しやすくしていきます。そういう仕組みが、非常に重要になってくると思います。

ハイドリッヒ　現在、ヨーロッパでは大規模なプロジェクトが行われています。EMCスクエア、EMC二乗と書くプロジェクトです。これは組込み型のマルチ・コアのシステムで、アーキテクチャに加えて、安全のためのエンジニアリングとセキュリティのためのエンジニアリングを含んだメカニズムになります。

私たちは、このプロジェクトにおいても、とくに安全、セキュリティを統合した形でエンジニアリングを行っていくという作業にかかわり、その部分をリードしています。

ベッカー　私どもにとって課題となるのは、単に安全とセキュリティの組み合わせにならない、というだけではなくて、やはりシステム自体がランタイムで変わっていく、複数のものがランタイムに統合されていく、一緒になっていくということが起きますので、そこでの品質保証やアシュアランスの考え方を変えていくということも必要になります。

ランタイムにおいて、フィールドでコンポーネントが組み合わされていくということになれば、エンジニアリングにおいても、全く新しい仕組みが必要になるということです。

松本　それは非常に難しいところですね。動的に、ダイナミックにアダプテーションしていくというのは、具体的にどういう仕組みで実現しようとされているのでしょうか。

ベッカー　そのための専用のモデルがあります。そこで使われているのが、コンサートと呼ばれている技術で、これは、システムの挙動、システムのプロパティをモデル化するというものです。ランタイム中にシステム同士が組み合わさっていくというような場合に、全体のシステムのクオリティをシステム自身がチェックします。現在でもシステム開発中に使われるモデルというのがありますが、それをランタイムでも動かすということです。

システムがコンビネーションを変えていくときに、このモデルが必ず使われ、システム全体のプロパティに対して検証をするというような形を取れるようにします。

松本　そうすると鍵になってくるのは、システムをいかにうまくモデル化するかということでしょうか。

ハイドリッヒ　おっしゃる通りです。そこで全体的な大きな課題であるモデルベースのシステムズエンジニアリングとか、モデルフロー型のシステムズエンジニアリングの話に戻ることになります。モデルベースのシステムズエンジニアリングというのは、システム単体だけをモデル化すれば良いか、というとそうではありません。システムに加えて、そのシステムが置かれているコンテキスト、周りの状態―その中には人間も含みますが―その全体をモデル化していくことが必要となります。

松本　システムズエンジニアリングが、最近色々な場面で注目され始めた理由は、やはりIoTによってシステムを取り巻く環境がダイナミックに変化しているためだと言えるでしょうか。

今回、私たちの主催するセミナーでお話しいただいている内容※1の中で、ドイツにおけるシステムズエンジニアリングの適用状況のご紹介をいただいていますが、端的に言って、ドイツではシステムズエンジニアリングの活用は、かなり進んでいるのですか。

ハイドリッヒ　多くの企業が、自分たちの目的に対してシステムズエンジニアリングがどう適用できるのかを検討しているとは言えます。私どもの調査の結果でも、大多数の企業が、将来的な課題に対して対処していくために、大変重要なトピックとして捉えていることが明らかになっています。

とくに中小企業において顕著です。彼らにとっては今後の大きな変化を意味することであり、システムズエンジニアリングに関しての能力を持つためには、そのリソースが限られているからです。

システムズエンジニアリングが求める組織構造の変革

松本　システムズエンジニアリングというのは、非常に幅広い概念で色々な領域にまたがっています。それを取り入れようとすると、相当なスキルや能力が必要になってきます。企業にとっては、負担が大きいのではないかと思います。

ハイドリッヒ　システムズエンジニアリングの能力を構築していくための努力はされつつありますが、おっしゃる通り、システムズエンジニアリングは、かなり幅広い意味を持つ概念です。一人の人間がすべての領域の知識を持つということを、考えるべきではなく、会社の中の様々な領域の仕事をしている人間が、お互いにインターフェースを持ち、お互いに議論をし、コラボレーションするというアプローチが必要だと思います。ハードウェアの製品をソフトウェアと統合していくということになれば、やはりソフトウェアのエンジニアはハードウェアの部分に関して理解が必要になり、反対に、ハードウェアのエンジニアたちがソフトウェアの部分を理解することも必要になります。

私たちの時代、それは組織にとって大きな変革を意味します。様々な領域が相互に作用をしなければなりません。従来型の企業は、それぞれのプロセスが独立をし、いわゆる縦割りであったという状態が多かったのですが、それでは対応できません。

松本　単に技術的な問題だけでなく、組織的な変革も必要になってくるというのが、システムズエンジニアリングの考え方につながっていくということですね。

ハイドリッヒ　その通りです。調査でも、組織的な変更管理が一番難しいということが明らかになりました。所長

がおっしゃった通り、競争力を持つために―そのためには効率的なプロセスが必要になりますが―適切な組織構造が必要になります。単に技術的な方法論で済む話ではないのです。例えばボッシュ社は積極的にシステムズエンジニアリングを活用していますが、同社では、組織変革も積極的に行っています。ただ、多くの企業がその変革を始めるには、ある程度の時間がかかるでしょう。

課題となるのは、適切な人材を見つけていくということです。それがなかなか難しいです。ドイツにおいては、ソフトウェア・エンジニア、またはソフトウェア・エンジニアリングの能力を持つ人の労働市場がかなり逼迫(ひっぱく)し、適材を確保するのは難しくなっています。

松本　それは非常に重要なポイントですね。まだ日本では、システムズエンジニアリングというと、やはり技術的な側面の議論が多くて、日本の企業に「システムズエンジニアリングを導入していますか」と聞いても、その答えの中には「はい、導入しています。私たちはSysMLで書いていますから」というものがあるのです。

ベッカー　やはりシステムレベルで重要になってくるのは、システムズエンジニアリングをきちんと、その会社のビジネス上の目標とつなげて考えていくということです。また、新たな製品の機能であるというような明確な牽引要因となるものが必要です。それによって、部門を超え領域を超えたコラボレーションが必要だ、となっていけば、そこからシステムをどう構造立てていけば良いのか、何が必要なのか、ということを決めていくことができるようになると思います。

そのためには、まず、システムズエンジニアリングを行っていく、部門をまたがるチームを編成していくことになるかと思います。そして、時間が経つにつれて、必要とされるシステムズエンジニアリングの能力に基づいた組織編成に変えていく、というリオーガニゼーションを行っていくことだと思います。

松本　そもそもシステムズエンジニアリングという言い方が、あまり良くないのかもしれませんね。エンジニアリングというと、日本ではどうしても「工学」になってしまいます。システムズ・シンキング、システムズ・オリエンテッド・シンキングと捉えたほうが良いのかもしれない。

ベッカー　既に、システムズ・シンキングという考え方はあります。エンジニアたちはきちんとシステム全体のことを考えましょう、と。また、製品のスケールを考えましょう、という意味で、このシステム思考という言葉は使われています。そして、システムズエンジニアリングの中の一部に、このシステム思考というのがある、と考えられています。また、それと共にエンジニアリングには、システム・アーキテクチャ、システム・アーキテクト、またその役柄というものが必要になると言えると思います。

ハイドリッヒ　補足すれば、かなり早い段階から考え始めていくことが必要です。先程、ビジネスモデルの話も出ましたが、やはりビジネスモデルからかかわってくるということです。どういうチャンスがあるのか、ということを見出して、そのチャンスに対して必要な能力は何であるのか、と考えていく。そして、システムは、どういうアーキテクチャを持つことが必要なのか、と考えていく。そこがシステム思考ということだと思います。システム思考、システムシンキングにはベースの部分が必要であり、どういうオポチュニティーに対してやっていくのか、どういう方向性に向かっていくのかということが、基礎、根本になければならないと思います。

ベッカー　今、彼が言ったことは、本当にその通りだと思いますけれど、企業としては社内の組織のことも、きちんと考えていかなければなりません。その中では、ビジネスをやるマーケティングとエンジニアリングのチームが、お互いに話し合い、密な関係を持って協力作業をしていくことが必要です。ビジネス上のチャンスをきちんと手中にし、それを活用しながら、そこではベースとなる技術的なオポチュニティーが下支えしていなければならない、ということです。しかし、その両方を一人で考えるような役割は、現在の典型的な組織の中にはない状態ですから、いわゆるインターディシプリナリー、部門の垣根を超えた協力作業が必要になる、ということです。それをすることでチャンスを活かすことが可能になります。

ハイドリッヒ　やはり企業が考えなければならないのは、これまでの確立されてしまっている、いわゆるサイロ型、縦割り型の組織の垣根を取り払って、お互いにきちんと議論ができるようなものにしていく、ということでしょう。

イノベーションを行うためには、どこかのグループが単独で実現できるかと言えば、そうではありません。イノベーションを実現していくためには、技術的な専門知識も必要ですが、ビジネスの専門的な知識、つまり、どの方向に向かっていくのか、ということを見定めるための知識も必要になります。

松本　正に最近言われているように、IoT時代になってきたことによって、今までのクローズドなイノベーションでは、もう競争力がなくなってくる。オープンなイノベーションにしていかないといけない、ということですね。

ハイドリッヒ　その通りです。

ベッカー　そして、オープンなイノベーションというのも、大きな企業の中の、社内でやれるようなイノベーションだけではなく、組織の垣根を超えるようなものになってきている、と言えると思います。

実際に、私たちが提唱するスマートエコシステムという考え方を検討し始める企業の数も増えています。そこでは企業がプラットホームを提供し、外部の他社が、それに対して追加的にハードウェアのデバイスを提供したり、ソフトウェアのアプリケーションを提供したりするというような仕組みになります。

松本　お話を伺っていると、システムズエンジニアリングというものには、かなり広い意味があるという気がします。今までシステムズエンジニアリングというと、どちらかと言うと、How to makeが中心かと思っていたのですが、What to makeのところも領域の中に入ってくる、というイメージを持ちました。
ハイドリッヒ　その通りです。

システムズエンジニアリングにおける人の問題をどう捉えるのか

松本　システムズエンジニアリングが、それだけ幅広くなってくると、単にハードウェアとソフトウェアが対象になるだけではなく、人間、ヒューマン・ファクタのところも大きな要素になってくると思いますが、いかがでしょうか。
ハイドリッヒ　人がシステムズエンジニアリングの考え方の中心にくると思います。と言うのも、システムが提供するサービスを使うのは、最終的には人であり、そのシステムによってサポートされるのは人だからです。
松本　しかし、人間というのはモデル化しづらいというか、ほとんどできないのではないでしょうか。人間の要素をどうモデル化するかというのは、すごく難しいと思います。
ハイドリッヒ　そこが大きな課題です。プライバシーにもかかわってきます。システムが大量のデータを収集する中で、人についてのデータも多く収集する。ドイツではとくに注目されているトピックとなっており、集められたデータが誰に属するのか、誰が所有するデータになるのか、また、そのデータに対して何ができるのか、どこまでできるのか、ということが議論になっています。人のモデル化をするときには、その人が何をやっているのか、どういう状況にあって、何を行っているのか、ということを分析することが必要になります。それは、やはりプライバシーと大きくかかわるわけです。
松本　プライバシーの問題について、具体的にこう考えていこうという方向性はあるのでしょうか。
ハイドリッヒ　考え方としてあるのは、一人の人が、自分に関するデータに対して、より良いコントロールができるようにしていこう、というものです。提供されたデータに対して、何ができて何が許可されないのか、というのがきちんとコントロールできる環境にしていこうということですね。そのために、二つのソリューション・アプローチが考えられています。一つ目が技術的なソリューションで、そのデータに対して何ができるのか、何が許可されないのか、というポリシーを作り、そのポリシーのモデル化をしていく、ということです。そのために、IESEではフレームワークを作るというプロジェクトになっています。
　二つ目のアプローチは、技術的なものではありませんが、より人々の意識を上げていこうというものです。どんなデータを提供しているのか、そして、そのデータに対して何ができるのかということを、より慎重に考えるような意識向上です。
ベッカー　業界をまたがってトレンドとして出てきているのは、これまでのように、データをプッシュ型でクラウドに送り込んで、中央一元化して処理をするのではなく、ローカルなマシーン上に持たせ、処理をしていくというものです。つまり、組込みシステムのほうにその機能を渡していく、という考え方です。それをすることによって、機密性があり価値のあるデータは、全員が共有するのではなく、ローカルなマシーンに持たせていくという考え方です。
ハイドリッヒ　それが、ビッグデータの原則にも合うと思います。機能をデータのほうに動かしていくことで、データを機能のほうに動かすのではない、という考え方です。
ベッカー　それが、新たな市場機会を開いていくことにもつながっていくでしょう。例えば、スマートな組込みシステムを構築できる企業というのが、そのスマートな組込みシステムの中で、ユーザのニーズを理解し、また、振る舞いや挙動を学習することができる、ということになれば、そのシステムはより市場で成功するチャンスが高いということになります。インテリジェンスはクラウドに任せて、組込みシステムのほうには何も持っていない、というものと比べると、大きなチャンスになると思います。
松本　なるほど。例えば、これから色々なウエアラブル・デバイスができたときに、人間の身体的な情報、脈拍や血圧なども取れるようになりますが、いずれは自分にとって、これはクラウドに上げてヘルスケアで管理して貰ったほうが良いというデータと、これは個人情報だから自分のデバイスに残しておこう、というものと、ユーザがそれを管理できるような仕組みが必要になるかもしれないですね。

要件だけでなく、システムの挙動、振る舞いもモデル化し共有していく

松本　システムズエンジニアリングについて、先程のようにダイナミックに自分を適合していくようなシステムという発想になってくると、開発そのものも、今までのようなウォーターフォール型の開発ではなくて、アジャイル的な開発が主流になってくるように思うのですが、それについてはどう考えていますか。
ハイドリッヒ　イノベーションにかけられる時間が短くなり、いわゆるイノベーション・サイクルが短縮化されてきてしまうと、どんどん変わっていく顧客の需要やニーズに対応していくために、企業には俊敏性が求められると思います。そのために、ドイツの多くの企業は反復性の高いプロセスに移行しようとしています。どんどん変わっていく顧客デマンドに対応していくためです。そのため、アジャイル開発の手法はドイツではかなり多く使

われています。とくに情報システムと言われる部分に関しては多用されています。

少し前に私どもは、企業がどういうアジャイルの手法を活用しているのか、という調査をしました。その結果として出てきたのが、スクラムとかエクストリームのプログラミングを、教科書通りに使うのではなく自分たちのニーズに合わせて、適応させて使っているということでした。

例えばスクラムでは、セキュリティとか機能安全とか、それからシステム・プロパティに関してのところまで記述されていません。アジャイル開発をしようと言っても、それをそのまま使うのではなく、その側面を取り入れて、プロセスの中でより成熟化させて活用していく、ということだと思います。まず目標を見据え、目標に対して適切なプロセスは何であるのか、そういう考え方をしていくことが必要です。

松本　日本では、なかなかアジャイルの開発が広まりません。私はその大きな理由に、日本の産業構造があると思っています。ITの場合ですが、ユーザ企業とベンダ企業がある時に、開発はほとんどベンダ側が行うのです。アジャイル的に、先程のリクワイヤメント・エンジニアリングで考えるときには、やっぱりユーザ側とベンダ側が一体になってやらないといけないのですが、それがなかなかできません。ドイツの場合その点はどうでしょうか。ユーザ企業の中に開発部隊を持っているケースが多いのでしょうか。

ハイドリッヒ　それは、その企業により異なるという状態です。外部で開発されたソフトウェアを、供給を受けて使う企業も多く、ソフトウェア開発専業の企業もたくさんあります。ただドイツでは、IP※2やUSP※3が何なのか、ということによって変わってきます。IP、USP、その企業が持つ資材などがソフトウェアである、という場合には、社内的な開発能力を持つ場合が多いと思いますが、そうではない場合にはアウトソースするという場合が多いと思います。

松本　アウトソースする場合もシステムズエンジニアリングでそれをやろうとすると、契約などいろいろな問題が出てくると思いますが、そこをうまく解決する方法は、あるのでしょうか。

ハイドリッヒ　大変難しい問題だと思いますが、どういう開発プロセスを取るのか、ということによっても変わってくるでしょう。アジャイル手法であった場合には、ソフトウェアを供給する側と、そのソフトウェアの開発の注文を出した側との間に、信頼関係が必要になります。というのも、最初から大きな要件を規定した文書を持つのではなく、かなりの反復が行われていきます。その反復の作業の量によって支払いをしていく、ということになるからです。これまでと違う協力体制が必要だ、ということになります。

ベッカー　以前であれば、企業は要件、情報を共有するだけだったのですが、今出てきているのは、とくにモデルベースのシステムズエンジニアリングというやり方をしていく際には、OEMのメーカーが、システムモデルのパーツも共有していく。それをサプライヤー側に渡していく。そして、両方がコラボレーションをやっていく、ということです。文書ベースで要件の受け渡しをする、ということではなくて、それに追加してモデルもサプライヤーのほうに提供していきます。それによってより開発がやりやすくなりますし、両方が何が開発されるのかということに対して、共通の理解を持つ助けにもなります。

しかも、構造型のモデルを渡すだけではないんです。構造に関してのモデルというと、システムのエレメントやシステムのインターフェースだけがモデルに入っていると思われるのですが、そうではなく、システムの挙動、振る舞い、これもモデル化し、それを渡していく、という形になります。それをつかって、いわゆるバーチャル・エンジニアリングを行うことが可能になります。最初は、セントラル・システムに関してのシミュレーション、そして、いわゆるヘテロジニアスなシミュレーションを実行できるようにしていく、ということですね。OEM側のパーツのシミュレーションと、サプライヤー側のパーツのシミュレーション、これを混在させたヘテロジニアスなシミュレーションを実行していく、ということです。

松本　それは非常に興味深いお話ですね。今後の重要な方法論になってくるような気がします。

ベッカー　とくに自動車産業においては、既にかなり多用されています。

まずどういう付加価値を求めるのか、という方向性を定める

松本　今回、様々な調査をドイツで進められたわけですが、その結果から日本のインダストリに対しての提案あるいは、助言をいただけますか。

ハイドリッヒ　重要なのは、IoTやデジタル化の中で、まず何がオポチュニティー、チャンスであるのか、ということを考えるということです。その機会を見据えた上で、それに即した戦略作りをし、それに基づいて必要な組織構造の変革を行っていく。

そして、戦略がきちんと確立できた上に、ではその戦略のために、どういう能力が必要なのか、ということを考えることが必要です。システムズエンジニアリングの能力、そしてまた、とくにソフトウェア・エンジニアリングの能力として何が必要なのか。

更に、エンジニアリングのプロセスを考えていくこと

脚注

※1　イェンス・ハイドリッヒ博士、マーティン・ベッカー博士のSEC特別セミナー講演模様、講演資料はこちらから
http://sec.ipa.go.jp/seminar/20161024.html

※2　IP：Intellectual property 知的財産権

※3　USP：Unique Selling Proposition 独自の売り

が必要になります。どういう意味かと言うと、いわゆるインターディシプリナリー、色々な領域の人が、共同開発作業をしていくことが必要だ、ということです。様々なステークホルダの人たちを、開発のプロセスの中にどう統合していくのか。それによって、より効率良く、効果的にクオリティの高い製品を提供していくと考える必要があります。

調査から、私たちは、とくに確立しなければならない領域が３つあると考えました。

一つ目がモデルベースの開発、二つ目が要求工学、そして三つ目がシステムの検証及び妥当性確認、V&Vの領域です。これが出発点になっていくと思います。そして、先程お話に出た更に模索していくべき領域としては、バーチャル・エンジニアリングにメリットがあると思います。更にツールチェーンの統合です。

ベッカー　どこから始めていくのか、という場合には、ビジネスの視点から何が付加価値として必要なのか、ということを理解することが重要だと思います。そして、その与えるべき付加価値に対して、会社としてどういう追加的な能力が必要なのか、その能力をどう追加していくことが可能なのか、ということを見極めていく。単にツールを買って使えば良いというものではないと思います。

ハイドリッヒ　最初の部分が一番大変なところだと思います。何をしたいのか、ということが分からなければ、それはどのように実現できるのかということは全く分かりません。

ベッカー　その助けになると思うのが、ドイツの組込みシステム、サイバーフィジカル・システム、そしてインダストリ4.0に関してのロードマップの文書です。その中には、多くの新たなアプリケーション・シナリオが含まれています。それが、何が付加価値として、また、新しいアプリケーション・シナリオとして、自分の会社に考えていけば良いのかを模索する際に、インスピレーションを与えてくれる良い情報源になると思います。そこからシステムズエンジニアリングに必要な能力が何なのか、ということを導き出していくことができるでしょう。

松本　システムズエンジニアリングが分かる人間を、どうやって育てていくか、ということでは、何か助言がありますか。

ハイドリッヒ　二つの道を区別して考えていくことが必要だと思います。

まず一つ目は、早い段階のもので、大学レベルでの人の教育です。システムズエンジニアリングを学習していく動機を与えていく。それに対して相応するようなコースやカリキュラムを提供していく、ということです。

二つ目がいわゆるOJTで、仕事をやりながらの教育という部分です。社内で行うトレーニングも、外部で提供されるトレーニングもあるかと思います。

また、システムズエンジニアリングに関して遠隔学習のプログラムを行い、それに参加させていくということも、能力を付けていく一つのやり方だと思います。

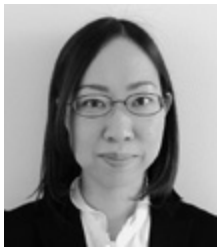
そして、もう一つ、教育・啓発といったときに、社内で行うものに限らずに、他社との経験の共有、または交換ということも考えていくべきでしょう。システムズエンジニアリングに関してのコミュニティに参加をし、積極的なメンバーとなってコミュニティから知識を獲得していくというのも、組織にとって良い戦略になると思います。

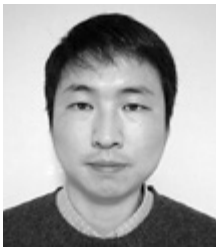
松本　ドイツにはそういったコミュニティがかなりあるのですか？日本にはINCOSEという国際的な団体の日本版であるJCOSEがあるのですが、メンバーがそれ程多くなく、有効なコミュニケーションを図りづらいという問題があるようです。

ハイドリッヒ　ドイツでもINCOSEのドイツ版であるGFSEというシステムズエンジニアリング協会のようなものがあります。それだけではなく、システム要件、システム・アーキテクチャ、システムの実装、またテスト、検証、妥当性確認というようなトピックで、様々なカンファレンスも数多く行われているので、そういうところに参加をする、というのも一つの方法だと思います。

松本　システムズエンジニアリングはこうだと教え込んでも、なかなか難しいという気がします。また、概念が非常に幅広いので、お話のようなベスト・プラクティスやセミナーなどを通じて、みんなで共有してとにかく自ら実践してみるということが重要なのでしょうね。今日は非常に有意義なお話をお聞かせいただきありがとうございました。

自動運転車を取り巻く System of Systemsの安全性要求の妥当性確認と検証






木下 聡子※　西村 秀和※　ユン ソンギル※　北村 憲康※

より安全な交通環境の実現を目指し，2020年を目処に自動運転車を社会へ導入するための取り組みが行われている．しかしながら，自動運転車自体で安全性を実現することは困難であり，周辺システムと協働することにより全体の安全性を確保することが重要となる．本論文では，自動運転車とその周辺システムを含む交通システム全体をSystem of Systems（SoS）と見なし，そのシステムモデルに基づき導出された安全性要求をもとにGSN（Goal Structuring Notation）によりアシュアランスケース記述を行うことでその妥当性確認を行うとともに，モデル検査を用いて検証を行う．モデル検査では，SoSの構成要素である自動運転システムとドライバ間の相互作用に関する安全性要求を検証するためCSP（Communicating Sequential Processes）モデルを用いる．

Validation and Verification of Safety Requirements for the System of Systems involving Automated Vehicles

Satoko Kinoshita, Hidekazu Nishimura, Sunkil Yun, and Noriyasu Kitamura
Graduate School of System Design and Management, Keio University

There are efforts to introduce automated vehicles for the realization of safer traffic environments. However, this cannot be achieved just by introducing automated vehicles into the environments. Accordingly, it is necessary to ensure the safety of a traffic environment considering a traffic system including automated vehicles and the surrounding systems. This paper treats those systems as a system of systems (SoS) and introduces the system models. First, safety requirements derived from the system models are validated with descriptions of assurance cases. Second, by using model checking for a CSP (Communicating Sequential Processes) model, we verified that the system model satisfies the safety requirements relating to interactions between the automated driving system and the driver, which are constituent systems of the SoS.

1　はじめに

2020年に自動運転車を社会に導入することを目指し，自動車会社やIT関連企業が自動運転車の開発を進めている．内閣府が進める戦略的イノベーション創造プログラムでは，交通事故の死者数の低減及び交通渋滞を緩和することを目指し，自動走行システムに関する技術開発や国際連携に向けた準備が進んでいる［内閣府2016］．自動

※慶應義塾大学　システムデザイン・マネジメント研究科

運転車を含む交通システムは，多様な利害関係者を巻き込むこととなるため，自動運転車の社会受容性を検討する必要があると考えられる．

［BCG 2015］によると，自動運転システムへの最大の懸念は安全性にある．社会受容性にとって安全性は極めて重要な要素である．しかし，自動運転システムを搭載する自動運転車の安全性が独立して成り立つことはない．それは，情報システムや交通インフラなどと連携して動作することによって確保できるものである．すなわち，自動運転システムの安全性要求を明確にするためには，交通システムを構成するその他の独立したシステムと協働して動作をすることを前提に検討する必要がある．自動運転車を取り巻く情報システム，交通インフラシステム，歩行者，自動車や自転車などのモビリティ，道路や障害物などの物理環境，天候や高度などの自然環境はそれぞれ独立して動作している．更に，地理的に離れた場所に存在するため，このようなシステム全体はSystem of Systems（以下，SoSと略す）として定義できる［Jamshidi 2011］．［DeLaurentis 2005］は既に交通システム全体をSoSとみなすことを提案しており，自動運転車の安全性を検討する際にもSoSの問題として捉えることで，自動運転システムに関係するシステムやそれらの相互関係を明確に検討することが可能となる．

COMPASS（Comprehensive Modelling for Advanced Systems of Systems）［COMPASS］では，SoSに対してモデル検査を行い，構成システムの相互作用の結果として現れるSoSの創発的な振る舞いを検証する研究が行われた．そこでは，VDM（Vienna Development Method）とCommunicating Sequential Processes（CSP）に基づく独自の形式仕様記述言語（CML，COMPASS Modeling Language）を提案している．CSPは並行システムを形式的に記述し，検証するための理論である［Shneider1999］．

本論文では，自動運転車とそれを取り巻く交通システム全体をSoSとみなし，SoSアーキテクチャを構築する過程での安全性要求の妥当性確認と検証を行う．自動運転システムの自動化レベルとしては，SAE International（Social of Automotive Engineers International）の定義［SAE2014］でレベル３を仮定する．このレベルでは自動運転システムがドライバに運転権限の移譲を要求する際には，ドライバがそれに応答する必要がある．このため，本論文では自動運転車のドライバと自動運転システムとのコミュニケーションが安全性に大きく関与するものと考え，双方の状態変化及び相互作用を対象とする．まず，SysML（Systems Modeling Language）［Friedenthal2014］を用いたシステムモデル記述［西村2016］，［ユン2016］に基づき導出された安全性要求をもとにGSN（Goal Structuring Notation）によりアシュアランスケース記述を行うことでその妥当性確認を行う．そして，SoSアーキテクチャで定義されるドライバと自動運転システムの状態変化と相互作用をCSPモデルで表現し，妥当性確認をした安全性要求が満たされることをモデル検査により検証する．

2　SoSアーキテクチャの検証

2.1　自動運転車を取り巻くSoSアーキテクチャ

自動運転車が導入される交通環境には，渋滞情報や地図情報などを提供する情報システム（ICT System，Information Communication Technology System），信号機や道路標識などの交通インフラシステム（TIS，Transport Infrastructure System），ほかの車両（Surrounding Mobility），歩行者（Pedestrian），障害物などの物理環境（Physical Environment），天候などの自然環境（Natural Environment）など様々な独立したシステムが存在する．自動運転システムはこれらの独立したシステムと相互作用することで動作し，安全性を実現する必要がある．文献［西村2016］では自動運転システムとその周辺システム全体を自動運転車を取り巻くSoSとして見なし，全体として安全な交通システムを保障するためのアーキテクチャ定義の記述にシステムモデルを用いた．そこでは安全性を検討するために現状の交通事故をもとにユースケース分析を行い，SoSの構成システム間の関係性を構造と振る舞いのシステムモデルで示すと共に，安全性に関する機能要求，インターフェース要求を定義した．

2.2　安全性要求の妥当性確認と検証のアプローチ

SoSアーキテクチャを定義するため，導かれた安全性要求の妥当性確認と，安全性要求に基づくアーキテクチャの検証を行う．本節では，そのアプローチについて述べる．

自動運転車を取り巻くSoSの安全性を確保するというミッションに対する安全性要求の妥当性確認に際して，本論文では，アシュアランスケース［山本2014］を用いる．アシュアランスケースの記述により，システムモデルから得られる安全性要求をゴールとして設定し，議論をすることでそれらの妥当性を確認する．議論の論理展開を記述として残すことができるアシュアランスケースは，様々なステークホルダやシステム間の合意形成が必要なSoSの妥当性確認の方法として有効である．既に文献［Goodger2012］では，重要な情報基盤セキュリティの脆弱性を評価するため，評価再検討サイクルとアシュアランスケースの記述の併用を提案している．

本論文では，SoSアーキテクチャを構築する初期段階で，システムモデルをもとにアシュアランスケースの記述を行い，要求の妥当性確認を進める方法を提案している．また，システムモデルで明確にされる安全性要求に関係するSoSの構成システムを関連付けるため，［Saruwatari2014］で定義されているアクターというノードを導入し，SoSの構成システムをこのアクターとしてアシュアランスケースを記述する方法を提案する．SoSでは，各構成システムの相互作用の結果，安全性要求が達成されるか否かを検討することが重要である．構成システムをアクターとする記述方法を用いることにより，ゴールの達成に必要な構成システムを明確にできる．これにより，関連する各構成システムの運用者や利害関係者でゴールの達成に関する責任を議論することが可能となる．

次に，妥当性確認をした安全性要求に対し，SoSアーキテクチャがこれらの安全性要求を満たすことを検証する．SoS

の構成システムは独立に動作し，相互作用することによってSoS全体の機能が実現する．従って，SoSの構成システムの振る舞いと相互作用をCSPモデルで表現することによって，安全性要求を満たさない振る舞いのパターンを検出する．著者らは文献[Kinoshita2015]で，自動運転システムとドライバの相互作用を表すCSPモデルを提案した．本論文では，これにSoSアーキテクチャで追加された振る舞いを加え，かつ自動運転システムとドライバの状態が周辺システムから得られる情報によって変化することをCSPモデルに導入する．状態の変化が振る舞いと並行して起こることを表現することにより，より詳細な安全性要求を検証できる．

3 アシュアランスケースを用いた安全性要求の妥当性確認

3.1 SoSのアシュアランスケース記述

自動運転車を取り巻くSoSの相互接続の検討をもとに安全性に関する議論を行った結果を図1に示す．図1の最上位のゴールとして「自動運転システムがドライバと共に安全な運転を実現する」を設定する（ゴール：G_1）．このゴールを詳細に分析する戦略として，運転の基本動作である認知･判断･操作に基づいて論じるという戦略を設定した（戦略：S_1）．これは，自動車事故の多くはヒューマンエラーによるものであり，ヒューマンエラーが原因となる事故を減少させることで安全性を高めることができると判断したためである．

図1に示すアシュアランスケースで自動運転システムへの要求として，ドライバが通常の状態ではないことを検知することが求められ（ゴール：G_7），ドライバの状態を回復することが要求される（ゴール：G_8）．自動運転システムの働きかけによってもドライバの状態が回復しない場合は，自動運転システムが交通事故を起こす危険を回避する必要がある（ゴール：G_9）．これらの議論の結果より，自動運転車の導入により安全性を実現するためには，ドライバの状態を正確に把握する必要があり，必要に応じて交通事故を起こす危険を回避することが要求されることが明らかとなった．とくに，レベル3の自動運転システムの定義で最終的な責任を負うドライバを安全な状態に回復させるという機能はSoS全体の安全性を実現するために重要な要求である．

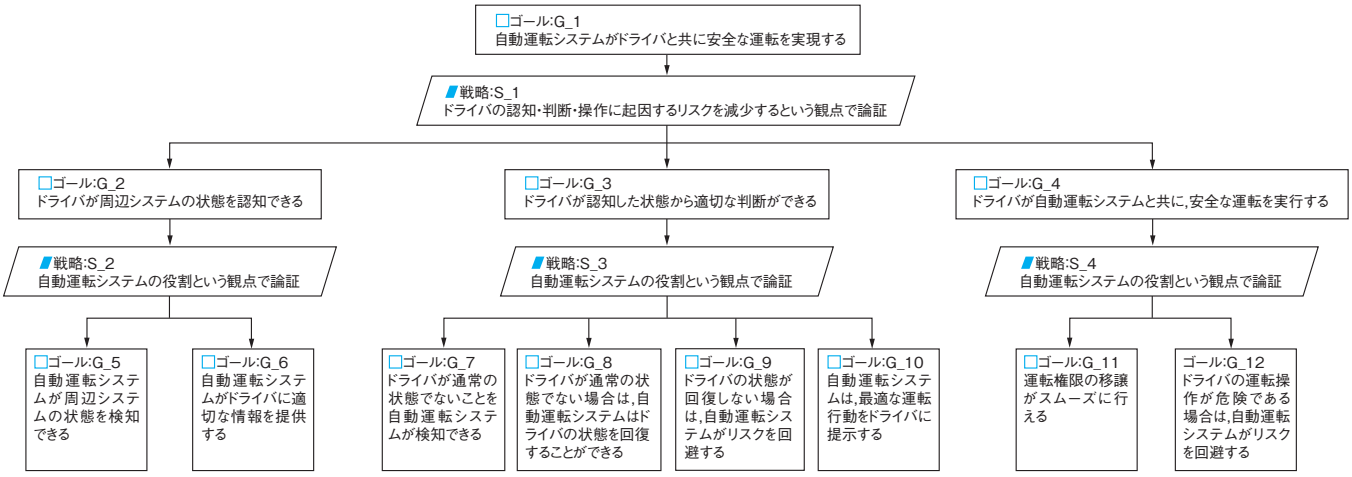


図1　自動運転車を取り巻くSoSの相互接続の検討をもとにしたアシュアランスケース

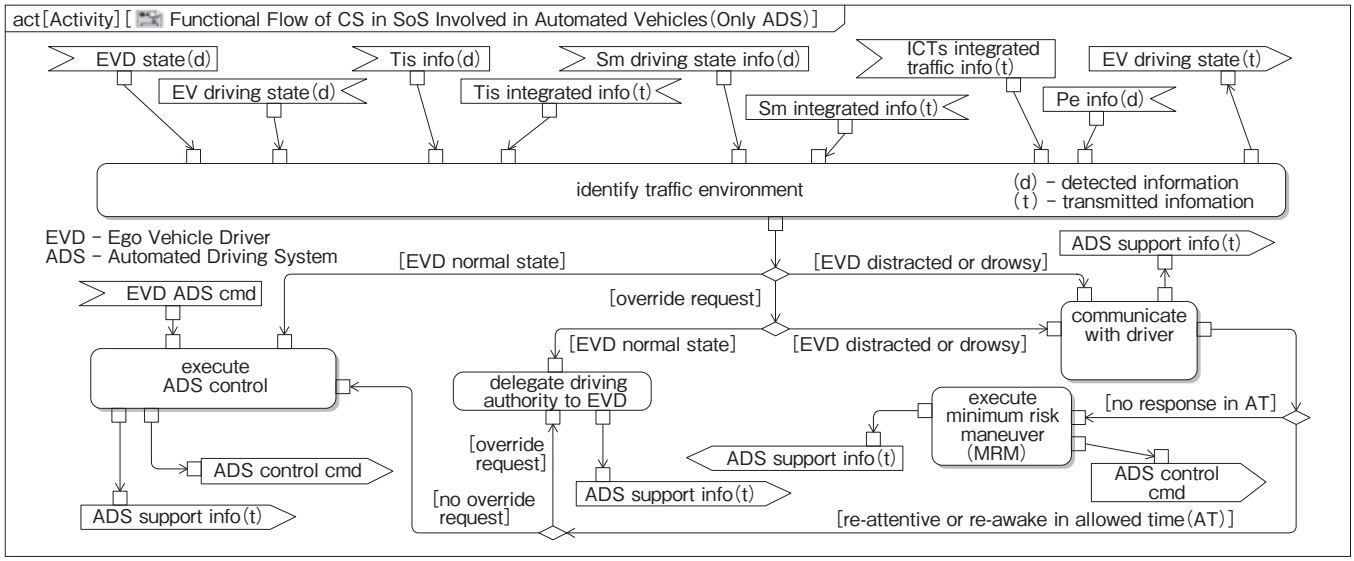


図2　自動運転システムの機能フローを示すアクティビティ図

3.2 アクターを用いた構成システム表現の導入

SoSの構成システム間の相互接続の検討ののち，ユースケース分析やインターフェースの検討の結果として，自動運転車を取り巻くSoSの機能の流れを表すアクティビティ図を記述した（図２）．このアクティビティ図より自動運転システム（Automated Driving System，ADS）に求められる安全性要求をアシュアランスケースにより記述する（図３）．このアシュアランスケースを用いて，図１に示す自動運転システムへの要求が図２より導かれる要求に引き継がれることを確認することによって，図２から導かれる安全性要求の妥当性を保証する．また，ゴールの達成に関連する構成システムをアクターのノードを用いて表現する．アクターは，図３のモジュールというフォルダの矩形で示されている．

まず，最上位のゴールとして「自動運転システムがドライバと共に安全な運転を実現する」というゴールを設定する（G_1）．次に，このゴールを詳細に議論するための戦略として，図２に示すアクティビティ図より，「自動運転システムのアクションから導かれる安全性要求を論じる」という戦略を設定する（S_1）．この戦略は，図２を参照するため，コンテキスト（C_1）として対象のアクティビティ図名を明記することで妥当性を確認するシステムモデルを関連付ける．

戦略（S_1）に基づき，ゴールG_2からG_6を設定する．アクターのノードに関連付けられている二重の矢印はdepend on（依存）の関係を示す．例えば，ゴールG_2「ADSは交通状況を特定できる」の達成に関して，自動運転システムはある構成システム群（CSs 01）に依存するということを示している．ここで，CSs 01は，自車両（EV），自車両のドライバ（EVD），交通インフラシステム，歩行者，周辺モビリティ，情報システムを表す．これらは，図２のアクティビティ図上で示されている情報の流れをもとにして設定する．このアクター間の依存関係は，自動運転システムにとって必要な情報がアクターCSs 01から得られる必要があることを示している．

今後，ADSによる交通状況の特定のサブ機能が詳細に定義されることにより，この構成システム群（CSs 01）は単体のアクターとして示される予定である．なお，このゴールG_2は図１のアシュアランスケースに示されるゴールG_5，G_6，G_7を実現する機能であることをコンテキストのノードで示している．一方，ゴールG_4に示されている「ADSはドライバに運転権限を移譲することができる」に関して，自動運転システムがドライバに権限を移譲するためには，自動運転システムの機能のみならず，ドライバが運転を代わることができる状態にあるのか，また適切な操作を判断できるかなど，ドライバの状態に依存することがアクターを関連付けることで示されている．このように，単体のシステムの機能だけではなく，独立して運用される他のシステムと相互に補完し合いながら動作することを検討できる．また，ゴールG_4は図１のゴールG_11を実現する機能であることをコンテキストで示している．

アクティビティ図に示される安全性要求は，図３のコンテキストのノードに示すように，全体の安全性要求として検討した結果を引き継いでいる．従って，ミッション達成のために実現すべき振る舞いがアーキテクチャの検討の中で妥当に検討できていると言える．また，アクターを用いることにより，ゴールの実現に必要な構成システムを確認することが可能となる．SoSは，構成システムが常に一定であるとは限らないという特徴を持つ．SoSの構成システム，すなわちアクターに変化がある場合にこのアシュアランスケースに立ち戻ることにより，再検討すべきゴールを見つけることができる．

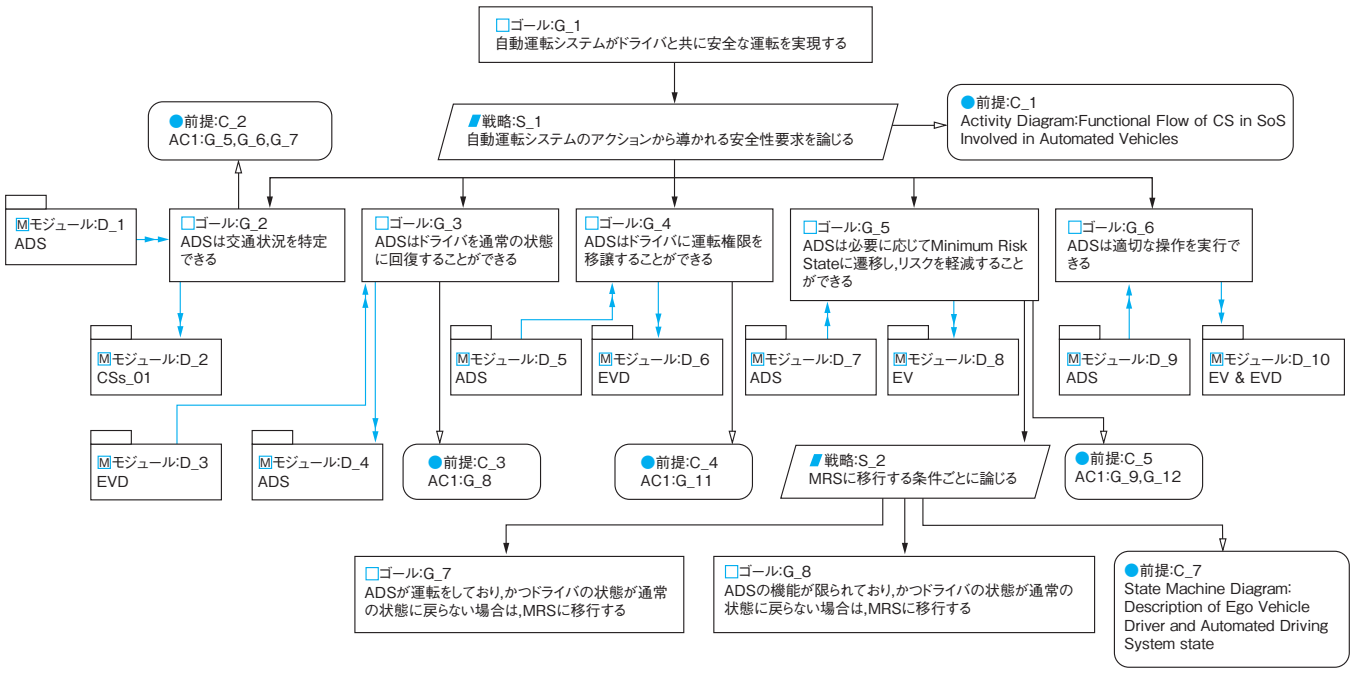


図３　アクターを構成システムとして記述したアシュアランスケース

4 CSPによるSoSアーキテクチャの検証

4.1 自動運転車を取り巻くSoSアーキテクチャのCSPモデル

3節で示した安全性要求の妥当性確認を受け，システムモデルの図で定義された振る舞いと状態の変化が安全性要求を満たすことを検証するためにCSPモデルを作成し，モデル検査を実施する．SoSの各構成システムの相互作用の結果，安全性要求が常に満たされることを検証する必要がある．CSPモデルは並行して動作するシステムの振る舞いを形式的にモデル化することができる．本論文では，自動運転車を取り巻くSoSの各構成要素を並行システムの要素とみなし，CSPモデルを構築する．CSPモデル作成のため，図2のアクティビティ図及び図4に示す状態機械図を用いる．ただし，図4の①の状態遷移はのちに示すモデル検査の結果を反映したものである．図4では，自動運転車が走行中のドライバ及び自動運転システムの状態遷移を示している．レベル3の自動運転システムでは自動運転システムとドライバとの相互作用が最も安全性に影響するという観点から，それらの相互作用に関して安全性の検証を実施する．

CSPモデルの構成を図5に示す．このCSPモデルは構成システム(CS)，自動運転システムの振る舞い(ADSProc)，ドライバの振る舞い(DriverProc)の大きな3つのプロセス群から構成されている．これらのプロセス群はCSPの並行合成を用いてSoS全体のプロセスを構成する．四角い矩形は，サブプロセスを表し，破線は各プロセスの情報交換による相互作用を表すためのチャネルを表す．また，実線の矢印は，プロセスの流れを表す．まず，自動運転システムとドライバが構成システムからの情報を認知し，その結果，それぞれの状態が変化する．ここでは，自動運転システムとドライバの状態の変化が安全性要求を常に満たすことを確認するために，それぞれの状態を変数として取り扱う．ドライバの状態の変数(evdstate)は，図4の状態の定義より，通常の運転ができる状態(normal)と眠い・注意散漫という運転をするには危険な状態(abnormal)を値として持つ．また，自動運転システムの状態(adsstate)は，状態遷移の条件より，自動運転が可能な状態(automated)と自動運転が困難な状態(limited)を値として持つ．図4上で定義される自動運転システムの状態である自動運転システムが運転をする状態(adsdriving)，ドライバが運転をする状態(evddriving)，及び最小リスク状態(Minimum Risk State)を自車両(EV)の運転の状態(evmode)として定義す

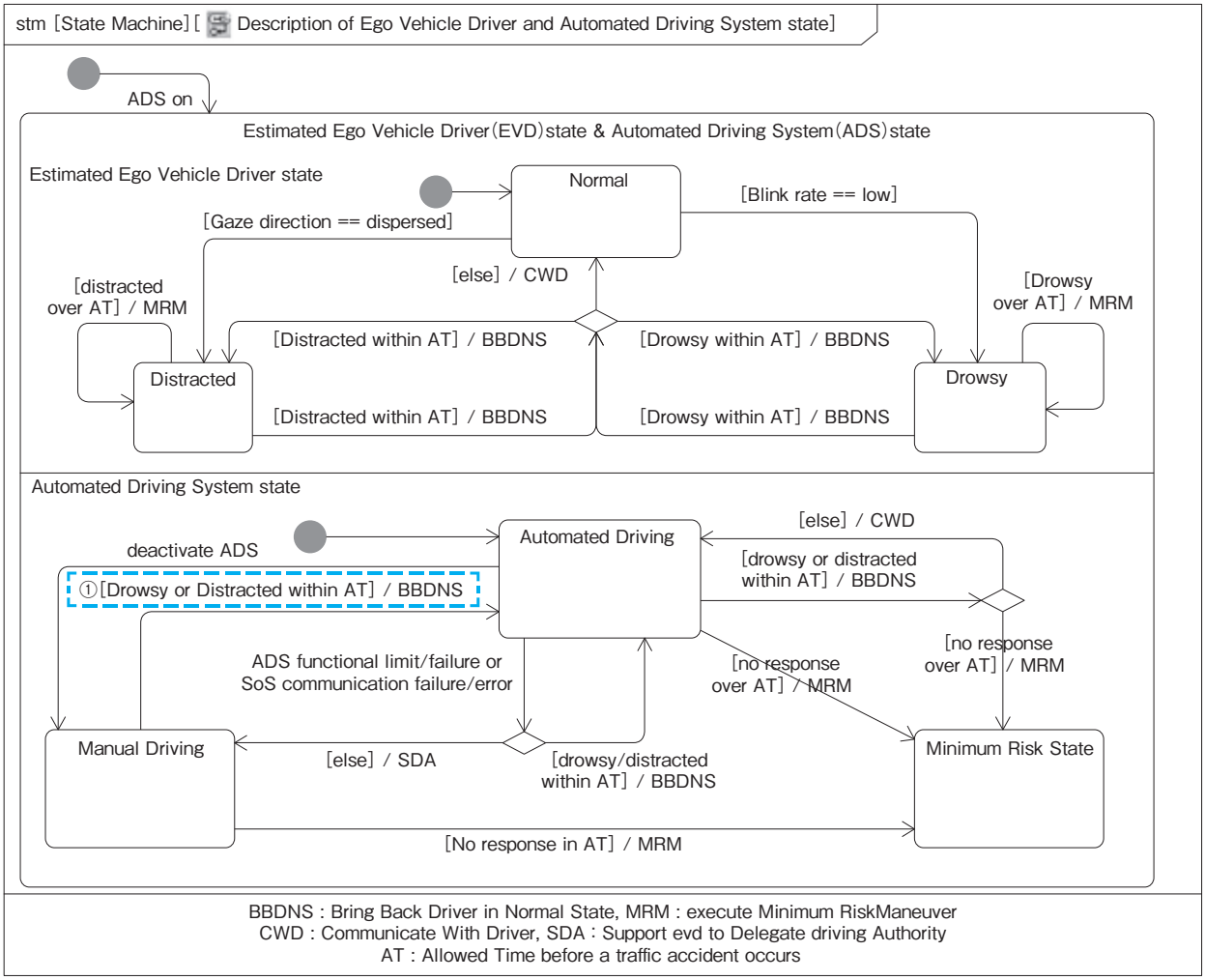


図4　自動運転車を取り巻くSystem of Systemでのドライバと自動運転システムの状態遷移

る．なお，最小リスク状態は自動運転システムが致命的な危険を避けるための状態である．

次に，図５のADSDecisionに該当するCSPモデルの自動運転システムのプロセスを表す部分を図６に示す．自動運転システムは，先に示した３種類の状態をもとに次のプロセスを選択する．これらの判断の条件文は，図２及び図４より導くことができる．例えば，自動運転システムが運転権限を持ち，かつ自動運転システムの機能では運転が困難である場合は，ドライバの状態を確認し，ドライバに権限を移譲するプロセス，またはドライバを通常の状態に戻すプロセスに移行する．ドライバが運転権限を持つ場合は，ドライバの状態を確認し，ドライバが危険な状態にある場合にドライバを通常の状態に戻すプロセスに遷移する．既に最小リスク状態に遷移している場合は，このCSPモデルでは危険を回避する自動運転システムのプロセスが続く．また，ドライバのプロセスを表すCSPモデルの記述の一部を図７に示す．図７では，ドライバが自動運転システムの振る舞いを認知することをチャネルadstoevdを介して情報を受け取るとして表現している．adstoevd を介して情報をドライバが認知したのち，ドライバ自身が運転操作をする，または自動運転システムに従うことを選択し，実行する．

4.2 モデル検査

システムモデルをもとに構築したCSPモデルに対して，検証すべき安全性要求をLTL（Linear Temporal Logic)式を用いて表現する．CSPモデルのモデル検査器であるPATはLTL式で表された条件に対して，対象のモデルが違反することがないことを検証できる[Sun2008]．LTL式の定義に反するプロセスがある場合は，PATが違反するプロセスの流れを反例として示す．この反例を解析することにより，安全性要求を満たさないシステムモデル上の対応する記述を検討する．

まず，安全性要求の妥当性を確認した図３から，検証すべき安全性要求を明確にする．図３のゴール：G_4及びG_8から求められる安全性要求をまとめると，自動運転システムが運転の継続をできない場合は，通常の状態のドライバが運転をする，または最小リスク状態に移行するという要求が導かれる．一方，自動運転システムが運転をしており，かつドライバの状態が不安全な状態の場合は，ドライバを正常な状態に戻そうとしたのち，状態が戻らないときに最小リスク状態に移行する．（図３のゴール：G_3，G_7)．これらの安全性要求を表現したLTL式の例を図８に示す．

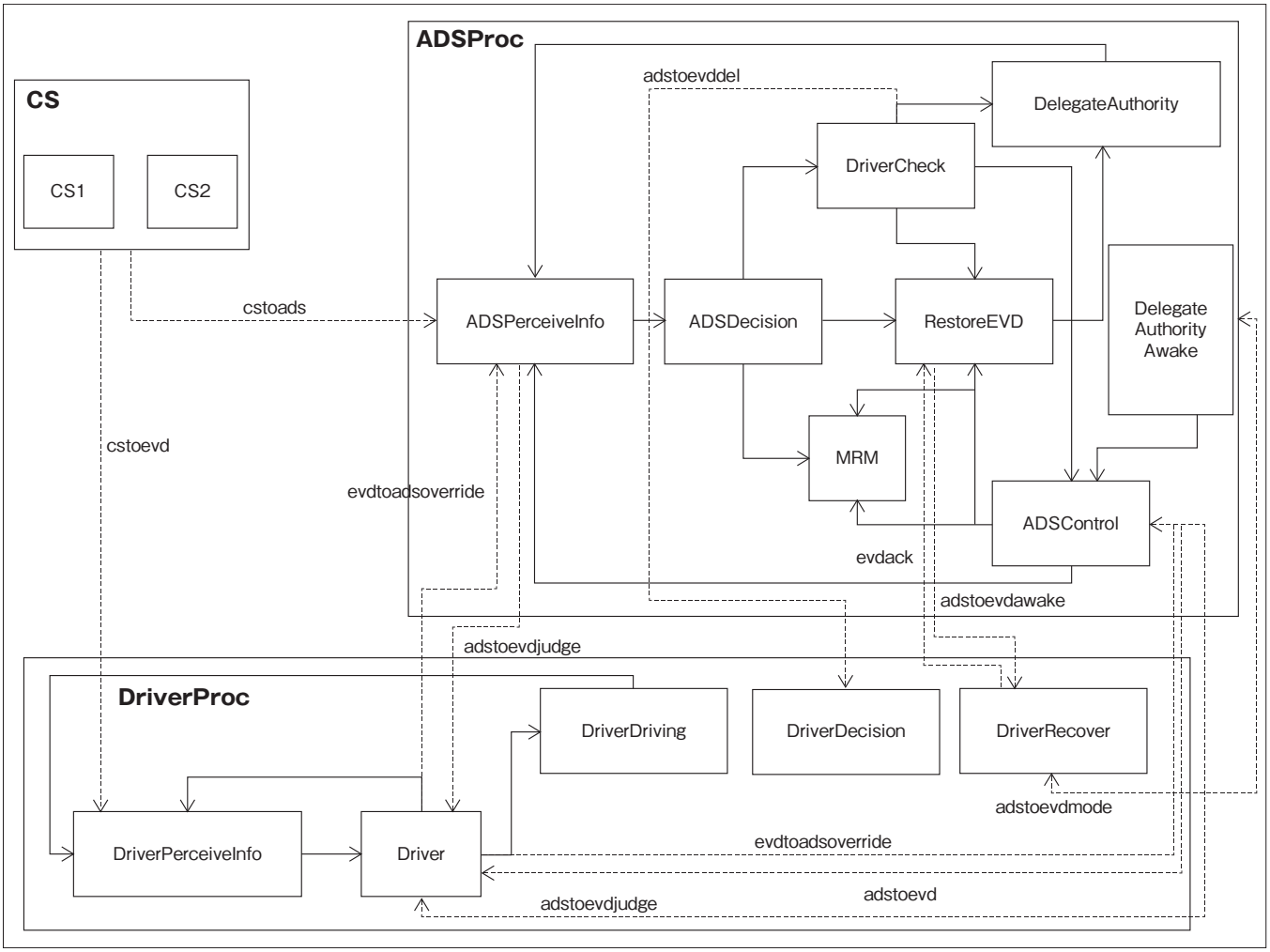


図５　CSPモデルの構成

```
ADSDecision = [evmode==evddriving && evdstate==normal && adsstate==automated](adsdecision01 -> ADSPerceiveInfo)
              [][evmode==evddriving && evdstate==normal && adsstate==limited](adsdecision02 -> ADSPerceiveInfo)
              [][evmode==evddriving && evdstate==abnormal && adsstate==automated](adsdecision03 -> RestoreEVD)
              [][evmode==evddriving && evdstate==abnormal && adsstate==limited](adsdecision04 -> MRM)
              [][evmode==adsdriving && evdstate==normal && adsstate==automated](adsdecision05 -> DriverCheck)
              [][evmode==adsdriving && evdstate==normal && adsstate==limited](adsdecision06 -> DriverCheck)
              [][evmode==adsdriving && evdstate==abnormal && adsstate==automated](adsdecision07 -> DriverCheck)
              [][evmode==adsdriving && evdstate==abnormal && adsstate==limited](adsdecision08 ->MRM)
              [][evmode==evmrs](adsdecision09 -> MRM);
```

図6　自動運転システムのプロセスを表すCSPモデルの一部

```
Driver = (adstoevd?y -> evdtoadsoverride!1 -> adstoevdjudge?z ->
          ([z==0](overriderefused -> DriverPerceiveInfo) [][z==1](driverctr1{evmode=evddriving;} -> DriverDriving)))
          [](adstoevd?y -> followads01 -> DriverPerceiveInfo)
          [](followads02 -> DriverPerceiveInfo)
          [] evdtoadsoverride!1 -> adstoevdjudge?x ->
          ([x==0](refusecmd -> DriverPerceiveInfo)[][x==1](driverctr2{evmode=evddriving;}->DriverDriving));
```

図7　ドライバのプロセスを表すCSPモデルの一部

```
#define dangerous3(evdstate == abnormal);
#define ExceptForDriver(evmode == evmrs|| evmode == adsdriving);
#assert SoS |= [](dangerous3 -> <> ExceptForDriver);
```

図8　ドライバが不安全な状態の場合に安全性要求が満たされることを検証するためのLTL式

最初にCSPモデルに対して，自動運転システムとドライバのプロセスがそれ以上進まなくなるデッドロックが起こらないことを検証し，それぞれのプロセスが予期せず停止することがないことを確認した．次に，LTL式を満たさないプロセスの遷移があるか否かを検証した．その結果，自動運転システムが運転を継続することが困難な状態で，かつドライバが不安全な状態の下で，ドライバの手動運転モードが続くパターンが検出された．

PAT上で示された反例を解析すると，まず自動運転システムの機能がほかの構成システムの変化を受け，制限された状態となる．そして，ドライバに運転権限のオーバーライドを要求し，正常な状態のドライバが要求に応答して運転権限を得る．これは，システムモデルで想定されている通り，安全な権限移譲と言える．ところが，そのほかの構成システムの影響で，ドライバの手動運転下でドライバの状態が不安全な状態となる．そして，ドライバの状態が一度は回復するが，その後もドライバが手動運転を続けている．この結果から，自動運転システムが運転を続けることが困難となり，ドライバが運転を代わったのちに危険な状態となる可能性があることが分かる．ドライバと自動運転システムが協働するという観点からは，自動運転システムがドライバの手動運転時まで自動運転システムが介入する必要性を検討する必要があると言える．ほかの安全性要求に関しては，その要求に反するプロセスは示されなかった．なお，このモデル検査の結果を受けて，ドライバの状態が不安全な場合に「Manual Driving」の状態から「Automated Driving」の状態に戻るという状態遷移が図4の状態機械図に反映されている．

5　得られた結果からの考察

3節では，システムモデルに基づいてSoSの構成システムをアクターと定義したアシュアランスケースの記述を行い，自動運転車のみならず関連する周辺システムを含めた交通システム全体の安全性要求を明確に議論することができた．とくに，どの構成システムが安全性要求に対して責任を持つべきであるかを明確に記述できることは大きな利点となる．SoSには，構成システムが時間や環境の影響により変化する特徴があり，こうした変化に対してトレーサビリティを確保した上で，関連する安全性要求の議論を再度行うことが可能となる．

次に，4節では，安全性要求が構成システム間の相互

作用によって満たされることを検証するため，システムモデルに基づきCSPモデルを作成しモデル検査を施した．SoSの構成システムはそれぞれ独立して運用されるという性質を持つことから，並行して動作するシステム要素間の相互作用を表現できるCSPモデルを用いることが有効である．また，本論文では，周辺システムからの情報に依存して自動運転システムとドライバの状態が変化することを考慮した．今後はより複雑な構成システム間の相互作用を扱うための方法を研究する必要がある．

本論文で提示したSoSの検証・妥当性確認のためのアプローチを実行するためには，システムアーキテクチャを構築し，それをシステムモデルとして記述する必要がある．それには，これらの活動に関係するメンバがシステムズエンジニアリングを理解した上で，対象とするSoSの性質を把握するために議論を重ねることが前提となる．このような体制のもとで，妥当性確認された要求に基づいてアーキテクチャの検証を行うことが重要である．とくにSoSアーキテクチャには，構成システム間の複雑な相互作用があるため，アシュアランスケースの記述やモデル検査を同時並行的に，あるいは繰り返し実施することが有効である．

なお，本研究は，大学教員４名（本務を企業に置く特任准教授１名を含む）が中心に行ったものであり，合計で約40人月を要した．これには本論文に示した内容のみならず，システムモデルの記述，過去の事故事例に基づく安全性分析などが含まれる．とくに当初は試行錯誤もあり，数多くの繰り返し検討があったことを注記しておく．

6　おわりに

本論文では，自動運転車を取り巻くSoSに対し，安全性要求の妥当性確認，及びそれに基づくアーキテクチャの検証方法を提案した．まず，システムモデルに基づき構成システムを関連付けたアシュアランスケースを記述することで安全性要求の妥当性確認を行った．次に，システムモデルに対応するCSPモデルを用い，妥当性を確認した安全性要求に対するSoSアーキテクチャの検証を行う方法を示した．ここではとくに，自動運転システムとドライバの相互作用に着目し，周辺システムからの情報による状態変化を考慮した．

SoSアーキテクチャに対するアシュアランスケースの記述では，ゴールに構成システムを関連付けるためにアクターのノードを利用し，構成システムの変化に対するトレーサビリティの確保を意図した．また，文献［西村2016］で定義したSoSアーキテクチャの自動運転システムとドライバの相互作用について検証を行ったところ，自動運転システムによる運転の継続が困難である状態で，かつ手動運転に移行したのち，ドライバが不安全な状態になる可能性があることを見出すことができた．

今後は，ドライバ特性や人とシステムのコミュニケーションなどをCSPモデルに導入した上で，モデル検査による安全性要求の検証を行いたいと考えている．

謝辞

本論文は，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（SEC：Software Reliability Enhancement Center）が実施した「2014年度ソフトウェア工学分野の先導的研究支援事業」の支援を受けたものである．

【参考文献】

［内閣府2016］戦略的イノベーション創造プログラム（SIP）自動走行システム　研究計画，内閣府，（2016）．
［BCG2015］自動運転車市場の将来予測，ボストンコンサルティンググループ，2015，（http://www.bcg.co.jp/documents/file197533.pdf）．
［Jamshidi2011］M. Jamshidi, System of systems engineering: innovations for the twenty-first century, John Wiley & Sons, 2011.
［DeLaurentis2005］D. DeLaurentis, Understanding transportation as a system of systems design problem, presented at the 43rd AIAA Aerosp. Sci. Meet., Reno, NV, 2005.
［COMPASS］COMPASS, Available: http://www.compass-research.eu/index.html
［Shneider1999］S. Schneider, Concurrent and Real-time Systems: The CSP Approach, John Wiley & Sons, 1st Edition, 1999.
［SAE2014］SAE International, Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems, Tech. Rep. SAE J3016, January 2014.
［Friedenthal2014］S. Friedenthal, A. Moore, and R. Steiner, A practical guide to SysML: the systems modeling language, Morgan Kaufmann, Elsevier, 2014.
［西村2016］西村秀和，ユンソンギル，木下聡子，北村憲康，自動運転車を取り巻くSystem of Systems のシステムモデル構築，日本機械学会2016年度年次大会，2016.
［ユン2016］ユンソンギル，木下聡子，北村憲康，西村秀和，自動運転車を取り巻くSystem of Systems の構成システムに対する安全性要求の明確化，日本機械学会2016年度年次大会，2016.
［山本2014］山本修一郎，アシュアランスケース入門，名古屋大学，2014.
［Goodger2012］A. C. Goodger, N. H. M. Caldwell, and J. T. Knowles, What does the Assurance Case Approach deliver for Critical Information Infrastructure Protection in cybersecurity?, 7th IET International Conference on System Safety, pp. 1-6, 2012.
［Saruwatari2014］T. Saruwatari, and S. Yamamoto, D* framework creation procedure from collaboration diagram, IT CoNvergence PRActice（INPRA）, Vol. 2, No. 2, pp. 43-54, 2014.
［Kinoshita2015］S. Kinoshita, S. Yun, N. Kitamura, and H. Nishimura, Analysis of a driver and automated driving system interaction using a communicating sequential process, 2015 IEEE International Symposium on Systems Engineering（ISSE）, pp. 272-277, 2015.
［Sun2008］J. Sun, Y. Liu, and J. S. Dong, Model checking CSP revisited: Introducing a process analysis toolkit, in Leveraging Applications of Formal Methods, Verification and Validation, Springer Berlin Heidelberg, 2008, pp. 307-322.

システムズエンジニアリングの推進

SECソフトウェアグループリーダー　中尾 昌善　　SEC調査役　室 修治

1 はじめに

ICT技術の進展に伴い、利用者からの要請や期待も高度化し、製品やサービス（以降本稿ではこれをシステムと呼ぶ）にとって、対応すべきことが複雑化、高度化の一途をたどっている。更に近年、従来の業種をまたぐような新たな領域のビジネスが注目され、多様な専門領域にまたがった複雑な取り組みが増加しており、システムの企画や開発を進める上での影響範囲が広がり、困難度が高まっている。

このような状況においては、従来までのアプローチでは解決できない、期待される成果を十分に得られない、思わぬ不具合を発生させるなどの問題が発生し得る。これらは今求められているシステムと従来までのモノづくりの間に何らかのギャップがあるためと考えられ、そのギャップがどこにあるかを見極め対策していく必要がある。IPA/SECではこれらを考えていくベースとして「システムを全体として捉える」ことによるメリットに注目し、そのプラクティスの集積であるシステムズエンジニアリングの理解を進め新しい時代のシステム開発に対する考え方、適用のための有益な情報を発信すべく活動を開始している。

2 システムズエンジニアリング推進の計画

2015年度より活動を開始したシステムズエンジニアリングの推進は図１のように進める計画としている。本計画は2015年中に実施した有識者による検討会において設定したものであり、そこで整理された課題と取り組むべき項目は図２に示す内容となっている。（図１）（図２）

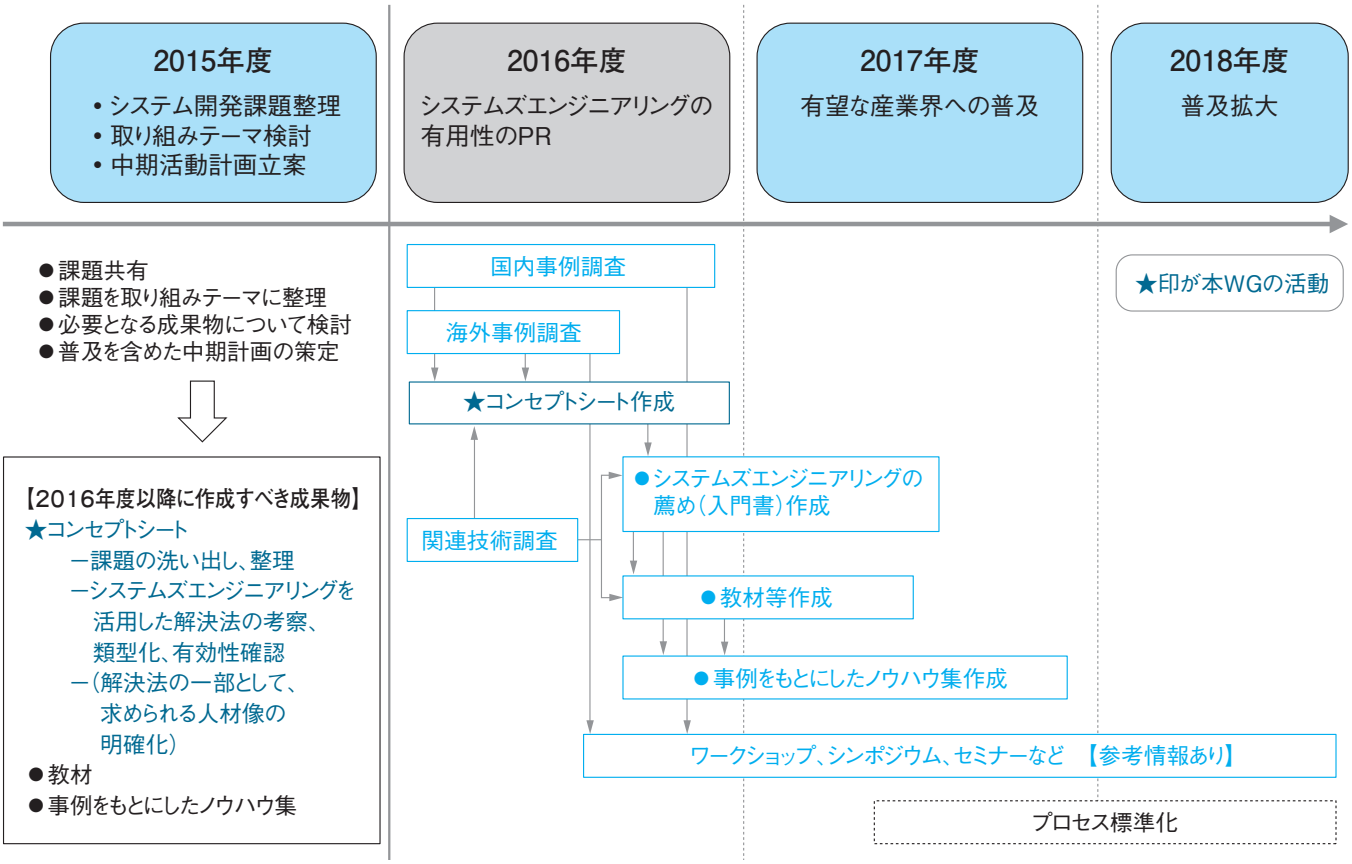


図1　システムズエンジニアリング検討のロードマップ

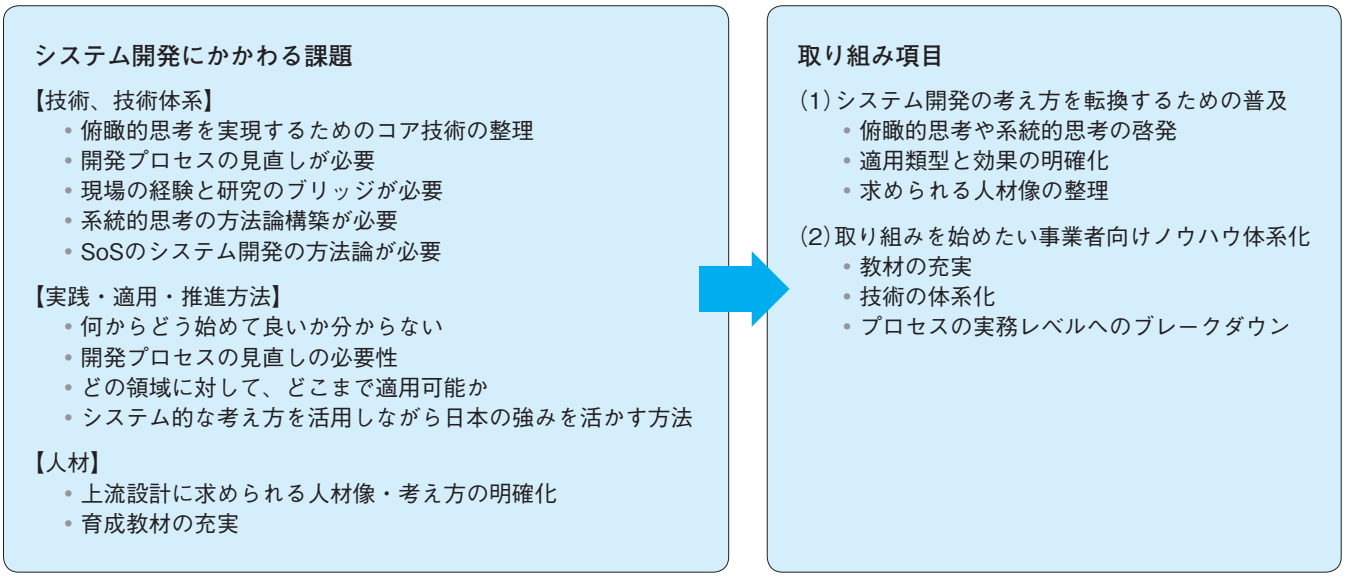


図2　「システム開発にかかわる課題」と「取り組み項目」

2016年度はシステムズエンジニアリングの有効性を紹介するためのパンフレット及びシステムズエンジニアリングをシステム開発に適用するための入門書を作成し、その理解とモチベーションを喚起できるよう計画した。

システムズエンジニアリングについてはISO 15288:2015（システムライフサイクルプロセス）、SEBoK（システムズエンジニアリング知識体系）、INCOSE SEハンドブックなどで一定の標準、知識体系が整備されているが、本活動は単にそれらを紹介、解説することを目的としたものではない。日本の産業界が今、そして将来の課題に対応し成功するために共に考えていくための有用な情報を発信し、産業界のフィードバックも得ながら共有知としていくことを目指している。

3　コンセプトシートの作成

システム開発を成功に導くための情報とするため、具体的に直面している課題を広く認識し、それらの本質的な問題について分析・整理することを活動の起点としている。2015年度より行っている国内企業などへのヒアリングも継続しながら、更に海外の先進事例の調査、ドイツ フラウンホーファ IESE研究所からの事例収集、国内企業に自社状況を提供いただいたワークショップでの情報などをもとに、2016年度より組織したシステムズエンジニアリング推進WGのメンバに議論いただき整理してきた。WGでの議論と成果のまとめ方は図3のように実施している。（図3）

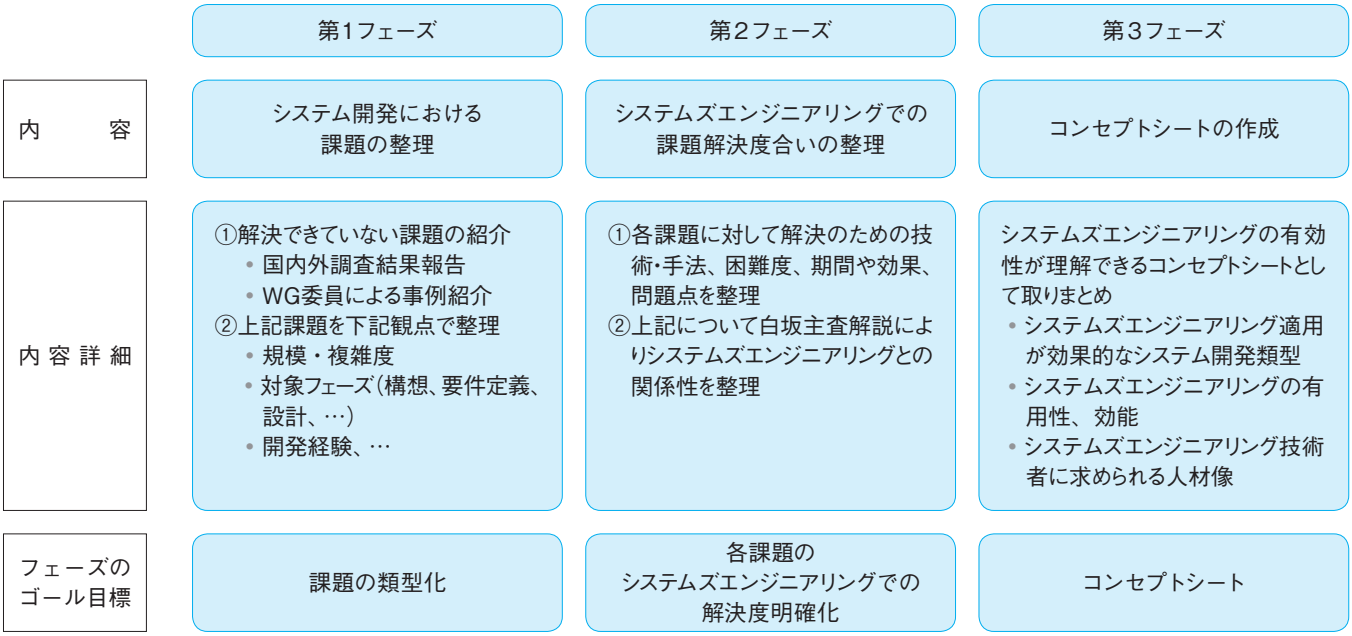


図3　システムズエンジニアリング推進WGの進め方

・課題整理の状況

国内調査結果について整理したものの一部を図４、５に示す。（図４）（図５）

従来までのシステム開発における課題が要求の高度化に伴う複雑化に伴いより難度が増している状況に加え、新しいビジネス形態への対応についての意識が見て取れる。

海外の事例などの収集が進む中で、切り出し整理した課題では機微が伝わらない＝実感を伴った情報とできない可能性の指摘を受け、図６に示すような実事例→事例中表現での課題→本質的な課題→事例中での対策→システムズエンジニアリングではどのようになるかを一覧形式で把握できるような情報としシステムズエンジニアリングの適用場面、手法、効果などの整理を進めた。（図６）

また、ワークショップでの成果については図７のような整理を行っている。

これらを総合して整理すると解決すべき優先的な課題は図８に集約されそうである。

コンセプトシートには上述した情報に加えそれらを実際のプロダクト／プロジェクトとして実施された事例も理解を助けることを目的に掲載していく。

#	業種	課題認識	左記に関連する補足事項	事業者の取り組み方針
1	交通（鉄道）	お客様に提供する新しい価値の創造と顧客サービスの向上（安全性に加えて利便性、快適性などの追求）	今はスパイラル状の継続的な発展（技術改良）	（非連続的な）飛び越える技術革新（イノベーション）を行う
2	交通（鉄道）	特定の駅間での混雑が恒常的であることへの対応	新しいシステムの開発：デジタル式自動列車制御方式	対応路線拡大中
3	自動車 OEM	車と車の外とのつながりが増えていく →従来にないインターフェース統合ニーズに対応		必要な人材の確保
4	自動車 OEM	｛従来にない要求｝ 自動運転、ネットワーク化への対応	スコープ設定が甘い→検討漏れ→大きな手戻り	
5	自動車 TEAR1	単品デバイスの技術向上では勝てない時代になってきている →単品デバイスをつなげた（システム）ソリューションに持っていきたい	単品開発をやってきた人材ばかりで上位レイヤーで考えられる人材が不足している すり合わせ開発に慣らされて時代に遅れている	抽象化能力の育成 IoTやOpen-Systemでは密でなく疎に作るセンスが必要 （組込み系とWeb系のような育ちの違い？）
6	自動車 TEAR1	要求開発は重要でやりたい →シミュレータを使った合意形成		
7	電機	｛製品・サービスの｝ SoS化への対応	今までのやり方ではうまくいかない全体を俯瞰できる人材が不足している	
8	ヘルスケア	個別のBtoBで要件の厳しいものが出てきた際の対応	測定結果を時系列でクラウドに蓄積するだけなら重いSLA㈱は不要 散発的にBtoB案件があり個別に苦労	
9	SI	単純なシステム開発にとどまらないケース（ステークホルダの多様化・増加、SoS、IoT、IT/OT融合など）への対応		｛IoT、SoSなどに対応できる開発標準などの強化拡充｝

※｛　｝で囲んだ記述はIPA/SECによる推察　　（注）SLA（Service Level Agreement）

図４　課題例 従来にない新たな要求

#		#	分類
1	要求事項の変化	1.1	従来にない新たな要求
		1.2	要求の複雑化
		1.3	要求の多様化
		1.4	要求高度化に伴う 新たな品質要求
		1.5	機能高度化による コスト・工期への影響
		1.6	要求に関する早期合意形成
2	その他	2.1	人材育成
		2.2	社内標準化、知識管理
		2.3	ビジネスのグローバル化、 ビジネスの可視化、 組織構造、ルール、風土

図５　国内調査における課題の整理

	A	B	C	D	E	F	G
	事例	事例で述べられている事実（課題、従来までのやり方では対応できない壁）	なぜそうなってしまうのかなぜそのような状況なのか（課題の原因）	事例で述べられている対策または事務局で推察・整理した対策	事例で述べられている効果または事務局で推察・整理した効果	対策（課題解決方法）を決定、及び実施するための活動・技術（システムズエンジニアリングの活動・技術）	システムズエンジニアリングが効果を示す場面・条件の分類
1	チェレンコフ望遠鏡アレイ開発	国際的プロジェクトであり物理や電気など専門領域も多岐にわたる巨大プロジェクト 言語や設計ドキュメントの表現などもまちまち	国際的、多数の専門領域を束ねる巨大プロジェクトにおける設計情報、技法が統一されていない	モデルによる可視化	言語の違いを克服 相互理解に有効	モデル駆動システム開発	検討中
2	ドイツ国防省 モジュール式多目的戦艦（MKS180）開発	モジュール式システムの構築には各モジュール担当者が対応可能か、スキルを持ち合わせているか、確認できなければならない	要求が明確でなければスキルを持ち合わせているか確認できない	要求の明確化のための抽出技術（要求抽出技法マトリクスによる多視点の確認、モデルベース要求開発）	（明確に確認できず）	モデルベース要求開発	検討中
3	輸送ドメインの 統合品質保証	自動車、航空、鉄道など多国間、多組織を跨る巨大システムの検証が膨大 品質保証が困難	（←）	形式検証、レビュー、テスト技法など、静的品質保証の技法と動的品質保証の技法を組み合わせた体系的な手法を実施	検証と妥当性確認の作業コストは、平均32%削減が可能（効率化） 後工程での残存欠陥により発生するコストは、平均27%削減が可能（品質向上） テスト・カバレージ、出荷後品質など８％は向上	モデルベース開発	検討中
4	トラックのリモート制御リフト用安全機構開発	リフトはスマートフォンで制御しなければならないが、スマートフォンは安全なデバイスとは限らない	スマートフォンの安全基準が低い	システム及びセーフティに関するコンセプトを仮想開発による評価を行いながら、モデルと実装開発を段階的に行った	・仮想プロトタイプを使用して機能を評価することで、実装／試作を使用した場合よりもはるかに早い段階で定量評価が可能になる ・欠陥や誤った仕様が早い時点で検出されるため、複雑なシステムを開発する際のリスクが減り、必要なやり直し作業も少なくなる ・仮想開発を使用することで、既存のアプローチではなく、新しいアイデア・革新的な概念を創出することが可能になる ・シミュレーションによって、開発者は経験を集め、様々なアプローチの性能を定量的に評価して比較できるようになる	モデルベース開発	検討中

（参考：IESE調査事例）

図６　事例にみるシステムズエンジニアリングの特徴

参加3企業課題から抽出された重要課題の共有とその推奨ソリューション領域

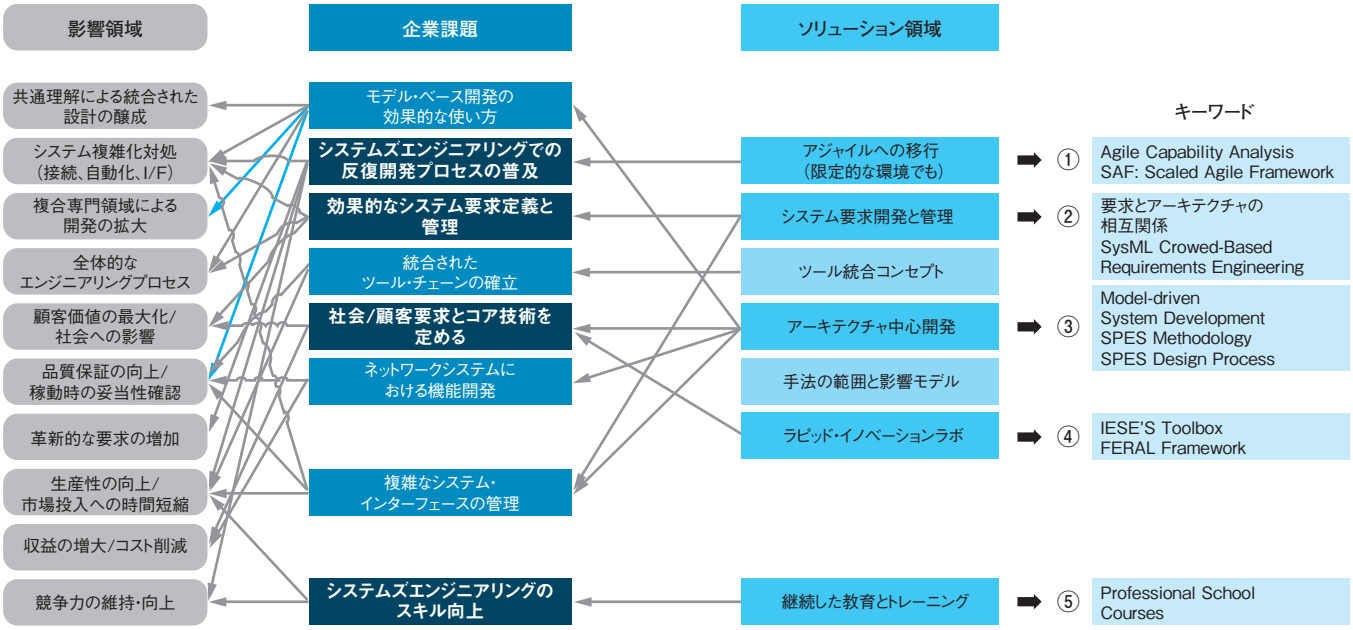
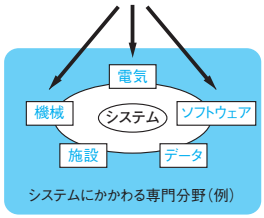


図7　ワークショップ 実施結果まとめ

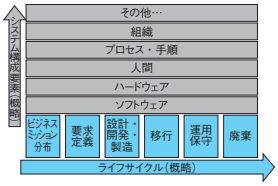
多様な専門分野にまたがる

特定の専門分野を深めるだけでは達成できない高い付加価値を実現すべく多様な専門分野をまたいで知見を総合し、全体最適を実現する考え方が求められています。



ゴールに向けた全体の俯瞰

限られた経営資源（ヒト、モノ、カネ）でゴールに到達するために、システムの検討（企画・開発）においては構成要素のすべてとライフサイクル全体を対象に、それぞれの要素間相互の関係を考慮することが必要です。ビジネス面、技術面を総合した知見が求められます。



抽象化・モデル化

抽象化・モデル化による品質向上や企画、開発の効率化が期待されています。留意点と期待される効能として以下があります。

- 抽象化の視点を柔軟に設定し、多視点からみることで「中核となる意味の理解」と、「アイデアの展開応用」を促進
- 可視化したモデルを通して異なる専門領域の人たちの間での情報共有と共通理解を促進
- モデル化を介して自動化につなげることで開発効率を改善

イテラティブな推進

再評価とフィードバックを反復して、段階的に明確化する考え方です。期待される効能として以下があります。

- 環境変化への柔軟な対応
- 避けられない「当初のあいまいさ」への対応

図8　課題の整理

4　入門書の作成

システムズエンジニアリングの有効性について一定の認知、理解をコンセプトシートなどで得られた後にはどのように自分たちのシステム開発に適用するかの情報が必要となる。コンセプトシートではそれら実現手法の詳細までは触れていない。実際にシステム開発の実務を担当される方を対象として技術者の観点・視点で解説する目的で入門書を作成している。

5　教材の作成

具体的にシステムズエンジニアリングを実施していくために必要となる技術などについては教材としてまとめていく予定である。2016年度は上述した活動を元に技術を整理し必要な教材を選定していく。作成は2017年度以降優先順位をつけて実施していく予定である。

6　調査事例の公開

活動中調査、収集した事例は2016年度中に公開していく。

IESEでの収集事例については既に2016年12月に公開されている。コンセプトシート、入門書で触れたその他事例について2017年３月中の公開を予定している。

7　2017年度の取り組み

2017年度の活動は上記教材の作成を進めつつ、これらコンセプトシート、入門書の普及の実施及び有効な適用方法の検討を実施していく計画である。

システムズエンジニアリング概要
～ VUCA時代のシステムデザインアプローチ～

慶應義塾大学大学院システムデザイン・マネジメント研究科 准教授　白坂 成功

ますますシステムを構築することが難しくなっている時代において、システム構築の基本的な考え方であるシステムズエンジニアリングの概要について説明する。とくにシステムとはどういったものであり、システムズエンジニアリングはどのようなものであるかを概説する。

1 背景

VUCAという言葉をご存知だろうか？ VUCAとは、Volatility（変動性）、Uncertainty（不確実性）、Complexity（複雑性）、Ambiguity（曖昧性）から構成された言葉である。現在の世の中はVUCAワールドと呼ばれ、先の予測ができない、計画通りにならない時代であると言われている。このような時代だからこそ、どのように環境を捉え、どのように考えてシステムを構築するかが重要となる。そのとき役に立つのがシステムズエンジニアリングである。例えば、どのように環境を捉え、そのためにどのような要求を持ち、それをどのように考えて設計したのかが分かっているから、環境が変化したときに、何が影響を受けるのかを把握できる。また、つながるシステムの代表であるIoTシステムのように、当初の想定とは違うものがつながり、複雑さがどんどん増す中で、その複雑さをコントロールする手段があるからこそ、対象を俯瞰的に捉えることができる。本稿では、VUCAワールドと言われる時代のシステムを考えるためには欠くことができないシステムズエンジニアリングの概要について説明する。

2 システムズエンジニアリング概要

2.1 システムとは

世の中にあるほとんどのシステム、サービスのいずれも、複数の要素から構成されているという点で、システムである。では、そのシステムというものをもう一度考えておくことは重要である。

システムは、「ハードウェア、ソフトウェア、ファームウェア、人、情報、技術、設備、サービス及びほかの支援要素を含む、定義された目的を成し遂げるための、相互作用する要素を組み合わせたもの」（INCOSE Systems Engineering Handbook）であると定義されている。つまり、いわゆるハードウェアやソフトウェアだけがシステムなのではなく、その中に人などを含むことがもともと想定されている。いくら詳細に個別の要素を見ても決して理解できないのがシステムである。システムとしてものごとを理解するためには、ものごとの個別要素にとらわれるのではなく、全体を一つのものとして捉える必要がある。

このようなシステムには幾つかの性質が存在する。ここでは、前述した定義に含まれている性質から説明し、その後、定義には書かれてないが、よく言われるシステムの持っている性質について説明する。これにより、システムがどういったものであるかについて、より明確に分かるのではないかと思われる。

目的性：システムは、それ固有の目的を持つ

前述した定義には「定義された目的を成し遂げるための」という記述がある。つまり、システムはそれ固有の目的を持つことが前提となっている。実際には、“システム”というものを一般的に扱うと必ずしも目的を持たないものも存在する。例えば、太陽系というシステムは自然システムの代表例であるが、目的を有していないため、目的性がないと言える。しかしながら、ここで扱うシステ

ム（人が作り出す人工システム）は、何らかの目的を持っており、それを実現するために存在するものに限定する。実際に、人間が作り出すシステムは何らかの目的を有しており、ここで扱う範囲においては上記性質を持っていると考えて問題ないはずである。

集合性：システムは、目的を果たすために必要な複数の要素の集合である

前述した定義では「要素を組み合わせたもの」と書かれている。つまり、要素が単体として存在しているわけではなく、複数の要素から構成されていることを示している。

相互関係性：システムを構成する要素間にはシステムの目的を果たすために必要な何らかの関係が存在する

前述した定義では、「相互に作用する要素」という記述がある。上述した通り、複数の要素が集まればシステムというわけではなく、複数の要素が相互に関係することにより、結果として新たな特質が生まれたとき（これを“創発”と呼ぶ）に、それをシステムと呼ぶ。

階層性：システムの構成要素は、それ自体がシステムであっても良い

例えば、パソコンを考えてみる。もちろん、パソコンはシステムである。このパソコンというシステムは、モニター、本体、キーボード、マウスなどから構成されている。では、本体という要素を考えると、その中にはハードディスクやCPU、メモリなど多くの要素から構成されている。つまり、システムの要素がシステムとなっていると言える。実は、この考え方は大変重要なものである。なぜなら、このシステムの階層性という考え方を理解すると、その考え方を繰り返し使うことで、全体から詳細に至るまで統一的な考え方で設計を進めていけるようになる。この階層性という考え方を「Building Block」と呼び、システムの構成要素のことを「サブシステム」と呼ぶ。図1では、システムはサブシステムから構成されており、このサブシステムもまたシステムであるということを示している。これがシステムの階層性である。

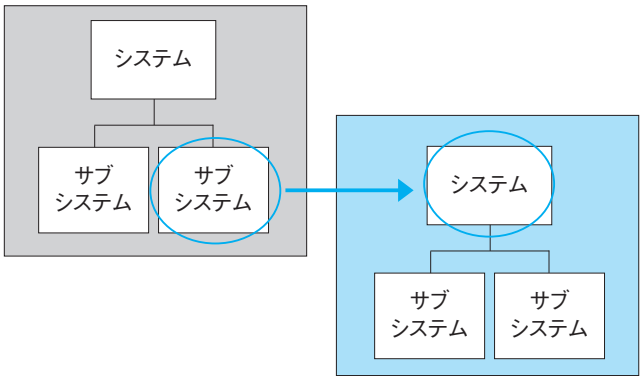


図1　システムの階層性を表すBuilding Block

任意性：システムの範囲はあらかじめ普遍的に決められたものではなく、個別に定義され決められるものである

システムというものを考えると、そこには明確な範囲というものが存在するように感じるかもしれない。しかし実際には決してそうではない。どこまでがシステムの範囲であり、どこからがシステムの外部であるかは、その目的や立場から定義するものである。例えば、同じシステムを見るときでも、その利用者も含めてシステムとして捉える場合もあれば、利用者はあくまでもシステムの外部にいる人として捉える場合もある。前者の場合には、利用者にも積極的に役割を与え、訓練などを実施することでシステムの構成要素としての役割を担うことを期待することなどがある。

上述したようにシステムは幾つかの代表的な性質があり、これらの性質を知っておくことは、システムを理解し、扱うために重要なことである。

2.2　システムズエンジニアリングとは

システムズエンジニアリングとは、「システムの実現を成功させることができる複数の専門分野にまたがるアプローチ及び手段」と定義される。（INCOSE Systems Engineering Handbook）つまり、システムズエンジニアリングとは、複数の専門分野（例えば、電気工学、機械工学、ソフトウェア工学など）を統合し、束ねるためのアプローチである。単一の専門分野だけで課題を解決したり、価値を創造したりすることは難しく、複数の専門分野の統合が必要となることを考えると、システムズエンジニアリングは課題解決や価値創造のために必要な、基本的な考え方であると言える。

次にシステムズエンジニアリングを実施するために役に立つポイントを紹介する。

① 多視点から見る

システムズエンジニアリングの実施のためには、対象を多視点から見て考えることが重要である。対象によって最適な視点は異なるが、一般的に役立つ視点がある。それは、時間の視点、空間の視点、機能の視点、物理の視点である。時間の視点は、対象となるシステムの観点ではライフサイクルと呼ばれ、ユーザの観点ではカスタマージャーニーなどと呼ばれる。機能と物理は別の視点であることを認識して、分離して考えるだけで有益である。

② 俯瞰的に捉え、系統的に考える

多視点で見たものを、それぞれの視点ごとに俯瞰的に捉え、系統的に考えることが必要である。例えば、時間の視点で見て、俯瞰的に捉えることでライフサイクル全体を視野に入れ、系統的に考えることで、ライフサイクルをもれなく分割して系統的に考えることがこれに当たる。

③ 抽象度をコントロールする

ある視点から見て、俯瞰的に捉え、系統的に考えるときには、抽象度をコントロールすることが重要である。人は、あまり多くの数を一度に把握することが難しい。このため、いきなり全体を20や30に分割するのではなく、まずは５つに分割し、それぞれを更に５つに分割する。こうすれば、最終的には全体を25に分割することになるが、全体感を失わないように徐々に細部を考えていくことが可能となる。

以上のように、システムズエンジニアリングは、複数の専門分野の統合のためのアプローチであり、この実施時には３つのポイントを意識しておくと良い。

2.3　Vモデル

Ｖモデルは、システムズエンジニアリングの基本的な考え方を表す。図２に示したように、Ｖモデルの左側はシステムデザインと言われ、要求分析とアーキテクティングを実施することでシステムを構成するサブシステムへと分解することを表している。Ｖモデルの右側は、実現されたシステムを構成する要素を統合（インテグレーション）してシステムを実現することを表している。そして、Ｖモデルの左側・右側の両方で、評価・解析を実施する。Ｖモデルの左側で実施する評価・解析としては、例えば、設計が正しいかどうかを確認するためのシミュレーションや、複数の設計候補案から一つを選定するために実施するトレードオフ分析などが該当する。Ｖモデルの右側で実施する評価・解析としては、試験が挙げられる。このとき、左側と右側のレベルが合わせてあり、Ｖモデルの左側での設計に対応した試験が、Ｖモデルの右側で実施される。また、右側で実施される試験のことを考えて、左側で設計を実施する。

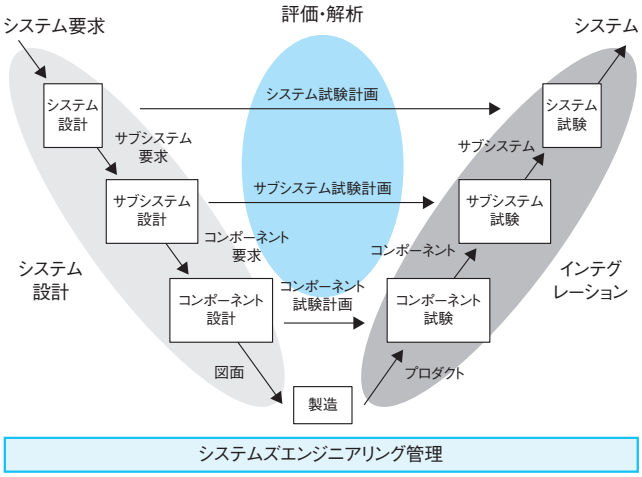


図２　Vモデルとシステムズエンジニアリングの構成

Ｖモデルは、システム開発全体プロセスを表すもの、と誤解されることがある。そのような場合もあるが、基本的にはシステム開発全体のライフサイクルとＶモデルは独立に考えられるものである。つまり、システムのライフサイクルを通じて、Ｖモデルが何度も繰り返されることもあり、また、対象となるシステムの“部分”を表すときにも使われる。例えば、システムを構成する構成要素の一つが、これまでに使ったことのないような技術を活用したもので実現することを考えてみていただきたい。そのような場合、新しいことをいきなりやってうまくできない可能性を減らすために、試しに作ってみるということを考えるであろう。つまり、部分的に設計、製造、試験をして、うまくできることを確認する。その上で、

全体を別のＶモデルに従って設計、製造、試験をする。このようにＶモデルは、ライフサイクルを通じて複数となることもあり、またその範囲がシステム全体のときも、システムの一部分であることもあり得る。

Vモデルは、色々なところで使われるようになった結果、「作業の順番を定義している」と捉えられているケースがあるが、実際には、順番を規定しているのではなく、あくまでも考え方を表現したものであるという点は注意が必要である。ここを間違えると、より進化したシステムズエンジニアリングを理解できなくなってしまう。あくまでも、Vモデルはシステムズエンジニアリングの概念を表したモデルであり、例えば、システムの要求は、サブシステムの要求の統合として実現されるということであり、Vの左側で行われる活動結果が、Vの右側で行われる活動で確認可能であるということである。

2.4 システムエンジニアリングプロセス

システムズエンジニアリングは、大きく４つのプロセスから構成される。上述したVモデルの説明と重複するところもあるが、ここでもう一度まとめて説明する。

① システム設計

まず、ステークホルダの要求を分析し、全体が整合し、抜け漏れがない形にすることで、要求を定義する。次に、その定義された要求を実現するために、アーキテクチャ設計を実施する。具体的には、複数の視点（viewpoint）からシステムを見て、それぞれの視点から見えるもの（view）における要素と要素間の関係を定義し、異なる見えるもの（view）間の関係性を定義することで、下位への要求を導出する。この活動はVモデルの左側として実施される。システム設計の実施時には、Vモデルの右側で実施されるインテグレーション及び試験のことを考慮して行う。

② インテグレーション

検証の終わったサブシステムを統合する活動である。

③ 評価・解析

エンジニアリング活動における解析及び検証（verification）・妥当性確認（validation）などの活動である。Vモデルの左側では、システム設計をシミュレーションで評価したり、代替案をトレードオフ分析をすることなどがこれに当たる。また、Vモデルの右側では、できあがったサブシステムを試験したり、インテグレーションされたシステムを試験することがこれに当たる。

④ システムズエンジニアリング管理

QCDを満たすために、ライフサイクルを通じて各種活動の計画・実施・評価を行う活動がこれに当たる。この過程において、どのような範囲で何回Vモデルを実施するかを決めていくことも行う。例えば、初めて利用する技術であれば、評価のために、その範囲のVモデルをほかの部分よりも増やして実施することを計画し、実施するなどがこれに当たる。

3 最後に

本稿では、システムズエンジニアリングの基本的な考え方を紹介した。今後、VUCAが進んでくると、対象となるシステムを捉えることが容易でなくなるからこそ、システムを扱う基本的な考え方であるシステムズエンジニアリングの基本をしっかりと理解し、実施できるようになっていることが重要である。そのような基礎をおさえておいてはじめて、システムズエンジニアリングから進化したアプローチを活用できるようになる。そのためにも今一度、システムズエンジニアリングの基本を理解しておいていただければと考えている。

システムズエンジニアリングの上流設計プロセスについて

株式会社コギトマキナ 代表取締役　鈴木 尚志

1 はじめに

1.1 本稿の背景と目的

近年、様々な複数のシステムが相互に結合された新しいサービスや価値が生まれている。これらのシステムの開発を成功させるためには、これまで行ってきた単独システム開発とは異なる開発視点が要求されることも多い。

システムズエンジニアリングは、対象とするシステム（System of Interest）全体を見渡すことにより、システムの開発や運用などを成功に導くための工学である。学問としてのシステムズエンジニアリング自体は、新しいものではないが、近年の複雑であったり大規模であったりするようなシステムの成功のためには重要であると近年注目を浴びている。

一方、これらのシステムの成功には、組み込まれたソフトウェアが大きく寄与していることも多く、そのため、ソフトウェアエンジニアがシステムズエンジニアとして活動することを期待されることも多くなってきている。

そこで本稿では、ソフトウェア・エンジニアリング領域で活動される諸氏を主な対象とし、まずはシステムズエンジニアリングプロセスのうち上流の「システム設計活動」を紹介する。

また、最近は、便利さのため「モデル」を用いるシステムズエンジニアリングであるモデルベースシステムズエンジニアリング（以下MBSE）に脚光が当たっている。本稿後半ではモデル化言語SysMLを用いたMBSEについても触れる。

2 システムズエンジニアリングにおけるシステム設計活動とは

2.1 システム設計活動の目的

システム設計活動の目的は様々な利害関係者のシステム把握に関する合意である。

主な合意の内容は、以下の２点に関するスコーピングであり、詳細な実装方法ではない。

1. そのシステムが解決する課題（What）
2. 課題に関する解法（How）

この目的を達成するために、対象システムのスコーピングを定義した最終成果物と合意を目的としたものも含む、議論を効率よく行うための資料としての補助的成果物という２種類の成果物が存在する。

2.2 システム設計活動の概要

システム設計活動は、大きく２つの活動、要求分析とシステムアーキテクチャの構築から構成されている。

成果物であるシステムアーキテクチャは、システムの企画時から、開発、運用、そして破棄に至るまで、すべてのライフサイクルにおける活動の基盤である。

2.2.1 合意について

システム開発に当たっては様々な議論が行われる。これらの議論の目的や内容は多分野にわたる（Multi-discipline）ことも多い。議論の例を挙げれば、要求や機能の追加や変更、新規テクノロジーやCOTS品の採用や既存システムとの共存、開発期間の短縮を含む様々なコスト削減、運用時や廃棄時の環境への省影響性、クリティカル性能などへの対応、相互運用性などである。

また最近ではとくに、自動車業界におけるISO26262のような機能安全規格のトレーサビリティや障害対応性などの安全設計、またセキュリティへの考慮も多くなっている。

これらの議論は、ある課題についての複数の解法のトレードオフを考慮しなければならない部分で紛糾することが多い。そのための、トレードオフスタディについての資料を作成することもシステムズエンジニアリングにおける重要なミッションの一つである。

2.2.2 システム設計活動を適用するタイミング

また、よく誤解されているところだが、システムズエンジニアリングの上流プロセスは、開発初期時にのみ実行されるものではない。

システムのライフサイクルは、概念検討から始まって、

開発、運用、利用、そして廃棄に至る。

しかし、現実として、システムの開発は、完全に上流で要求を固め、アーキテクチャを一度だけ構築するという、いわゆる理想的なV字プロセスではない。事実、システムのライフサイクル後期の製造や運用、保守の際に発見され、考慮された事項が、設計や開発、すなわちライフサイクル上流の成果物に反映されることも多い。システムズエンジニアリング各活動は、「どのライフサイクルにおいても「手戻り」が発生する可能性はあり、そして、システム成功のためには「早期の迅速な手戻り」は重要である」という思想を根底に持つ。

3 システム設計活動の詳細

ここではシステム設計活動の中心である「要求分析」「システムアーキテクチャの構築」という２つの活動について紹介する。なお、この基本的な２つの活動はウォーターフォール的に行われるわけではなく、開発ライフサイクルを通じ、何度も実行されるイテレーティブな活動であることをあらためて記しておく。

3.1 要求分析

システム設計活動は、通常、そのシステムを使用する、あるいは購入する顧客の要求を獲得することから開始される。

しかしシステムの使用者や操作者のようなユーザが、開発者がすぐに開発可能なような完全な要求を定義することは不可能である。また、提示された要求が、ほかの利害関係者の提示した要求と不整合を起こすことや、開発者自身が開発者制約と言われる要求を入力することも多い。

要求分析とは、つまり、このような様々な利害関係者の要求を、設計者にとって十分な品質を持った技術的仕様へと変換する作業である。

要求分析の作業は、２つの側面を持つ。１つはシステムをブラックボックスとして定義することで、対象システムとシステム外部との間のシステム境界を明確にするという側面。もう１つが、対象システム内部を明確に記述するというホワイトボックス的側面である。

3.1.1 要求の種類と領域

システムへの要求は多岐にわたる。主な要求の種類として

目的を実現するために持つべき機能
目的を実現するために持つべき性能
システムと外界との外部インターフェース
動作環境
目的を実現するために持つべきリソース
システムに対する物理的要求（容積、サイズ、重量など）
設計（設計基準など）
プロセスと手法（製造プロセスや品質管理など）

などが存在している。

その一方で、要求分析の技術領域もとても広範である。これらは様々なシステムズエンジニアリング標準においても様々な定義がある。

そこで、以下にシステムズエンジニアリングの世界標準の一つであるIEEE1220のシステムズエンジニアリングプロセスにおける要求分析タスク領域の例を挙げて要求にかかわる領域を簡単に紹介する。

IEEE1220における要求分析タスクには以下の15の領域がある。

1．顧客の期待
2．プロジェクトと会社の制約
3．外部制約
4．運用シナリオ
5．効果指標（MoE）
6．システム境界
7．インターフェース
8．利用環境
9．ライフサイクルプロセス概念
10．機能要求
11．性能要求
12．運用モード
13．技術的性能指標
（Technical Performance Measures）
14．物理特性
15．ヒューマンファクタ

そして上記15領域の要求分析タスクを入力として、「要求ベースライン統合」タスクが実行されることとなっている。

3.1.2 要求分析の手法

要求の種類も多岐にわたり、考慮すべき領域も広範であるため、現実には要求分析作業の多くが属人的なものである。効率的に要求を抽出するためには、経験的に様々な手法がある。ここではその一部を例示する。

3.1.2.1 ブラックボックス的な要求分析手法

ブラックボックス的な要求分析手法は、主にシステム境界を明確にするという側面のために行うことが多い。そのためにはそのシステムの使用方法（ユースケース）の文書化や、そのシステムのコンテキストやミッションプ

ロファイルなどを記述した運用コンセプト文書の作成などを行うことによって要求を洗い出す。

これらの文書作成や要求の洗い出しのためにはユーザや近似システムの経験者などの参加が基本的に必須である。

3.1.2.2　ホワイトボックス的要求分析手法

ホワイトボックス的な要求分析手法は、対象システム自体を明確にするという側面のために行われることが多い。そのために、洗い出しを中心としたブラックボックス的要求分析手法とは異なり、明確に「システムの機能」について、様々な観点から考慮することが多くなる。

ブラックボックス的要求分析において、ユースケースなどで洗い出された機能をより明確にすることを機能分析と呼ぶこともある。ここで言うユースケースはあくまで、その「システムの使い方」であり、「ユースケースが機能を使用する」と考えれば理解しやすい。そして、この機能分析が、要求分析とシステム設計を接続するための軸となる活動であることに注意されたい。機能分析を通じて、システムは何を行うべきであり、行うべきでないか、という判断が決定され、明確化される。

機能を明確にするためには、従来からFFBD(Function Flow Block Diagram)のような機能フロー図や、状態モード分析、規定外の条件への対応を規定する範囲外動作分析などを行うことも多い。

一方、属人性を可能な限り廃するために、検証方法を先に定義することによって要求のヌケモレなどを検出することもある。そのため、最近では形式的仕様記述や構文解析による要求抽出などの形式的方法が用いられることもある。これもホワイトボックス的要求分析の一つである。

上記の活動などから得られた要求を統合したものがシステム仕様書である。作成の際には、各要求の妥当性、明確性、完全性、無矛盾性、単一性、検証可能性、変更容易性、追跡可能性、重要度などの要求そのものの品質についても意識しなければならない。

3.2　システムアーキテクチャの構築

システムズエンジニアリングにおいて「システムアーキテクチャの構築」は、トップダウン・アプローチである。

「システムアーキテクチャの構築」活動は、目的としている機能を規定のパフォーマンス制約の制限内で提供可能にするようなシステムの物理アーキテクチャの開発に重点を置いている。

ここでは要求分析の成果物であるシステム仕様書を入力とし、成果物としてシステムアーキテクチャ仕様書と補足文書を作成する。ここで言うシステムアーキテクチャは、デバイスやチップレベルのハードウェア設計や、物理的なプラントや社会インフラ開発などを対象としたものであり、明らかにソフトウェアアーキテクチャと言われるものよりも抽象度が高いものを指す。

また、システムアーキテクチャでは対象となるシステムと、関係するシステムとの相互作用も定義する。

アーキテクチャの定義は様々であるが、システムズエンジニアリングのためのシステムアーキテクチャでは、システムの機能やサブシステム構造のみならず、システムの安全性、信頼性、可用性、保守性、セキュリティといったシステムを横断するような重要な関心事についての定義は必須である。

3.2.1　観点と要素の割り当て

システムアーキテクチャの構築活動は、システムの目的を達成できるように独立したアーキテクチャ要素同士を関連付ける活動である。システムアーキテクチャには関心事ごとの様々な側面(ビュー)がある。その側面を理解するための観点(ビューポイント)がある。アーキテクチャ要素の種類も、観点ごとに存在し、例えば要求、機能、振る舞い、構造、フロー、プロパティなどと多岐にわたる。

アーキテクチャ要素同士の関連付けは、同種のもの同士とは限らないことに注意が必要である。異種のアーキテクチャ要素同士を割り当てること(アロケーション)によって、異なったアーキテクチャの側面間の整合性が構築されることになる。

3.2.2　システムアーキテクチャの構築活動の手法・手順

「システムアーキテクチャの構築」は、アーキテクチャ分析とアーキテクチャ設計という２つの大きな活動から構成される。

3.2.2.1　アーキテクチャ分析の活動

アーキテクチャ分析はトレードオフスタディと呼ばれることもある活動である。この活動によって、システムアーキテクチャ設計のコンセプトが構築される。

なお、現実には完全なアーキテクチャ分析が実行されることは少なく、既存アーキテクチャを使用するときにはその一部が省略されることも多い。

主要なシステム機能を明確化する

アーキテクチャ分析の活動は、基本的にシステム機能分析を中心として行われる。そして、概念的にはシステムの主要なシステム機能は、副次的な機能サブセットや論理的なシステム構成要素で構成されているため、これらを洗い出す必要がある。

機能サブセット発見の手がかりとして以下のような例が考えられる。これらが最初に考慮すべき大粒度の論理的なアーキテクチャ要素の候補となる。

- 機能や構造が密接に凝集している
- 単一アーキテクチャコンポーネントによって実現される
- 既存コンポーネント（HW/SW）の再利用によって実現される
- システム内で再利用される場合がある
- 特定の設計制約に対応している

これらを用いて、ブラックボックスを、システムの目的を実現できるよう、論理的なアーキテクチャ要素へ、あるいは、物理的なアーキテクチャ要素へ分解することになる。

ところで、通常、システムの目的を実現可能な解の候補として、複数のアーキテクチャ要素を想定可能であることが多い。これらの要素は、ハードウェアやソフトウェア要素であったり、あるいはその両方を含む大粒度のものであったりする。

そこで、その候補の中から合理的に最適な要素の選択を行うために、選択の基準に従った選択を行うが、通常は、相対的な重要性に基づいて重み付けされた選択の基準（有効性の指標（Measures of Effectiveness: MoEs）など）を定義し、それに基づいて選択を行うことが多い。

これらに基づき、主要な要求の集大成として、システムアーキテクチャ設計コンセプトが生成される。

3.2.2.2 アーキテクチャ設計の活動

この活動は、ブラックボックスとして定義されていたシステムを、必要な関心事が割り当てられたサブシステムに分解し、最終的にはシステムアーキテクチャ構成を得るためのものである。

この活動は、アーキテクチャ分析から得られたシステムアーキテクチャ設計コンセプトに基づいて、システムを論理サブシステム、物理サブシステムに分解することから開始する。

この活動は、以下のようなワークフローで実行されることが多い。

システムレベルの操作をサブシステムに割り当てる

このサブシステムは、前述のアーキテクチャ分析から得られたもの、あるいは既存アーキテクチャの要素である。

品質要求や安全性要求など、要求分析活動で捕捉されたシステムレベルで解くべき課題について、様々な割り当てを行うことになる。

割り当て活動は、反復プロセスであり、通常はドメインエキスパートと協力して行う。

一つの操作を単一に割り当てることができない場合は、操作をサブ操作に分解し、複数のサブシステムに割り当てる場合もある。

フォールトトレランス要求を満たすためにアーキテクチャの冗長性が必要な場合などは、同じ操作を異なったサブシステムに割り当てる場合もある。

複数のサブシステムにまたがった割り当てには、インターフェースが存在することになる。このタイミングで個々の通信チャネルへの負荷見積もりを行うことができる。

品質要求を関連パーツに割り当て、トレーサビリティを確立する

システムの機能要求を満たすようなシステム操作がシステムサブシステムに割り当てられた後、品質要求（安全性のための設計制約や性能要求など）が関連するアーキテクチャ要素を割り当てる、これによって機能要求と品質要求のトレーサビリティが確立される。

機能サブセットをサブシステムに割り当てる

前述の、従来からシステム機能を明確にするために、機能フロー図や、状態モード分析などが使用され、機能サブセットへの分解が行われてきた。

ここで、システム機能を実現できるように機能サブセット（機能フロー図における機能ブロックに相当）をサブシステムに割り当てる。

サブシステム間インターフェースを定義する

機能サブセットのフローをもとに、サブシステム間のトポロジーとインターフェースを明確にする。

インターフェースは、ソフトウェア的なサービスインターフェースだけでなく、物理的なものも定義する。

その結果、システム内部のサブシステム構造に関するアーキテクチャが構築されることになる。

各サブシステムについて分析する

サブシステム間インターフェースを定義した結果として生じる各サブシステムの状態遷移の定義を行う。

システムアーキテクチャを検証する

システムアーキテクチャの正確性や完全性の検証は、まず、機能要求の側面の検証を行い、その後に品質要求の検証を行う。

また、安全性要求の観点から、故障モードとその影響の解析（Failure Modes Effects Analysis: FMEA）やミッション・クリティカリティ分析などを行う。

成果物を作成する

システムズエンジニアリングにおいて利害関係者の合意を得るための文書の作成は必須である。記述される範囲と内容はプロジェクトの特性によって異なるが、通常、システムアーキテクチャ仕様書や補足文書としての要求

文書（HWやSW要求仕様書など）が作成される。

特筆すべき要求文書としてICD：インターフェース管理分書（Interface Control Document）がある。

ICDには要素間の物理的、機能的なインターフェースのみでなく、それぞれの要素を担当する組織間の作業や責任所掌のインターフェースなども記述されることもある。通常ICDは単独の文書ではなく、外観図、インターフェース項目表、電気的情報、CADモデルなど多くの図や表を含む複数の文書から構成されるパッケージ構成となっている。

4 モデルベースシステムズエンジニアリング（MBSE）によるシステム設計活動について

MBSEは自然言語記述だけでなく、便利さのためにモデル記述言語による記述を併用するシステムズエンジニアリングである。本節ではモデル記述言語SysMLを用いたMBSEによるシステム設計活動について簡単に述べる。

4.1 なぜモデル化言語も併用するのか？

これまで、基本的なシステムズエンジニアリングにおけるシステム設計活動について解説してきたが、これらの活動の主目的はMBSEにおいても同様である。

しかし、これまでのシステムズエンジニアリングには２つの「不便」があった。

一つは自然言語そのものの記述力を原因とする不便である。

例えば、自然言語でシステムズエンジニアリング領域で多用される階層構造や並行動作を同時に記述することは困難である。また、入り組んだ情報の俯瞰や、複数の原因と結果を持つ因果関係の説明をすることも難しい。SysMLはシステムズエンジニアリングのために策定された準形式的図式言語であり、これらの不便を解消しやすい。

もう一つの不便は、記述内容の一貫性の維持という問題である。

自然言語で記述された文書間をまたがるような情報の完全性や一貫性の維持は困難であるし、文書に記載された情報が変更されたときのシステムのライフサイクルへの影響の見通しも困難である。

これらについてもモデリングツールを併用しながらSysMLを使用することでこれらの不便を解消できる。

4.2 SysMLのダイアグラム

MBSEにおいては要求、構造、振る舞い、パラメトリック制約という４つの観点を4 Pillarsと呼んでいる。これを表現するためのSysML図はそれぞれ要求モデル、構造モデル、振る舞い/機能モデル、制約モデルと呼ばれる。

以下、それぞれにごく簡単に紹介する。

要求モデル：要求図

要求図は要求の分類や、要求同士の包含、派生、因果などの関係をビジュアル化できる。

要求図の主なモデル要素は要求を表現した要求ブロック、依存関係などである。

機能安全規格準拠のためのトレーサビリティ表現に最近よく使用されている。

構造モデル：ブロック定義図
（BDD: Block Definition Diagram）

ブロック定義図は、システムの基本構造要素（ブロック）、及びそれらの全体-部分関係や依存関係などを示す。

主なモデル要素は、構造単位であるシステムブロック、コンポジット関連など。

システムブロックは、ハードウェア、ソフトウェアなどに限らず、関心事一般に使用できる。

全体-部分の階層構造の表現にとくに多く使用される。

構造モデル：内部ブロック図
（IBD: Internal Block Diagram）

内部ブロック図は、BDDで定義された全体を表すブロックを構成する部分（パート）をどう結合して全体システムをどう実現するか、すなわちサブシステム同士がシステム内部でどう内部接続されているかを示す。

主なモデル要素は、パート、ポート、コネクタなど。

パートは、コネクタを介して他のポートと接続される。

ポートは、パート間の相互作用ポイントである。サービス呼び出しや、入出力するデータや物理的実体（液体、固体、気体、エネルギーなど）など、相互作用の種類がポートごとに決まっている。

このような明確に定義されたインターフェースを使用するポートをシステム要素として指定することで、再利用可能モジュールとしてのブロックの設計が容易になる。

振る舞い/機能モデル：シーケンス図

シーケンス図は、ユースケース図やアクティビティ図で示された要求に対し、アクターやブロックがどのような順序で相互作用しながら振る舞うかをビジュアル化する。

主なモデル要素はライフライン、メッセージ。

ユースケースのシナリオを詳細化するためによく使用されている。

振る舞い/機能モデル：アクティビティ図

アクティビティ図は、機能の実行フローをコネクタによって結合された複数のアクションに分解することで、ユースケースのワークフロー、ビジネスプロセス、アル

ゴリズムなどを記述する。

主なモデル要素はアクション、コントロールフロー、決定ノード、スイムレーンなど。

アクティビティ図は、単純なアクティビティも、条件付き分岐や並行性を使用した、複雑なアクティビティも記述できる。

アクションはグループ化され、個々のサブシステムに割り当てることができる。この場合、個々の責任範囲を示すスイムレーンを用い、アクティビティ図を分割する。

振る舞い/機能モデル：ステートマシン図

ステートマシン図は、ブロックの状態ベースの振る舞いを記述する。

主なモデル要素は状態、遷移、イベント、アクションなど。

振る舞いの階層構造や並行性を記述可能である。

アクティビティ図（機能フロー）とシーケンス図（環境との相互作用）の両方の情報を集約し、それをブロックにおけるイベント駆動の振る舞いとして定義できる。

ステートマシン図は、遷移線によって結合された状態によって構成されている。イベントは、ある状態から別の状態への遷移のトリガーとなる。アクションは、状態遷移時や状態への入場時、退場時に実行される。

振る舞い/機能モデル：ユースケース図

ユースケース図は、システムのユーザとシステム自体との間の相互作用を記述することで、システムの機能要求を捕捉する。

主なモデル要素は、ユースケース、アクター、システム境界など。

対象システムのスコーピングに使用される。

ユースケース図においてシステムはブラックボックスとして扱う。

制約モデル：パラメトリック図

パラメトリック図は、システムプロパティ間のパラメトリック関係を視覚化するための、特殊なタイプの内部ブロック図である。

主なモデル要素は制約ブロックやコネクタなどである。

技術品質測定やトレードオフスタディーのためによく使用される。

4.3 SysMLを用いたシステム設計活動

ここでは、MBSEで用いられるSysMLの特徴的な使用方法について簡単に述べる。

4.3.1 要求分析時

前述のように、要求分析の作業は２つの側面を持つ。MBSEでは、それぞれの側面を表現するため、それに適した図を用いる。

システムをブラックボックス的に定義する

MBSEでは対象システムとシステム外部との間のシステム境界を明確にするためにユースケース図を使用し、対象システムのスコーピングを行う。

システムをホワイトボックス的に定義する

システム機能を明確にするため、シーケンス図、アクティビティ図、ステートマシン図などの振る舞い図を使用し、システムレベルユースケースの振る舞いを定義する。これは機能分析にも使用可能である。

4.3.2 システムアーキテクチャの構築時

システムアーキテクチャの構築活動においては、モデルを用いた様々な割り当てを行う。グラフィカルなモデル要素同士を様々な方法でリンクすることによって表現する。最も視覚的な方法は《allocate》ステレオタイプのついた破線矢印で２つのモデル要素を結ぶことである。以下にMBSEで使用することの多い割り当ての例を挙げる。

要求の割り当てについては、要求ブロックを割り当ての単位とする。要求図は要求ブロックと設計要素である論理ブロックや物理ブロックとの実現関係を示すことが可能である。

また、要求が満たされたことを検証するためのテストケースとの検証関係なども表現可能である。

振る舞いと構造の割り当てについては、振る舞いモデルの要素（アクティビティ図のモデル要素であるアクティビティやアクション）を構造モデルのサブシステム要素（ブロック定義図のブロック）に割り当てることで機能割り当てを表現する。割り当てられたモデル要素ブロックやパートは、振る舞いを実行する責務があるという意味となる。

また構造要素間のフローはアクティビティ図でアクティビティ間を接続するオブジェクトフローや内部ブロック図におけるアイテムフローで表現される。

構造同士の割り当ては、論理-物理割り当てと言われる、論理アーキテクチャの論理ブロック階層と物理アーキテクチャの物理ブロック階層を対応付けるときによく使用される。同様にHW-SW割り当てにも使用される。（ソフトウェアのより詳細な割り当てにはUMLの配置図を使うことのほうが多い）

プロパティの割り当ては、とくに性能や物理プロパティのような値プロパティ（システムの重量や部品コストなど）をブロックのプロパティに割り当てる。

パラメトリック図を使い詳細化することも多い。

システムレベルの操作をアーキテクチャ構造の要素へ割り当てたい場合、もし、一つの操作を単一ブロックに割り当てることができない場合は、操作を分解する必要がある。

その際は、ユースケースレベルのホワイトボックスアクティビティ図を作成し、アクションを、それぞれ分解階層のブロックを表すスイムレーン内に分配する。

ユースケースレベルのホワイトボックスアクティビティ図は、開発イテレーションによるアーキテクチャ分解を反映し、再帰的に、より詳細になる場合がある。

そして、アクティビティ図におけるスイムレーンをまたがったリンクはインターフェースに相当するため、ホワイトボックスアクティビティ図を使用して、個々の通信チャネルへの負荷について最初の見積もりを行うこともできる。

一方、システムレベルの一つの操作を異なった複数のブロックに割り当てなくてはならない場合もある（フォールトトレランス要求を満たすためのアーキテクチャ冗長性など）。

4.3.3　アーキテクチャ分析の活動時

システムアーキテクチャ設計のコンセプトを構築するためには主要なシステム機能の明確化を行わねばならない。

MBSEにおいて主要なシステム機能は、ユースケース図で捕捉し、機能サブセットへの分解は前述のようにアクティビティ図を用い、操作をBDDで分解したアーキテクチャ構造の要素であるシステムブロックへ割り当てる。この割り当ては論理ブロックに対しても、物理ブロックに対しても行うことができる。

この作業は、非常に容易な上、理解性にも富んでいる。

また、その際、もし複数のアーキテクチャ要素が割り当て選択の候補に上がっている場合、選択の基準である有効性の指標（MoE）を定義、計算する際に、パラメトリック図を用いると便利である。

重み付け目標計算の結果、決定したソリューションをもとにしたシステムアーキテクチャ設計コンセプトには、ここまで使用してきたSysMLの各モデルが含まれる。

4.3.4　アーキテクチャ設計の活動時

この活動ではアーキテクチャ分析から得られ、SysMLモデルで記述されたシステムアーキテクチャ設計コンセプトに基づいて、システムを論理サブシステム、物理サブシステムに分解し、システムアーキテクチャ仕様書を作成する。

システムレベルの操作をサブシステムに割り当てる

ユースケースレベルのホワイトボックスアクティビティ図を作成し、BDDを用いて分解したシステムブロックごとに、アクティビティ図のスイムレーンを作成、サブアクションを各スイムレーンに割り当てることで実行の責務を割り当てる。

要求分析活動で捕捉された、品質要求や安全性要求などのシステムレベルで解くべき課題について、様々な割り当てを行うことになる。

割り当て活動は、反復プロセスであり、通常はドメインエキスパートと協力して行う。

一つの操作を単一ブロックに割り当てることができない場合は、操作をサブ操作に分解し、複数のサブシステムに割り当てる場合もある。

フォールトトレランス要求を満たすためにアーキテクチャの冗長性が必要な場合などは、同じ操作を異なったサブシステムに割り当てる場合もある。

複数のサブシステムにまたがった割り当てには、インターフェースが存在することになる。このタイミングで個々の通信チャネルへの負荷見積もりを行うこともできる。

品質要求を関連パーツに割り当て、トレーサビリティを確立する

システムの機能要求を満たすようなアクションがシステムサブシステムブロックに割り当てられた後、要求図上で品質要求を表す要求ブロックとサブシステムブロックを《satisfy》依存関係で割り当てる、これによって機能要求と品質要求のトレーサビリティが確立される。

そして、アクティビティ図におけるスイムレーンをまたがったリンクはインターフェースに相当するため、パラメトリック図を用い、サブシステム間通信の負荷についての見積もりを行う。一度、性能評価のモデルを構築しておけば、将来、振る舞いや構造、あるいは操作などの変更依頼への対応が容易になる。

機能サブセットをサブシステムに割り当てる

従来の機能フロー図における機能ブロックはアクティビティ図におけるアクション相当、状態モード分析は、階層化されたステートマシン図相当とみなせる。

サブシステム間インターフェースを定義する

IBDを用い、サブシステムパート間のトポロジーとインターフェースを明確にする。

ポートを使用することで、サービスインターフェースだけでなく、物理的な入出力も定義可能となる。

その結果、システム内部のサブシステム構造に関するアーキテクチャが構築される。

各サブシステムについての分析

ステートマシン図を用い、サブシステム間インターフェースから通知されるイベントをトリガーとした各サブシステムブロックの状態遷移の定義を行う。

システムアーキテクチャの検証

様々な抽象度におけるブロックのステートマシンを構築することで、検証項目のヌケモレを回避できる可能性が向上する。

成果物の作成

MBSEにおいても従来のシステムズエンジニアリング同様、通常、システムアーキテクチャ仕様書や補足文書としての要求文書（HWやSW要求仕様書など）が作成される。

ただし、これら文書中には理解性、明瞭性などの観点からSysML図が含まれている。

通常、SysMLモデリングツールを使用してSysML図を記述した場合は、後工程で特定要求のトレーサビリティや詳細の確認、トレードオフスタディの再検証などの際に構築したモデルが必要になることも多いため、ツールで記述した全モデルレポジトリのデータを補助的成果物として扱うことも多い。

5 おわりに

本稿では、従来のシステムズエンジニアリングにおける上流工程である要求分析活動とシステムアーキテクチャの構築活動について紹介した。また、近年話題に上ることの多いSysMLを利用したMBSEの上流工程で、実際にモデルをどのように使用しているかについても簡単に述べた。

今回紹介したのは、要求を起点とした古典的システム開発、言い換えればQCDを守って成功裏に開発するためにアーキテクチャを構築するためのシステムズエンジニアリングのごく一部にすぎない。

基本的にMBSEを含むシステムズエンジニアリングは、主にそのシステムの目的を達成するためのシステム機能を入力とし、様々な観点を持つアーキテクチャを構築することが「上流工程」となる。

また、MBSEにおいては観点ごとのビルディングブロック（要求ブロック、システムブロック、アクティビティなど）を用いた図式表現を行ったり、再帰的なシステム-サブシステム構造を多用したりすることで、効果的なアーキテクチャ表現が可能となる。

ところで、最近の巨大で複雑なシステム開発プログラム/プロジェクトにおいて、そもそも多岐にわたる膨大な要求にどう対応すべきか不明となり、十分な成功を収められないことが多くなってきた。

そして、巨大で複雑なシステムは個々人の要求を満たすという目的より、ビジネス戦略を成功裏に遂行するという目的が重要と考えられる。

この目的達成のためには、ビジネスプロジェクト開始時からビジネス戦略だけでなく、ビジネス戦略遂行の仕掛けとしてのシステムアーキテクチャを同時にマネジメントしなければならない。これをアーキテクチャファーストの考え方という。

そして、アーキテクチャファーストに基づき、ビジネス、運用、情報、アプリケーション、制御などといった異なる機能ドメインからの視点を持つシステムアーキテクチャを構築するためのアーキテクチャがアーキテクチャフレームワークである。

現在IoTに対する関心がますます高まってきている。ここで最も懸念されているのは、異なるハードウェアやソフトウェアであっても、IoTサービスやデバイス間の連携が可能な相互運用性と言われている。そしてアーキテクチャ不在で、プロプライエタリなセンサデータ形式とサービスインターフェースだけではIoT業界は協調不能なサイロを作り続けるだけなのではないか、という声も聞こえてくる。

一方、Industrie 4.0においても、2030年の実現に向け、サプライチェーン、生産の効率化にデジタルモデルを活用することの重要性を強調しているが、このリファレンスアーキテクチャもアーキテクチャフレームワークそのものである。

これらの新潮流の考え方は、明らかにアーキテクチャファーストであり、強くアーキテクチャフレームワークを意識している。これら領域での事業を計画されている諸氏は、ぜひMBSEを含むシステムズエンジニアリングを再確認していただければと思う。

自動車のパワーバックドアシステム開発のためのモデルベースシステムズエンジニアリングの適用※1

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　西村　秀和
日産自動車株式会社　中本　貴之　日産自動車株式会社　宮下　真哲

1　はじめに

自動車の電気／電子アーキテクチャ（以下、E/E（Electrical/Electronic）アーキテクチャ）では、ユーザからの要求の多様化に対応するため、追加開発を余儀なくされている。この追加開発にモデルベースシステムズエンジニアリング（以下、MBSE）を適用することによる効果を明確にするため、本編では、アクセス・盗難防止システムの一つであるパワーバックドアシステムの追加開発を事例に取り上げる。MBSEに基づきSysML（Systems Modeling Language）を利用してパワーバックドアシステムのシステムモデルを記述しアーキテクチャを構築するまでのプロセスを示すと共に、部分として考えるのではなく、システムとして考えることによる技術者の気づきや明らかになった点を示す。とくに、SysMLを用いたシステムモデルの記述により、従来の“部品ありき”の設計スタイルでは見出すことのできなかった、組織の枠にとらわれない“対象とするシステム全体”としての最適化の検討が行えたことは大きなメリットとして認識された。また、従来の開発では、成果物をどのような形でどのように残していくかの判断やその質が個人のスキルに依存していたが、MBSEのプロセスでSysMLを用いることにより、文書として残すべき成果物と、関連するシステムモデルを明確にできることの優位性が評価された。ここでは、既存のパワーバックドアシステムの解析から、「操作性の良さ」と「おもてなし※2」に関するユーザニーズを引き出し、ユーザの自然な動作によって操作可能な新しいパワーバックドアシステムを設計する。新しい機能のコンポーネントへの割り当てをSysMLダイアグラム上で検討する中で、新・旧パワーバックドアシステム間での幾つかのコンポーネントの統合を図った設計を試みている。

2　概要

ボディ、シャーシ、パワートレインなどから構成される自動車は、近年電子化が急速に進んでいる。この電子化を支えるECU（Electronic Control Unit）の数は、1台当たり50とも100とも言われている。これらのECU間の様々な情報や信号のやり取りを行うために、複数のECUをネットワークで構成するE/Eアーキテクチャが必要とされている。また、様々なユーザニーズの変化に応じて新たな機能を追加することが求められ、その度にECUの増加を招き、E/Eアーキテクチャはますます複雑化してしまう。こうした機能の追加に伴って実施する追加開発では、意図しない手戻りなど様々な問題を引き起こし、業務上の効率を落とすことにつながる。

既存のE/Eアーキテクチャに対してシステムの追加開発が求められた際には、既存システムのどこを活かし、何を追加し、改良すれば良いのかを明確にする必要がある。このためには、システムズエンジニアリング[1][2]を活用し、対象とするシステムを明確にすると共に、追加するべき機能との関係性を明らかにしなければならない。また、追加開発のもととなる要求が、設計上どのように関係しているのかが不明確となっていることも効率の低下を招く。サブシステムやコンポーネントを担当する技術者にとっては、追加開発のもとになるシステム全体を見

脚注

※1　ダイアグラム作成協力　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　ユンソンギル氏
※2　利用者に対して歓迎する、または何かの準備を事前にしていることを示すこと

渡すことができないために、ほかのサブシステムやコンポーネントとどのような関係性を持つのかが分からないということが大きな問題となっている。

システムズエンジニアリングでは、関係部署をまたぐ技術者間のコミュニケーションが重要となるが、必ずしも、こうした観点でのアプローチが系統的に行われていない。MBSE[1]では、システムモデルを記述することにより、関係部署間でこれを共有し、コミュニケーションを取ることで問題に対処することができる。欧米の航空宇宙産業、自動車産業、メディカル産業などでは、MBSEをサポートするためにSysML[3][4]が近年注目されているが、日本での導入は欧米に後れを取っている。もちろん、当然ながら、SysMLをツールとして導入すればそれで業務の効率化が図れるというものではない。MBSE及びそれをサポートするSysMLを導入することによる効果と、その際の課題を明らかにする必要があると考える[5]。

本編では、既存のE/Eアーキテクチャの中でアクセス・盗難防止システムの一つであるパワーバックドアシステムに対して、ユーザニーズとしての“ユーザフレンドリー（操作性の良さ）”や“おもてなし”などの新たな要求に基づく機能を追加する際に、MBSEを適用して検討した事例を示す。この事例では、システムモデルを記述するためにSysMLを初めて企業へ導入する際の、技術者の気づきやそこから明らかになった点を示している。具体的には、SysMLを用いてパワーバックドアシステムのアーキテクチャを構築するためのシステムズエンジニアリングプロセスを示す。まず、ボディ制御モジュールと非接触センサから構成される既存のパワーバックドアシステムのE/Eアーキテクチャの解析を行う。そして、「操作性の良さ」と「おもてなし」を求めるユーザニーズを考慮し、ユーザの自然な動作によって操作可能な新しいパワーバックドアシステムを設計する。新しい機能のコンポーネントへの割り当てを行う中で、新・旧パワーバックドアシステム間での幾つかのコンポーネントの統合を検討する。

3 システムモデルを活用したシステムズエンジニアリングプロセス

システムズエンジニアリングでモデルを用いることのメリットは何か？

システムズエンジニアリングに関する国際協議会（INCOSE、International Council on Systems Engineering）では、システムズエンジニアリングに関し、

「システムを成功裏に実現するための複数の分野にまたがるアプローチ及び手段」

「システムズエンジニアリングでは、開発ステージの初期の段階で顧客のニーズを明確化し、要求される機能性、システム要求を定義し、関連する問題をすべて考慮しながら設計のための総合とシステムの妥当性確認を進める。」

「システムズエンジニアリングでは、ユーザニーズに合致した品質の製品を供給することを目的とし、ビジネスとすべての顧客の技術的要求を考慮する。」

と定義している。

MBSEでは、システムモデルを記述することにより、関係部署間でこれを共有し、システムライフサイクル[2]全般にわたる検討のもと、コミュニケーションを正しく取ることで、様々な問題に対処できる。モデル化することのメリットをまとめると、次のようになる。

- 仕様書など文書だけではすぐに理解できないことが、図的に表現することで理解が容易になる。
- 複数の分野にまたがり協働してシステムを開発するには、共通言語が必要であり、それをサポートするには図的な言語が有効である。
- モデルを再利用することにより開発の効率化が期待できる。
- モデルを用いて抽象度を上げることにより革新に導く。

そして、MBSEでシステムをモデルで記述する言語としてSysMLがある。図１に示すように、SysMLダイアグラムは振る舞い図、要求図、構造図の３つに分類することができる[3][4]。振る舞い図には、ユースケース図、シーケンス図、アクティビティ図、状態機械図があり、構造図にはブロック定義図、パラメトリック図、内部ブロック図、パッケージ図がある。

ユースケース図は、目的を達成するために開発するべきシステムが外部システムの中でどのように用いられているかを機能性として表す。シーケンス図は、内部システムと外部システムの間、あるいはシステム内部のパート間でやり取りされるメッセージの順序を表す。このメッセージのやり取りがブロック間の相互作用を表すことに注意されたい。アクティビティ図は、入力、出力、及び

制御を用いたアクションの順序付けと、アクションによる入出力間の変換によって振る舞いを表す。状態機械図は、イベントによって引き起こされる状態間の遷移に関するエンティティの振る舞いを表す。

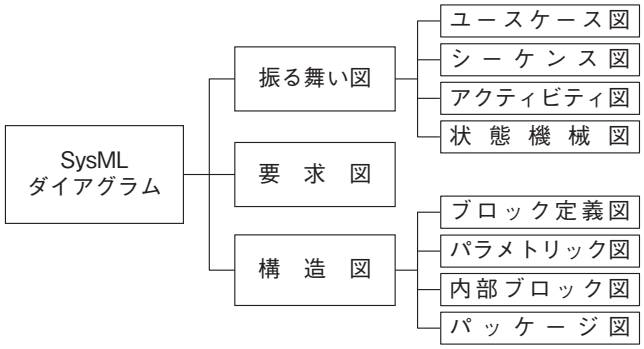


図1　SysMLダイアグラムの分類

要求図は、テキストベースの要求とそれらに関連するほかの要求、設計要素、テストケースとの関係を表し、もとの要求から機能要求、性能要求、制約、要求される品質特性やそれらを実現する構成要素などの間のトレーサビリティをサポートすることができる。

構造図の一つであるブロック定義図は、ブロックと呼ばれる構造的要素間の関係性を明確に記述することができる図で、これによりブロック間のインターフェースが明確となり、機能アーキテクチャ、物理アーキテクチャを表すことができる。パラメトリック図は、ブロックのプロパティに制約を与える方程式、例えば$F=ma$による属性値に関する制約を表すことができ、エンジニアリング解析をサポートすることができる。内部ブロック図は、ブロックのパート間の相互接続とインターフェースを表す。パッケージ図は、モデル要素を含むパッケージに関してモデルの編成を表すことができる。

これらのSysMLダイアグラムを用いることで、MBSEを円滑に進めることができる。まず、コンテキストレベルでの開発するべきシステムの運用シナリオを明確にするため、対象システムが外部システム（アクター）との関係性の中でどのように用いられるかをユースケース図で記述する。そして、シーケンス図を用いて機能性を表すユースケースを記述することで、対象とするシステムが外部システムに対して持つべき機能を明確にすることができる[3] [4]。

次に、システムが持つ機能を更に分析及び分解する。ブロック定義図で仮にシステムを分解してサブシステム（システム構成要素）を定め、このサブシステム間の相互作用をシーケンス図で検討する。これにより、機能分解を行いながら、システム構成要素を検討する。更に、アクティビティ図で機能の分解を進めつつ、システム構成要素への割り当てを明確にしていく。機能のシステム構成要素への割り当てに際しては、性能の割り当ても検討する必要があり、その際には、パラメトリック図をもとにしたシミュレーションなどのシステム解析が必要となる。また、状態機械図により、システムの状態遷移を確認する。このようにして、システムの振る舞いと構造が明確化され、アーキテクチャの候補が明らかになる。更に、システム要求の優先順位をもとに、制約や要求される品質特性、性能要求などのバランスを取るアーキテクチャを候補の中から選定する。

以降では、ここで示したSysMLを用いたMBSEのアプローチに従ってパワーバックドアシステムの開発を行っていく。

4　パワーバックドアシステムのコンテキスト分析

4.1　現行パワーバックドアシステムの分析

最初に、図2に示す現行のパワーバックドアシステムについて分析を行う。現状の問題点を明らかにし、そこから要求を明確に導くためである。現行パワーバックドアシステムを、以降で議論する新しいパワーバックドアシステムと区別するため、非接触パワーバックドアシステム「Contactless Power Back Door System」（以下、CPBDS）と呼ぶ。

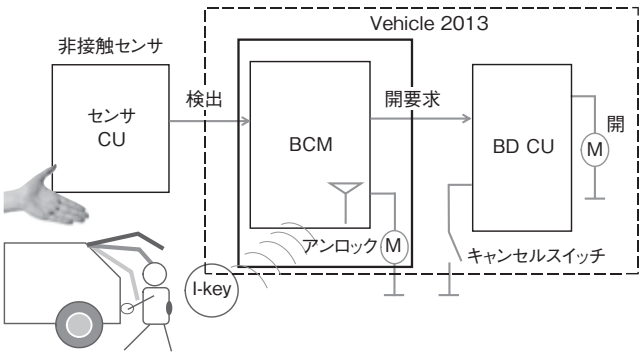


図2　非接触パワーバックドアシステム（CPBDS）

図2に示すように、CPBDSは、認証機能を備えたキーフォブ（以下、I-key）を携帯するユーザの手を非接触セン

サで検出することによってバックドアを開くことができる。CPBDSには、ボディ制御モジュール（BCM）、バックドア制御ユニット（BD CU）及びセンサ制御ユニット（センサCU）が含まれる。点線の長方形で囲んだ「Vehicle 2013」は、BCMとBD CUで構成され、新しいパワーバックドアシステムでは、既存のコンポーネントとして機能することを想定している。CPBDSでは、I-keyを携帯するユーザが手に荷物を持ったまま、バックドアを開けたいとき、ユーザが手をセンサの前に差し出すような操作をしなければバックドアを開くことができない。これは必ずしも容易なことではなく、ユーザにとって操作性の良いものではない。こうした検討から、「操作性の良さ」に関する要求が導かれる。更に、図３のような絵を用いた検討から、I-keyを携帯するユーザを自動車が「おもてなし」するといった機能が求められていることに気づいた。本編では、「操作性の良さ」と「おもてなし」の観点で新パワーバックドアシステムがユーザから望まれているものと仮定して展開していく。

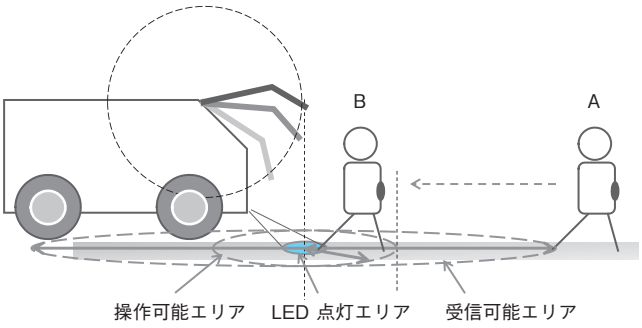


図３　ユースケースシナリオ“ドアを開く”

4.2 「ワンステップ・バックドアシステム」のコンテキスト分析

最初に、4.1で述べた「操作性の良さ」と「おもてなし」のユーザニーズに基づく新しいパワーバックドアシステムの要求を整理するため、対象とするシステムの利用場面を様々な観点から検討していく。図３のような絵を多用し、利用場面の想定を行った。その結果、以下の４つの要求を導くことができた。

① I-keyを携帯するユーザは、手を使わずにバックドアを開閉できること

② バックドアの開閉は、ワン・アクションの自然な動作で行えること

③ バックドアの開閉の際に手や服を汚すことのないこと

④ パワーバックドアシステムは、ユーザへの「おもてなし」の姿勢を示すこと

これらの元の要求は「操作性の良さ」と「おもてなし」のカテゴリーに分類することができる。「操作性の良さ」には、上記の①、②、③が含まれる。一方、「おもてなし」は、上記の④が関係し、「バックドアに近づくI-keyを携帯するユーザに対して、自動車が歓迎する意思をユーザに示すこと」を導いた。これらの検討結果から、この新しいパワーバックドアシステムを「ワンステップ・バックドアシステム（One-Step Back Door System）」（以下、OSBDS）と名付けた。

コンテキスト分析では、まず、外部システムとの関係性の中で、OSBDSはどのように用いられるか、あるいはどのように動作するかを考えることになる。そこで、「バックドアを開ける」場合と「バックドアを閉める」場合の両方のユースケースについて考察した。ユースケース「バックドアを開ける」のシナリオ場面を図３に示し、そのユースケースを記述したシーケンス図を図３に示す。ただし、図３や図４を得るために、対象システムの利用を想定した簡易なプロトタイピングなどにより反復的な議論を行った。

ユースケースシナリオを記述した図３に示す通り、地点AにいるI-keyを携帯するユーザが自車に近づくと、システムはI-keyを検出し、LEDを地面に反射させて「おもてなし」の意思を示し、ユーザを誘導する。更にB地点までユーザが近づくと、LEDを点滅させて、ユーザを更に自車に近寄らせる。ユーザが点滅するLEDに足をかざすと、ソナーが足を検出し、バックドアが開くということを示している。図４は、この一連の振る舞いをOSBDSと外部システム間の相互作用を表すシーケンス図で記述している。図４のシーケンス図に示されるOSBDSの生存線※3上にあるメッセージのやり取りから、OSBDSの機能が導出される。

なお、これらの一連の検討の中で、「ユーザは、バックドアの動作領域から退くことなく、バックドアの背面からバックドアを開閉できること」という要求を導いた。これは、バックドア開閉時の安全性に関連することである

脚注

※3　四角で表されるブロックから下方につながる点線を生存線（lifeline）と呼び、そのブロックの生存期間を表す。

が、ユーザが荷物を抱えたままバックドアを開けるための操作をし、バックドアが開く際に後ずさりをしなければならないようでは操作性が良いとは言えない。「おもてなし」という観点からもこの新たな要求を導出しており、非接触センサの配置に関連する要求となる。

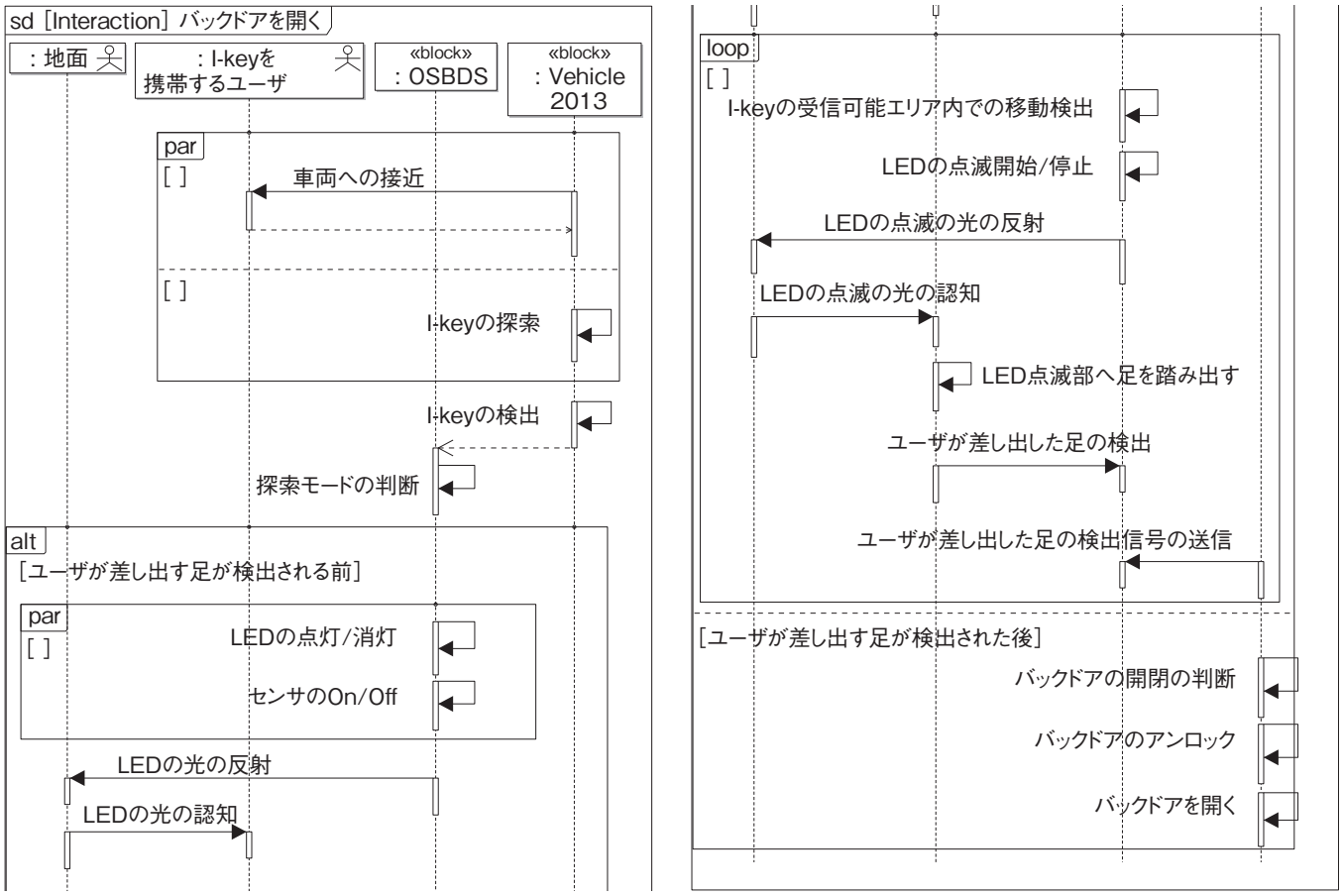


図4　ユースケース“バックドアを開く”を記述したシーケンス図

5　システムズモデリングのための機能分析と統合

5.1　機能分析と機能のコンポーネントへの割り当て

「4. パワーバックドアシステムのコンテキスト分析」で導出された機能を実現するためにシステムOSBDSを構成するコンポーネントを検討する。まず、OSBDSがLED、I-key移動検出器（以下、移動検出器）、足を検出するための非接触センサ（以下、非接触センサ）、及びバックドア制御ユニット（以下、BD2020 CU）から構成されているものと仮定する。また、非接触センサの候補としては、例えば、赤外線センサ、静電容量センサ、レーザー・レーダ、ソナーシステムなどがあり、これらのどれを採用するか検討する必要がある。図5のブロック定義図には、OSBDSが4つのコンポーネントから構成され、その中の非接触センサについては、代替案として4つあることを表している。

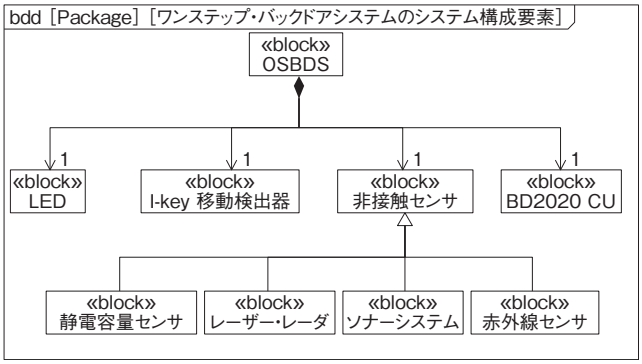


図5　ワンステップ・バックドアシステムのシステム構成要素を表すブロック定義図

各コンポーネントに要求される機能を導出するため、図4から得られた各機能をシーケンス図で記述する。図6は、

「I-keyの受信可能エリア内での移動検出」について記述したシーケンス図である。

図６の「I-keyの受信可能エリア内での移動検出」のシーケンス図より、移動検出器の機能を導出することができる。

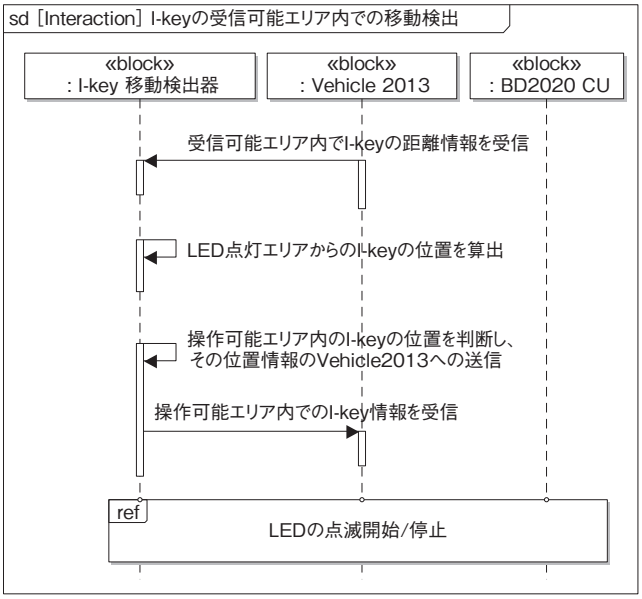


図６　「I-keyの受信可能エリア内での移動検出」を表すシーケンス図

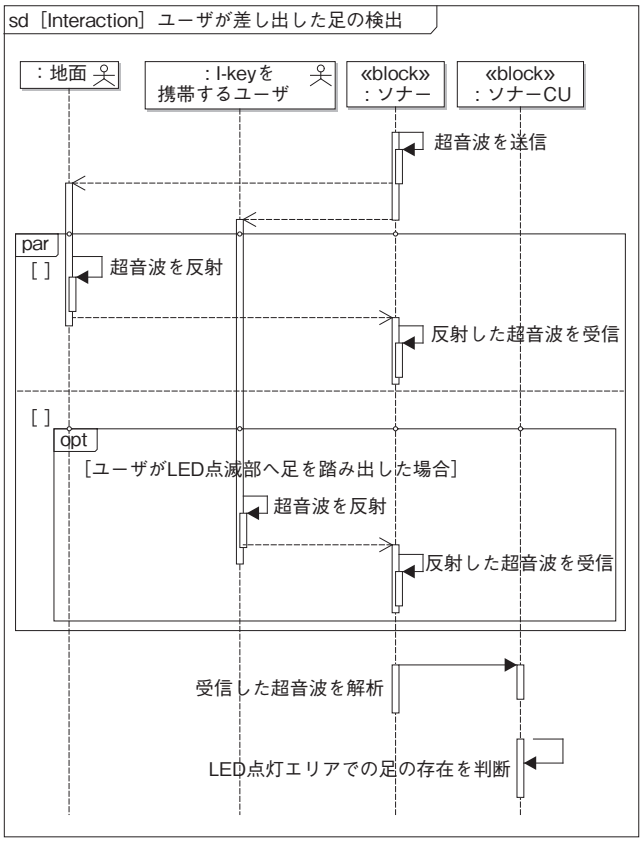


図７　「ユーザが差し出した足の検出」を表すシーケンス図

同様に「ユーザが差し出した足の検出」に関してもシーケンス図で詳細化した結果を図７に示す。ここでは、非接触センサの候補の中からソナー及びソナー CUからなるソナーシステムが選択されたものとして、シーケンス図を記述している。ソナーシステムを選択した理由については、5.2で述べる。図７より、ソナーシステムの機能を導出することができる。

このような機能分析により、各コンポーネントが持つべきすべての機能を明らかにすることができる。また、地面、I-keyを携帯するユーザ、及び既存のバックドアシステムVehicle 2013で定義された外部システムを含めて、コンポーネント間のインターフェースを図８のブロック定義図にまとめることができる[6]。図８から「移動検出器」の機能の実現には、「Vehicle 2013」が関連していることが分かる。このことから「移動検出器」と「Vehicle 2013」との統合について検討する余地があると考えることができる。なお、BD2020 CUの機能「ユーザが差し出した足の検出信号の送信」の実現には、送信先であるVehicle 2013が関連するが、「移動検出器」と「Vehicle 2013」の統合にこの機能は影響しないと考えられる。図８のブロック定義図では、Vehicle 2013はアクターとして定義しているため、BD2020 CUからの足の検出信号を受信し、バックドアの開閉の判断にこれを利用していることを明記していない。

次に、アクティビティ図により、各コンポーネントの振る舞いを検討しながら確認する。Vehicle 2013の機能も含めて検討した結果として得られたアクティビティ図を図９に示す。これによりOSBDSがどのように機能するかについての全体のビューを捉えることができる。また、これらの検討には、状態機械図（以下、ステートマシン図）を確認しながらの作業が有効となる。技術者の多くは通常の業務の中で状態遷移を検討している場合が多く、ステートマシン図や状態遷移表を活用している。

図９のアクティビティ図には、Vehicle 2013は、低頻度探索モードでI-keyを探索し、I-keyが検知されると高頻度探索モードへ移行させる機能を持つことが示されている。また、タイマーにより高頻度探索モードの状態で１分が経過し、その後I-keyを５秒間検出できなかった場合には、低頻度探索モードに戻るといった機能を持たせている。これは停車中の消費電力を抑制する要求を実現する機能である。

またBD2020 CUは、Vehicle 2013から低・高頻度探索モードの情報を受けた後、ソナーCUとの間で、両探索モードへの切替えコマンドの受信機能を二重に持っていることが分かる。これらの信号と振る舞いは同じタイプであり、２つのコンポーネントはいずれも制御ユニットであるため、統合の可能性を見出せる。こうした検討をSysMLダイアグラム上で事前に行うことは、プロトタイプを作成してからの手戻りを防ぐ上で大変効果的である。

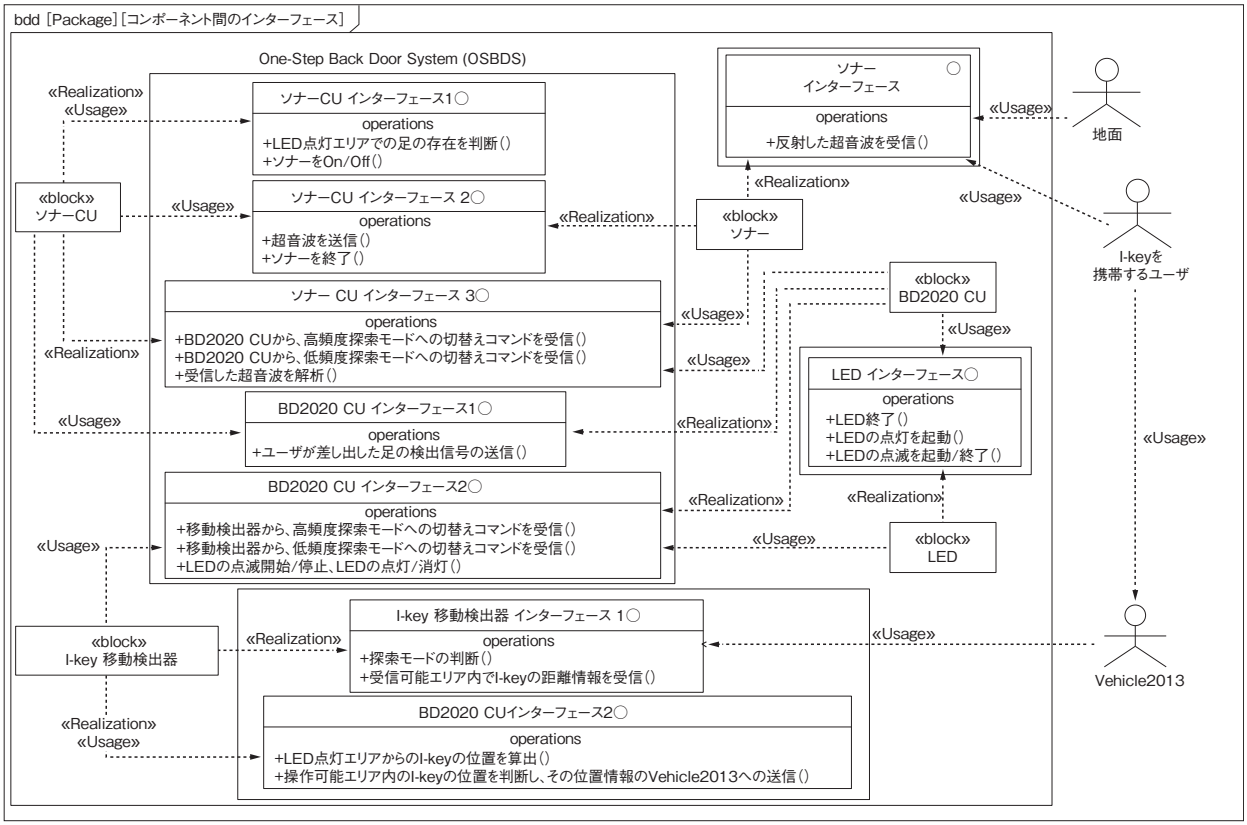


図８　コンポーネント間のインターフェースを表すブロック定義図

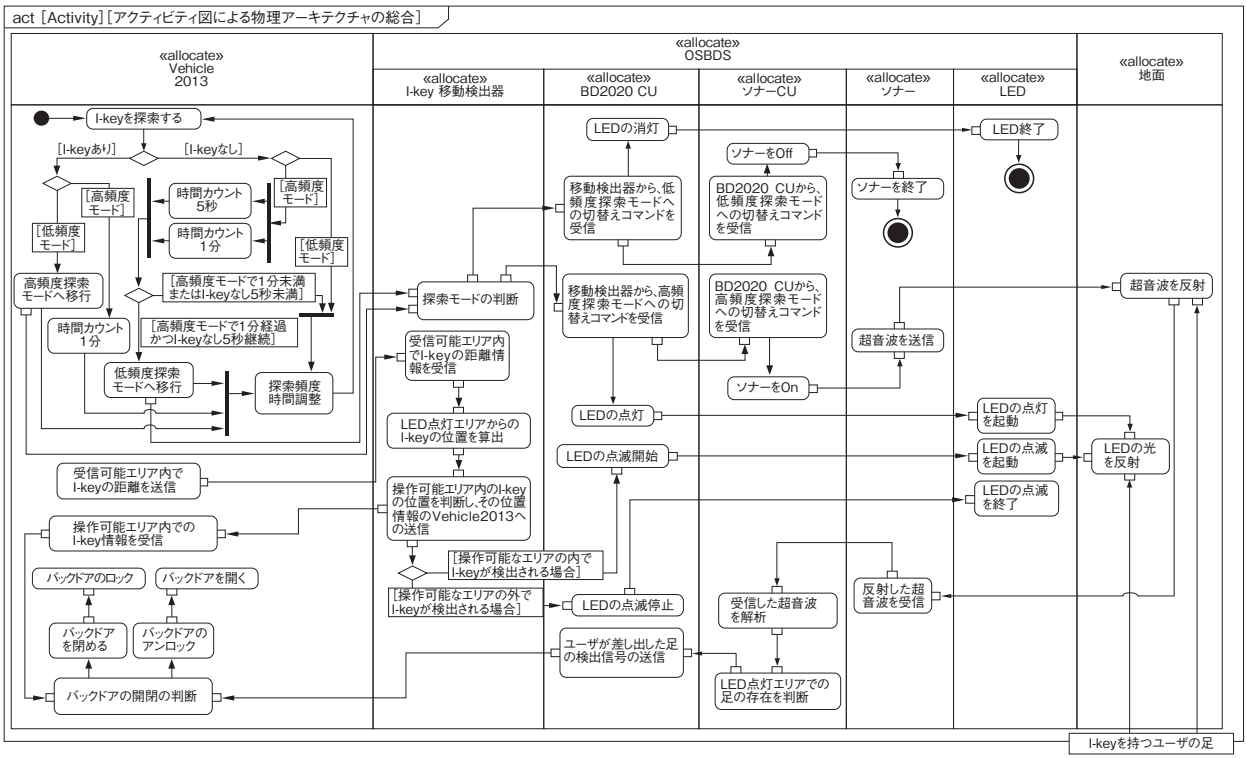


図９　アクティビティ図による物理アーキテクチャの総合

5.2 要求とトレーサビリティの詳細化

前述した通り、各システムコンポーネントの機能は、ユーザニーズである「操作性の良さ」と「おもてなし」の要求を満たすために、振る舞い図を用いて明確に導出される。また、機能のコンポーネントへの割り当てについても議論してきた。要求に関するこのような詳細化の関係性は、図10に示される要求図で表すことができる。最上位の要求は、4.2で述べた４つの要求から構成され、これらの要求は、「操作性の良さ」、「おもてなし」などの幾つかの主要な要求に分解される。「操作性の良さ」は、更に５つの要求に分解され、その機能要求は、「4. パワーバックドアシステムのコンテキスト分析」で示した振る舞い図によって導出される。こうした分析を経て、OSBDSでは、I-keyを携帯するユーザが車に近づいてくるとLEDが点滅し、LEDで示されたエリア内に置かれたユーザの足を検出し、パワーバックドアを開くものと決定することができた。こうした分析の中で、機能要求「センサは、センサと地面の間に位置するユーザの足を検出することができる」が導出される。図５で、ユーザの足を検出するためのセンサとして、赤外線型、静電容量型、レーダ型、ソナー型を想定した。次に足の検出をするためのセンサの選定を検討しておく。

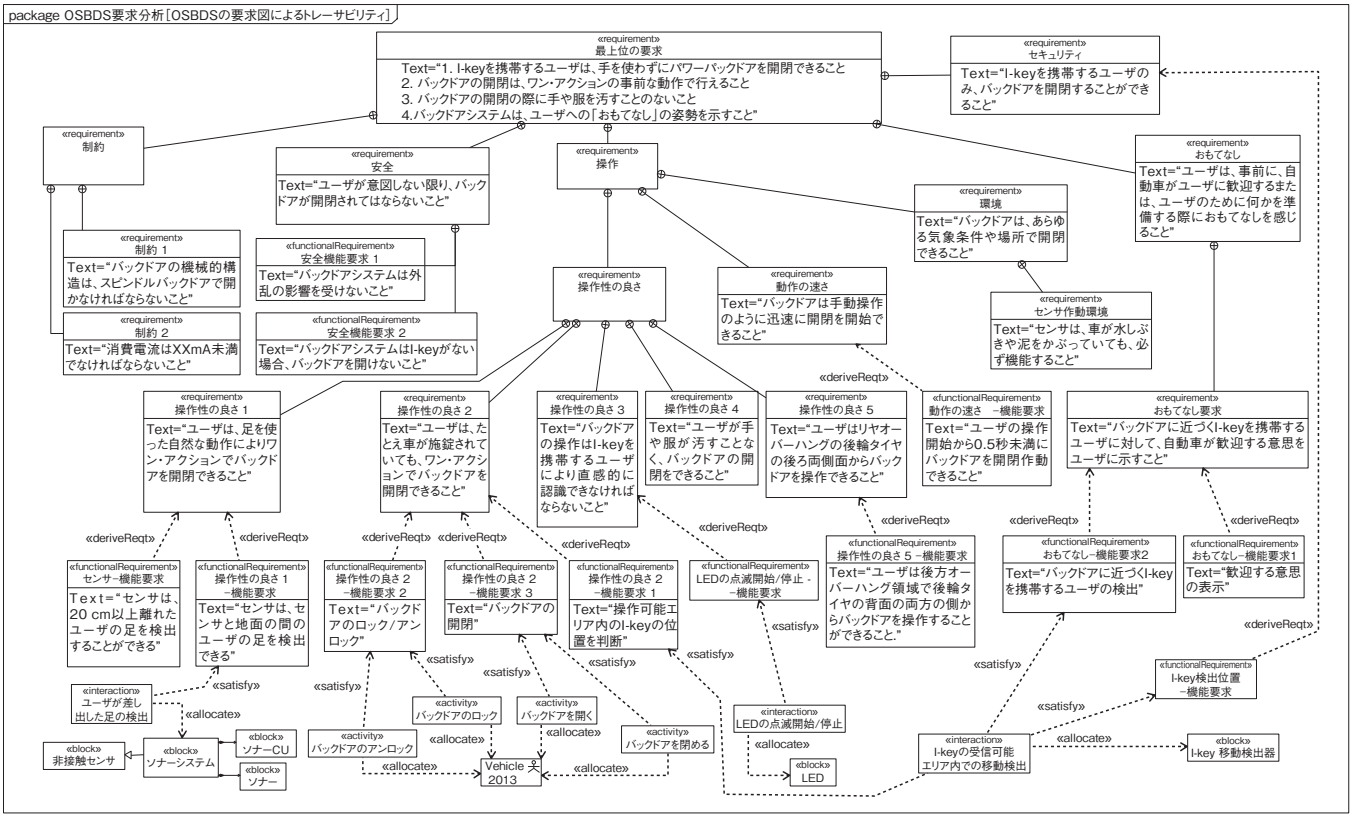


図10　要求図によるトレーサビリティの確保

「操作」に含まれる「環境」に関する要求の中の「センサは、車が水しぶきや泥をかぶっていても、必ず機能する」という要求から、先に導出した1)機能要求に加えて2)性能要求が導出される：

1)機能要求：センサは、センサと地面の間のユーザの足を検出できること。

2)性能要求：センサは、20 cm以上離れたユーザの足を検出できること。

性能要求の「20 cm以上」は、ユーザの手や服が汚れないようにするという要求からきている。センサは、たとえ車が水しぶきや泥で覆われていても機能しなければならないため、「赤外線型」や「レーダ型」は汚れに対して脆弱なため選択肢から除かなければならない。また、「20 cm以上」という性能要求から、静電容量型は、選択肢から外される。このような検討結果から、ソナー型が候補の中から選定される。

上述の議論の結果から、図10に示される要求図を得ることができる。要求図には、元の要求、導出された機能要求、及び機能要求に合致したコンポーネントの間の多

くの関係性が描き出されている。このように要求図には要求のトレーサビリティが明確に示されており、このトレーサビリティによって、構成管理及び変更管理を正しく行うことができる。

6　得られた気づきと明確になった点

SysMLを用いたパワーバックドアシステムのアーキテクチャを構築するシステムズエンジニアリングプロセスから、次の通り気づきを得ることができた。

(1) 要求のトレーサビリティ

すべてのシステムモデル及び情報は要求図に示されている元来の要求にさかのぼることができるため、エンジニアはシステムのすべてのビューを把握することができる。また、要求に関連するシステムモデルはSysMLツールに保存されているため、例えば、ECUを追加して新たな機能を実現する追加開発の際には、要求及びコンポーネント間の関係性を要求図で追いながら検討できる。要求のトレーサビリティを確保しておくことは非常に重要である。

(2) コンテキスト分析におけるシーケンス図による効果

コンテキスト分析で、図３のような絵を描き、またシーケンス図を用いた検討を行う中で、LED光の地面からの反射があることに気づくことができ、ユーザへの「おもてなし」あるいはLEDとしての機能が、地面の状態に依存することを見出せた。また、バックドアの開閉の動作をイメージすることにより、I-keyの検出を行うためのアンテナの配置に関する検討の見逃しを防ぐことができた。非常に初歩的な話ではあるが、アンテナをバックドアに配置してしまうと、バックドアが開いたときに、アンテナ自体もバックドアと一緒に上に上がっていくので、そうならないように、車体側にアンテナを配置する必要があるということに気づきを得た。また、バックドアが閉まっているときと、開いているときでは、電波環境が異なるので、I-key位置検出のチューニングをバックドア開閉状態それぞれで実施する必要があるという気づきを得ている。

また、Vehicle 2013が持つべき機能と新たに追加するOSBDSが持つべき機能を明確にすることができた。すなわち、既に示した通り、移動検出器とVehicle 2013への機能の割り当てやそれらの統合の可能性、及びBD2020 CUとソナー CUの統合の可能性をそれぞれ示唆することができた。SysMLを用いたシステムモデルの記述により、従来の“部品ありき”の設計スタイルでは、見出すことのできなかった組織の枠にとらわれない、対象とするシステム全体としての最適化の検討が行えたことは大きなメリットとして認識された。

(3) 機能の割り当て

アクティビティ図は、機能がどのようにシステムコンポーネントに割り当てられているかを示すことができ、これはシステムに関与するシステムズエンジニアだけではなく、コンポーネントの設計者にとっても、設計するコンポーネントがシステムの中でどのように機能するのかを知ることができるため、有用である。また、技術者がよく活用するステートマシン図や状態遷移表と併せて検討することによって、コンポーネントの機能に漏れがないかを検討することができる。

暗黙知化された振る舞いや機能をアクティビティ図で書き表すことで、機能のコンポーネントへの割り当てやコンポーネント間の統合をシステムとして俯瞰して見ることができるようになる。また、このことにより、既存の担当領域や組織の枠にとらわれないシステムとしての最適化の検討が行えるようになった。コンポーネントに依存する形で分割された組織では、機能の割り当てやコンポーネント間の統合を検討する際に、技術者自身の組織内のコンポーネントのみに影響がとどまり、自身が属する組織外のコンポーネントに対して影響が出ないように設計をしがちであるが、そうした思考の枠にとらわれずに全体を俯瞰することで、実プロジェクトにおけるQCDの最適化が図れる可能性がある。こうした検討がシステム開発の早い段階で行えることは、試作やプロトタイプを減らすことができ、意図しない手戻りの減少に大いに効果があるものと期待される。

(4) システムズエンジニアリングとSysMLの導入

従来の開発では、成果物をどのような形でどのように残していくかの判断やその質が個人のスキルに依存していたが、MBSEのプロセスでSysMLを用いることにより、文書として残すべき成果物と、関連するシステムモデル

を明確にできることの優位性が評価された。ただし、企業における製品の開発をモデルベースに移行させ、システムモデルの記述言語としてSysMLを導入するには、幾つかの乗り越えるべき壁がある。企業での開発業務を行う組織では、システムレベルでエンジニアリング活動を行うための組織がそもそも存在しない場合が多い。システムレベルでエンジニアリング活動を行うために組織を編成すると共に、SysMLを導入するためのコスト、エンジニアをトレーニングするためのコストなどの負担にどのように対処するべきかが大きな課題となる。また、追加開発も含んだエンジニアリングプロセスの中での活用方法を検討しなければならない。

7 結論

本編では、SysMLを用いてパワーバックドアシステムのアーキテクチャを構築するシステムズエンジニアリングプロセスを示した。ECUを追加して新機能を実現しようとすることに起因するE/Eアーキテクチャ設計の複雑さの課題を解決するため、MBSEの導入を検討した。パワーバックドアシステムに「操作性の良さ」と「おもてなし」に関する新たな要求を追加する際のモデルベースシステムズエンジニアリングによる設計プロセスを示した。具体的にはユースケース図とシーケンス図を用いて、新しいパワーバックドアシステムの要求を分析した。コンテキストレベルで、対象とするシステムと外部システムとの関係性を明確に示すことにより、システムが持つべき機能性を明確にすることができた。更に追加開発のもととなる要求と、詳細化されていく機能要求や性能要求とのトレースが取れる形で要求図に明確にすることができた。全体のシステムを構成するそれぞれのコンポーネントに機能を割り当てる際にはアクティビティ図での検討を行い、併せて状態機械図、シーケンス図を用いた。アクティビティ図とブロック定義図を用いてコンポーネント間の相互作用を確認することができ、幾つかのコンポーネントを統合することの可能性が示された。

システム解析、トレードオフ分析をサポートするパラメトリック図を用いたアーキテクチャの選択についてはここでは示さなかったが、システムモデルを記述することで、システムレベルでの議論を経てどのようなシミュレーションを実施する必要があるのかを明確にすることができる。システムモデルにより、技術者は、システム全体を見渡すことができ、担当するサブシステムやコンポーネントがほかのサブシステムやコンポーネントとどのような関係を持つのかが明確に把握できる。システム開発の初期段階で、コンポーネントやサブシステムの統合に関連する有用な気づきが与えられることで、意図しない手戻りを減少させる効果は大きい。

参考文献

[1] Systems Engineering Handbook, A Guide for System Life Cycle Process and Activities, Fourth Edition, International Council on Systems Engineering, 2015
[2] INTERNATIONAL STANDARD, ISO/IEC/IEEE 15288:2015, First edition, 2015-05-15
System and software engineering - System life cycle processes
[3] Sanford Friedenthal, Alan Moore, Rick Steiner, A Practical Guide to SysML, The System Modeling Language, Third Edition, The MK/OMG Press 2014
[4] 西村秀和：監訳、システムズモデリング言語SysML、東京電機大学出版局、2012
[5] モデルベースシステムズエンジニアリング導入の手引き、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、2013
[6] Laurent Balmelli, An Overview of the Systems Modeling Language for Products and Systems Development, The Journal of Object Technology, Vol. 6, No. 6, pp.149-177, 2007
[7] Chi Lin, Dave Nichols, The Application of MBSE at JPL Through the Life Cycle, INCOSE IW 2014, http://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:06-iw14-mbse_workshop-application_of_mbse_at_jpl_through_the_lifecycle-nichols-lin-final.pdf
[8] 先進的な設計・検証技術の適用事例報告書
http://www.ipa.go.jp/sec/reports/20151118.html

未来のイノベーションを牽引するシステムズエンジニアリング
German Industrie 4.0の事例から

Fraunhofer IESE Dr. Jens Heidrich※1　Fraunhofer IESE Dr. Martin Becker※2
Fraunhofer IESE Dr. Thomas Kuhn※3　Fraunhofer IESE Dr. Thomas Kleinberger※4
Fraunhofer IESE Dr. Markus Damm※5　Bosch Rexroth Anne Duell※6

1 要約

デジタル化に向けた一般的な傾向により、システム開発に関する様々な課題に取り組む必要が生じている。自動運転、Industrie 4.0（ドイツが推進している製造業の高度化）、IoT（Internet of Things：モノのインターネット）、ビッグデータでは、ソフトウェアがイノベーションを牽引している。そのことから、ハードウェア及びソフトウェア開発プロセスを一体として理解することが効率的なシステムズエンジニアリングの基本的な前提となっている。このようなソフトウェア・エンジニアリングからシステムズエンジニアリングへの移行というトレンドが、従来の領域を超えて統合されている。本稿では、今後の製品トレンドに対応するためにはなぜ、システムズエンジニアリングが重要なのかを説明する。そしてFraunhofer IESEによるシステムズエンジニアリングについての調査結果を詳細に検討し、システムズエンジニアリングの実践において企業が直面している課題、及びそれに対処するために使用している技法、手段、ツールなどに関するベストプラクティスを明らかにする。最後に、Industrie 4.0のための共通プラットフォーム開発を目指すドイツの実施プロジェクトの具体的な例を紹介し、いかにシステムズエンジニアリングの持つ機能性がイノベーティブなシステムソリューションの実現に役立つかを示す。

2 はじめに

すべての適用業務領域においてデジタル化が進んでいる。製造分野では、IoS（Internet of Services：サービスのインターネット）と、「スマート」オブジェクト（マシンや製品）を含むIoTを組み合わせることでパラダイムシフトが起き、Industrie 4.0として知られている第４次産業革命と認識されている。ネットワーク接続され、サイバー環境で相互に連携する物理的な製品の統合によって、イノベーションが期待できるため、今後は製品の製造方法だけでなく、製品そのものや関連するビジネス・モデルも変化することになるだろう。

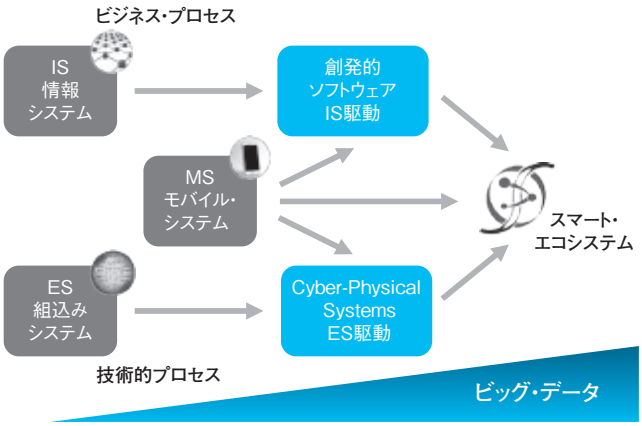


図１　スマート・エコシステムに向けたトレンド

スマート製品のライフサイクル全体を通してこの種のアプローチや技術の導入が増えているため、ハードウェア中心の開発や製造プロセスを適用している企業は、拡大するデジタル市場で生き残るために、ビジネス戦略の再考と、組織能力強化が必要になっている。一方、ソフトウェア開発やソフトウェアプロビジョニング分野に強い企業にとっては、これは大きな可能性を意味している。従来

※1　ドイツ フラウンホーファー研究機構 実験的ソフトウェア工学研究所（IESE）にて、プロセスマネジメント領域の部門長を務めている。システムとソフトウェア・プロセスの改善についての研究・教育を行っている。
※2　同研究所組込みシステム・エンジニアリング部門責任者。組込みシステムアーキテクチャ、モデルベースのアプローチ、バリアント管理についての研究を行っている。
※3　同研究所組込みソフトウェア部門責任者。組込みソフトウェア・アーキテクチャと仮想エンジニアリングについての研究を行っている。
※4　同研究所組込みシステム部門のプロジェクト・マネージャ。システム要求エンジニアリングについて研究している。
※5　同研究所組込みソフトウェア部門のプロジェクト・マネージャ。組込みソフトウェア・アーキテクチャと仮想エンジニアリングについて研究している。
※6　ボッシュ・レックスロス社において、コンピューター技術者としてテクニカル・ソリューション アーキテクト兼システムエンジニアの任にあり、製造IoTでのコンセプト開発とユースケース実装を担当している。

からあるITやソフトウェア企業は、製品ポートフォリオを拡大して市場に参入し、従来のソフトウェアとサービス製品を物理的な製品と組み合わせるようになるだろう。

2.1 スマート・エコシステムに向けたトレンド

最終的には、Industrie 4.0によって、現在ほとんどが閉ざされた環境にある製品やシステムは、高度に統合されたSoS（systems of systems）に移行することが予想されている。これはスマート・エコシステムと呼ばれ、組込みシステム、情報システム、モバイルシステムが融合し、ランタイム時に新しいサービスを生成するようになる（図1参照）。この市場への新規参入も増えるだろう。提供する製品やサービスの互換性と柔軟性が、主なビジネス推進要因となる。

スマート・エコシステムに主要な情報システムを統合し、組込みシステムによって異なる組織の技術目標を達成することで、ビジネス目標を達成する。このようなシステム統合を推進するトレンドは、自動車業界（Car-2-X通信やSmart Mobilityなど）、エネルギー業界（スマート・グリッドやSmart Energyなど）、医療技術（Smart Health）、農業技術（Smart Farming）、その他多くの分野で見ることができる。

2.2 システムズエンジニアリングの必要性増加

相互接続とネットワーキングの増加は、イノベーションの主要な要因で、成功持続のためにも重要である。潜在的なイノベーションを推進するためには、組織の境界を超えて、ハードウェア（機械、電気、電子）とソフトウェアのエンジニアが統合したチームとなり、緊密に連携していくことである。

ただし現実的には大きな課題もある。エンジニアリング領域ごとに異なるエンジニアリングアプローチを適用しており、それぞれがばらばらに機能するからである。更に、将来的なシステムには典型的な特性があり、適切なシステムズエンジニアリングの実践によってこれに対処しなければならない。(a)今後、システムの複雑性は引き続き高まるにつれ、たとえば、テキストによる記述ではなく、モデルベースのエンジニアリングアプローチの適用、適切なシステム要求のエンジニアリング、柔軟性の高いスケーラブルなアーキテクチャ、成熟したシステム開発プロセスなどが必要になる。(b)将来的なシステムは多種多様なシステムと、異なる企業や様々な領域のステークホルダから構成され、これらを統合する必要がある。これには、例えば、統合を簡単にする相互に運用可能なアーキテクチャ、インターフェースの標準化、QoS（Quality of Service：サービスの品質）保証が必要になる。(c)将来的なシステムは、環境の変化や新しいステークホルダに対応できなければならない。そして彼らとの付き合い方も、時間の経過と共に変化する可能性がある。そのため、例えば、順応性の高いシステム、運用時に所定の品質（システムパフォーマンス、機能上の安全性、セキュリティ、プライバシーなど）を認定する機能、シミュレーションと仮想開発アプローチによって、開発時間とランタイムの連携を密にする必要がある。(d)非常に重要な組込みシステムを機密情報システムに統合する場合、結果的に構成されたシステムは、機能安全とセキュリティ上の問題に同時に対応しなければならない。このような対応が行われていない場合は、セキュリティ障害によって安全上の問題が発生する。そのため、例えば、セキュリティと機能安全に同時に対応する統合モデルが必要になる。(e)複雑性が急激に高まる中、システムの可用性を維持する必要がある。明確なユーザエクスペリエンスを保証するためには、システムとユーザインタラクションの統合戦略が必要になる。(f)スマートデータを使用することで、将来的なシステムは自動または半自動で機能するようになる。これには、例えば、高度な人工知能と、各システムの順応性が必要になる。(g)将来的なシステムのインテリジェンスは、異なる情報源から正しいデータに接続し、適切に分析して、モデルを構築することによって得られる。例えばそのためには、一方で確かなレベルの品質のデータを収集しながら、他方では個人のプライバシーを保護するための強力なメカニズムを導入する必要がある。

専門的なアプリケーション領域内のシステムズエンジニアリングコミュニティは、各分野で総合的なアプローチを推進する有望なアプローチ、技法、ツールを提供している。この種のアプローチを導入することで、開発する製品やサービスだけでなく、対応する開発プロジェクトや組織全体にも対応できる。従ってシステムズエンジニアリングは、将来的な製品開発のイネーブラとみなすことができる。

3 システムズエンジニアリングのベストプラクティスに関する研究

システムズエンジニアリングに関する様々なトピックについて、各種文献が提供されている。例えば、参考文献[1]（以下[1]と表記）はモデルベース・システムズエンジニアリング（MBSE）技法を中心に書かれたもので、企業が使用している主なアプローチに関する調査結果も示している。[2]では、プログラムマネジメントとシステムズエンジニアリングの統合について調査している。3,000人のシステムエンジニアと5,000人のプログラム・マネージャを対象に、組織内におけるこの2分野の統合状況について調査した。[3]では、33人のエキスパートへインタビューを行い、システムズエンジニアリング産業界での実践について調査している。システムズエンジニアリングの持つ能力と活用度合について確認することが、この調査の目的としている。[4]では、企業のシステムズエンジニアリング能力とプロジェクトパフォーマンスの関連性を示す証拠を得るため、システムズエンジニアリングによる効果を調査している。

ここで参考文献について言及する。本調査はFraunhofer IESEに対しIPA/SECから委託研究として実施したものである。

2016年６月から８月にかけて実施された[5]の調査は、ドイツ・欧州企業におけるシステムズエンジニアリングの実施状況について調査したものである。これに基づき、ベスト・プラクティスの観点から課題とソリューションアプローチについて分析している。最終的には18の企業に属する20人へインタビューを実施できた。このうち６つの組織／部門は中小企業、14は大規模な組織と分類された。この調査では複数のドメイン（業種や分野）を対象としており、単一のドメインに特化したものではなく、また大企業だけでなく中小企業も対象としている。この調査はベストプラクティスに関する情報を収集する目的で行ったため、過去のコラボレーション経験から、システムズエンジニアリング分野に積極的な企業に参加を募った。ドイツ・欧州すべての企業の状況を総合的に把握するための調査ではないが、調査結果から、システムズエンジニアリングに積極的な企業が直面している課題や、ベストプラクティスとして確立した分野を知ることができる。

3.1　今後の製品とシステムズエンジニアリング

（1）　企業はシステム要件の複雑化（60%が指摘）と、顧客の要望に応えるために、製品バリエーション（多様性）がかつてないほど増加していること（50%が指摘）に直面しており、更に商品化のスピードアップ（55%）も加わることで、現在のシステムズエンジニアリングに大きな負荷がかかっている。

（2）　全体の85%以上の企業が、ソフトウェアが自社製品において大きな役割を担っていると回答している。回答者の約70%は、もともとはハードウェア開発出身である。更に、85%の企業が開発予算の30%以上（最も多い回答は90%）をソフトウェア開発に費やしていると回答した。

（3）　システムズエンジニアリングの重要度合を１（重要ではない）から10（企業存続に不可欠）で評価した結果、平均は7.6だった。そして今後５年の間に8.7に増加する見込みという回答であった。ほとんどの企業は、重要性が高まった理由として、顧客が更なる高品質な製品を求めるようになったことと、製品の複雑性が上がっていることを同時に挙げている。

3.2　エンジニアリングにおける課題

（4）　80%の回答者が、変化に対応できる組織改革が最大の課題だと答えており、その次に複雑な要件とインターフェースの管理を挙げている（SoS：system of systemsの場合は特に）。

（5）　適用しているプロセスモデルに関しては、中小企業と大規模組織合わせて45%以上が、アジャイル・モデルを採用していると回答している。一方、大規模組織の50%以上では、ウォーターフォール・モデル、または反復型のウォーターフォール・モデルが採用されている。

（6）　企業規模に関係なく、あらゆる組織でシステムズエンジニアリングプロセスには多くの部門や該当するステークホルダが多数関わっていると回答している。「システムエンジニア」の明確な役割は、大規模組織でのみ定義されている。また、60%～70%の企業が合同チームを確立し、ワークショップやミーティングを一緒に開催して調整を行っている。

（7）　60%近くの組織は、製品部品を社外から調達している割合が25%未満である。それでもなお、３分の１の組織では社外調達率が最大50%に上っている。しかしながら社外で調達したコンポーネントの知的財産の重要度合は、平均的にかなり低い。

3.3　エンジニアリング・ソリューションのアプローチ

（8）　既に確立されているプラクティスの中で（図２参照）、50%以上の企業が挙げた方法、技法、アプローチは、モデル駆動開発、要求開発、テスト駆動開発、システム検証と妥当性確認に関連していた。１つ以上の組織が言及したその他のプラクティスには、統合ツールチェーン、仮想開発、そして全体論的システムアーキテクチャが含まれている。大規模組織はモデル駆動開発（60%が選択）と検証と妥当性確認（80%が選択）が中心だが、中小企業の約80%が最も確立された実践としてテスト駆動開発を選択した。

（9）　回答者の大半が、システムズエンジニアリングの実施で影響を最も受けているのが、技術及びソフトウェアの実装エンジニアリング・プロセス分野（ISO/IEC 15288及び12207）だと回答した。

（10）　80%以上の回答者が、UMLを主な仕様言語として挙げた。大規模組織は、SysMLをシステム・モデリング用の言語として使用する傾向がある。更に、ドメインに特化した言語についても広く言及している。挙げられたシステムズエンジニアリングツールの50%以上は、システム全体、またはソフトウェアの異なる部分をモデリングするためのものだった。40%近くは、近い将来、非形式的なテキストによる仕様の代わりに、形式手法とモデルベースのシステム開発アプローチを導入する予定だと回答した。

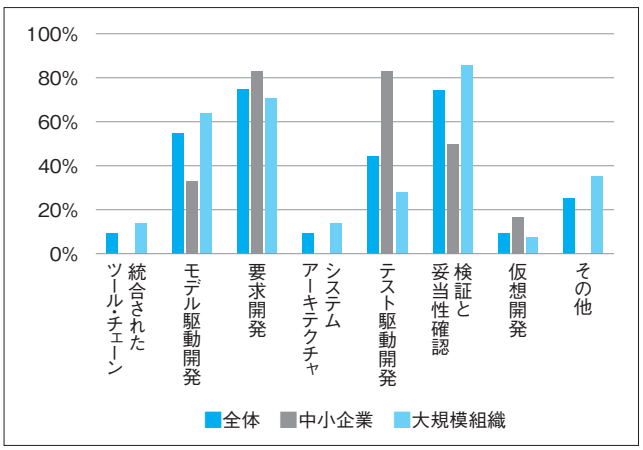


図２　確立されたシステムズエンジニアリングの実践

3.4 将来的な方向性と能力構築

(11) システムズエンジニアリングにおいて最も改善が期待されるものは、仮想開発と、使用するツールチェーンの統合強化で、回答者の50%がこの２つを挙げた。中小企業のほうがこれらに対するニーズが高いようである。大規模組織の40%近くは、プログラムマネジメント(プロジェクトポートフォリオマネジメント)の改善も挙げていた。

(12) 大半の企業が社内外のトレーニングプログラム(社内は100%近く、社外も60%以上)を活用して、システムズエンジニアリング能力を高めていることが分かっている。更に、回答者の50%以上が、国際的なシステムズエンジニアリングカンファレンスへの参加を挙げた。

3.5 企業向けのアドバイス

以上12項目の主な調査結果に基づき、システムズエンジニアリングに取り組んでいる組織に対して、いくつかのアドバイスを行うことができる。(A)組織改革については、企業は適切な組織改革戦略を確立して、システムズエンジニアリングの実践方法を導入する必要がある。この戦略は、企業のビジネス及び組織目標と、それら目標達成のためにシステムズエンジニアリングがどのような機会とメリットをもたらすことができるのかによって決定し、推進する必要がある。不足している能力に関しては、システムズエンジニアリング全般と、とくにソフトウェア・エンジニアリング能力が挙げられ、これらの向上が必要なことが分かった。とくに大規模組織は、異なるシステムズエンジニアリングプロジェクトのポートフォリオ管理方法についても検討しておく必要がある。(B)技術的な開発に関して、企業はシステムズエンジニアリングのアプローチとプロセスを開発し、すべてのステークホルダも含めて統合する必要がある。とくに要求開発、モデル駆動開発、そしてシステム検証と妥当性確認の分野において、プラクティスの確立を検討する必要がある。より高度な企業は、正しいモデルに基づき仮想開発の分野で実践を確立し、統合されたシステムズ開発のツールチェーンを提供する必要がある。

4 アプリケーション領域としてのIndustrie 4.0

Industrie 4.0は第４次産業革命である。第３次産業革命では、製造の機械化、大量生産と組み立てライン、そしてコンピュータ化とオートメーションが始まった。第４次産業革命のポイントは、製造システムと製造ラインのネットワーク化である。そうして誕生したスマート製造システムは、システムリソースのオーバーヘッドを抑えて変更することができる。そのため、Industrie 4.0の主な推進要因の１つになっているのが、個別化した製品の大量生産だ。これにより、カスタマイズされた大量生産品を、カスタマイズしていない製品と同じ価格で製造できるようになる。

これを達成するためのIndustrie 4.0環境の主要機能の１つとして、製造変更機能が挙げられる。つまり、事前に予定された変更だけでなく、製造環境における予定外の変更にも確実に対応することができる機能である。Industrie 4.0では、ネットワークに接続されたデバイスによってこのようなニーズに対応している。この種のデバイスは、提供されたサービスを、標準化されたエンティティやオブジェクト・モデルで記述し、インスタンスのプランニングによって、最適な製造ラインに関する決定をすぐに下すことを可能にする。選択した製造ラインは必ずしも物理的に同一の場所になくてもよく、同じ工場や企業にすら属している必要がない。従って、物流業者、工場、企業を網羅したデータの相互のやり取りが必要になる。

これはIndustrie 4.0に関連する次の３つの大きな課題につながる。第１に、データとサービスを標準化された方法で記述する、共通のインターフェースの定義が必要である。Industrie 4.0コミュニティでは、管理シェル(Administration Shell)とデジタル・ツインズ(Digital twins)という用語が登場するが、いずれも製造に必要と判断したエンティティをデジタル化して定義したものである。管理シェル／デジタル・ツインズでは、その名が示す通りで、リアルタイムデータを持つ現実のデバイスと、複数の工場にまたがる製造ジョブのプランニングに必要な仮想モデルの両方にアクセスできなければならない。これは、分散化された製造にはデータ・セキュリティが必要だという、Industrie 4.0の２番目の課題につながる。本番プランニングシステムのプランニング対象が複数の企業の場合、サプライヤや、場合によっては競合他社のデータが必要になる場合がある。このデータの所有権とアクセス権は明確に定義しておき、Industrie 4.0プラットフォームでは、データ所有者が自身のデータを常に制御していることを保証する必要がある。更に、プラットフォームは強力なセキュリティメカニズムを実装し、データの盗難や許可なく変更されることがないようにする必要がある。そして、製造ラインを移動するための技術的な機能が、Industrie 4.0の第３の課題となっている。製造制御コードは現在、特定の製造ラインに合わせて詳細にカスタマイズされている。このラインを変更するには手間と時間がかかり、物理的に同一ではない別なラインへの移動はほぼ不可能だ。理由の１つは、製造制御コードが非常に低いレベルで実行しており、実際のハードウェアをほとんど理解していないという点にある。効率良く製造環境を変更するために、Industrie 4.0プラットフォームは、リアルタイムパフォーマンスと柔軟性は提供しているものの、ハードウェアへの依存度が低い製造制御コードをサポートする必要は残っている。

上記のようなIndustrie 4.0における３つの課題を解決するには、業界内の多くのステークホルダ、研究機関、大学が関与する手間のかかる作業が必要になる。これは難しい課題だが、取り組む価値がある。アナリストは、

2013年から2025年の間に、第４次産業革命によってドイツにおける生産性が23%高まると予測している[6]。

Industrie 4.0におけるドイツのフラグシップ・プロジェクトの１つが、BaSys 4.0プロジェクト[6]である。これは、Fraunhofer IESEがコーディネーションを行い、Industrie 4.0プラットフォーム戦略の一環として、ドイツ連邦教育研究省（German Federal Ministry of Education and Research：BMBF）が出資したプロジェクトである[7]。12の著名な産業及び科学パートナーから構成され、Industrie 4.0の基本システム開発を目指している。そのコンセプトは自動車業界のAUTOSARプラットフォームに似ているが、Industrie 4.0デバイスの基本サービスを提供する。BaSys 4.0の目的は標準と、オープン・ソースのリファレンス・ソフトウェア実装の両方を開発することにある。これによって、中小企業だけでなく大規模組織によるIndustrie 4.0の使用を促すことができる。

5　Industrie 4.0の実践的ユースケース

ボッシュ社（Robert-Bosch GmbH）は、BaSys 4.0開発において重要な産業界パートナーの一社である。提供されたユースケースセットに、多目的製造ラインの一つの可能性として、多品種製造ラインへの改良に焦点を当てたものがある。

大量生産ラインが、相反する少量ロットの製造に対応するためには、多品種製造ラインへの改良に向けた基本的アイデアである「製造での柔軟性をより高めること」が必要である。ボッシュ・レックスロス社（Bosch Rexroth）では、ハンブルグの工場に合計200品種以上になる６つの異なる製品群を組み立てるラインを構築した。そこでは、たとえロットサイズが１個であったとしても対応できる。個々の製造部品はRFID技術を使って、部品自体が通るべき複数の組み立て工程を正しく通るようガイドしながら製造される。それぞれの組み立て工程では、製品タイプ、組み立ての進捗状況、品質情報を問いながら、固有の要件に沿うよう製造していく。もし、とある工程の品質ゲートをパスできなかった場合、その工程は自動的に停止され、パスできるまで改修が要求される。この“ペーパーレス工場”は、MESシステムとERPインターフェースを用いた縦統合によって実現するコンセプトである。

この実装は、最先端の製造の姿を示すと同時に、常に要求が変わるという選択されたアーキテクチャの不都合さを示す。例えば、新しい製品やツール、もしくは製造機器が加わることで、すべてのソフトウェアに対応しなければならない労力、最悪の場合ダウンタイムが発生する。製造工程のオートメーション部分、ツールのインターフェース、MESの作業計画、その他あらゆる更新が必要になる。それらを解消するための大きな要求として、製造にかかわる機器類の“Plug-&-Produce”機能による多目的製造機器の優れた柔軟性の確立が挙げられる。

一つの方向性として、直接の機器間通信がある。それにより製造工程での部分的変更が可能になり、製造上の集中管理の必要もなく、より柔軟な生産量を製造できるようになる。しかしながら、機器間通信に関するデータモデルの標準化の不足と、合意されているPeer-to-peer通信標準の不足により今日に至るまで実現はされていない。

BaSys 4.0において、多品種製造ラインに関するこのユースケースは、製造工程内のすべての機器へのAdministration Shell（BaSys 4.0で仕様化されている実機と仮想空間をつなげるCPS）の実装を示している。このAdministration Shellは、Digital twins技術として知られているIndustrie4.0の主要コンセプトであり、機器間通信を可能にするイネーブラーである。このAdministration Shellは、通信にかかわる機器、オーダー、作業員、製造される製品など、それぞれのエンティティー（実体）の仮想化された代理人である。ある製造部品が、Industrie4.0多品種製造ラインを通過する場合、その部品の仮想化されたDigital twin（双子の片割れ）が、その部品に関するすべての情報を一緒に運んでいく。この情報には、組み立て完了までに必要な製造工程、設計図、CADデータ、その他製造に必要なデータが、この部品の品質データと共に含まれている。このDigital twinは、実部品（物理的）とリンクしており、もしこの部品が新しい製造機器に投入された際、この部品のDigital twinはすぐさま存在し、例えば、その製造機器から直接この部品の品質情報の問い合わせが可能になる。Digital twins技術では、自身で書き込みができ、実体の属性、提供するサービスなどの情報を、機器が判読できる様式で提供される。機器が以前使用された機器と同型かどうか、製造者が同じかどうかなどは、Digital twinsにとっては一切関係なく生成される。

Administration Shellは、何度でも即座に生成される。多品種製造ラインの場合、ある部品を使って製造する予定の作業台は、まだ実製品が他の製造工程で使用されていたとしても、即座にその部品のAdministration Shellにアクセスできるようになっている。Administration Shellのインスタンスはネットワーク化されており、ひとつのインスタンスの属性が変更されても、すべての他のインスタンスに伝達される。よって実製品の流れに沿うように直接機器間通信を可能にしている。

Digital twins技術の実現には、製造ラインの高度なネットワーク化が要求される。機器間の通信には標準化が必要であり、物理的ネットワークのインフラ支援が必要である。Administration Shellを実現するInduetrie4.0の通信プロトコルは、既存通信技術の上に実装される必要があり、かつ合意されたサービス品質の定義に従ったリアルタイム性のある通信をサポートしている必要がある。更に、同時にベンダー非依存のデータモデルでなくてはならない。

Industrie4.0のもう一つの方向性は、製造工程での迅速な変更をサポートすることである。今日の多品種製造ラインでの更新作業は、すべての機器の更新作業とテスト

が完了するまで、製造を一旦止めなければならない。更新後ソフトウェアの不具合が発見された場合、不具合が修正されるまで、以前の状態に戻す必要がある。テスト作業についても、しばしば多品種製造ラインの実作業台で実施されるため、更新作業期間を極端に長くしている。Induetrie4.0では製造工程及び対象となる機器の仮想化によって、この問題に取り組んでいる。Digital twins技術は、単に機器や部品の現状を問い合わせるためだけに使われるのではなく、シミュレーション・モデルともリンクしている。使われているシミュレーション・モデルに従って、Digital twinsは、振る舞いやインターフェースを確認する機能シミュレーション、あるいは物理シミュレーションとして、ロボットの不適切な動きのような物理的欠陥を見つけることを可能にする。このことから、製造工程でのソフトウェア更新のテストに際し、実製造ラインや製造工程に適用する前に、仮想製造ラインで検証できることになる。また製造工程の最適化を仮想化で行えるようになる。

このシステムズエンジニアリングのユースケースでは、次のような異なる要求にも取り組んでいる：

- 製造における機器の仮想化は、製造工場の複雑さを管理する最適化アルゴリズムを持つ自己拡張型工場モデルの策定や、工場スタッフの判断を支援するシミュレーション・モデルを提供するのに役立つ。企業内での仮想化では、データ・セキュリティーや機密保全の考慮の必要性が挙げられている。
- 多品種製造のスケーラビリティーや割り振りを、複雑な製造エコシステム内での容易な実装を可能にするアプリケーション。理想的には、実装の労力をとって代わるソフトウェア・コンポーネントの自動構成である。

6 結論

この記事では、デジタル化やシステム統合の増加といった将来的なトレンドに対応するための、システムズエンジニアリングの重要性に着目した。システム・エンジニアリングは、将来的なシステムで一般的になるだろうシステムの複雑性、多様性、ランタイムにおける不確実性、安全性とセキュリティ脅威の合致、優れたユーザエクスペリエンス、自律機能の強化、そしてデータ駆動性といった特性に対応することができる。システムズエンジニアリングに関する12の主な調査結果に基づき、企業は2つの基本的なアクションが必要であることが分かった。1つ目は、組織改革戦略の確立や、一般的なシステムズエンジニアリング適用能力の強化など、組織変更の必要性だ。2つ目は、統合システムズエンジニアリングプロセスの確立、手法、技法、ツール専用分野の実装など、技術的な変更の必要性だ。具体的には、システム要求開発、モデル駆動開発、システムの検証と妥当性確認などが挙げられる。Industrie 4.0により、デジタル化とシステム統合の増加トレンドが明確になった。これは開発時に設計されるスタティックなソリューションから、ランタイムに自動的に調整及び最適化されるダイナミックなソリューションへの移行を意味している。工場における製造ラインの可変性を示した具体的な使用事例からも分かるように、将来的な製品イノベーションを成功、実現させるためには、システムズエンジニアリングの適用能力とその進化が不可欠だ。

参考文献

[1] J. A. Estefan著、『Survey of Model-Based Systems Engineering（MBSE）Methodologies（モデルベースシステムズエンジニアリング（MBSE）技法の調査）』、INCOSE MBSE Focus Group、2007年。

[2] E. Conforto、M. Rossi、E. Rebentisch、J. Oehmen、M. Pacenza共著、『Survey Report：Improving Integration of Program Management and Systems Engineering（調査報告書：プログラム管理とシステムズエンジニアリングの統合強化）』、PMI及びINCOSE、Philadelphia、2013年。

[3] P. D.-I. J. Gausemeier、R. Dumitrescu、D. Steffen、A. Czaja、O. Wiederkehr、C. Tschirner共著、『Systems Engineering in industrial Practice（インダストリアル・プラクティスにおけるシステムズエンジニアリング）』、Heinz Nixdorf Institute、Fraunhofer Institute for Production Technology、Unity AG、2015年。

[4] J. P. Elm、D. R. Goldenson共著、『The Business Case for Systems Engineering Study：Results of the Systems Engineering Effectiveness Study（システムズエンジニアリングのビジネスケース調査：システムズエンジニアリングの効果に関する調査結果）』、Carnegie Mellon University、Software Engineering Institute、AESS、NDIA、2012年。

[5] J. Heidrich、B. Tanveer、R. van Lengen、T. Kleinberger、L. Gorodilova、T. Kuhn、M. Becker、T. Bauer、A. Morgenstern共著、『Systems Engineering Booklet 2016：Challenges and Best Practices（システムズエンジニアリングブックレット2016：課題とベストプラクティス）』、Fraunhofer IESE、Kaiserslautern、2016年。

[6] S. S. D. M. Wilhelm Bauer著、『Industrie 4.0 – Volkswirtschaftliches Potenzial für Deutschland』、BITKOM、Fraunhofer IAO、Germany、Berlin、Stuttgart、2014年。

[7] 『Base System for Industrie 4.0：Project Page（Industrie 4.0の基本システム：プロジェクト・ページ）』、Fraunhofer IESE、Fraunhofer-Gesellschaft、Germany、［オンライン］。以下に掲載：http://www.basys40.de/。［2016年12月 2 日にアクセス］。

[8] Federal Ministry for Economic Affairs and Energy（BMWiE）及びFederal Ministry of Education and Research（BMBF）、『Plattform Industrie 4.0』、2016年９月23日。［オンライン］。以下に掲載：http://www.plattform- i40.de.

[9] 『Functional Mock-up Interface（FMI）（機能モックアップインターフェース）』、Modelica Association、Sweden、［オンライン］。以下に掲載：https://www.fmi-standard.org/［2016年12月 2 日にアクセス］。

システムズエンジニアリング実践調査の分析結果報告

SEC研究員　杉崎　眞弘

1　はじめに

2016年度上期、IPA/SECがドイツ フラウンホーファ研究機構IESEへの委託調査として、欧州企業20社でのシステムズエンジニアリングの実践状況について調査を行った。

調査目的：ドイツ・欧州の企業における最近のシステムズエンジニアリングの実践状況について、とくにその実践上の課題と解決策（実践的方法論、適用技術、組織的取り組み、将来予測など）を調査・分析し、その有効事例を探ることにある。

調査対象：異なるドメインの企業20社を抽出して、29項目の質問票によるヒアリングを実施した。20社の内訳は図１の通り。（大規模企業14社、中小規模企業 6 社）

調査対象とした企業は、これまでフラウンホーファ研究機構IESEとの共同プロジェクトの実績を持つ企業から、特定ドメインに偏らないよう考慮して抽出された。それら企業の特徴として、もともとハードウェア開発主体企業が多く、近年ビジネスあるいは技術的必要性からソフトウェア開発を取り込んでいった経緯を持っている（図２）。

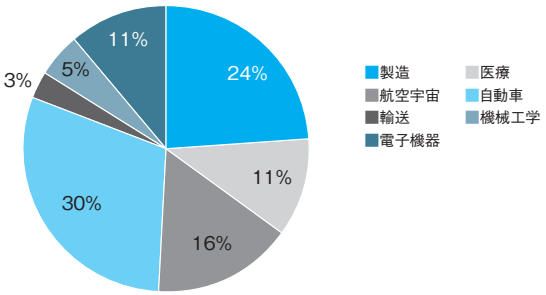


図１　調査対象企業のドメイン分布

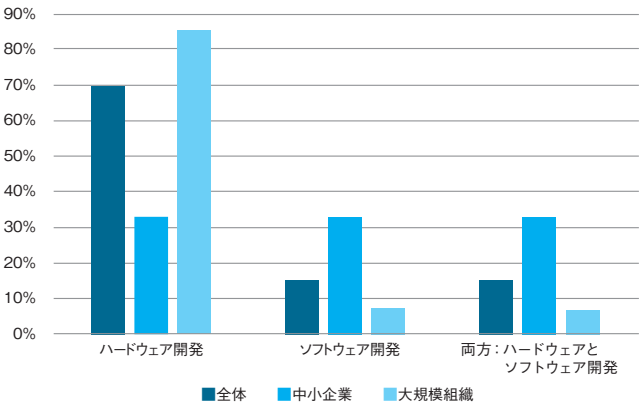


図２　調査対象組織の出身母体

2　質問票に基づく主たる調査・分析結果

2.1　システムズエンジニアリングに転換した動機は?

■システムズエンジニアリングに転換した動機については図３の通り。
ここでは、次のことが上位に並んだ。

- セーフティ・セキュリティ要件（ドメイン要求）を満たすため
- 顧客要求の多様化
- 製品の複雑化
- H/W、S/W、サービスが一体となったソリューションが求められるようになってきたため

■ドイツでは、90年代に自動車業界からシステムズエンジニアリングへの転換が始まった。自動車の制御が機械制御からアナログ制御システムを経て、完全にソフトウェア制御のシステムに転換していった時期と重なる。

■ここ数年（2010 ～ 2015年）で、全産業においてシステムズエンジニアリングが普及していったことが確認されている。（IESE調査報告書より）

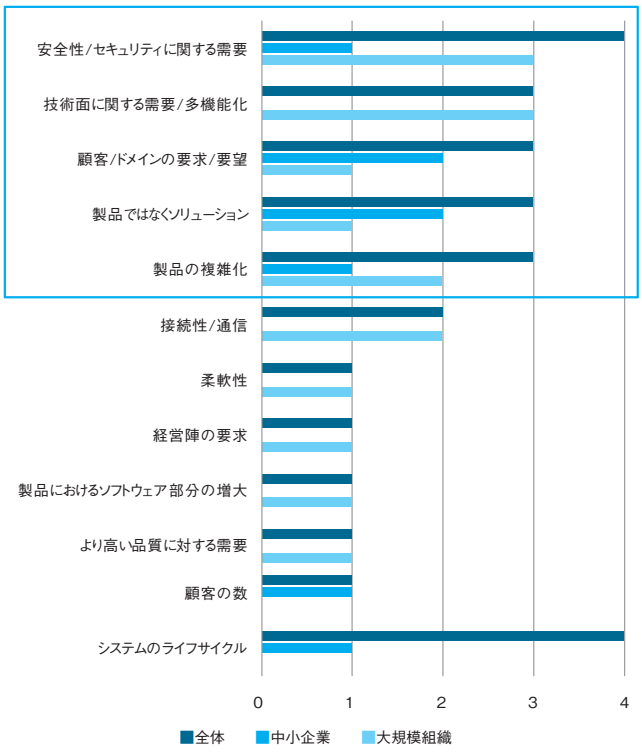


図３　システムズエンジニアリングへの転換動機

2.2　直面する最近の製品開発の傾向?

■現状多くの企業が直面している開発傾向について、次のことが上位の回答にきた。(図４)

- システム要求の複雑化
- 市場投入までの時間(TTM)の短縮(研究開発時間の短縮)
- 製品の多様化の増大

■今後(５年以内)の変化の見通しとして、現状に加え、更に、

- 複数の専門分野にまたがる開発の増加

が、挙げられた。(図５)

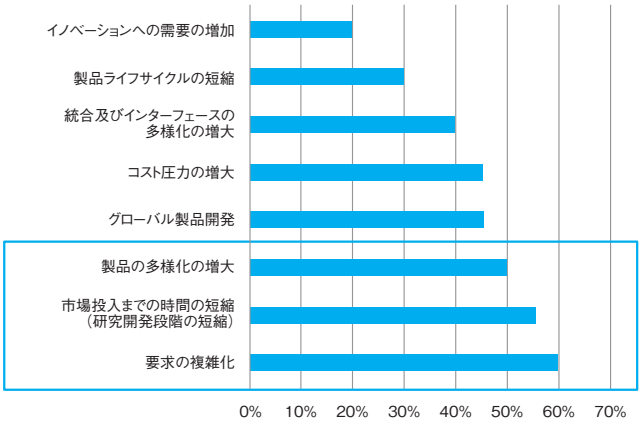


図４　現在のシステム開発の傾向

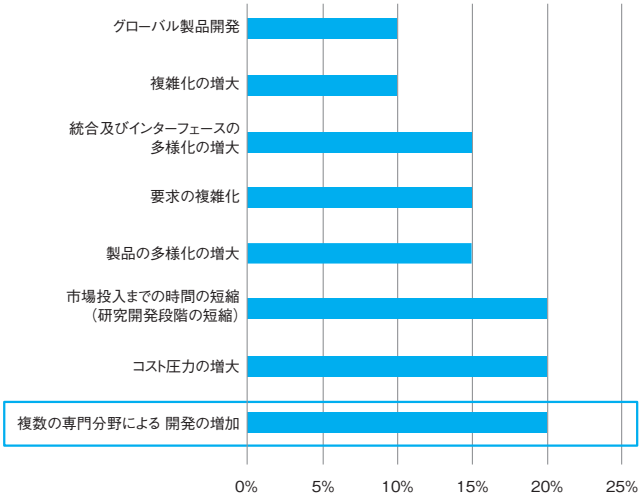


図５　５年以内の変化の見通し

2.3　システムズエンジニアリングの組織にとっての重要性はどの程度か？

■現状、組織にとってシステムズエンジニアリングの実装(プロジェクト･プロセス、技術プロセス、合意プロセス、組織的プロセスを含む)の重要性を10点満点(10：不可欠～１：重要でない)で評価すると、
平均：7.6となった。　(図６)

- 不可欠(重要度10～９)：25%
- 重要(重要度８～７)　：45%
- 中程度(重要度６～５)：30%
- 重要でない(４以下)　：　０%

という回答であった。

■今後(５年以内)の重要度の変化の見通しとして、
平均：8.7という結果となった。(図７)

■今後ますます重要性が増す理由として、主に次のことが挙げられている。(図８)

- 大企業：製品が更に複雑化する
  (多機能化、System of Systems、新しい技術)
- 中小企業：顧客の要求の高度化
  (製品品質確保、新しい技術)

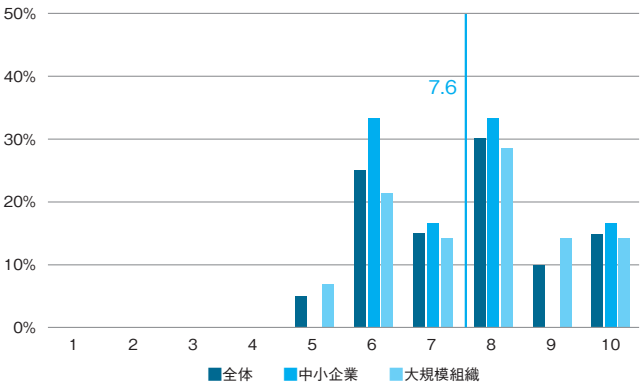


図６　現状のシステムズエンジニアリングの重要度合

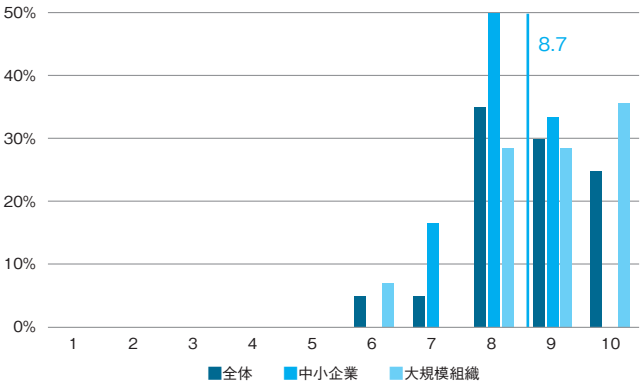


図７　今後の重要度の変化の見通し

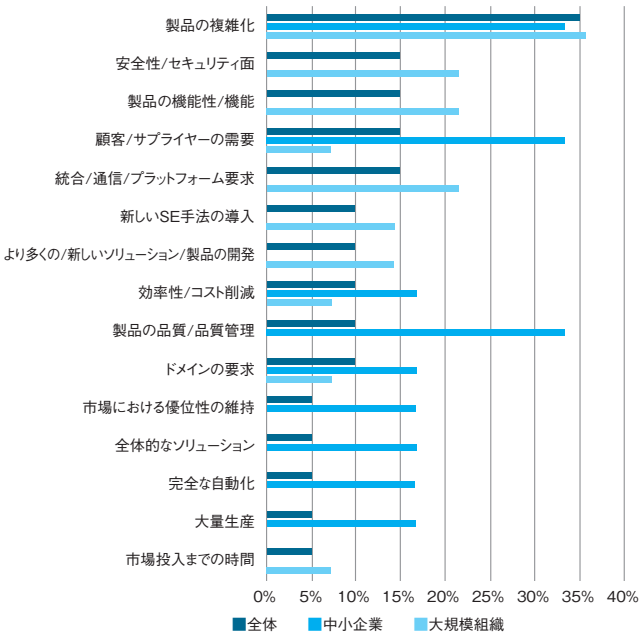


図８　重要性が変化する理由

2.4　現在直面しているシステムズエンジニアリングの課題は何か？

■80%の組織が直面する課題として、次のことが上位に挙げられている。（図９）
- 課題に対応できる組織への変革
- 要求管理、関連インターフェースの管理
- モデリングとシミュレーション
- 品質確保（セーフティ・セキュリティ含む）
- 組織間での一貫性のあるツール・チェーン

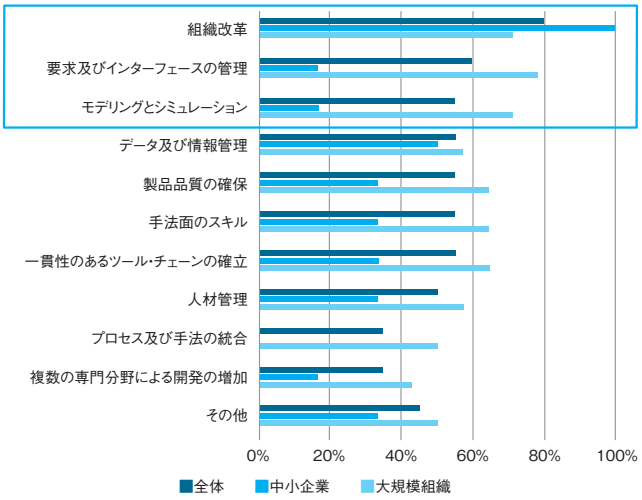


図９　現在直面しているシステムズエンジニアリングの課題

2.5　システムズエンジニアリングの実践として確立しているものは何か？

■多くの企業で確立している手法、技術、取り組みは、次の領域に関することである。（図10）
- モデル駆動開発
- 要求開発
- テスト駆動開発
- 検証と妥当性確認（V&V）

■大企業では、モデル駆動開発と検証と妥当性確認に注力している。

■中小企業では、更にテスト駆動開発に注力している。

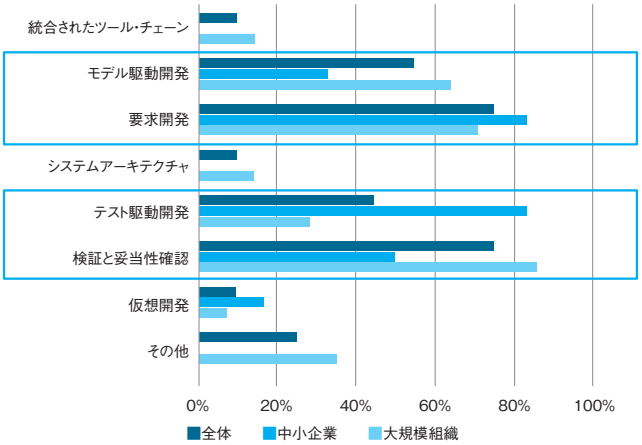


図10　確立したシステムズエンジニアリング実践の上位

2.6　システムズエンジニアリングは、どの分野で最も大きな改善の可能性があるか？

■システムズエンジニアリングによって最も改善の可能性がある領域。（図11）
- 仮想開発の増加
- より良いツール・チェーンの統合
  その需要は、中小企業ほど大きい。
- 約40%の大企業で、プログラム・マネジメントの強化（とくに、プロジェクトにおけるポートフォリオ・マネジメント）での効果に言及している。
- 約40%の中小企業で、自動化を更に進めることが、改善に重要と考えている。

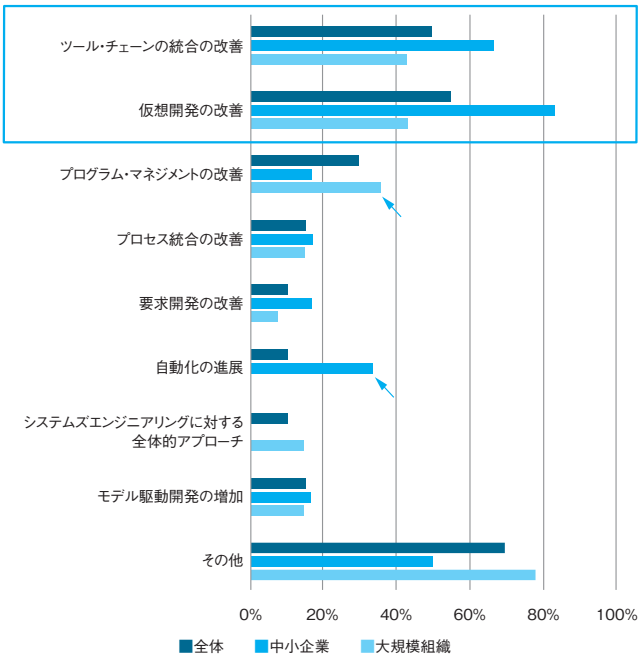


図11　システムズエンジニアリングによる改善の可能性が見られる分野

3　調査結果の分析から導き出された推奨事項

■組織開発

●組織改革戦略：
企業の80%が、システムズエンジニアリング実践の主要な課題は組織改革であると答えている。
どのような組織構造及びプロセスが変化に対応するのに最適かをオープンに考えることが重要である。
とくに、改革の動機付けと伝達を的確に行って、あらゆるステークホルダをそのプロセスに取り込むことが重要である。

●システムズエンジニアリング能力：
システムズエンジニアリングに関して内部トレーニングプログラムを作成し、外部トレーニングプログラムを購入することが、ほとんどの組織で義務化されていた。

更に、最新の開発情報を得たり知識・経験を共有したりするために、国内外のシステムズエンジニアリング関連のコンファレンスに参加し、各コミュニティーの積極的なメンバになることを推奨する。

●ソフトウェア開発能力：
もともとの業種がハードウェア開発寄りであるにもかかわらず、企業の85%以上が製品でソフトウェアが主要な役割を果たすと答えている。また今後も伸びると考えていることから、企業が適切なソフトウェア開発能力を構築する、あるいは維持するには、製品がソフトウェアに依存する度合いと、企業の主要なIP（知的財産）及びUSP（独自の売り）がどこに存在するかによって異なる。IP/USPがソフトウェアそのものに存在する場合、ソフトウェア開発を自組織内のリソースで構築することが重要である。またソフトウェアが１つの目的を達成するための手段にすぎない場合は、外部のソフトウェアサプライヤーあるいはパートナーを管理するための能力を構築することが、少なくとも理にかなっている。

●プロジェクト・ポートフォリオ管理：
大規模組織が、改善のための着目点として挙げているように、プロジェクトのポートフォリオ全体の管理と、プロジェクト間の相互連関及び依存関係についてとくに重点を置くべきである。

■技術開発

●システムズエンジニアリングの統合アプローチ：
新製品を市場投入するまでの時間（TTM）が短縮されるのと並行して、製品の複雑化が進んでいるため、システムプラットフォーム及びシステム統合の重要性が高い。これには、関与するすべての専門分野間で十分に検討され、また調整されたアプローチが必要となる。

●システム要求開発：
時間と共にシステム要求はますます複雑化し、製品の種類も増加している。システムレベルでどうやって要求を引き出し、開発し、長期にわたり系統立てて管理するかについてやり方を検討する必要がある。また、どう下位レベル（とくにソフトウェア）の要求に落とし込むかの手段を考えなくてはならない。

●モデル駆動システム開発：
システムのモデル駆動開発が組織にとって重要な実践と見なされていることが確認された。
組織はシステム仕様のどの側面をモデリングするか、妥当な範囲でどんな言語とツールサポートを利用できるかを評価する必要がある。ここでのツール選定は、開発プロセスのツール環境においてシームレスな統合を実現する上でツールによって提供されるインターフェースの適切さにも影響を受ける。

●システムの検証と妥当性確認：
システムの検証と妥当性確認、並びにテスト駆動開発のために適切な技法及び手法の確立を検討する必要がある。とくに、システムの検証と妥当性確認を、常にシステム要求と適切に関連付けて考える必要がある。

●仮想システム開発：
製品がますます複雑化し、複数の専門分野にわたる開発が進むにつれて、物理的に様々なシステム部品を構成することは難しくなり、コスト負担も非常に大きくなる。そのため、モデルに基づく仮想システム開発を適用できるかどうかについて検討する必要がある。これは開発速度を上げるという点でも改善が見込まれる。

●統合されたシステムズエンジニアリングのツール・チェーン：
組織においてはシステムズエンジニアリング実践のために多様なツールが使用されている。
ツール・チェーン統合は、主要な改善点と考えられる。ツール・データの相互運用性とツール・チェーンを統合することにとくに重点を置くべきである。

4 まとめとして

今回のシステムズエンジニアリングの実践に関する調査・分析結果から抽出された現場での実践について、次の５つの分野の実践が有効と結論付けられる。

■既に確立した実践のうちで、企業が広く（50%近く、またはそれ以上）適用しているのは、以下の分野に関連する手法、技法及びアプローチである。

1. モデル駆動システム開発
2. システム要求開発
3. システムの検証と妥当性確認

■適用・実践は初期段階ではあるがシステムズエンジニアリングで最も大きな改善が期待されるとして、企業の約50%が挙げている分野は下記の２つである。

4. システムズエンジニアリング・ツール・チェーンの統合
5. システムの仮想開発

本稿では調査分析報告の主要部分を紹介した。本稿のもととなっている詳細な解説については、事例紹介を交えて、下記IPAのサイトからダウンロード可能となっている。お役立ていただけることを期待して結びとする。
http://www.ipa.go.jp/sec/reports/20161219.html

第1回 STAMP Workshop in Japan 開催報告

SEC調査役 石井 正悟

IPA/SECは、2016年12月５日～７日の３日間、九州大学稲盛財団記念館、九州大学西新プラザにおいて九州大学、有人宇宙システム株式会社（JAMSS）、一般社団法人組込みシステム技術協会（JASA）、日本MOT学会と共催で第１回 STAMP Workshop in Japanを開催した。

1 開催の背景

九州大学稲盛財団記念館

MITのNancy Leveson教授が提唱する、システム理論に基づく新しい安全性分析方法論STAMP（Systems-Theoretic Accident Model and Processes）が欧米を中心に産業界で注目されている。IPA/SECもSTAMPに注目しつつ、我が国の産業界に有効なシステム安全性向上手法の調査・検討・普及を行うべく、2015年度にシステム安全性解析手法WGを立ち上げ、活動を進めてきた。欧米では既に毎年STAMPワークショップが開催されているが、日本でも、IPA/SECが今年度立ち上げたシステム安全性・信頼性分析手法WGが主体となり、STAMPに深く関心を持つ九州大学、日本でSTAMPの先駆的な経験・知見を有するJAMSS等と共に、「第１回STAMP Workshop in Japan」を開催することになった。本ワークショップは、日本でのSTAMP普及促進にとって重要なイベントになるとIPA/SECは期待している。

2 ワークショップ概要

初日は九州大学稲盛財団記念館にて、２日目からは九州大学西新プラザに会場を移して実施した。

開催日	時間	プログラム
12/5（月）	午前	キーノートスピーチ：MIT　John Thomas
	午後	イントロダクション：九州大学　荒木教授
		招待講演　４件
12/6（火）	午前	招待講演　２件 ショート講演　２件
	午後	一般講演セッション　９件
12/7（水）	午前	一般講演セッション　５件

１日目：キーノートスピーチとしてMITのDr. John ThomasによるSTPAチュートリアル（初級）、STPAチュートリアル（中級）、STPAの事例研究を紹介いただいた。Nancy Leveson教授のもとで研究されているSTAMP第一人者であるJohn Thomas氏から直接、長時間にわたるチュートリアルを実施いただき、多くの参加者から感謝の声をいただいた。

九州大学西新プラザ

続けて、STAMP実践に関して日本の第一人者であるJAMSS 星野様の講演など４件の招待講演が行われた。

２日目：STAMPと並び今後の安全性解析に有効性が期待され、またSTAMPとの組み合わせも期待されるレジリエンスエンジニアリングについての２件の招待講演が行われた。JAMSS 野本様による招待講演では、新宿駅コンコース歩行を対象として構築したFRAM（Functional Resonance Analysis Method：機能共鳴分析手法）モデルをSTAMPモデルに対応させるという、新たなSTAMP活用方法が紹介された。その後、ショート講演セッション、一般講演セッションA、B、Cで計11件の発表があった。

３日目：一般講演セッションD、Eで計５件の発表が行われ、クロージングでは次回以降のSTAMP Workshop in Japan開催について議論し、多くの参加者から次回以降の開催を楽しみにしているとのご意見をいただいた。

また、SECの事業を中心にIPAで取り組んでいるIoTや組込み系に関連する事業の資料配布も実施した。

3 本イベントを振り返って

今回のワークショップは、日本では初開催であり、開催地も首都圏以外であることから参加者数を心配したが結果としては約130名の皆様に参加いただき、STAMPへの関心の高さがうかがえた。また一般講演への応募も多数あったためプログラムを追加し５つのセッション、16件の発表・意見交換の場を持つことができた。

当初２日間と見込んでいた会期を３日間に延長したり、直前になって会場を広い部屋に変更したりするなど、予想を大幅に上回る大盛況であった。日本におけるSTAMPへの関心・期待の大いなる盛り上がりを感じ、ぜひ次回以降のSTAMP Workshop in Japanにつなげ、我が国のシステム安全性向上に寄与すべく、STAMP普及・定着を推進したい。

１日目の会場風景

John Thomas氏のチュートリアルを含めた基調講演をはじめ、本ワークショップでの講演資料をIPA/SECのWebサイトで公開しているのでぜひご参照いただきたい。

第１回 STAMP Workshop in Japan　IPA/SEC Webサイト
http://www.ipa.go.jp/sec/events/20161205.html

- イベント概要、プログラムの紹介
- 講演資料のダウンロードが可能です
- 講演の動画も随時公開予定です

GQM+Strategiesによる組織目標と戦略の整合化及び目標定量管理の実践と拡張
–SEC WG及び早稲田大学ゴール指向経営研究会の活動より–

早稲田大学　国立情報学研究所　株式会社システム情報　鷲崎 弘宜／新谷ITコンサルティング　新谷 勝利
早稲田大学　青木 耀平／早稲田大学　志村 千万輝／伊藤忠テクノソリューションズ株式会社　野村 典文

IPA/SECにおいては2007年10月から、フラウンホーファー研究機構実験的ソフトウェアエンジニアリング研究所（IESE）と共同でGQM+Strategies（目標・質問・メトリクス＋戦略）の研究に取り組んできた。GQM+Strategiesは、組織のあらゆる箇所や階層において目標と戦略を、定量管理可能な形で整合させ、改善させ続ける手法である。IESEの名称が「実験的」と冠するように、SECとの取り組みにおいて企業における複数回の実践を経て手法の洗練化が図られた。また、一般へと展開すべく、SECにおけるWG（作業部会）にて教材を作成し、SEC主催セミナーを実施している。更に2013年４月からは、活動主体を早稲田大学グローバルソフトウェアエンジニアリング研究所・ゴール指向経営研究会に移行させて、更なる研究、実践、教育・普及に取り組んでいる。具体的には、GQM+Strategiesにおいて作業成果の質が分析者の経験や観点に大きく依存するため、その改善と発展を意図した様々な手法を研究し、企業における適用を経て有効性を確認している。本稿では研究成果を中心としてゴール指向経営研究会の活動を紹介するとともに、システムズエンジニアリングの観点からGQM+Strategiesを解説する。なお、SECにおけるWG活動成果についてはSECジャーナルの過去の解説[1][2]を参照されたい。

1 ゴール＆ストラテジー入門

2004年SEC創設と共に始まったプロセス共有化WGにおいて、開発及び運用における諸問題の多くが、ソフトウェア開発プロセスの初期段階である超上流段階に起因していると看破されている[3]。そこでこの諸問題を解決する方法はないものかという調査をしていた段階で、IESEのGQM+Strategies[4]に遭遇した。

GQM+Strategiesは、以下の文書化を行い、これらを表現するモデル（「グリッド」と称する）を作成し、戦略を実践し、データを収集し、分析し、改善活動を行う手法である。ゴール指向の測定手法であるGQM（Goal-Question-Metric）パラダイム[5]を組織目標や戦略の整合化へと発展させたものと捉えることができる[6]。

- 企業が取り組みたい目標と戦略
- 企業部門を超えて目標と戦略を結び付けるにあたっての論理的根拠（事実と仮定）
- 目標の達成度を評価する測定モデル
- 意思決定のために測定結果を解釈するガイドライン

GQM+Strategiesグリッドの例を図１に示す[6]。グリッドは通常、組織構造に沿って、目標・戦略を連鎖させ、各目標の達成可否判断に必要なメトリクス（測定の方法と尺度）をGQMグラフにより表現する。また目標や戦略の導出の根拠として、事実（Context）や仮定（Assumption）を明示する。この仕組みを適用することで、組織構造上の上位において決定した戦略を下位部門が実現していないことや、逆に下位の戦略が組織に貢献していないこと、あるいは、せっかくの測定データが組織目標と結び付いていないと

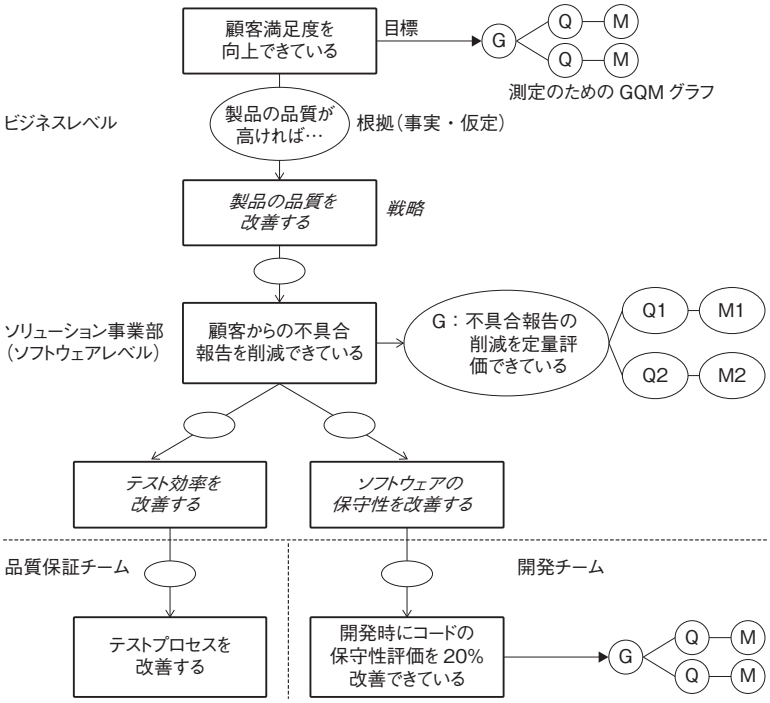


図１　GQM+Strategiesグリッドの例（[8]を一部改変）

いった不整合を可視化できる[7]。その上グリッド上で、目標と戦略を整合化させて、更に目標と戦略へと寄与する測定を実施するように改善可能となる。

GQM+Strategiesにおける活動の流れを図２に示す。最初に「初期化」を実施する。以降は、「環境の特性化」「目標と戦略の設定」からなる開発、続いて「実行計画の策定」「計画の実行」からなる実行、更に「結果の分析」「改善パッケージ化」からなる学習を実施し、これを繰り返すというものである。この実施の流れは、組織レベル及びプロジェクトレベルの取り組みを通じて継続的なプロセス改善を実現するQuality Improvement Paradigm（QIP）[9]のサイクルモデルに従っている。

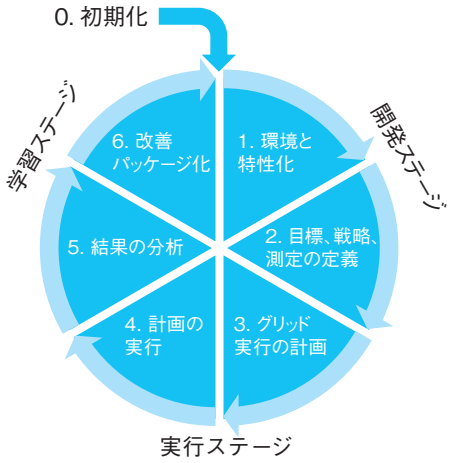


図２　GQM+Strategiesの活動の流れ

GQM+Strategiesの初の解説書『ゴール＆ストラテジー入門 – 残念なシステムの無くし方 –』[4]には活動の詳細が解説されているため、ぜひ参照されたい。更に同書には、SECにおけるWGの活動成果も含めて、GQM＋Strategiesの国際的な適用事例及び得られた知見が含まれている。以下にその例を示す。下例で、最後の２つは日本における適用事例である。

- 大学や軍における適用も含めて業界に制約なし
- 極めて大規模な国際的企業における戦略の見直しについても有効
- メトリクスの項目を導出可能
- 複数のプロジェクトの実施優先位の決定を数値的に説明可能

2 GQM+Strategies適用の展開 – システムズアプローチ

SECにおいては、2016年度からシステムズエンジニアリングの推進が始まっている。ソフトウェア・エンジニアリングもそうであったが、新しい開発方法論を必要とする理由として、ソフトウェア・エンジニアリングであれシステムズエンジニアリングであれ、その対象が従来よりも「大きく」かつ「複雑」になってきたことがある。システムズエンジニアリングでは、それに加え、ステークホルダの多さが対象とするものへの変更要求をより多く、より複雑なものにしている。

1968年にソフトウェア・エンジニアリングが必要とされてから今日で50年近く経過している。この間独立して開発されたソフトウェアは、ビジネスの変更、組織の変更などにより、その用をなくし運用を停止されたもの、修正され保守され今も生きているもの、機能は変わらないが運用のプラットフォームなどが変わったもの、等々様々な形態を取っており、独立して開発されたものも保守の過程を経て連携されて運用されているものもある。

運用のトップレベルの目標の実現は、より下位のシステムの構成要素から実現されることになる。GQM+Strategiesは、このトップレベルの目標とそれを達成する構成要素であるストラテジーの整合性を取る方法論であり、まさしくシステムズアプローチとの関係が深い。

システムの構成の概念を図３に示す。図３において明白なことは、もしトップレベルの達成目標が明確でなければ、より下位で目標を実現するべきストラテジーが決まらないことになったり、トップ目標がそれを分割して実現するサブ目標も定まらないことになる。

これはまさしく超上流という用語で議論されたことである。また、前図のような構造はユニークなものではなく、ビジネスの視点、環境の視点など、で多くのバリエーションが可能である。これらのうち、どのような構造がより最適なのかを特定する方法論が必要になる。GQM+Strategiesは目標の達成を測定し、評価する仕組みを持っているので、今後は更にこの仕組みをシステムズアプローチと連動させてゆく実践が必要と考えている。

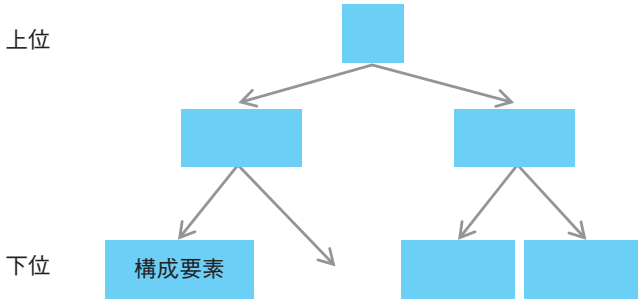


図３　システムの構造図

3 GQM+Strategies適用の展開

IPA/SECにおけるWG活動を早稲田大学グローバルソフトウェアエンジニアリング研究所（所長：鷲崎）へ移管する形で2013年４月に研究所傘下にゴール指向経営研究会[10]を設立して以降、実務家と研究者が総勢15名以上集い、活発な議論を繰り返しながら以下の研究、教育・普及、実践を深めている。

3.1 研究

GQM+Strategiesは組織における目標と戦略及びデータの整合化に優れているが、その作業成果の質は分析者の経験や観点に委ねられるところが大きい。また、識別する目標や戦略の観点はしばしば当該組織や組織内箇所の一視点に限定されていた。

そこで研究会では、以下の拡張・関連研究(1)-(4)に取り組み、トップカンファレンスに採択されるなど国際的に高く評価される成果をあげ、後述の教育・普及や実践へと応用を開始している。

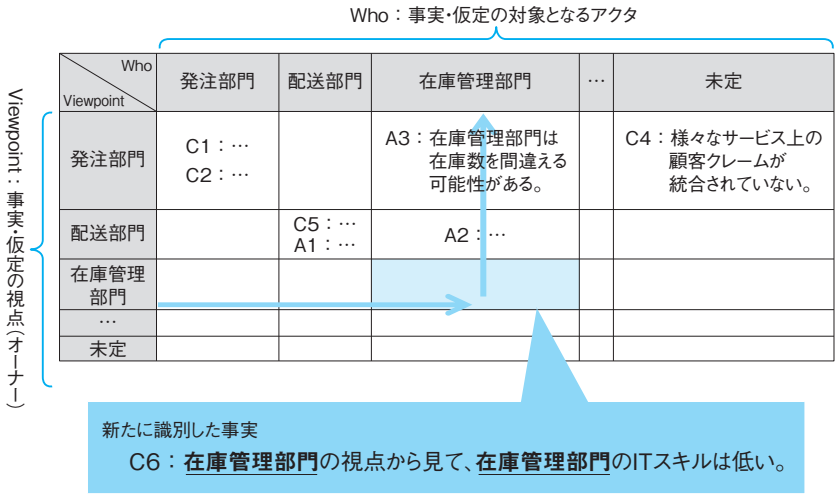


図4　CAMによる事実(A)・仮定(C)識別の様子

(1) 利害関係者間の関係分析を通じた事実・仮定の導出

目標や戦略の導出にあたり、その妥当性を裏付ける根拠として、客観的なデータなどのある「事実(Context)」と、不確かな「仮定(Assumption)」を識別しておく必要がある。しかしGQM+Strategiesそのものは事実・仮定の識別を支援しない。従って識別の漏れや、観点の偏りの問題が生じている。

そこで研究会では、利害関係者間のマトリクス上で、すべての利害関係者の組み合わせについて網羅的に事実及び仮定を識別する手法Context-Assumption-Matrix(CAM)を考案した[11]。CAMにおいては図4に示すように、ある観点(縦方向のViewpoint)から誰か(横方向のWho)を対象として、それぞれに把握済みの事実あるいは想定される仮定を網羅的に記述する。

研究会では大学における演習において、GQM+Strategiesを単独で用いるよりも、CAMを併用する方が様々な観点を網羅する形で効率的に事実・観点を識別できることを確認した。更に、後述のように企業における適用を通じて実用性も確認している。

(2) 戦略の衝突・重複・依存関係の特定

GQM+Strategiesは、原則的にはトップレベルの目標から下位の戦略、目標を定義していく。しかし、このようなトップダウン型アプローチは、下位に位置する戦略間において衝突や重複、依存といった関係が潜在することがある。これらの潜在的な関係は、目標達成や戦略実施の阻害要因になるため、戦略策定の段階で発見・検討されることが望ましい。

そこで研究会は、構造化モデリング手法を利用してGQM+Strategies内の要素間の関係を検査するHorizontal Relation Identification Method(HoRIM)を開発した[12]。HoRIMの適用の流れを図5に示す。HoRIMでは、GQM+Strategiesによるグリッドと、構造化モデリング手法により構築したモデルを比較することで、GQM+Strategiesグリッド内に潜在する関係を特定する。続いて特定した関係を検討し、その結果を用いてGQM+Strategiesグリッドの改善を図っていく。

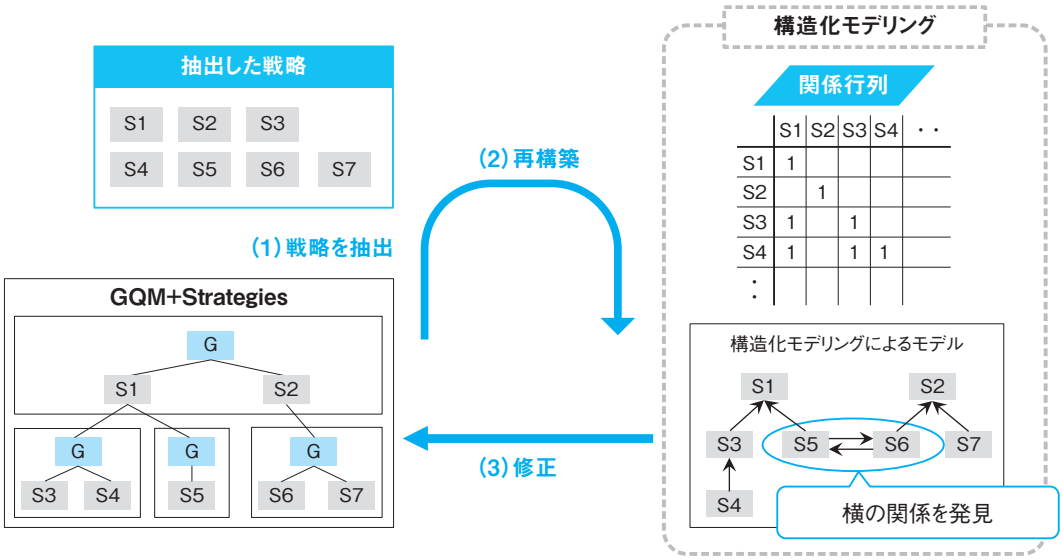


図5　HoRIM適用の流れ

研究会では、HoRIMの有効性について対照実験を行った。その結果HoRIMを使用した場合は使用しない場合よりも多くの潜在する関係を発見できることが判明した。また企業への適用事例から、HoRIMの有益性を確認した。

(3)形式的な定義と設計原則

現在、GQM+Strategiesグリッドとして取り得る構造は厳密には定義されていない。従って、作成されたグリッドの形式は作成者ごとに異なり、正当性の確認は不可能な現状にある。この現状はグリッドに潜在する構造上の問題やそれを要因として発生する戦略上のリスクの特定を阻害しており、改善が求められる。

そこで、研究会はGQM+Strategiesの構造をUMLクラス図で表現したメタモデルを考案し、グリッド作成のための形式的な定義とした。更に、メタモデルに基づいてグリッドを作成する際の各要素間の関係制約を設計原則として定義した。例えば「組織構造の原則」は、階層的な目標と戦略の繋がりは実際の組織構造に合わせて定義するという原則である。これらの原則はOCLによって記述され、グリッドの構造的な原則違反を自動検出可能とする[13]。原則違反検出の流れを図６に示す。

研究会では被験者実験を行い、設計原則を使用した場合のほうが未使用の場合よりも早く正確なグリッドを作成できることを確認した。更に、企業の事例に適用することで定義の妥当性、有効性を確認した。

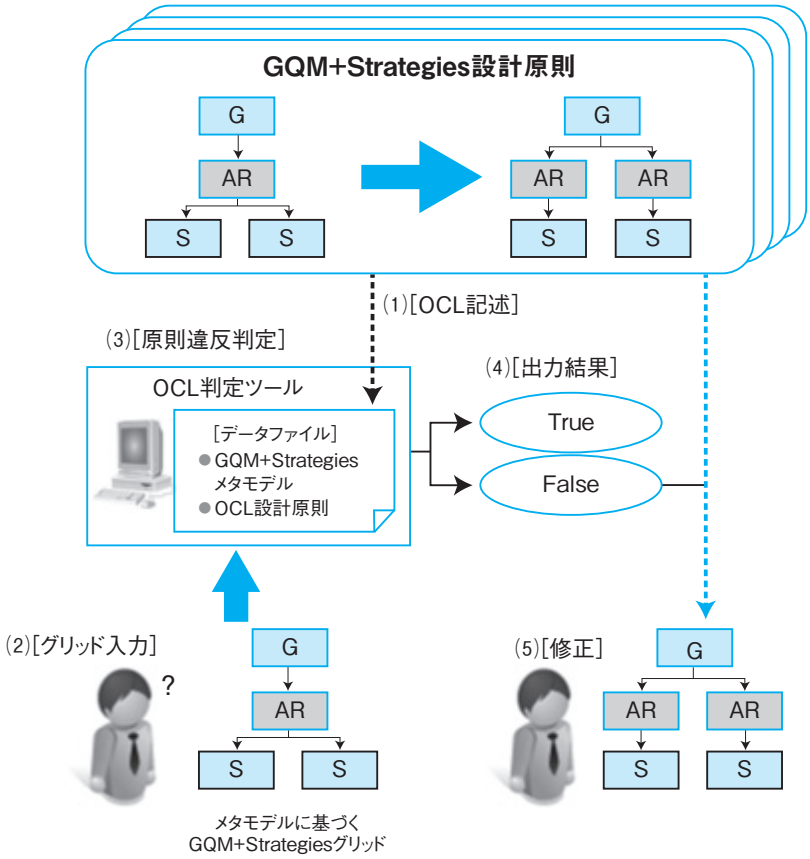


図６　原則違反検出の流れ

(4)ユーザ側の視点を組み入れた戦略設計

(1)-(3)の内容は、ドメインを限定せずに戦略実施側(組織側)の視点による利用を想定したものである。一方で、ユーザ向けのインターフェース及びユーザビリティが重要なサービスドメインにおいては、従来はユーザ中心の設計が進められてきた中、サービスを提供する組織側の目標との整合や調整が重要な課題となりつつある。

そこで研究会はYahoo! Japanとの共同研究により、GQM+Strategiesによるサービス提供側の目標・戦略と、ユーザ中心設計においてしばしば用いられるペルソナを起点としたユーザ側の目標・戦略とを整合させる手法GO-MUC(Goal-oriented Measurement for Usability and Conflict)を考案した[14]。

GO-MUCの概要を図７に示す。GO-MUCでは、ユーザ視点で識別した目標達成可否判定に向けたメトリクスと、ビジネス視点のメトリクスとの間の組み合わせを検討することで、両者間で起こり得る戦略上の衝突の関係を効率的に特定し、その改善・緩和策を検討する。Yahoo! Japanの実サービスの開発運用についてGO-MUCを適用し、サービス改善のための戦略を識別しその効果を確認している[14]。

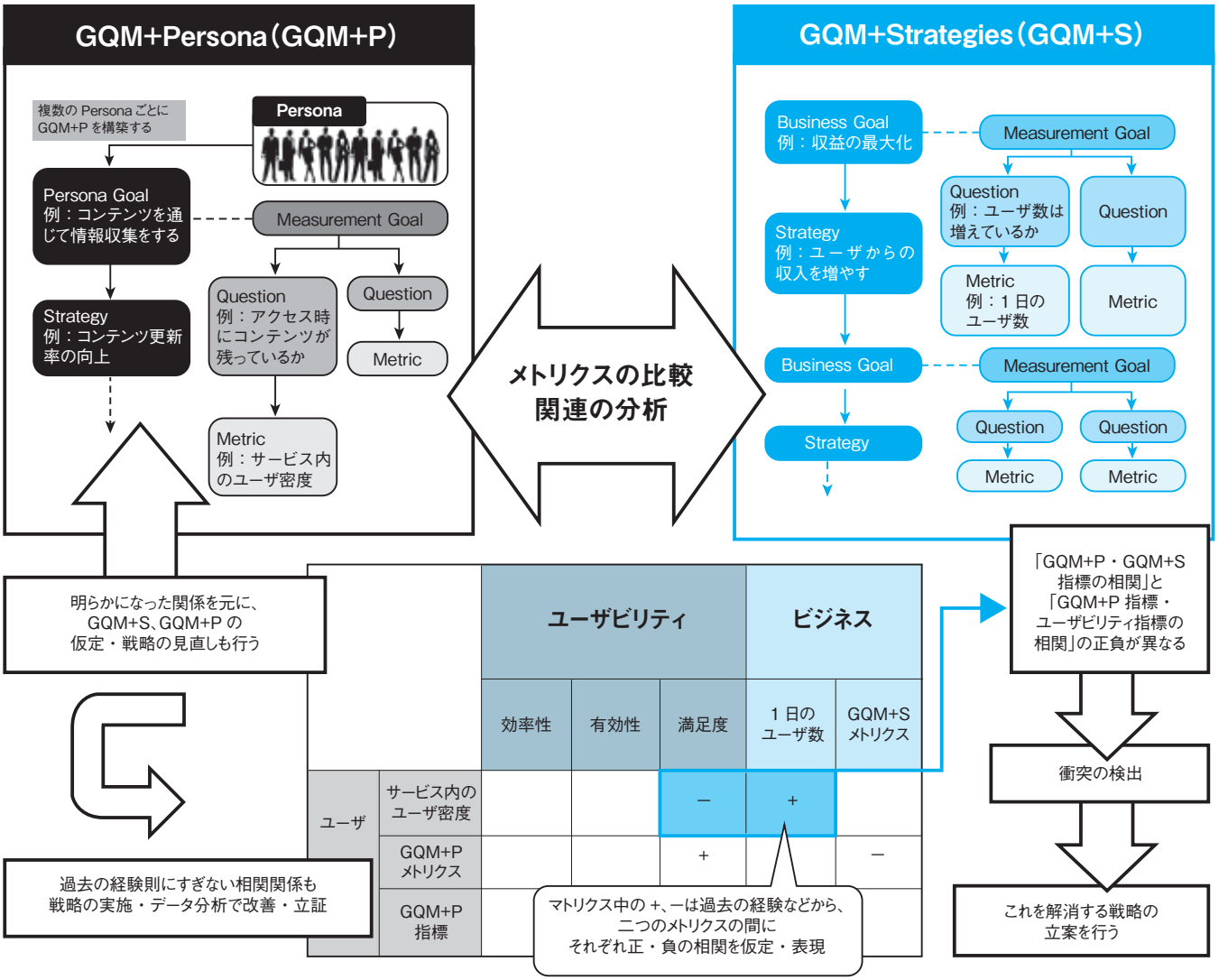


図7　GO-MUCの全体像

3.2　実践

訳書の出版やゴール指向経営研究会の活動に触発される形で、国内の数多い企業や組織においてGQM+Strategiesあるいは類似のゴール指向の目標・戦略の整合化の取り組みが進められている。以下に、研究会メンバが連携する形で進めている実践の取り組みの一例を紹介する。

- リクルート住まいカンパニー：目標と戦略の整理にGQM+Strategies、CAM及びHoRIM手法を適用実践し、整合化及び見落としがちな戦略の識別に有効なことを確認した[11][12]。
- 株式会社システム情報：GQM+Strategiesを適用実践し、組織構造上の構成単位を超えた目標と戦略の整合化に成功した。
- Yahoo! Japan：GQM+Strategiesを組み入れた形でGO-MUC手法を適用実践し、ユーザ側とサービス提供側の両者の目標を整理し、サービス改善の戦略を識別、サービスの改善に成功した[14]。
- 伊藤忠テクノソリューションズ：2013年度よりGQM+Strategiesを組織の目標管理（MBO：Management By Objective）に適用している。初年度は本部内で試行し有効であることを確認した。2014年度から事業グループ全体に適用し、来年度は全社展開を予定している。

3.3　教育・普及

大学における講義や企業実務家向けのチュートリアルを通じて、GQM+Strategies及び研究会において研究開発した手法の教育と普及に努めている。具体的には早稲田大学と島根大学でそれぞれ20-50名ほどの学部・大学院生向けに演習形式で講義を実施してきている。

またIPA/SECセミナーとして、演習中心の入門チュートリアルを、東京を中心に大阪、札幌など様々な箇所で毎年4回以上実施し、好評を博している。アンケートを通じて把握する参加者の共通の感想は「これまで目標と戦略の整合及びその組織連鎖を考えていなかった（あるいは軽視していた）」というものである。GQM+Strategiesという新たな解決策を紹介する過程で、自身や自組織に内在する問題の可能性に気付きを与えていることが分かる。

広く一般向けの啓蒙としては、GQM+Strategiesのもともとの発明者であるJens Heidrich氏らを招聘してIESE（及び第１回はIPA/SEC）と共同で2015年２月と９月にそれぞれセミナーを開催し、GQM+Strategiesや関連手法の紹介に努めた。更に前述の訳書出版[4]や記事執筆[15]による啓蒙も進めている。

更に2015年度より厚生労働省内のIT人材研修の１テーマ「業務見直し方針の策定」に採用され、今年度も継続して実施している。国立情報学研究所が進める人材育成プログラム・トップエスイー（TopSE）[16][17]においても要求工学の教育に採用されている。

これらの入門チュートリアルの反響を受けて更なる実践を支援する教育のニーズが高まっており、研究会では現在、研究及び実践結果を盛り込んだ実践チュートリアルを開発中である。

4 まとめ

本稿では早稲田大学ゴール指向経営研究会における活動を中心として、GQM+Strategies及び関連手法の研究、実践、教育・普及の様子を紹介した。本稿により、読者や関連組織における目標・戦略・データ（定量化）の整合について内在する問題を識別され、GQM+Strategiesや関連手法の適用実践を通じて整合化及びそれを通じた効率的・効果的な戦略実施と目標達成が進むこととなれば幸いである。

研究会では引き続き、GQM+Strategiesの実践と応用研究を進める予定である。例えば前述のようにシステムズアプローチとの連携は今後の研究課題の一つである。IoT時代における組織を超えて、そして変化しやすい目標や戦略との親和性も、検討すべき研究課題である。

ぜひ研究会やセミナーにご参加いただきたい。様々な組織においてGQM+Strategiesや関連手法を実践し、その取り組みを共有しつつ、研究により日本から世界へと優れた成果を共に発信していくこととなれば幸いである。

謝辞

活動機会をいただいたIPA、IPA/SECにおけるWG、ならびにゴール指向経営研究会の参加メンバ各位に御礼申し上げます。また、研究実践やセミナー開催を進める上でご協力をいただいた関係各位に御礼申し上げます。

参考文献

[1] 新谷勝利、平林大典、企業・組織の目標達成とIT導入計画の整合化を実現するための手法推進、SEC journal, No.30, 2012.

[2] 新谷勝利、平林大典、定量的な目標管理手法の普及活動の展開～組織目標達成とIT導入の整合性を図る「GQM+Strategies®」の活用～、SEC journal, No.33, 2013.

[3] IPA/SEC、経営者が参画する要求品質の確保～超上流から攻めるIT化の勘どころ～第２版、オーム社, 2006.

[4] Victor Basili, Adam Trendowicz, Martin Kowalczyk, Jens Heidrich, Carolyn Seaman, Jürgen Münch, Dieter Rombach 著、鷲崎弘宜、小堀貴信、新谷勝利、松岡秀樹 監訳、早稲田大学グローバルソフトウェアエンジニアリング研究所ゴール指向経営研究会 訳、ゴール&ストラテジ入門：残念なシステムの無くし方（GQM+Strategies）、オーム社、2015.

[5] Victor Basili, G. Caldiera, Dieter Rombach, Goal, Question, Metric Paradigm, Encyclopedia of Software Engineering, Vol.1, pp. 528–532, 1994.

[6] 鷲崎弘宜、ゴール指向の測定評価と留意 – GQMパラダイムと拡張 –、メトリクス公団、Vol.1、TEF東海メトリクス勉強会、2013.

[7] 鷲崎弘宜、実践的ソフトウェア品質測定評価のための４つの「落とし穴」と７つの「コツ」：ゴール指向、不確実性、機械学習、実態調査ほか、品質、Vol.46, No.3, pp.137-140, 品質管理学会、2016.

[8] IESE 制作、IPA/SEC訳：「GQM+Strategies®」のワークショップ教材（IPAソフトウェア高信頼化：IESE共同研究資料）、2012.

[9] Victor Basili, Quantitative Evaluation of Software Engineering Methodology, 1st Pan Pacific Computer Conference, 1985.

[10] ゴール指向経営研究会（GQM-RG）GQM+Strategies手法の普及活動と研究活動：https://gqmstrategies.wordpress.com/

[11] Takanobu Kobori, Hironori Washizaki, Yoshiaki Fukazawa, Daisuke Hirabayashi, Katsutoshi Shintani, Yasuko Okazaki and Yasuhiro Kikushima, Exhaustive and efficient identification of rationales using GQM+Strategies with stakeholder relationship analysis, IEICE Transactions on Information and Systems, Vol.E99-D, No.9, pp.2219-2228, 2016.

[12] Yohei Aoki, Takanobu Kobori, Hironori Washizaki, Yoshiaki Fukazawa, Identifying Misalignment of Goal and Strategies across Organizational Units by Interpretive Structural Modeling, 49th Hawaii International Conference on System Sciences (HICSS-49), pp.4576-4585, 2016.

[13] Chimaki Shimura, Hironori Washizaki, Takanobu Kobori, Yohei Aoki, Kiyoshi Honda, Yoshiaki Fukazawa, Katsutoshi Shintani and Takuto Nonomura, Identifying Potential Problems and Risks in GQM+Strategies Models Using Metamodel and Design Principles, 50th Hawaii International Conference on System Sciences (HICSS-50), pp.4857-4866, 2017.

[14] Chihiro Uchida, Kiyoshi Honda, Hironori Washizaki, Yoshiaki Fukazawa, Kentaro Ogawa, Tomoaki Yagi, Mikako Ishigaki, Masashi Nakagawa, GO-MUC: A Strategy Design Method Considering Requirements of User and Business by Goal-Oriented Measurement, 9th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE 2016), pp.93-96, 2016.

[15] 早稲田大学ゴール指向経営研究会（鷲崎弘宜、平林大典、野村典文、井出昌浩）、連載記事「残念なシステム」のなくしかた、日経情報ストラテジー /日経ITPro、2014.

[16] トップエスイー：http://www.topse.jp/

[17] Shinichi Honiden, Yasuyuki Tahara, Nobukazu Yoshioka, Kenji Taguchi, and Hironori Washizaki, Top SE: Educating Superarchitects Who Can Apply Software Engineering Tools to Practical Development in Japan, 29th International Conference on Software Engineering (ICSE 2007), pp.708 – 717, 2007.

情報システムの事故データ

情報システムの障害状況 2016年後半データ

IPA顧問 松田 晃一
SECシステムグループ 主任 八嶋 俊介

2016年７月から12月までの情報システムの障害は19件であった。その中で、アクセスの集中をきっかけとする障害が今期も４件報告されている。そのほか、業務処理に誤りがあるまま気づかずに長期間運用されてきたが、今期になってその誤りが発覚した事例が３件報告されるなど、社会インフラシステムの障害発生数は相変わらず多い。

1. はじめに

2016年７月から12月までの2016年後半の半年間に報道された情報システムの障害の概況を次節で述べる。続いて３節では、今期に多数発生したシステムへのアクセス集中による障害の事例について報告する。４節においては共同利用型システムにおいて、一利用者の障害がほかの複数の利用者に影響を与え、被害が拡大した事例を取り上げ、共同利用型システムのリスクについて示す。

2. 2016年後半の概況

2016年７月から12月までの半年間に、表１に示す21件の障害が報道された。このうちの２件（事例1621、1622）は、2016年前半に発生したものが今期になって報道されたものであり、2016年前半の障害へ加算すると後半の障害件数は19件となる。一方、2016年前半の障害は２件増加して22件となり、通算では41件、月平均では3.4件となる。過去の年ごとの発生件数及び月平均件数の推移は図１に示す通りである。なお、事例1637と1641は後述の通り、いずれも2008年以前に発生した障害が今期に発見されたものであるが、前例に従い今期の障害件数に加えて報告する。

2015年10月に運用開始後、障害が多数発生し問題となっていたマイナンバー関連システムは、問題が解決したとの発表が2016年４月末にあり、その後は特段の障害の報道もなく問題が解消されている様子が見られた。しかし、残念ながら2016年後半に入って２件のトラブル（事例1630、1640）が報道されており、更なる安定稼働を期待したい。

また、業務処理の誤りに気づかずに長期間運用されてきたシステムにおいて、今期になって誤りが顕在化した事例が３件（事例1625、1637、1641）報告された。同種の事例は、2016年前半に３件報告されており、年間では６件となり、前年2015年の６件と同数となっている。更に、本報告期間外であるが2017年早々に、電力小売自由化に伴って運用が開始された、中部電力のシステムに不具合が発覚。電力取引の価格算定の基礎となる数値の計算に誤りがあったため、７カ月にわたって電力事業者間での取引価格が誤っていたとのことである［ITメディア 2017］。

この種の問題については前号でも取り上げた［松田2016］が、システムダウンを引き起こして直ちに検知される不具合とは異なり、（重大な誤りを抱えながら）一見正常に運用されているため、問題の発覚が遅れ影響が拡大する。これらの問題に有効な一般的な対策を見出すのは難しいが、コンピュータの処理結果だからと盲目的に信用するのではなく、以前の傾向と比較して異常な値ではないか確認するなど、ユーザ側でも検証し疑問があれば運用元に問合せることも必要なことかもしれない。

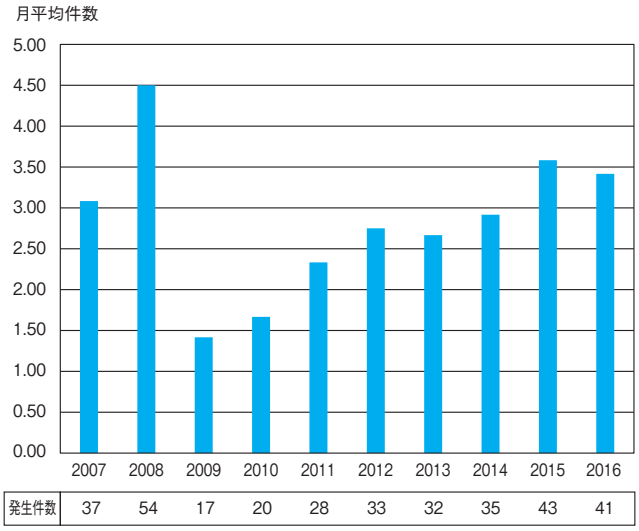


図１ 情報システムの障害発生件数の推移

表1　2016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				
1621	Yahoo! ショッピング 日次売上 速報メール	2016	5	25	08時00分	店舗に売り上げ情報を知らせるメールが他店に誤送信され、メールを通じて情報漏えいが発生した。 被害店舗は236店舗 誤送信先は1,113店舗	前日にリリースしたばかりの新機能「日次売上速報メール」が、システムの不備により、他店に送信された。テストケースに漏れがあり、不具合が発生した。 誤送信メールを受け取った店舗側も、送り先のメールアドレスがすべて「TO」欄に羅列されており、受信者すべてが見られる状態になり、不利益を被った。 ヤフーは、以下の再発防止策を講じた上でサービスを再開する。 • 30件を超える宛先へのメール送信は異常と判定する機能を追加（サービス利用者の登録上限が1店舗あたり30件のため） • 項目を見直したテストを再度実施。開発担当者だけでなく、別の担当者による二重のチェックを行う。	ソフトウェア障害	• 日経コンピュータ（2016.8.4）
1622	総務省 e-Stat	2016	6	30	10時45分	システム障害により、サイトが閲覧できなくなった。	政府の各種統計をまとめているインターネットサイト「e-Stat」が、API機能の不具合により閲覧できなくなった。直接の原因は機器の不具合。 このサイトでは、総務省の国勢調査や、厚生労働省の有効求人倍率といった情報が一元的に管理されている。	ハードウェア障害	• e-Statプレスリリース（2016.6.30） • 毎日新聞朝刊（2016.7.1）
		2016	6	30	18時00分				
1623	大阪取引所 J-GATE	2016	7	29	09時51分	国債先物、国債先物オプション、有価証券オプションが約定できなくなった。	原因は調査中。トラブルを起こしたシステム「J-GATE」は7月19日に稼働を始めたばかりだった。	不明	• 日本経済新聞夕刊（2016.7.29）
		2016	7	29	10時12分				
1624	大阪取引所 J-GATE	2016	8	1	午前	国債先物のオプション取引に関連した一部機能を停止した。	7月29日に発生したシステムトラブル（項番1623）は、この機能の処理が遅くなったことが一因と見られる。 処理遅延の原因や対策が判明するまで、投資家の利用を制限する。 制限するのはオプションを乗り換える際に市場価格を参考にする機能で、銀行など一部投資家が利用する。	不明	• 日本経済新聞電子版（2016.8.1）
		2016	8	1					
1625	東京電力 パワーグリッド 電気使用量 システム	2016	8	23		電力小売りを手掛ける事業者に、電気使用量を誤って伝えていた。 影響があったのは、家庭や商店などとの契約8,531件。 4～8月の使用量が最大で数百倍になっており、過大請求につながった可能性がある。	新電力や東電の新プランに切替えた家庭は、アナログ式の電力量計から新型のスマートメータに交換したが、その際、検針員が旧電力計の数値を読み間違えてシステムに登録した。検針システムの不具合も重なった。	• ヒューマンエラー • 他システムの不具合	• 朝日新聞朝刊（2016.8.24）
1626	三重銀行 ATM	2016	8	28	13時14分	システム障害によって、愛知県、三重県に設置したATM232台が利用できなくなった。 午後9時までに192台が復旧し、残りの40台についても翌日の営業開始までに復旧見込みと発表した。	中央のシステムと、個々のATMをつなぐネットワークに障害が発生した。	ネットワーク障害	• 日本経済新聞（名古屋版）朝刊（2016.8.29）
		2016	8	29	朝				

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				
1627	日本生命保険相互会社	2016	8	31	09時45分	保険金・給付金の支払いについて、顧客約5,000人の手続きが1日程度遅延した。 支払い期日に遅れる場合は、所定の遅延利息を付して支払うとしている。	オンラインシステムにおいて使用するコンピュータのハードディスク装置に障害が発生した。システムは15時45分に復旧したが、支払いの手続きが遅延するなどの影響が発生した。	ハードウェア障害	• 日本生命保険相互会社プレスリリース（2016.8.31） • 日本経済新聞朝刊（2016.9.1）
		2016	8	31	15時45分				
1628	横浜銀行 七十七銀行 北海道銀行 北陸銀行 勘定系システム「MEJAR」	2016	9	16	14時00分	ＡＴＭのシステム障害で、35,900件の振込みなどの取引ができなくなっていた。	磁気情報が消失したキャッシュカードをATMで利用した際、本来は取引を不成立とする。ところが、一部の処理においてプログラムミスがあり、取引を不成立にできないままシステム内部で繰り返しエラーが出る事態に陥った。 再発防止策として、磁気情報が消失している場合は、事前チェック段階で取引を不成立にするようプログラムを修正した。	ソフトウェア障害	• 日本経済新聞朝刊（2016.9.29） • 日経コンピュータ電子版（2016.10.3） • 日経コンピュータ（2016.11.10）
		2016	9	16	20時00分				
1629	東京国際映画祭電子チケット販売システム	2016	10	15	13時30分	10月15日12時からのチケット発売開始後、アクセスが集中し、1時間半あまりでシステムが停止した。	発売開始直後から、webサーバに1分間で3万7,000件を超えるアクセスが集中した。（想定の6倍以上） 新たにクラウド上に構築した本システムは、利用状況に応じて処理能力を増やせる仕組みだったが、最大限に拡張しても対応できなかった。チケット購入希望者のアクセスの仕方が、通常の映画ファンとは異なり、短い時間で何度もアクセスを繰り返すことを認識できていなかった。 対策としては、チケット販売期間を前半と後半に分けるなど、処理が集中しないよう運用面の改善も検討する。	アクセスの集中	• 日経コンピュータ（2016.11.24）
		2016	10	19	12時00分				
1630	J-LIS マイナンバーカード 交付システム	2016	10	22	08時00分	システムの不具合で、約3時間、自治体がカードを交付できなくなった。	中継サーバ1号機のハードウェア故障が原因。 ４台の中継サーバ（個人番号カード管理システムを構成するサーバで、市町村と機構との通信を中継するもの）のうち、1台（1号機）が正常に動作しなかった。 中継サーバ１号機を切り離し、残り3台のサーバに分散させることで復旧した。 今後、詳細な原因分析等のため、ログ解析等を行う。	ハードウェア障害	• J-LISプレスリリース（2016.10.24） • 朝日新聞朝刊（2016.10.25）
		2016	10	22	10時52分				
1631	JR東日本 モバイルSuica	2016	10	25	07時30分	同日から開始されたApple Payサービスの影響で、サービスへのアクセスが集中し、利用がしにくくなった。	SuicaのApple Payの登録や、モバイルSuicaを使ったグリーン券の購入など、オンラインサービスが利用しづらくなった。チャージ済みの残高を使っての改札通過や買い物など、サーバとの通信が不要なサービスへの影響はなかった。	アクセスの集中	• 朝日新聞電子版（2016.10.25） • 日本経済新聞夕刊（2016.10.25） • 毎日新聞夕刊（2016.10.25） • ITmediaニュース（2016.10.25）
		2016	10	25	11時00分				
1632	松井証券 取引システム	2016	11	7	09時00分	個人投資家向け取引システムに障害が発生し、注文受付や約定処理に数分の遅れが生じた。 1日約4万人の個人投資家らが売買したいタイミングで取引ができなかった。	原因調査中。	不明	• 朝日新聞朝刊（2016.11.8） • 日本経済新聞朝刊（2016.11.8）
		2016	11	7	16時00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				
1633	ANA 羽田空港手荷物 搬送システム	2016	11	10	18時25分	乗客から預かった荷物を数えるシステムに障害が発生した。羽田を出発する計18便が30分以上遅延したほか、12便で荷物を一部搭載できずに出発し、後続便で運ぶ事態となった。約4,500人に影響が出た。	原因調査中。	不明	• ANAプレスリリース（2016.11.10） • 毎日新聞電子版（2016.11.10） • Aviation Wire（2016.11.10）
		2016	11	10	19時17分				
1634	東京商品取引所	2016	11	10	03時15分 10時00分	全商品の取引を一時停止した。（同様の障害が2回発生）	注文件数が処理能力の上限に達したため。この日は米国大統領選でトランプ氏が市場予想に反して当選したことで、為替相場が大きく変動していた。	アクセスの集中	• 日本経済新聞朝刊（2016.11.15）
		2016	11	10	05時30分 10時30分				
1635	Jアラート （茨城県高萩市）	2016	11	22		11月22日の福島県沖の地震で、震度5弱を観測した茨城県高萩市の全国瞬時警報システム（Jアラート）が作動していなかった。 受信した情報を住民に伝える防災行政無線（市内34箇所）も起動しなかった。	通信障害が起きたためとみられるが原因は分かっていない。	不明	• 日本経済新聞朝刊（2016.11.23）
1636	Jアラート	2016	11	29		Jアラートの一斉訓練で、防災行政無線から訓練情報が放送されないなどのトラブルが13都道府県の24市区町村であった。	機器の設定ミスや配線不良、故障が原因。	ハードウェア障害	• 朝日新聞朝刊（2016.11.29）
1637	損保ジャパン 日本興亜保険金 支払いシステム	2016	12	1		自動車保険の等級適用を誤り、契約者343人から保険料を計1,800万円多く受け取っていたと発表した。	担当者の入力ミスが原因。運転手に過失がない事故など、本来は事故後の等級が下がらないか、1等級だけ下がる事故で、3等級下がる事例があった。契約者からの問い合わせで分かった。 さかのぼれる2008年4月以降の契約分で間違いがあれば返金する。 今後は入力ミスが起こらないように改善する。	ヒューマンエラー	• 損保ジャパン日本興亜プレスリリース（2016.12.1） • 朝日新聞朝刊（2016.12.2） ※障害発生は2008年以前であるが、それが判明した日時に基づき掲載。
1638	みずほ証券	2016	12	8	09時00分	株式や投資信託を取引する専用画面にログインしにくい状況が続いた。	利用者のアクセスが集中し、処理速度が低下したことが原因。	アクセスの集中	• 朝日新聞朝刊（2016.12.10）
		2016	12	8	10時00分				
1639	気象庁 ホームページ	2016	12	11	午前中	ホームページにデータを送信するシステムに不具合が発生し、一部のページが自動更新ができなくなるトラブルが起きた。 手動更新で同日深夜には最新の情報が見られる状態にし、12日朝までに復旧した。	原因調査中。	不明	• 日本経済新聞夕刊（2016.12.12） • 毎日新聞夕刊（2016.12.12）
		2016	12	12	朝				

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				
1640	横浜市 住基ネット	2016	12	7	夜	住民基本台帳ネットワークに障害が発生し、マイナンバーカードを受け取りに区役所を訪れた約1,200人に交付できなかった。	誤った手順書に従ったサーバのメンテナンス作業が原因。（作業のチェックシートから必要な項目が抜けていた）	作業ミス	• 日本経済新聞夕刊（2016.12.21）
		2016	12	8					
1641	後期高齢者 医療制度 保険料計算 システム	2016	12	27		厚生労働省は12月27日、後期高齢者医療制度で、保険料の過大・過小徴収があったと発表した。制度開始の2008年から続いており、対象者は約2万人、誤徴収額は約6億円に上る可能性がある。	保険料を計算する都道府県ごとの「後期高齢者医療広域連合」の一部が11年以降、システムの不備を厚労省に指摘した。同省は問い合わせのあった広域連合には個別に正しい計算方法を伝えたが、全国への周知やシステムの改修はしなかった。	ソフトウェア障害	• 日本経済新聞朝刊（2016.12.28） ※障害発生は2008年であるが、それが判明した日時に基づき掲載。

事例1635、1636はいずれもJアラートの障害である。幸い実害があったわけではないが、いざとい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。2015年にも緊急時に使われる自治体の防災情報システムのトラブルが4件報道されている（事例1506、1531、1532、1541）[松田2 2015][松田1 2016]。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、一般のシステムとは違った考え方が必要である。

また、システムへのアクセスの集中をきっかけとする障害が今期も4件報告されている。この種の障害は過去にも多数発生しており、本連載でもたびたび取り上げてきたが[松田1 2015]、次の3節で再度取り上げる。一方事例1628は、たった一枚のカードのエラーが大きな障害に拡大した、共同利用型のシステムで発生した特異な事例であり、第4節で取り上げる。

3．システムへのアクセス集中による障害

今期報告された事例の20%に当たる4件（事例1629、1631、1634、1638）は、どれも想定を超える大量のアクセスがシステムに集中したことによって発生した障害である。事例1638については、なぜそのようなアクセス集中が発生したのかは不明であるが、そのほかの3件については、それぞれアクセス集中の原因となる事象は明らかになっている。すなわち、事例1631のモバイルSuicaの管理システムの障害事例は、当日に新型iPhoneによるApple Payサービスが開始され、その登録のためのアクセスが集中したためである。事例1634は、アメリカ大統領選挙においてトランプ氏が当選を決めた日に発生したものであり、事前の大方の予想を覆す結果に為替相場が大きく変動したため、商品取引が急増したことが原因である。また、事例1629は東京国際映画祭のチケット発売開始直後に想定を超えるアクセスが集中し、発売開始後1時間半あまりでシステムが停止したものである。報道[日経BP2 2016]によれば、それ以前のチケット販売の実績データなどから6000件/分のアクセスを想定し準備を整えていたが、実際には3万7000件/分を超えるアクセスが殺到したことがきっかけとなった。アクセス集中によって処理能力が逼迫（ひっぱく）したため、料金支払いのための決済会社からの支払い承認通知が正常に受け取れず、タイムアウトになり、いったん確保できた座席をキャンセルする処理を行うなど、処理の滞留が更なる滞留を呼ぶ悪循環に陥った。このため、決済処理を行う機能を別システムに切り出し、座席予約システムの過負荷の影響を受けない形に変更した結果、決済会社からの承認通知をスムーズに受け取り処理できるようになったとのことである。クラウドサービスを利用していたため、このような構成の変更やハードの準備は短時間で可能であったが、アクセス急増に対して処理能力を最大限拡張しても対応できなかった模様である。

また、購入希望者はシステムの混乱のためにチケット購入ができなかったと思い、予約状態を確認したり、必要以上に購入しようと短時間に繰り返しアクセスするなど、予想を超えるアクセスをしたために、更にアクセス数が増大し混乱が拡大した。

この事例では、アクセス集中の見積りや利用者習性の見通しの甘さ、処理の停滞を加速させるような処理方式の問題、更にはクラウドのスケールアウト機能の限界など、多様な要因が重なっており、今後の対策に関する多くの示唆を得ることができる。

4．共同利用型システムのリスク

事例1628は４つの銀行で共同利用するシステムで発生した障害である。日経コンピュータの記事［日経BP1 2016］には原因の詳細が示されているが、それによると、今回の障害はある銀行のユーザがキャッシュカードを使ってATMから振込操作をしたところ、そのカードの一部の磁気情報が壊れていた。そのことが引き金となって、振込処理が停止、振込以外のほかの取引も停止、更には共同利用していたほかの銀行の取引も停止となり、全体で３万5900件の取引が不能となる大きなトラブルとなった。本来、カードの磁気情報が壊れていた場合には、その取引は不成立として処理を終わるべきところ、「他行カードを使う他行への振込」という特定の処理では、壊れていた磁気情報の部分のチェックがすり抜け、他行宛てに振込を依頼する処理へと進んでしまった。他行宛振込依頼は、銀行間を接続するセンターに対して振込依頼を送信する、別のプロセスが分担する。しかし、壊れた情報が渡ったために振込依頼電文が正常に作れず、センターへの電文送信をせずにプロセスは異常終了した。ところが、センターからの応答を監視するための監視プロセスでは、30秒の監視タイマーが誤って残ったままであったため、タイムアウトが発生。監視プロセスは、タイムアウトが発生した振込処理の取消を依頼したが、取消処理を行うべきプロセスは異常終了してしまっており取消ができなかった。一方、監視プロセスでは取消処理の完了を２秒ごとのタイマーで監視していたが、取消処理が完了しないために２秒ごとに繰り返しエラーを発報、そのたびに次々とメモリを食い潰し、そのほかの処理も停止してしまった。こうしてたった一枚の磁気情報が壊れたカードを使った振込が原因で、４つの銀行の取引を７時間以上にわたって止めてしまう大事故となった。

共同利用型システムでは、共通のリソースを食い潰すような不具合があると、自らのシステムだけではなく、共同利用しているほかのシステム全体に影響を及ぼすという事例である。利用者ごとに利用できるリソースの上限をあらかじめ定めておけばこのような事象は避けられるが、リソースの分割損を生み、共同利用型の利点である大群化効果による効率性・経済性を損なうことになる。

過去にも共同利用型のシステムでの事故は報告されており、事例1212はデータセンターの電源故障、事例1214は保守作業のミスによるデータ消失であり［松田 2012］、それぞれ原因は異なるが、いずれも影響が広範囲に及ぶ大きな事故となった。SECでは、障害事例からシステム高信頼化のための教訓を得る活動を継続しているが、本事例と類似の共同利用型システムの障害を取り上げ、教訓化（G5、G8など）しているので参考にしていただきたい［SEC1 2016］。

5．むすび

2016年後半６カ月間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。本文中でも触れたが、SECではこれらの障害事例を分析し、開発・運用に当たって参考にすべき教訓を汲み取る活動を進めている。そして、教訓がまとまるごとに逐次Webサイトで公開しているので参照していただきたい。

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

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IPA/SECの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。更に、年度を区切りとしてまとめ、教訓集として公開・出版しているので併せて参考にしていただきたい［SEC1 2016］［SEC2 2016］［SEC3 2016］。

失敗の経験を社会の共通の財産として共有し、少しでも事故を防ぎ、安心・安全なIT社会に向けて地道な努力を続けていく必要があり、関係者のご理解、ご協力を期待したい。

参考文献

［松田 2012］松田晃一・大高　浩：情報システムの障害状況　2012年前半データ、SEC journal No.30、Vol. 8, No3, pp.12-pp.14, Sep.2012
［松田1 2015］松田晃一・八嶋俊介他：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2 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1 2016］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2 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
［SEC1 2016］情報処理推進機構　SEC：情報処理システム高信頼化教訓集（2015年度版）（ITサービス編）、2016年３月
［SEC2 2016］情報処理システム高信頼化教訓作成ガイドブック（ITサービス編）、2016年２月
［SEC3 2016］情報処理推進機構　SEC：情報処理システム高信頼化教訓活用ガイドブック（ITサービス編）、2016年２月
［ITメディア 2017］ITメディア　スマートジャパン
http://www.itmedia.co.jp/smartjapan/articles/1701/06/news017.html
［日経BP1 2016］日経コンピュータ　動かないコンピュータ2016.11.10
［日経BP2 2016］日経コンピュータ　動かないコンピュータ2016.11.24

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、その取り組みの一つとして、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、2015年７月から2016年６月までに投稿された合計14編のうち、査読者による審査を経て、SEC journal に採録された６編の論文を候補とし、そこから更に選考委員会と表彰委員会による厳正な審査の結果、２編を選出いたしました。
各賞の発表と表彰式は2016年11月17日にEmbedded Technology 2016内で実施いたしました。本年は最優秀賞は該当なし、優秀賞１編、所長賞１編が選出されました。

SECjournal論文賞表彰委員会 委員

委員長	片山　卓也	北陸先端科学技術大学院大学 名誉教授
委　員（50音順）	有賀　貞一	AITコンサルティング株式会社 代表取締役
	岩野　和生	国立研究開発法人科学技術振興機構 研究開発戦略センター システム・情報科学技術ユニット 上席フェロー 東京工業大学 大学院 イノベーションマネジメント研究科 情報・サービスイノベーション分野 客員教授
	大島　啓二	一般財団法人日本科学技術連盟
	大原　茂之	一般社団法人スキルマネージメント協会 理事長
	土井　美和子	国立研究開発法人情報通信研究機構 監事
	松田　晃一	独立行政法人情報処理推進機構 顧問
	松本　隆明	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

SECjournal論文賞選考委員会 委員

委　員（50音順）	飯泉　紀子	株式会社日立ハイテクノロジーズ 経営戦略本部 専門部長
	兼本　茂	会津大学 コンピュータ理工学部 教授
	紫合　治	東京電機大学 情報環境学部 情報環境学科 教授
	新谷　勝利	新谷ITコンサルティング 代表
	寺中　勝美	NTTソフトウェア株式会社 経営企画部 エグゼクティブ アドバイザー
	古山　恒夫	東海大学 理学部 客員教授
	水野　修	京都工芸繊維大学 情報工学・人間科学系 准教授
	神谷　芳樹	みたに先端研合同会社 本店 代表社員
	峯　恒憲	九州大学 大学院 システム情報科学研究院 准教授
	森崎　修司	名古屋大学 大学院 情報科学研究科 准教授
	山城　明宏	東芝ソリューション株式会社 ソリューションセンター 主幹
	山本　修一郎	名古屋大学 大学院 情報科学研究科 教授
	山本　雅基	名古屋大学 大学院 情報科学研究科 附属組込みシステム研究センター 特任教授
	山本　里枝子	株式会社富士通研究所 システム技術研究所 所長
	鷲崎　弘宜	早稲田大学 理工学術院 基幹理工学部 情報理工学科 教授

選考委員会では、全委員の査読結果を含め、対象論文の審査を行った。
ただし、委員が著者の論文や委員の関係者の論文については、該当委員は審査を行っていない。

※委員は50音順に掲載。敬称略　　※所属、肩書きは2016年11月当時

【優秀賞】

プロセス改善技術者育成コースの設計と実装

久野 倫義、中島 毅、芝田 晃、近藤 聖久、小笠原 公一
（SEC journal 46号掲載）

【所長賞】

Goal Structuring Notationを用いた汎用的な安全要求の明確化と評価

柿本 和希、川口 真司、高井 利憲、石濱 直樹、飯田 元、片平 真史
（SEC journal 47号掲載）

SECjournal論文賞 2016

（敬称略）

上段左より、松田 晃一、大島 啓二、大原 茂之、松本 隆明
片山 卓也、久野 倫義、柿本 和希、川浦 立志（IPA 理事）

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
名誉教授
片山　卓也

SEC journalは我が国のソフトウェア産業政策の一環として発行されているジャーナルであり、ソフトウェア開発現場での先端技術の実践や開発の報告、論文の掲載などを通して我が国のソフトウェア産業、IT産業の技術力向上に貢献してきました。そして、そのような論文の中からとくに優れたものを毎年選び表彰を行ってきました。今回は、2015年７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いたしました。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いたしました。

「プロセス改善技術者育成コースの設計と実装」

企業におけるプロセス改善技術者（SEPG要員）の育成と教育コースの設計について述べたもので、SEPG要員が現場で直面する実際的な課題をベースとした、極めて実践的な内容である。QCD達成のために、本来重要な役割を果たすべきSEPG要員だが、企業の中でうとまれる存在になりがちであり、要員のモチベーションの低下が懸念される。

これに対し本論文で提案されている手法は、事例に基づき、現場で役立つスキルを身に付けることができる優れた育成コースであると共に、様々な産業界に横展開が可能であり、またベストプラクティスという意味で授賞に値すると評価された。

「Goal Structuring Notationを用いた汎用的な安全要求の明確化と評価」

国際規格などの汎用的に適用される安全要求は、比較的あいまいな記述を含んでいる場合が多くその解釈の誤りが安全性上の欠陥の混入や、コストの増大につながるおそれがある。これに対し本論文は、汎用的な安全要求が暗黙的に仮定する知識などの暗黙知をGoal Structuring Notation（GSN）を用いて明確化し、思い込みによる危険性の見過ごしを防止するというものである。安全性だけでなく、様々なリスクに対応でき応用範囲が広いこと、また、比較評価実験を行い、GSNの効果を具体的に検証し定量的に示した点などが高く評価され授賞に至った。

― 受賞者コメント ―

【優秀賞】 （SEC journal 46号掲載）

プロセス改善技術者育成コースの設計と実装

三菱電機株式会社 設計システム技術センター　**久野 倫義**


久野 倫義


中島 毅


芝田 晃


近藤 聖久


小笠原 公一

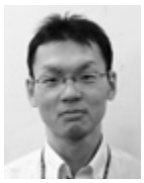
本論文は、ソフトウェア組織のプロセス改善に重要な役割を持つ「改善推進グループ」の要員（SEPG要員）を育成するための教育コースを設計・実践した事例をまとめたものである。一般に、SEPG要員に求められる技能は広範囲であり、プロセスにかかわる技術、開発技術（指導力）や管理技術（推進力）やパーソナルスキルなどを必要とする。このようなスキルを、いかに短期間の教育コースで効果的に獲得するかという点で困難である。この問題を解決するために、本コースでは、ソフトウェア開発現場でSEPG要員がよく経験する課題を解決できることを要求事項として設定し、受講者の職場の課題を題材に、３人１チーム＋指導者（リードアセッサ）の構成で、① 問題分析：問題設定、② 問題分析：課題抽出、③ 施策立案、④ ステークホルダへの説明（発表会）の４ステップを３回体験する演習を実施する。集中講義・演習は要求通り４日間という短期間に収めていると共に、アンケート評価により、受講者自身のスキル向上の自覚と、上司の期待通りの育成効果を上げていることが分かった。

【所長賞】 （SEC journal 47号掲載）

Goal Structuring Notationを用いた汎用的な安全要求の明確化と評価

奈良先端科学技術大学院大学 情報科学研究科　**柿本 和希**


柿本 和希


川口 真司


高井 利憲


石濱 直樹


飯田 元


片平 真史

汎用的な安全要求はあいまいな記述を含んでおり、解釈を誤ると要求の意図しない設計につながるため、手戻りによるコストの上昇や事故の原因となり得る。本研究では、汎用的な安全要求が過去の設計や事故事例などを、暗黙知として仮定していることに着目し、GSNを用いて暗黙知を明確にすることで、システムの安全性の向上を目指した。更にその有効性を評価するため、宇宙航空研究開発機構の技術職員を対象とした比較評価実験を行った。

比較評価実験の結果から、GSNを用いて明確化した安全要求を使用することで、システムの安全性が向上することが確認できた。この結果から、国際安全標準規格や、社内独自の安全規格及びチェックリストなどを用いている企業では、GSNを用いた暗黙知の明確化を検討する価値があると考える。また単に暗黙知の明確化という点に着目すれば、安全にかかわらず様々な領域に活用が可能であると考えられる。

本研究は、ご指導いただいた奈良先端科学技術大学院大学の先生方、並びに評価実験にご協力いただいた宇宙航空研究開発機構 第三研究ユニットの方々のご協力によって成果を発表することができた。深く感謝申し上げます。

Column

高生産性社会の到来

IPA顧問、学校法人・専門学校HAL 東京 校長 **鶴保 征城**

一昨年12月に電通の新入社員、高橋まつりさんが過労自殺したことが大きく報道された。労働基準監督署が労災認定し、直ちに任意の立ち入り調査が本社などに行われた。

立ち入り調査から１カ月も経たずに、労働基準監督官が本社と３支社に一斉に家宅捜索に入った。今回の行政のアクションは素早い。

更に、政府は働き方改革実現会議で議論を進めている。長時間労働の是正を目指して昨年10月に「過労死等防止対策白書」を公表、電通事件については「徹底的に究明する」姿勢だ。

高橋さんの事件は痛ましいが、「過ちては改むるに憚ること勿れ」だ。

電通は既に「全館22時消灯」の措置を実施しているが、表面的な対策だけでなく、石井直元社長が述べているように、新しい電通を作り上げるためには「意欲と真摯な姿勢」「プロフェッショナルの矜持」が欠かせない。

つまり、長時間労働と表裏一体である職場の体質改革が必要であるとの認識だ。

問題を指摘しても上司が握り潰したり、個人の能力に帰したりすべきではない。風通しが良く、社員が生き生きと働ける環境を作らなければならない。もとより各種のハラスメントがあってはならない。

以上のような問題はあるが、本質は日本企業における働き方の付加価値や生産性が低いことだ。

「自分たちのやっていることに価値があるのか、やり方の効率が低いのではないか」と疑ってみることが必要だと思う。

この背景には、世の中がかつてないほどの勢いで高付加価値、高生産性社会に向かっているという事情がある。

わかりやすい例として、UberとAirbnbが挙げられる。前者は、個人が自分の空き時間に自家用車で乗客を運び、賃料をもらう。後者は、個人の空き部屋や空き家を宿泊場所として貸し出す。

両者の利点は色々あるが、最大のものは不活用資産、Uberの場合は個人の時間と保有する自動車を活用するということだ。タクシーが街中を空きのままで走ったり、駅で何時間も待っていることを考えると、Uberの圧倒的な合理性、つまり生産性の高さが理解できる。

タクシー会社の運転手が不眠不休で働いても、売上高は伸びるが生産性は向上しない。重要なのは生産性であって、トータルの売上高ではない。

生産性が向上しなければ会社の業績が良くなることはありえず、無理して売り上げを上げようと思うと、どうしても過重労働にならざるを得ない。これが電通問題の根底にある。

UberやAirbnbが特殊な話かというとそうではなく、高生産性の動きはあらゆる分野に及んでいる。その手段はAI、ロボットの活用だ。

医療の分野は日進月歩だ。人間の医者と違って、コンピュータは年間数100万本以上の論文を容易に読み込める。AIを使ってそれらの情報を分析して診療に当たるということは、そう遠くない時期に実用化されると思われる。

ほかにも、判例検索、各種審査業務、更に自動運転等も考えられている。

では、どうすれば各企業は高生産性の波に対応できるのか。

日本ではAIやロボット出現のかなり前から、ITの利活用が提起されていたが、決して成果を上げているわけではない。むしろ、工場以外では日本はIT利活用後進国だと言っても過言ではない。

その原因だが筆者は、「経営者がIT利活用に熱心ではなかった」ことが大きいと思っている。工場は生産活動であり、品質や効率の指標が明確であったため、現場の積み上げである小集団活動が功を奏したのではないだろうか。

一方、ホワイトカラーを含む会社全体の生産性の改善は、「経営の改善」そのものと言える。にもかかわらず、工場の小集団活動と同じように現場に丸投げしていたのが問題だと思う。経営者の中には「ITのことは分からないから…」と逃げ腰になる人もいるが、これからは許されない。

経営者が先頭に立って、生産性改善、すなわち経営改善をリードすべき時代になった。

書籍紹介

新産業構造ビジョン

第４次産業革命をリードする日本の戦略

経済産業省経済産業政策局
産業再生課　編

ISBN：978-4-8065-2981-1
経済産業調査会刊
A5判・151ページ
定価1,800円（税抜）
2016年７月28日刊

本書は、経済産業省の産業構造審議会・新産業構造部会（部会長・伊藤元重東京大学教授）が、関係省庁と一体となって策定（平成28年４月27日に中間整理）した結果を書籍化したものである。

政策に関する解説書なのでじっくりと行間を読んでいく必要があるが、第４次産業革命（IoT、AI、ビッグデータ、ロボット）をリードし、日本のプレゼンスを上げる戦略の骨子がまとめられた野心作である。

「第４次産業革命の第一幕では、バーチャル空間は、GAFA（Google、Apple、Facebook、Amazon）支配下にあるが、第二幕では、リアルデータを生み出すリアル空間は、現場力の発揮如何で日本は競争優位なポジションを得る可能性を秘めている」と本書は主張している。経済社会システムの見直しはこれからであり、そこに、新産業構造部会は活路を見出している。

具体的には、基本戦略による産業構造の転換と、就業構造の転換を目指した７つの方針と７つの我が国の具体的な戦略が解説されている。しかし、参考資料集で指摘されているように日本の弱みとして、(1)パッケージソフトウェアの国際的な劣位、(2)データ利活用の人材不足、(3)ビジネスモデルの不足などの懸念点がある。これらは、まさにソフトウェア分野の弱みであり、我々自身が克服すべき課題である。

第４次産業革命の果実を取るには課題を克服する必要があるが、我々に与えられた時間はあまり多くない。スピード感溢れる実行戦略が求められている。

その意味で、真誠惻怛の思いを込め、マイケル・ポーターの競争戦略論と併せて現場レベルで熟読すべき本である。　（久保 忠伴）

アプレンティスシップ・パターン

徒弟制度に学ぶ熟練技術者の技と心得

Dave H. Hoover,
Adewale Oshineye　著
柴田 芳樹　訳

ISBN：978-4-87311-460-6
オライリー・ジャパン刊
A5判・216ページ
定価2,200円（税抜）
2010年７月刊

本書は「なりたて」のソフトウェア技術者が「熟練職人」になるための心得や行動パターンをまとめたものである。中世ヨーロッパに広く普及していた職人たちの「ギルド」における徒弟制度をモデルにして記述しているのが特徴である。

「無知をさらけ出す（Expose Your Ignorance）」、「学びを共有する（Share What You Learn）」、「自分の地図を描く（Draw Your Own Map）」、「練習、練習、練習（Practice, Practice, Practice）」など著者が出会ってきた熟練職人たちの「経験」から抽出された、徒弟に学んでもらいたいパターンがその解説とともに紹介されている。例えば、自分の知識不足を感じた時にどう振る舞うべきなのかについて、次のようにアドバイスがある。わからないことを質問することが大切。ただし、そのままだと生涯他のメンバに依存し続けることとなるので、「無知をさらけ出す」ことと「無知に向き合う」こと、そのバランスを取ることが重要である。

目から鱗が落ちるというものばかりではないが、eXtreme Programming、アジャイル、ペアプログラミングといったものが普及してきた現代こそ再認識すべきパターンがあると感じた。アプレンティスからジャーニーマン、今ではクラフトマンとなっている皆様にとっても、本書にあるいくつかのパターンは以前から持っていた疑問を解消してくれるものとなるのではないだろうか。　（遠藤 秀則）

編集後記

今号はシステムズエンジニアリングを特集しました。本誌ではシステム開発の解説が中心でしたが、システムズエンジニアリングは成功を手に入れるためのアプローチでその応用範囲は広く、今号巻頭言の狼先生が何年か前に次のような事例を紹介しています。“UAEの原子力発電所建設売り込みの際、ダークホースの韓国が受注したのは、売り込みの何年も前からシステムズエンジニアリングの定石を踏んだ周到な準備を進めていたから。”日本ではシステム的なアプローチや思考よりも情緒的に決定される傾向があるとも言われています。様々なモノやサービスが複雑につながるIoTの時代、俯瞰的に捉え超上流を意識しつつ、要素の細部にまで分解して目を光らせるシステムズエンジニアリングの考え方を参考にしていただければと思います。

さて、最近、ディープラーニングという言葉をよく見聞きします。多層ニューラルネットを利用した機械学習で特徴を自ら見付け出すようですが、ランダムな初期値では発散してうまく学習できないとのこと。目的を設定し、良い初期値を与えることがコンピューターが効率良く学習するポイントだとか。これって人間と一緒ですね。まもなく新入社員を迎える皆様も多いと思いますが、新入社員と接するとき、このディープラーニングの話を思い出してみてはいかがでしょうか。新年度を迎えましても、変わらぬお付き合いを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等、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

SEC journal 編集委員会

編集委員長　遠藤 秀則

編集委員（50音順）
荒川 明夫
石橋 正行
江野村 亮輔
日下 保裕
佐藤 康彦
中尾 昌善
長谷川 佳奈子
三原 幸博
室 修治
山下 博之
和田 恭

福寿草　撮影：M.Ooe

SEC journal 第12巻 第４号（通巻51号） 2017年３月１日発行

©独立行政法人情報処理推進機構 2017

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：http://www.ipa.go.jp/sec/　e-mail：sec-journal_customer@ipa.go.jp

※本誌は「著作権法」によって、著作権等の権利が保護されている著作物です。
※本誌に掲載されている会社名・製品名は、一般に各社の商標または登録商標です。

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。
http://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。http://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.48
第12巻第４号（通巻51号）
2017年３月１日発行

©独立行政法人情報処理推進機構

ISSN 1349-8622

VEGETABLE OIL INK
R70 古紙パルプ配合率70％再生紙を使用
リサイクル適性Ⓐ