安全性モデリングとSTAMP/STPA、その最新ツール紹介

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

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

1 モデルとは何か

モデルを定義することは難しいが、筆者なりに定義をすると、

現実世界の対象物を、ある目的で捨象し、
その目的下で扱いやすくした抽象物。

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

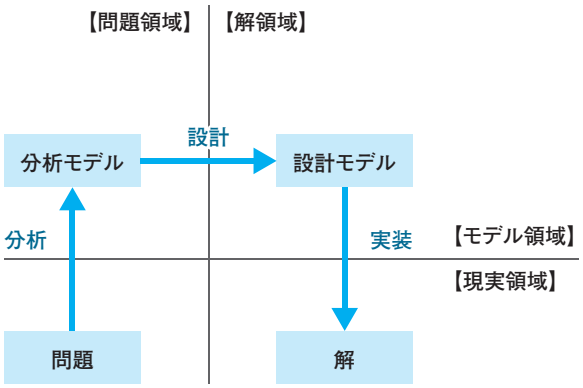

図1　モデルの役割

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

1.1 様々なモデリング言語

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

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

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

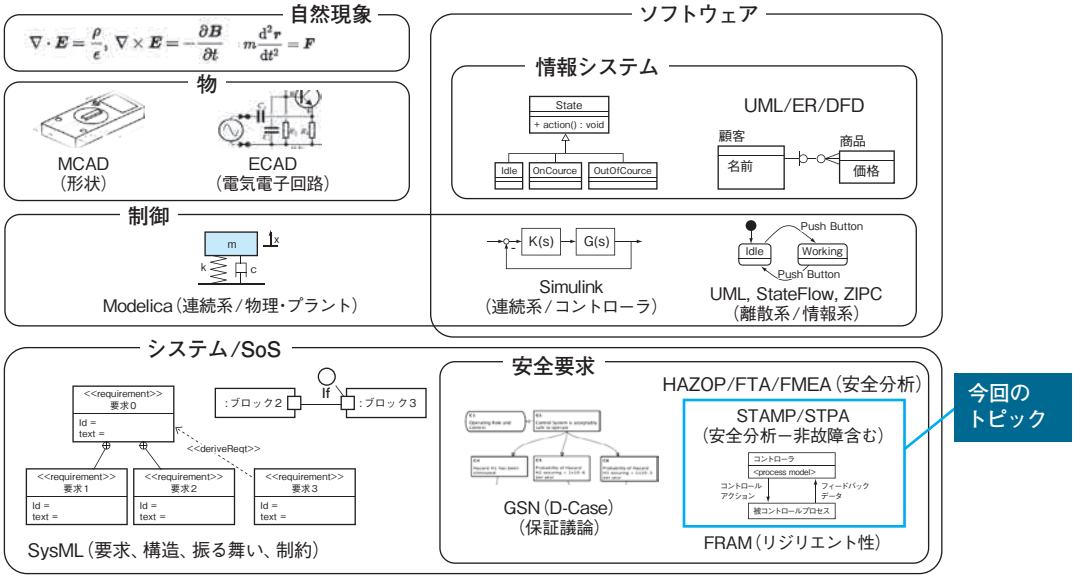

図2　システムのモデリング言語

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

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

1.2　安全分析のモデリング

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

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

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

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

2　STAMP/STPAと支援ツールの紹介

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

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

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

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

2.1　STAMP/STPA概説

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

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

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

2.1.1 Step0準備1

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

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

2.1.2 Step0準備2

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

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

表1　アクシデント、ハザード、安全制約の一覧表[2]

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

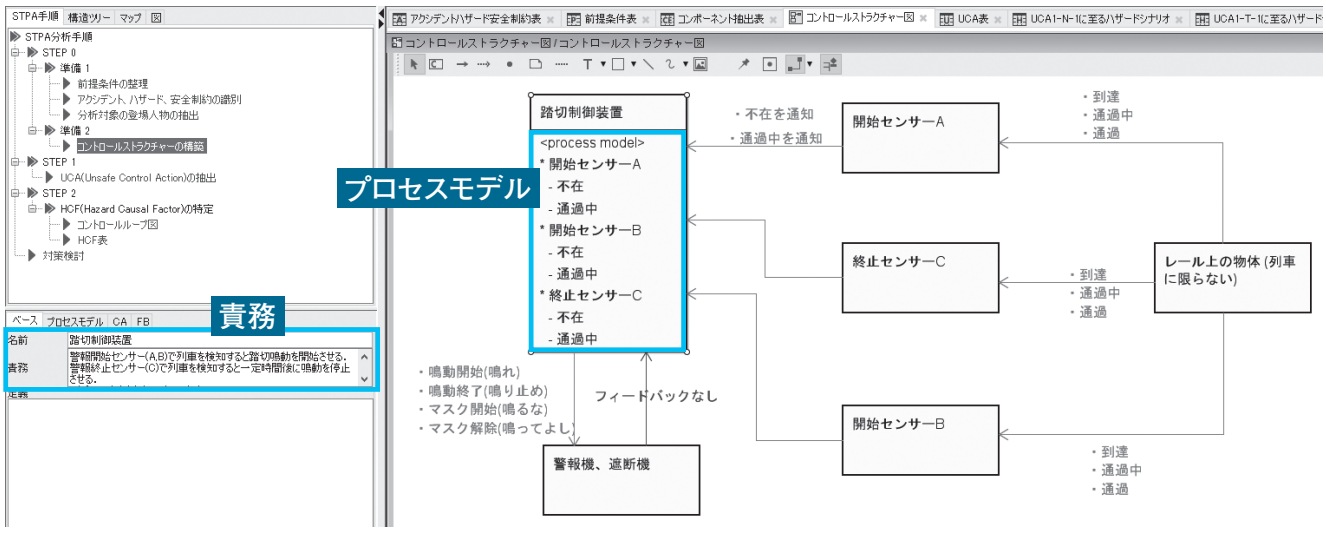

図3　コントロールストラクチャー図[2]

表2　UCA一覧表[2]

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

連携する場合には参照関係が壊れがちである。

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

2.1.3　Step1

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

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

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

2.1.4　Step2

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

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

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

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

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

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

2.2　STAMP/STPA支援ツール

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

表3　ハザード要因の一覧表

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

表4　ハザード要因の一覧表 [2]

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

2.2.1　XSTAMPP

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

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

2.2.2　SafetyHAT

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

2.3　STAMP Workbench

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

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

起こす修正作業など）から分析者を解放することを目指している。

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

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

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

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

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

2.4　2章のまとめ

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

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

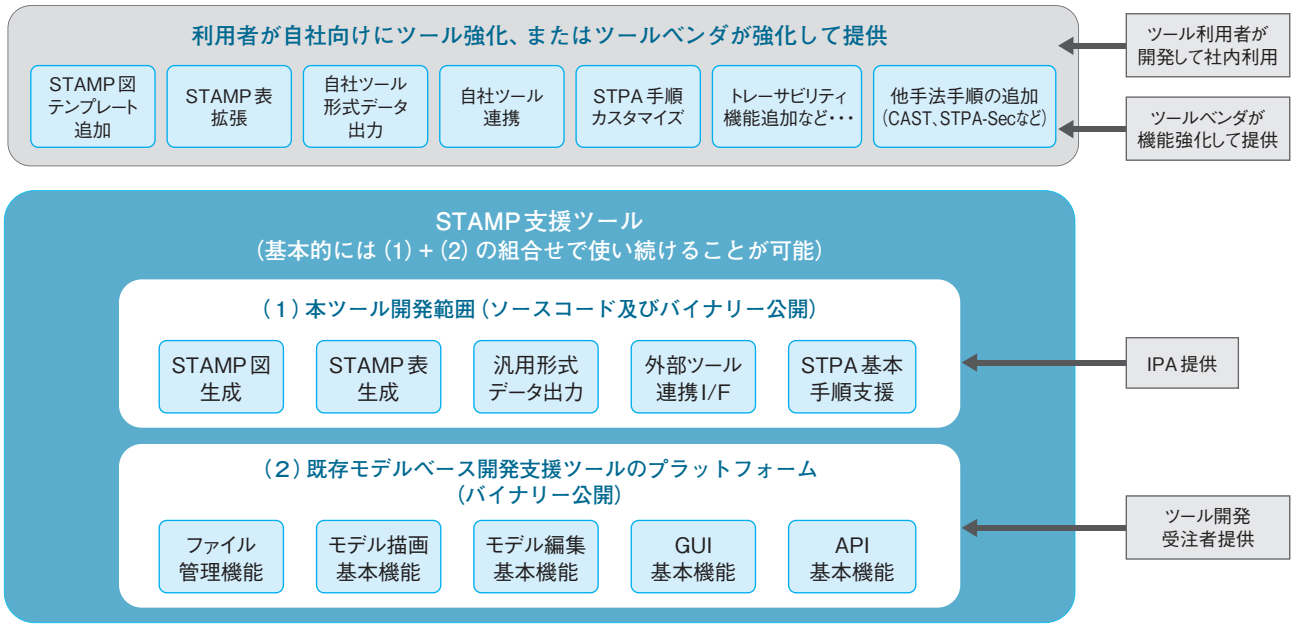

図4　STAMP Workbenchの構造

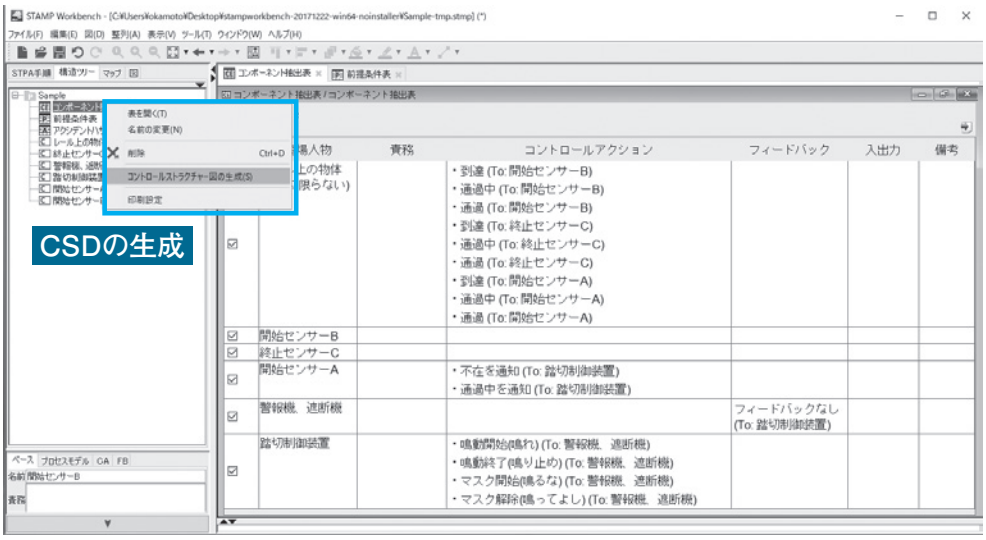

図5　コントロールストラクチャー図生成機能

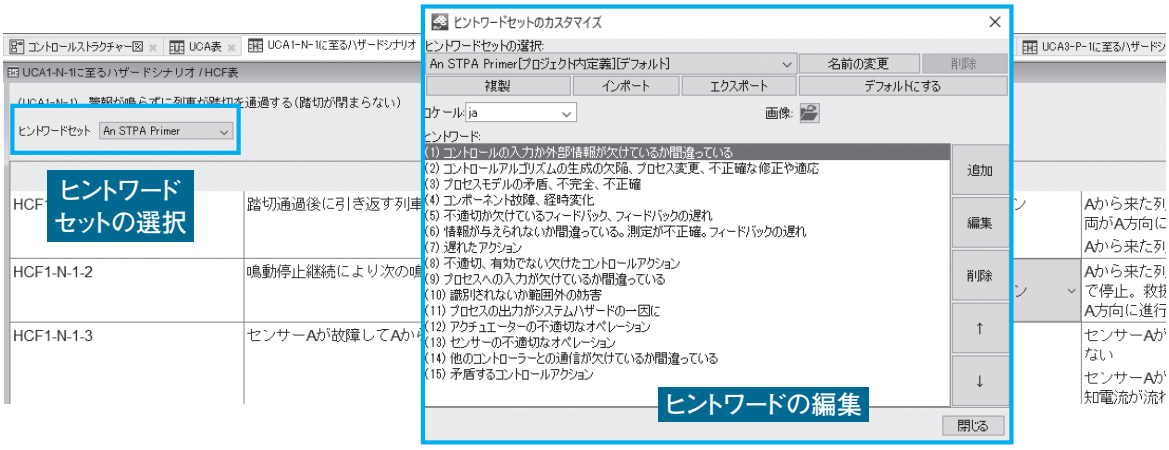

図6　HCFヒントの選択・編集機能

【参照文献】

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

[2] IPA, はじめてのSTAMP/STPA入門編, 2016.

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

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

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

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

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

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

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