システム思考によるソフトウェアの安全設計
～現状と今後～

2018年11月14日

余宮　尚志（（株）東芝　研究開発本部）
hisashi.yomiya@gmail.com
兼本　茂（会津大学　名誉教授）
shigeru.kanemoto@gmail.com

◆安全設計入門改訂版

電波新聞社 2010年7月発行

改訂版の主旨

企業でのソフトウェア技術者向けの安全設計の入門書として,
安全分析の基本手法(FTA,FMEA,HAZOP),
各種機能安全規格、
STAMP/STPA による新しい安全分析法などを広く学べる。

目次

1.安全の基本
2.安全規格体系と概要
6.自動車の機能安全(ISO 26262)
7.サービスロボットの機能安全(ISO 13482)
8.システム思考で考えるこれからの安全

2019 年 3 月発刊予定（電波新聞社）

本講演でもこの一部をお話します

国際安全規格の体系「ISO/IEC Guide 51」

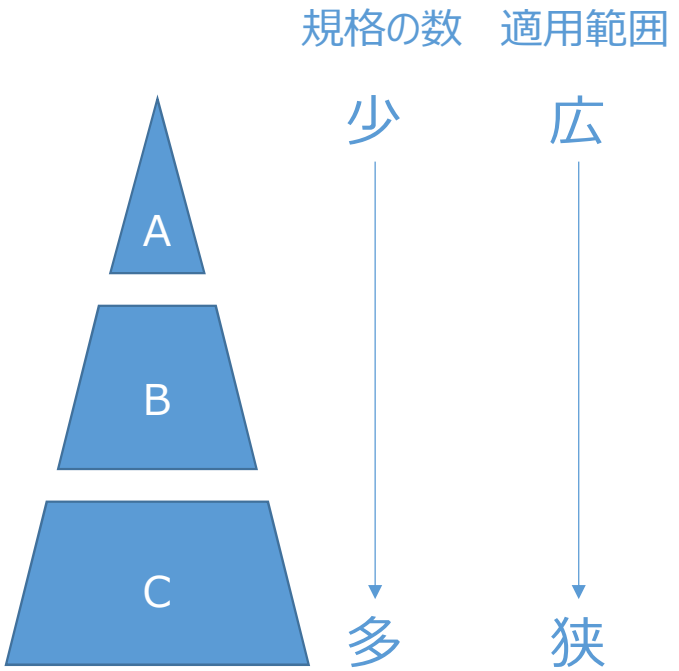
- ISOとIECが共同で策定した「ISO/IEC Guide 51」では、**安全規格の階層構造**や**基本的な用語**が定義されている

機械類の安全
リスクアセスメント
およびリスク低減

コンピュータ

自動車の電子制御系

規格の種類	ISO/IEC規格
基本安全規格 (タイプA規格)	• ISO 12100 • ISO 14121
グループ安全規格 (タイプB規格)	• ISO 13849 • IEC 60204 • IEC 61508 など
製品安全規格 (タイプC規格)	• ISO 12648 • IEC 60745 • ISO 26262 など



ISO/IEC Guide 51が定める安全規格の階層構造

国際安全規格を採用する理由(1)

- 各国が国際標準を採用する理由
  - 各国が自国の枠を超えて、規格の共通化を促す根拠にはWTO-TBT協定がある
  - この協定は貿易の**技術的障害**(Technical Barriers to Trade:TBT)**を**持たないように**取り除くため**のもの
  - 規格類を国際規格に整合化することで、工業製品や農業産品などの輸出入において、**不必要な貿易障害を取り除く**ことを目的としている

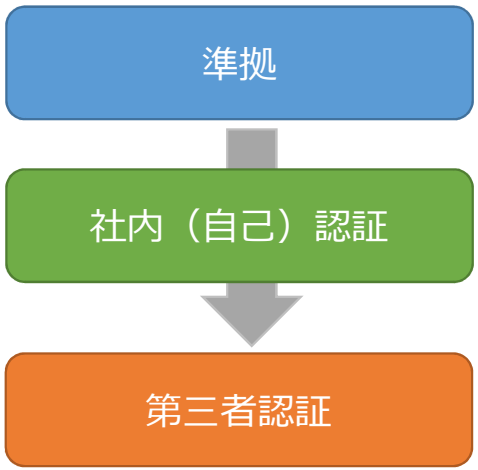
※電波新聞社「組込み系技術者のための安全設計入門」

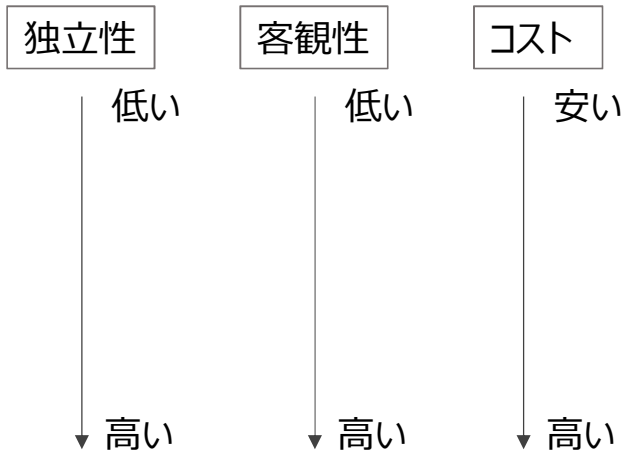
国際安全規格を採用する理由(2)

- 製造者や輸入者が国際標準を採用する理由
  - 製品が国際的な基準に適合していることを示すことができることがメリット
  - 製品が安全規格の認証を得ていることで、**製品の安全性を客観的に説明できる**ようになる
  - 万が一にも市場で事故が発生した場合など**説明責任を果たしやすくなる**

国際安全規格を採用する理由(3)

- 様々な立場によって国際安全規格に準拠する(参照する)目的は異なる
- 当社では…
  - 安全性確保の手段、説明責任遂行時の根拠とする　⇒ **<u>安全論証</u>**
  - 入札条件や顧客要求である
  - 他社との差別化や営業戦略として採用する





各々の安全規格の限界(適用範囲)

- 各々の安全規格には、適用範囲(Scope)が定められている
- IEC 61508
  - コンピュータを含む、安全機能が求められるさまざまな製品を対象としているが、コンピュータを含んでいない**製品(はさみなど)は対象外**であるし、**コンピュータで実現していない安全機能(逆流防止弁やクッション材など)の要求事項も規定していない**
- ISO 26262
  - IEC 61508の適用範囲であって、さらにISO 26262:2011では**車両総重量が最大3,500kgまでの量産される乗用車に限定されている**。ISO 26262:2018では、モータサイクルや、トラックやバス、トレイラーなどの商用大型車に適用範囲を広げているが、**モペットは除かれている**

安全規格の限界

- 各々の安全規格の限界(適用範囲)とは、あくまでその安全規格における要求事項を定めた適用範囲外を指す
- しかし、実際の製品では安全規格の限界にかかわらず包括的に安全方策を導入して、リスクを軽減する必要がある
- 現行における、ほとんどの安全規格の限界(適用範囲外の内容)　例
  - **ヒューマンエラーへの対応**

    合理的に予見可能でない誤使用や、予見可能であってもあらゆるヒューマンエラーへの対処を安全規格は要求していない
  - **新しい特性を持つ製品への対応**

    自動運転車や人工知能(AI)にかかわる製品等、これまで安全規格で扱いにくかった製品が市場に増えているが、こうした製品にも該当する安全規格がない。これとは別に、安全確保を図っていく必要がある

(参考) 安全規格とサイバーセキュリティ

- 安全規格は、使用者が(合理的に予見可能な誤使用も含めて)正しい使用方法を行うことを前提としている
- 実際には、製品が悪意のあるサイバーセキュリティの脅威にさらされることで、許容できないリスクが顕在化することが起こるかもしれない
- このような、サイバーセキュリティの脅威や脆弱性を適用範囲として取り込んでいない

自動運転車やAIに関連した製品のように、安全規格の発行よりも、新しい特性を持つ製品の開発が先行している事例がある。特にサイバーセキュリティによる脅威を考慮した安全確保は、安全規格の確立も含めて今後の新しい流れとなるかもしれない

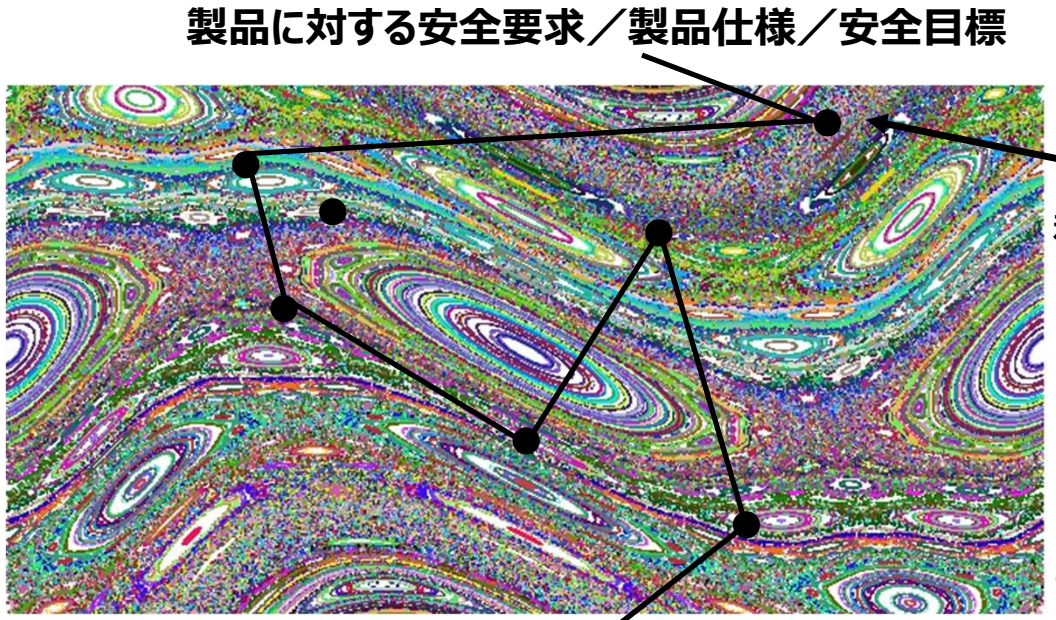
安全規格と製品安全を支える**安全論証**の考え方(1)

- 安全論証とは、**安全性を設計文章等のエビデンスを用いて、論理的かつ合理的に説明できること**(説明責任が果たせること)である

- すなわち、受容できないリスクがないことをエビデンスを用いて論理的に説明できることである

安全論証を成立させるために、最善を尽くしていることが、論理的な説明に有利となる

安全論証の(講演者の)イメージ図



安全論証とは、この実線を説明することである。説明するための構造化されたドキュメントの集まり(エビデンス)がセーフティケース。セーフティケースには安全目標やワークプロダクトのほか、議事録、電子メール、規格、動作環境、安全文化、コンピテンス管理、確証方策等(いずれもドキュメント化されていること)を含むことがある

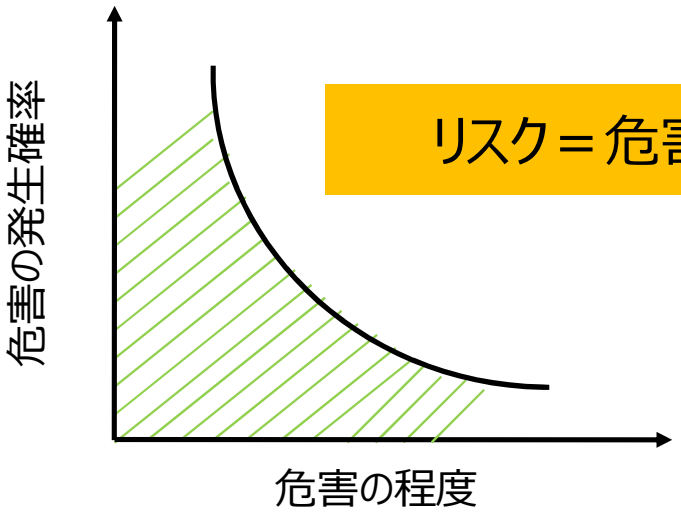
安全規格と製品安全を支える安全論証の考え方(2)

- これには、以下の二つの考え方が内在している

1. (想定する危険事象に対する)危害の発生確率と危害の程度の組み合わせが受容できるよう設計されていること　⇒　**安全目標(安全制約)の設定が可能**
2. 想定する危険事象に想定外がないこと

⇒　**想定外に対しては安全目標の設定ができない**

(想定外を減らす最善の努力をしたという論証は提示すべき)



リスク＝危害の発生確率およびその危害の程度の組み合わせ

- 「安全」とは、「受容できないリスクがないこと」（斜線部）
- 「組み合わせ」は、必ずしもかけ算とは限らない
- 受容できるリスクは、社会や時代によって異なる

※安全規格によっては異なる定義(文言)となっていることがあります

ソフトウェアにおける安全論証の考え方

- 一般論として、ソフトウェア特有の安全論証に対する考え方はない(共通である)

- 安全規格の要求事項を満たすこと、安全規格にある各フェーズの目的を達成すること
- ソフトウェアに関わる安全要求の実現を確実にすること
  - 特に、ソフトウェアにバグがないこと
  - ソフトウェアにバグがないことの確からしさを、客観的に説明できること

  ※ バグとは特に、安全要求の侵害に至るものを重視
- ソフトウェア業界の品質に関わる慣例(例:開発プロセスによる組織の成熟度モデル)を取り入れていること　etc

注）ソフトウェアに関わる安全要求に対するバグ

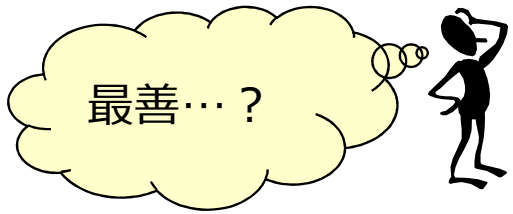
バグがないことを含めて、安全論証を支えている考え方は〝最善を尽くしているか〞

(コラム) 安全論証とSTPAに対する考察

- STAMP/STPAは…
  - 全ての部品や機能の故障(故障モード)を洗い出すといった分析に向いているわけではない
  - 例えば、ハザードに対して分析と対策が尽くされたことを説明できないかもしれない

STAMP/STPAの実施だけで安全論証は成立しない

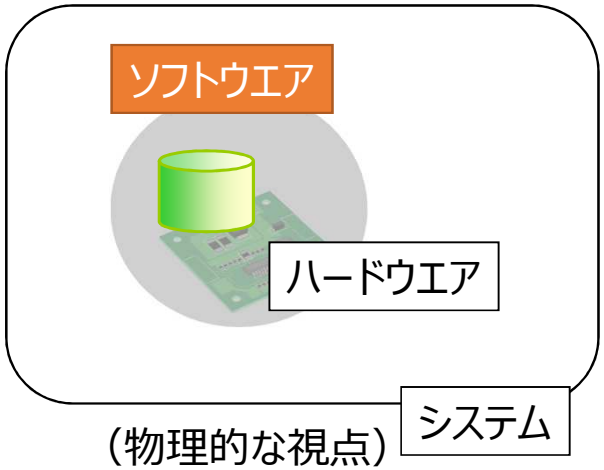
- ISO/IECは…
  - ある特定の制限/条件の下で、ハザードに対して分析と対策が尽くされたことを説明する
  - 新しい、今までに経験のない製品、ステークホルダー全員が素人であるような製品開発には対応しない (ex 自動運転車など)



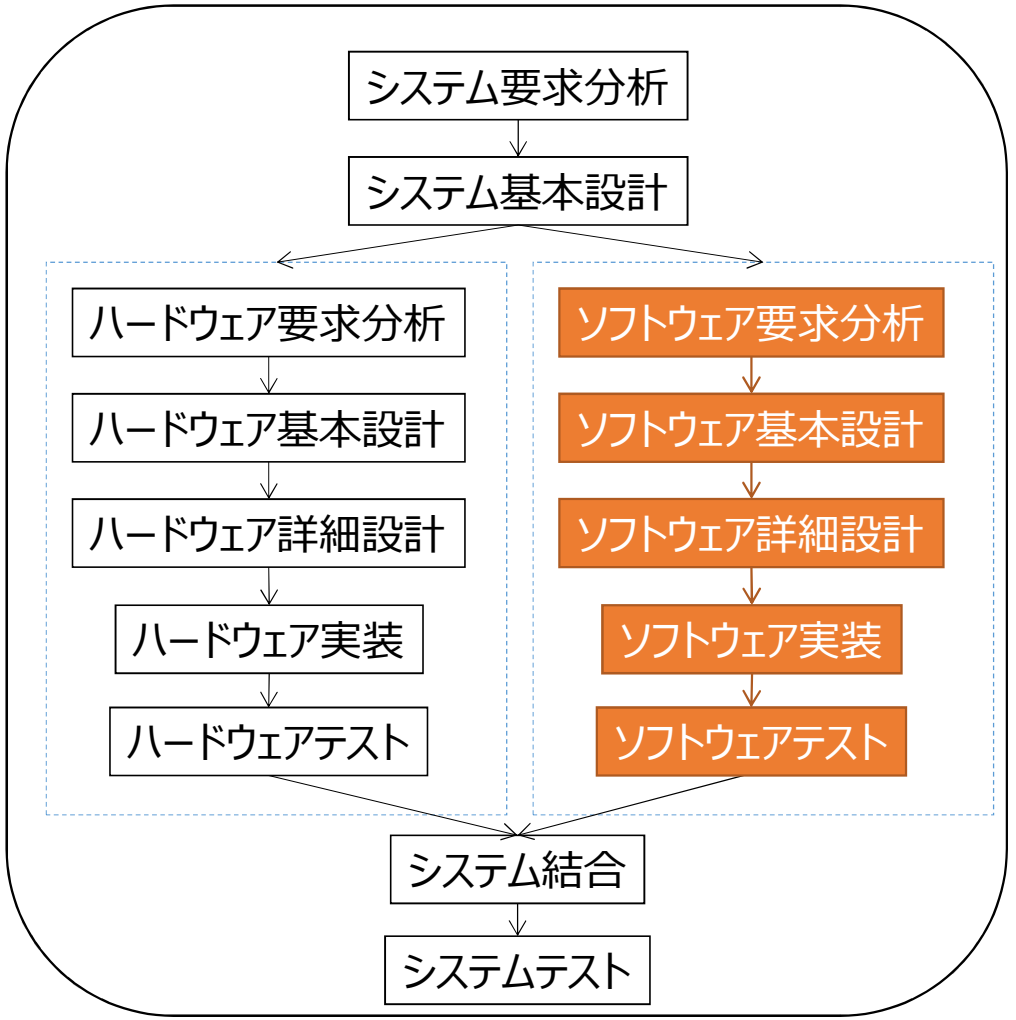
従来の分析手法だけでは難しい

※第二回STAMPワークショップの発表スライドより引用

ソフトウェアの安全設計



（物理的な視点）



（プロセスの視点）

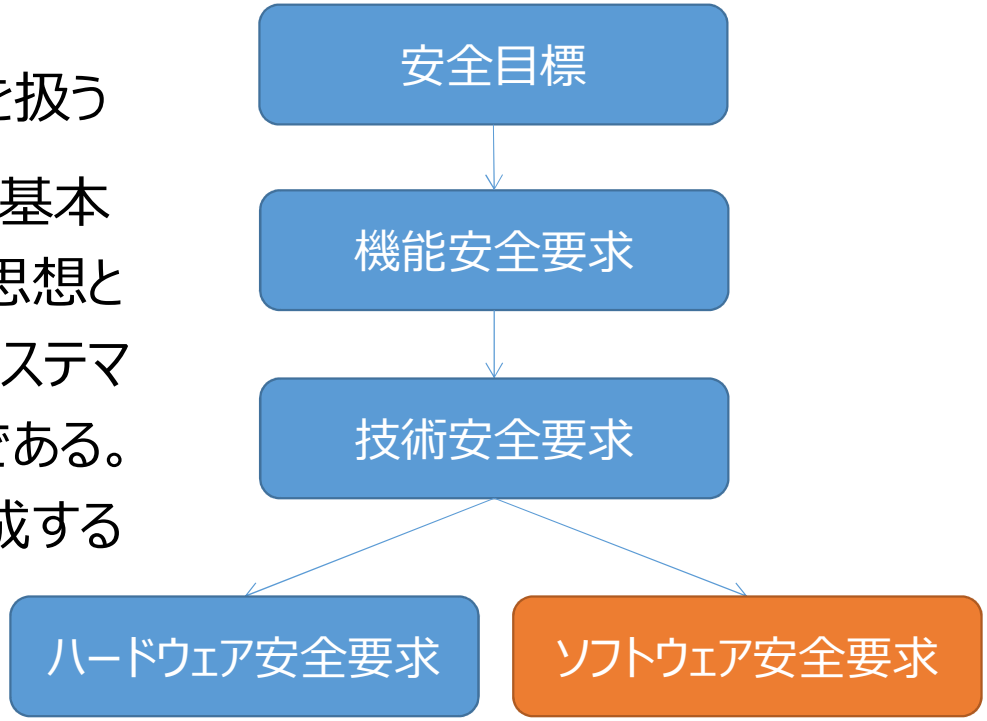
ソフトウェアだけで安全性を論ずることはできない

安全性を論ずるにはソフトウェアの考慮が不可欠である

ソフトウェアの安全設計~安全規格の目線から(1)

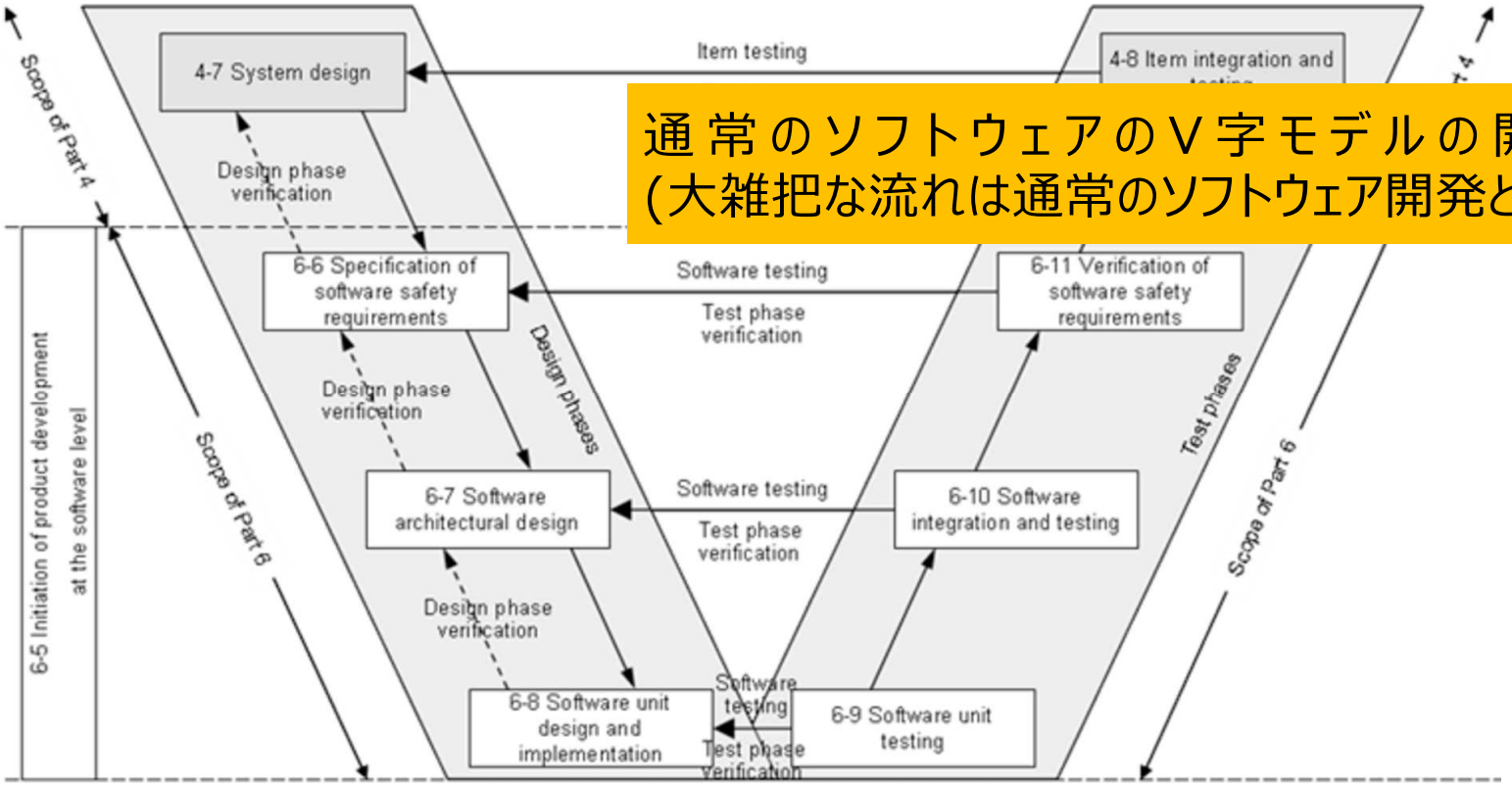
- ソフトウェア安全要求を(技術安全要求から)導出し、ソフトウェア設計を行う
- ソフトウェア設計では、システマティックフォールトを扱う
- 安全を達成するための安全分析も含まれるが、基本的にバグがないようソフトウェアを設計することが思想となっている。安全規格の要求事項の数々は、システマティックフォールトの混入を低減するための事項である。それによって、ソフトウェア安全要求を確実に達成することを説明する

注）ソフトウェアに関わる安全要求に対するバグ



ISO 26262の安全にかかわる要求

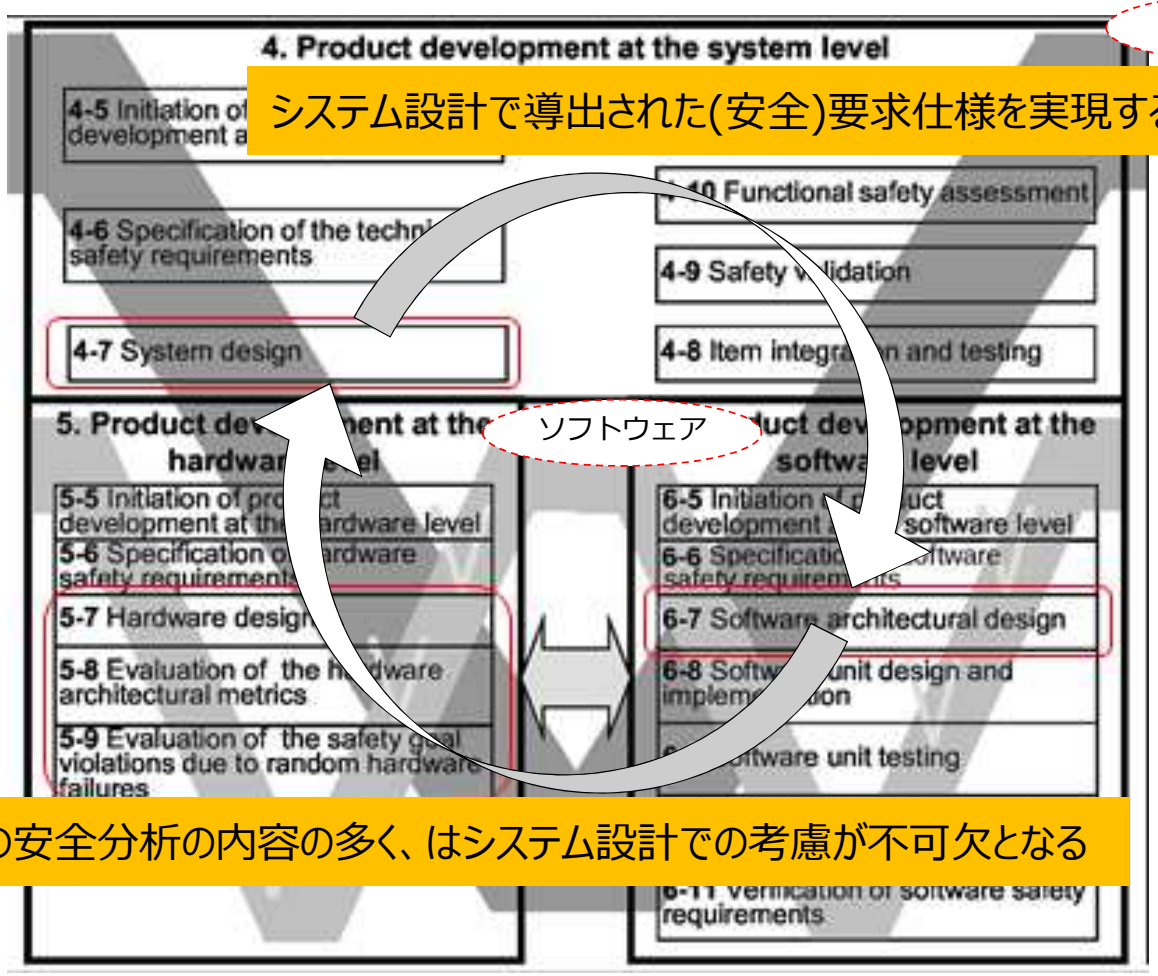
ソフトウェアの安全設計~安全規格の目線から(2)



通常のソフトウェアのV字モデルの開発である
(大雑把な流れは通常のソフトウェア開発と変わらない)

ISO 26262, Part6, Figure 2

(参考)ソフトウェアの安全設計~安全分析



システム設計で導出された(安全)要求仕様を実現するよう、ソフトウェアアーキテクチャ設計を行う

ソフトウェアでの安全分析の内容の多く、はシステム設計での考慮が不可欠となる

システム設計では、ソフトウェアの技術的な特性や、実現能力等を考慮する必要がある

ソフトウェアでの安全分析は、ソフトウェア安全要求を侵害しないことを、アーキテクチャレベルで実現すること、及びその検証を主な目的とする

(例)
イテレーションを許容するかシステムレベルの安全分析時に、ソフトウェアエンジニアが参加したりレビューをする

ソフトウェアの安全分析手法事例

- ソフトウェアの安全分析の主目的(主にアーキテクチャ設計レベル)
  - ① ソフトウェアにおける安全性の向上、安全の達成
  - ② ソフトウェアにおけるバグゼロの達成

次以降のスライドでは、ソフトウェア安全分析のひとつの考え方を紹介する
注) 安全論証の補強としても有効だが、安全論証には安全分析で最善を尽くすこと以外に、さまざまな視点がある(例：テストケースの十分性)

従来の安全分析手法の使用(当社の事例)

- FMEAでは、(還元的に)各部品・機能の故障モードがどのようにシステムに影響するかを分析する
  - 部品の網羅性

- FTAでは、安全上起きてはいけないことが起きていないことを検証する
  - 全ての安全目標・安全要求を侵害していないこと

- HAZOPは、FMEAやFTAの中で、ガイドワードの考え方を用いる
  - (例えば)故障モードの網羅性

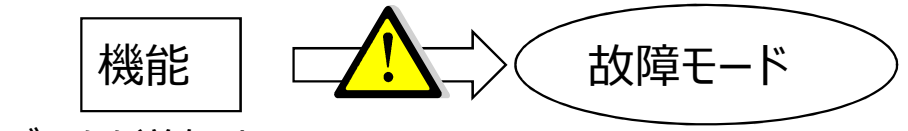
主に経験知に基づく網羅性によって安全性を論証する

ソフトウェアで安全分析をするときの課題(FMEAを例として)

1. 故障モード（故障）の抽出が難しい

【故障の捉え方(例)】

ハードウェア	ソフトウェア
経年劣化あり	経年劣化なし
物理特性の変化	条件の組み合わせ



データを送信する

データが送信できない

部品の果たす機能の裏返しになってしまう

×発想が広がらない
×壊れ方のパターンが定まらない
（影響の評価や対策の検討も十分にできない）

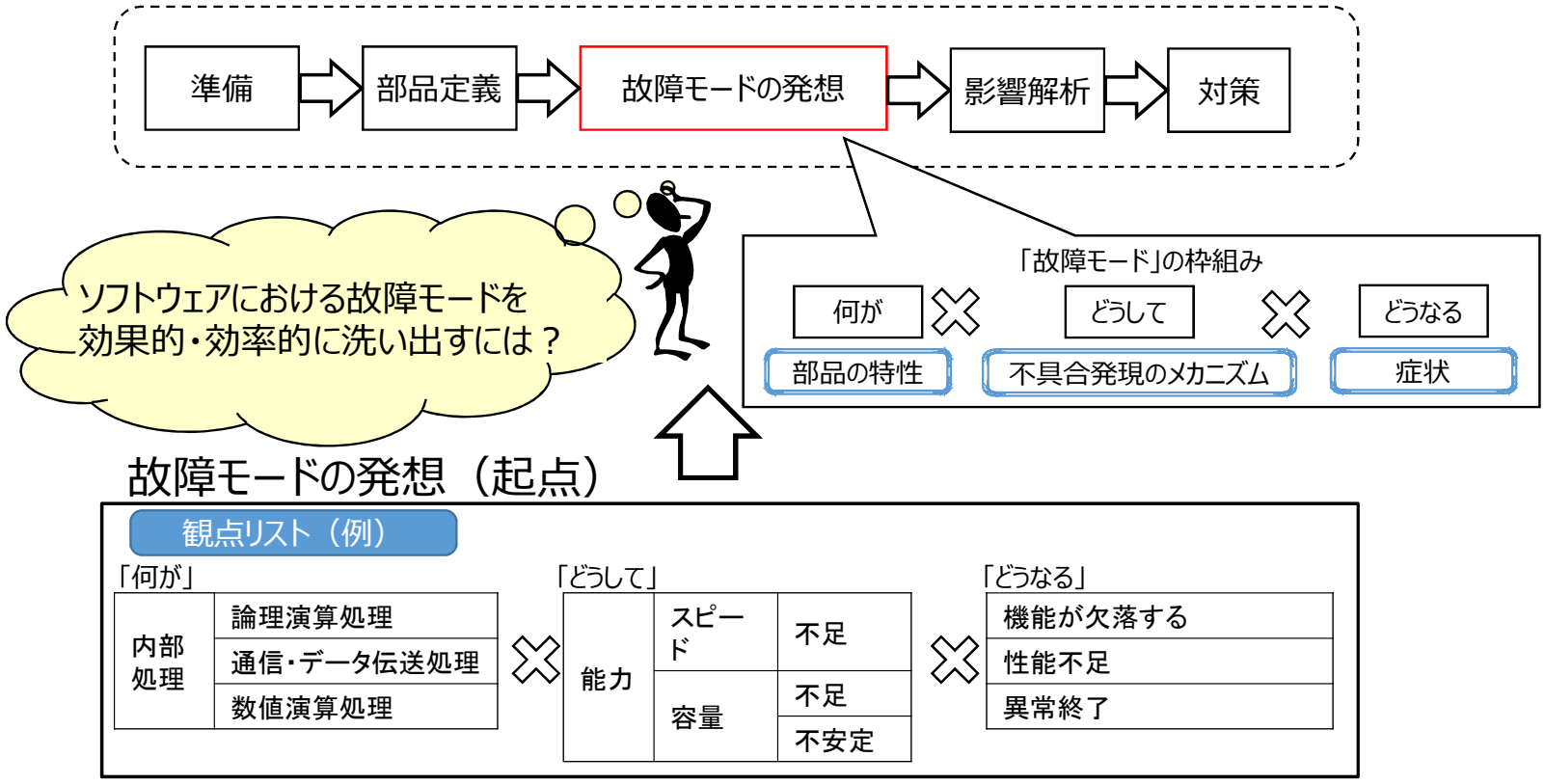
2. 限られた開発工数の中でやりきれない

ハードに比べて設計の自由度が高いため、考えるべき範囲や組み合わせが多い

⇨ 匠が最適な設計解を与えて解決してきた

SQiPシンポジウム2016発表資料を改変

ソフトウェアのFMEAと発想の観点(当社の事例)



過去トラをベースにしたシステマティック故障の洗い出しを実施

製品分野を考慮した観点リストの期待効果

- 製品分野を考慮した観点リストを使うことで、
  - 製品の特性によって起こりやすい不具合の再発・未然防止につながる
  - 開発組織やエンジニアの特性によって起こりやすい不具合の再発・未然防止につながる

観点リストとは、故障モードを発想しやすくするための、技術的な観点(起点)を記述した目録

(参考) 観点リストの開発手順(当社の事例)

（0）“組込みソフトウェア一般向け観点リスト”を用意する(全社でひとつ用意)

（1）**過去に発現した不具合や制限事項を真因解析**し、結果を故障モードにおける3つの属性の組（“なにが”、“どうして”、“どうなる”）で表現する

（2）**手順（1）に対する応用例・類推を考え、列挙する**

（3）手順（1）と手順（2）の結果を抽象化する

（4）“製品分野を考慮した観点リスト”に反映すべき結果を選択する
(抽象度の調整やグルーピングも行う)

（5）手順（0）における組込み一般向け観点リストを更新する

- 経験や実績等をより活かすために以下の内容も手順（1）に含める
  - **エキスパートの経験や知見**

(参考) 評価結果~エンジニアの予測に基づいた評価

項目		結果
故障モードの発現密度		13.0個/K step
新たに認識された故障モード		44%
新たに認識された故障モード(44%)の内訳	設計段階	5%
	実装段階	10%
	テスト段階	74%
	市場流出	11%
実施工数		68人時(7人)

- 新たに発見された不具合リスク： 0.45個／1人時
- 組込み一般向け観点リストによる場合の実績値（平均）
  ： 0.39個／1人時

15%高い

(参考) 評価結果~市場における評価

	ソフトウェアＦＭＥＡ実施部品(製品Ｐ)	不実施部品(製品P)	不実施部品(製品Q)
部品数	4	15	2
不具合の発現密度(PPM)	0	123	48

PPM: ソースコード100万行あたりの不具合個数を表す割合

- ソフトウェアFMEA実施部品　：　不具合無し
- ソフトウェアFMEA不実施部品　　：　48−123PPM

629PPM⇒0

再発防止及び
未然防止に効果

(参考) 2 つの観点リストの導出

（0）　“組込みソフトウェア一般向け観点リスト”を用意する

（1）　過去の不具合を真因解析し、その結果を3つの属性で表現する

（2）　手順（1）に対する応用例・類推を考える

抽象度が低い観点リスト

（3）　手順（1）と手順（2）の結果を抽象化する

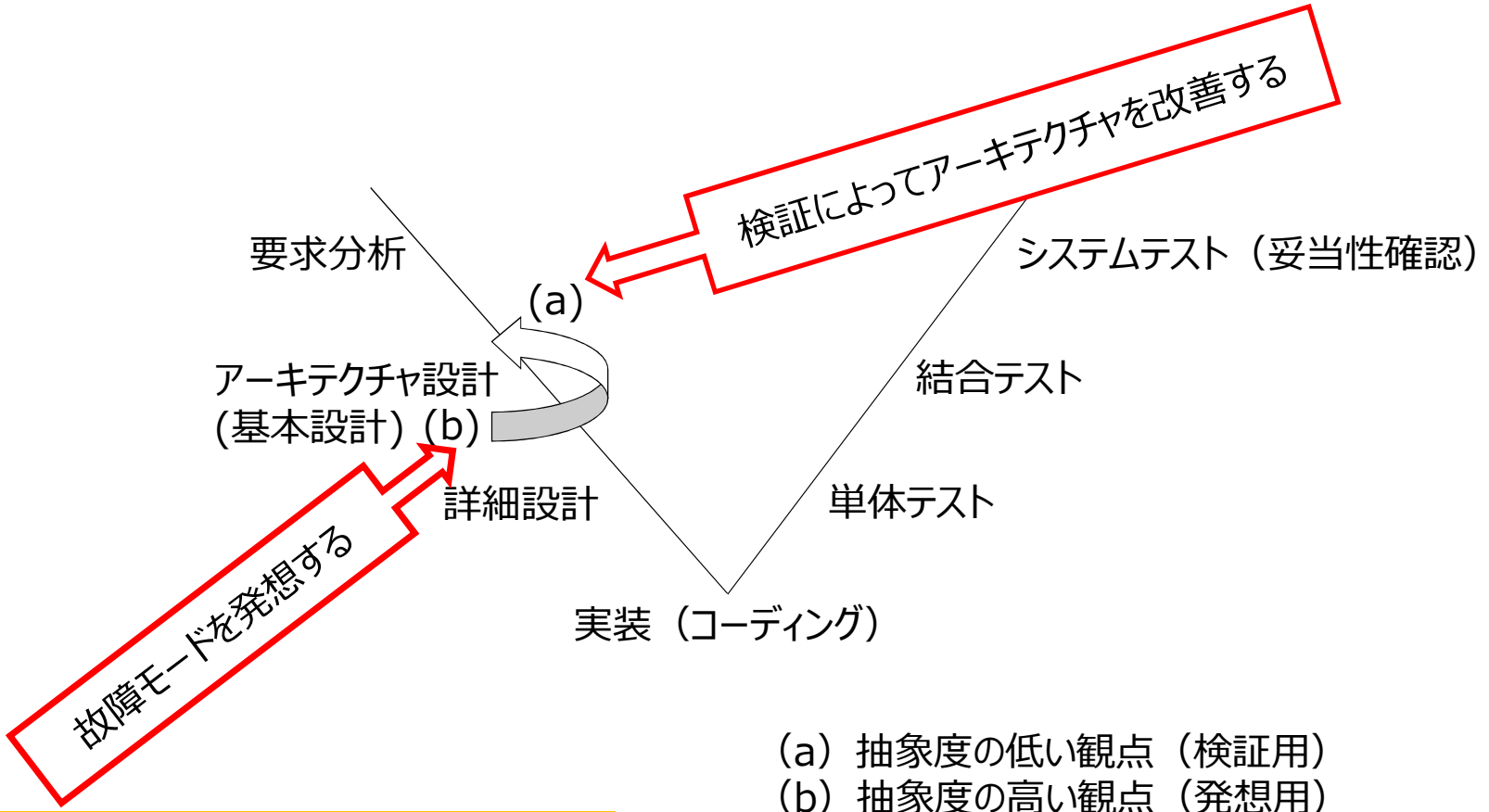
（4）　“製品分野を考慮した観点リスト”に反映すべき結果を選択する

抽象度が高い観点リスト

（5）　手順（0）における組込み一般向け観点リストを更新する

信頼性・保全性シンポジウム(第47回)発表資料を改変

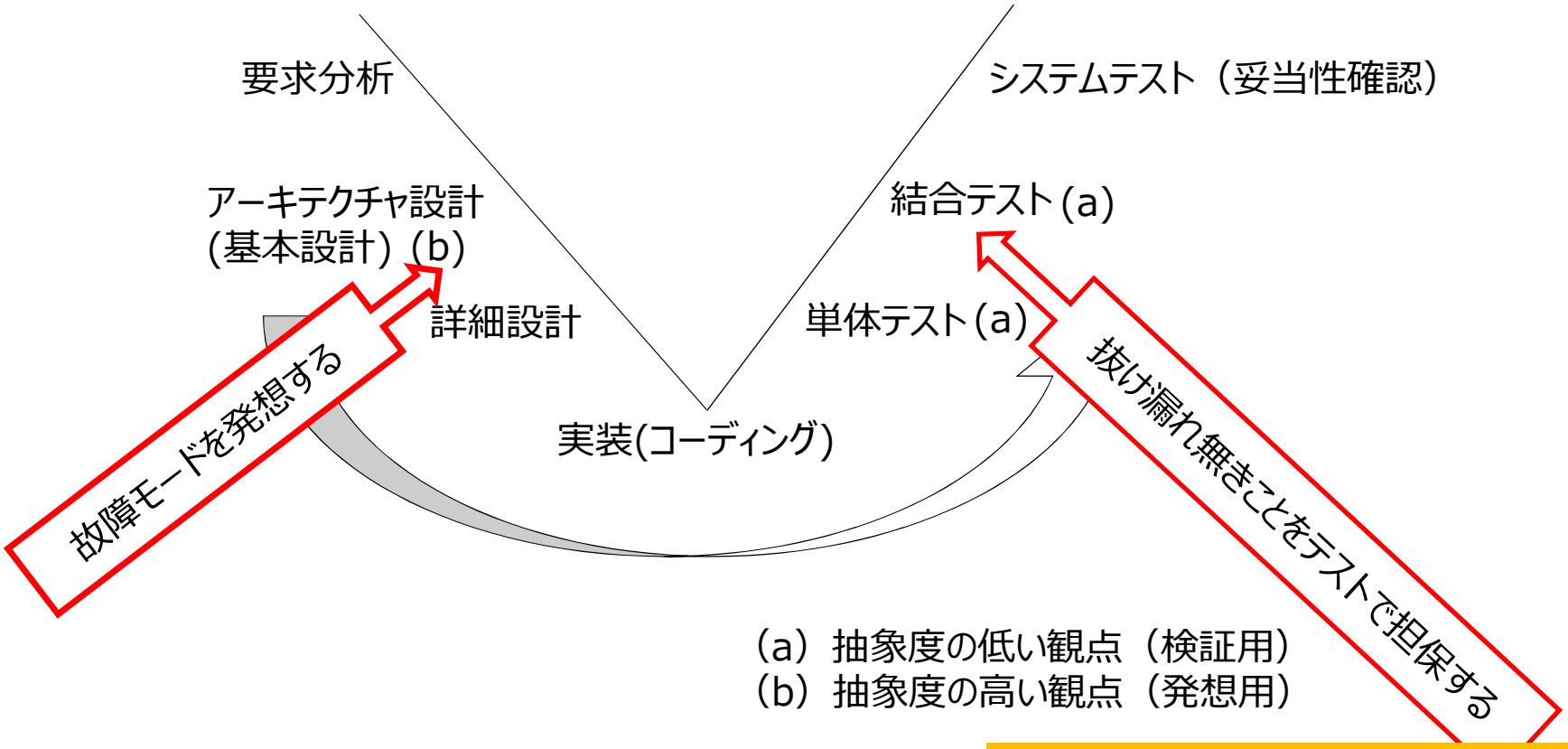
観点リストの適用ケース(1)



（a）抽象度の低い観点（検証用）
（b）抽象度の高い観点（発想用）

抽象度の高い観点リスト(発想用)で未然防止

観点リストの適用ケース(2)



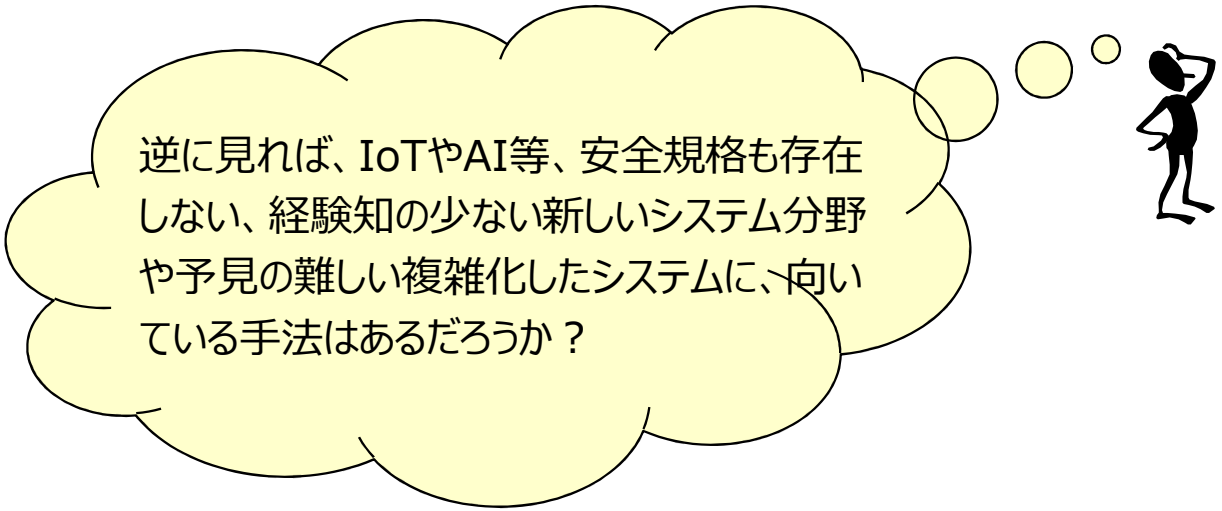
2つの観点リストまとめ

	抽象度が高い観点リスト	抽象度が低い観点リスト
メリット	未然防止につながる	発想が容易
デメリット	発想が難しい	未然防止につながりにくい
目的	発想用	検証用
適用フェーズ	V字モデルの左側上部	V字モデルの左側上部 またはテスト工程
抽象度の方針	十分に高める	具体的にする
観点の数	絞る（A4一枚程度）	制限無し（できるだけ多く）
注意事項	抽象度は徐々にあげる	観点は追加し続ける

観点リストによって、過去トラ、匠の知識、業界の慣例等を取り入れることが可能
アーキテクチャ設計では過去トラと類似トラブルも防ぎ、テストとしての十分性を補完できる

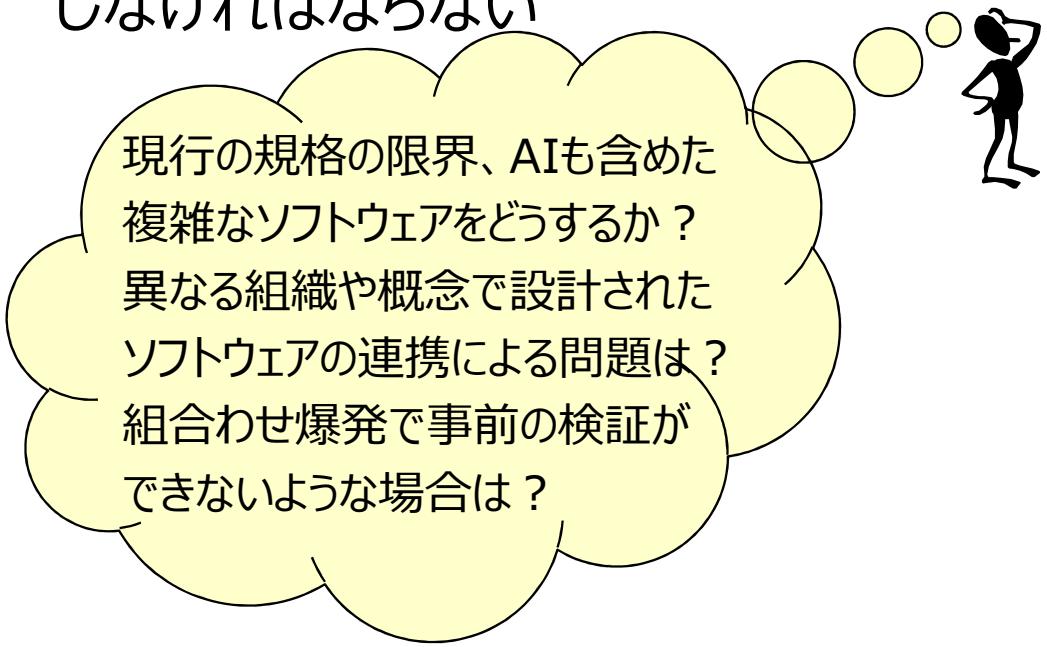
開発現場における従来の安全分析手法

- 事故やバグが起きたら対策を採るが、FMEAやFTAらの分析にはそれらの知見が反映されている　※ソフトウェアでも同じ
- この方法は、十分な過去トラがある等、経験知が高い製品では十分な安全確保のための手法と成り得る



これからの安全設計におけるソフトウェア

- IoTやAIの時代となって、大規模そして複雑化・多様化するソフトウェア
- 同時に、ソフトウェアは安全や信頼性だけでなく、サイバーセキュリティも考慮しなければならない



今すぐに答えを得るのは難しい

安全目標を網羅的に定義するには

- 他のシステムや、他の技術領域に対する理解も必要になるだろう
- 想定できる他のシステムや、他の技術領域を考慮した設計には、以下のような方法があるだろう
  - 各々を知っているエンジニアを育成する
  - 各々の技術内容を同時にレビューする
  - 各々がレビューしてイテレーションを繰り返す
- それ以外の、異なる組織や概念で作られたソフトウェアの連携によって、安全を脅かすようなことは？⇒　範囲を限定しない安全論証は、困難を極める

これらは今後の課題
予測できない組み合わせは、STAMPの活用も検討すべきだろう (後半に例としてご紹介)

安全目標を定義したあとでも…

- ソフトウェアの(主に連携による)不具合は起こる。要求仕様として、いかにインターフェースを定義できるかにかかるだろう
  - 複雑でも仕様が決めることができれば、形式手法などで対処できるかもしれない
  - 組み合わせが膨大な場合は、別途検討が必要

前半のまとめ

- 国際規格は製品の安全性を客観的に説明するのに有効である(安全論証の一部として活用できる)
- 現行の安全規格には限界があり、新しい特性を持つ製品への対応はできていない
- ソフトウェア固有の安全論証の考え方はなく、バグがないことを含めて〝最善を尽くしているか〟がひとつの視点である
- 現行の安全規格では、安全目標／安全要求を決定すれば、ソフトウェアはシステム設計、ハードウェア設計に沿ったバグのないオブジェクトとして期待されている
- ソフトウェアの安全分析方法として、故障モード発想のための観点リストに〝最善を尽くす〟ための考え方の一例を紹介
- 新しい特性を持つ製品、これからのソフトウェアに対する安全分析に、STAMPの考え方が役立つことの背景を説明

講 演 者 交 代

システム思考によるソフトウェアの安全設計
～現状と今後～

副題

新しい特性を持つ製品・ソフトウェアの安全分析としての、
STAMP・STPAへの期待

STAMP/STPAへの期待
～システムズ理論に基づく事故モデルとプロセス～

- 背景
  - ソフトウェアは複雑化する
  - システミックとシステマチックの違い
- STAMP/STPAのシステムズ理論とは？

現代の工学システム

自動運転車、生活支援ロボット、列車、航空機、船舶、プラント・・

- ほとんどの便利な機能はソフトウェアが担っている
- 付随する安全機能もソフトウェアが担っている

ナンシー・レバソンは、これを、
「ソフトウェア集約システム（Software-Intensive system）」
と呼んでいる。

より便利な機能を追い求めて、人の支援・代替機能や、
外部環境との協調・適応制御を十分な検証なしに実装
しがちになる（完全な検証ができない場合もある）。
→事故が起こった後、想定外との言い訳をしてしまう

ソフトウェア集約システム（Software-intensive system）の課題
～ソフトウェアは好むと好まざるとにかかわらず複雑化する～

- ソフトウェアコンポーネントとコンポーネント間通信は、市場競争の産業界では必然的に複雑化する
  - ソフトウェアは、人が行う機能を補助ないし代替しようとするが、エラーも含めて代替してしまう
  - ソフトウェアは、環境の状態を人間をまねて認識しようとするが、同等の認識はできない
  - ソフトウェアコンポーネントは、将来の拡張性やトラブル時の事後確認などを想定して、仕様にない機能も含めてしまう
  - ソフトウェア・コンポーネント間、人・環境とソフトウェアコンポーネント間の通信データは、送る側の意図と異なる意図で受け取り側が利用することがある

⇒ 複雑システムではコミュケーションエラーが事故を引き起こす

システムズ・アプローチ
～システミックとシステマチックの両面からの分析の大切さ～

「システミックxシステマチック」な問題解決を


SDM NEWS

2013年 1 月号

行事予定

2013年2月6日(水) 19:00～21:00
SDM研究所・GCOE共催公開講座
「ダイアログとデザインの未来
Vol.8 経済の未来」
@日吉キャンパス 協生館C3S10教室
http://www.sdm.keio.ac.jp/2013/02/06-132333.html
要事前登録 無料

2013年3月4日(月) 13:00～17:10
第5回環境共生・安全システムデザイン
シンポジウム「脳と心と幸福を考える」
@日吉キャンパス協生館 藤原洋記念ホール
http://www.sdm.keio.ac.jp/2013/03/04-164319.html
要事前登録 無料

研究科委員長兼研究所長からのメッセージ

「システミック×システマティック」な問題解決を

明けましておめでとうございます。今年のSDMのキャッチフレーズは"システム"。昨年は主にデザインという単語にフィーチャーした一年だったと言えるかもしれません。今年は初心に帰り、"システム"にフォーカスします。システムとは、要素間の[illegible]性によって創発する特徴を持つもの。技術システ[illegible]ステム、人間システムまで、インタラクシ[illegible]もシステムです。慶應SDMで重視する[illegible]、システムズエンジニアリングを基盤とするシステムズ・アプローチ。システムズ・アプローチには、ふたつの意味があります。システミックなアプローチと、システマティックなアプローチ。前者は、ものごとをシステムとして俯瞰的・全体的に見るということ。後者はロジカルに分解・統合するということ。慶應SDMでは両者を駆使してイノベーティブなデザインとサステナブルなマネジメントを実現します。システムズ・アプローチを身に着けた人材の育成、SDM学の基礎と応用に関する実践的な研究、そして企業や事業体との連携に基づく様々な階層での問題解決・社会変革。今年度もよりよい世界を構築するために邁進してゆく

慶応大学　SDMニュース
http://www.sdm.keio.ac.jp/pdf/sdmnews/SDM_News_201301.pdf

システミック・アプローチ
ものごとをシステムとして俯瞰的・全体的に見る
システマチック・アプローチ
ものごとをロジカルに分解・統合して見る

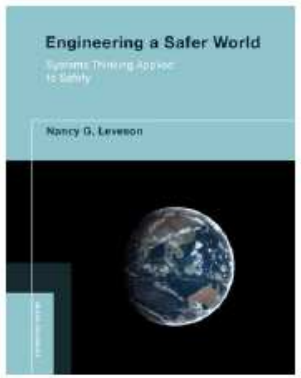
（参考）システムズエンジニアリング
「システムの実現を成功させることができる複数の専門分野にまたがるアプローチおよび手段 」（INCOSE SE Handbook,2000)

ソフトウェア集約システムの安全設計でのパラダイムシフト

- 従来の安全設計法（FTA、FMEA、HAZOPなど）は還元主義（Reductionism）である（N.G.Leveson)
  - 安全目標を要素に分解して設計する信頼性工学の手法に基づいて全ての故障をなくす
  - システマチックアプローチに対応するが、ソフトウェア集約システムでは、全ての欠陥を見つけて除去することは困難
- システム理論に基づく事故モデルと安全分析法STAMP/STPAは、トップダウンで安全制御構造を可視化して事故を防ぐ
  - システミックアプローチに対応し、事故シナリオを創発（Emergence)と捉え、安全制御工学の手法で事故を防ぐ
  - 仕様の欠陥や要素間の連係ミスにより起こされる事故シナリオを抽出することが主眼であり、還元主義的手法と異なるパラダイムシフトでもある

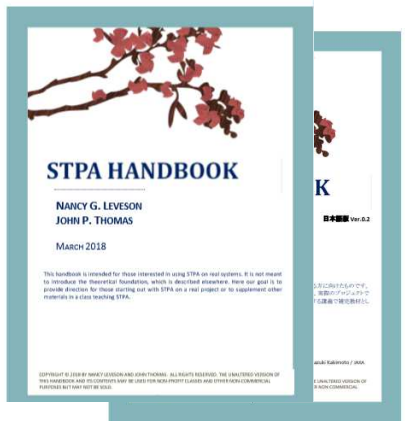
STAMP/STPAへの期待
～システムズ理論に基づく事故モデルとプロセス～

- 背景
  - ソフトウェアは複雑化する
  - システミックとシステマチックの違い
- STAMP/STPAのシステムズ理論とは？



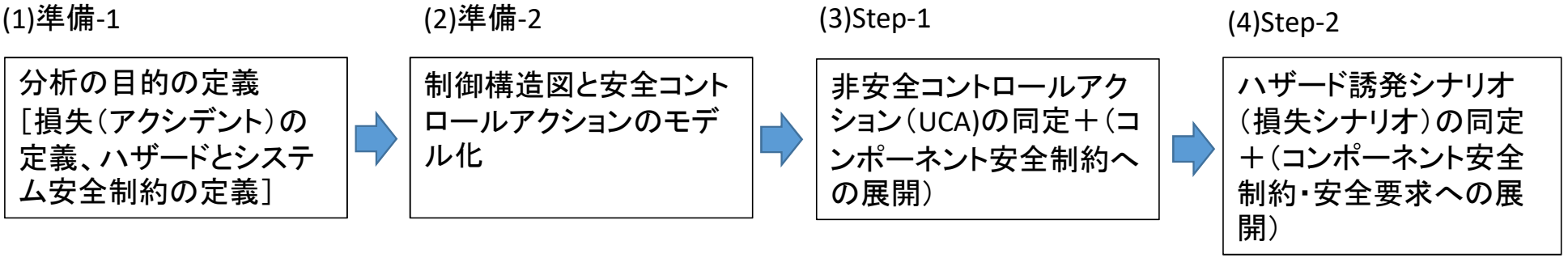
An STPA Primer
Version 1, August 2013
(updated June 2015)

http://psas.scripts.mit.edu/home/wp-content/uploads/2015/06/STPA-Primer-v1.pdf



http://psas.scripts.mit.edu/home/

STPA手順（Handbook）



(1)表面的には極めて簡単な手順で、4つの非安全コントロールアクション(UCA)を起点にしてハザード誘発シナリオを抽出し、事故を防ぐ方法を考える
(2)ハザード誘発シナリオは、従来のFTA,FMEA，HAZOPの思考法と大差ない。
しかし、
(3)アクシデント、ハザードの決め方、制御構造図の作り方などで、明示化されていないノウハウがある （これを理解するにはシステムズ理論の考え方の理解が重要である）

アクシデント、ハザード、安全制約

システム	損失（アクシデント）	ハザード	安全制約
ACC(自動追従運転)	L1:2台の車の衝突	H1:前方ないし後方の車との不適切な車間距離	SC1:二つの車は最小の車間距離を越えてはならない
化学プラント	L1:有害物質による人命の損失または危害	H1:プラントからの有害物質の気中や地中への放出	SC1:有害物質はプラントから過失によって放出されてはならない
自動車のエアバッグ	A1:運転者の死傷	H1:衝突したのにエアバッグが開かない H2:通常走行時にエアバッグが開いてしまう H3:エアバックの異常な爆発（部品飛散）	SC1:衝突時にはエアバックが開く SC2:通常走行時にエアバッグは開かない SC3:エアバック開の際に部品を飛散させない

アクシデント、ハザード、安全制約

システム	損失（アクシデント）	ハザード	安全制約
ACC(自動追従運転)	L1:2台の車の衝突	H1:前方ないし後方の車との不適切な車間距離	SC1:二つの車は最小の車間距離を越えてはならない
化学プラント	L1:有害物質による人命の損失または危害	H1 の	
自動車のエアバッグ	A1:運転者の死傷	H1 グ H2 グ H3 発	

損失（Loss)は、人命損失、環境汚染、経済的損失、ビジネスリスクなど何でもよい。

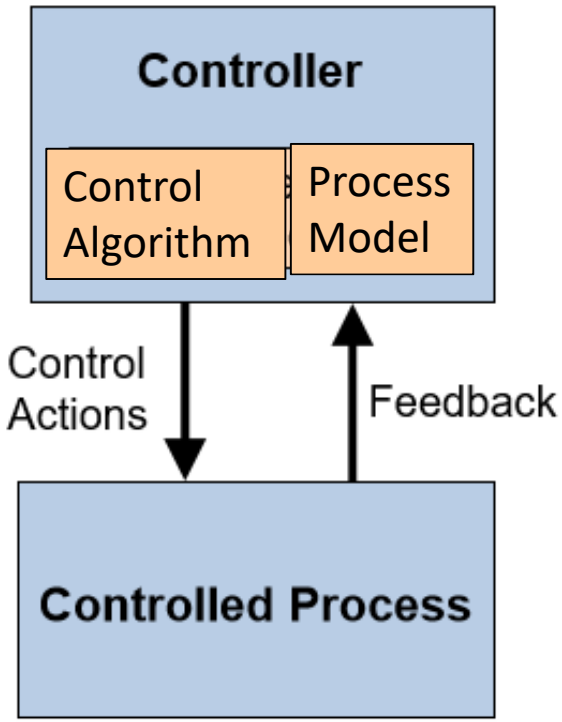
ハザードは、事象（イベント）や危険源ではなく、事故の一歩手前の放置してはいけない状態と考える。

◇システム外部の状態は制御できないので最悪状態を仮定する

◇ハザードを誘発する要因まで言及すると発想を狭める（例：ブレーキ故障ではなく、加速・減速という表現）

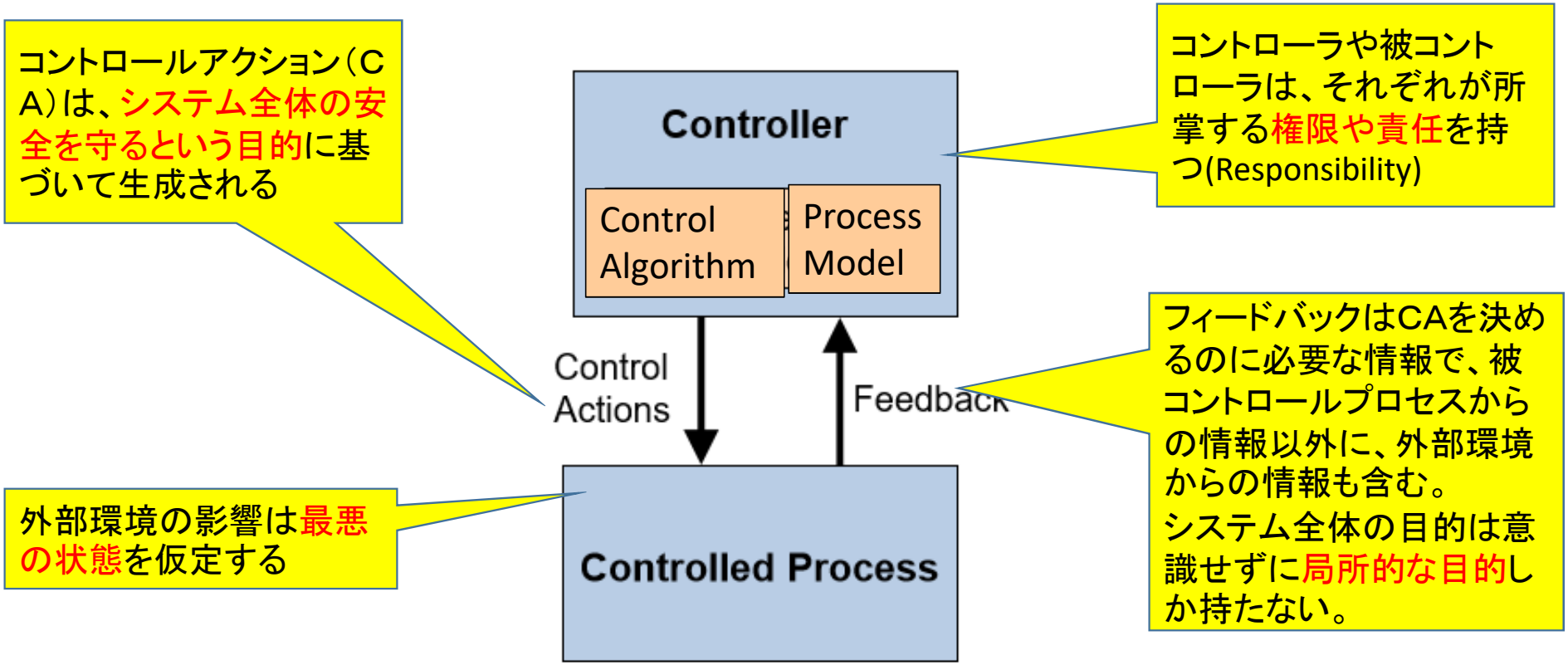
◇ハザードは、システム全体を見て10個程度以内に抑える。これが多すぎるのはシステムの理解の抽象度が足りないことを意味する

Basic STAMPの制御構造図



- コントローラは、その所掌責任・権限に基づいて、コントロールアクション(CA)によりシステムの安全を確保する。CAは、フィードバック情報とプロセスモデルに基づいて決められる。
- プロセスモデルが正しくないときに想定外の挙動が起こる(プロセスモデルは、被コントロールプロセスの挙動についての分析者のメンタルモデルでもある)
- 四つの不適切なコントロールアクション(UCA)
  - (N) Not providing
  - (P) Providing causes hazard
  - (T) Inadequate timing, too early or too late
  - (D) Inadequate duration, stop too soon or applying too long
- ソフトウェアと人間の挙動のモデルにより、ソフトウェアエラー、ヒューマンエラー、相互作用による事故などを説明する

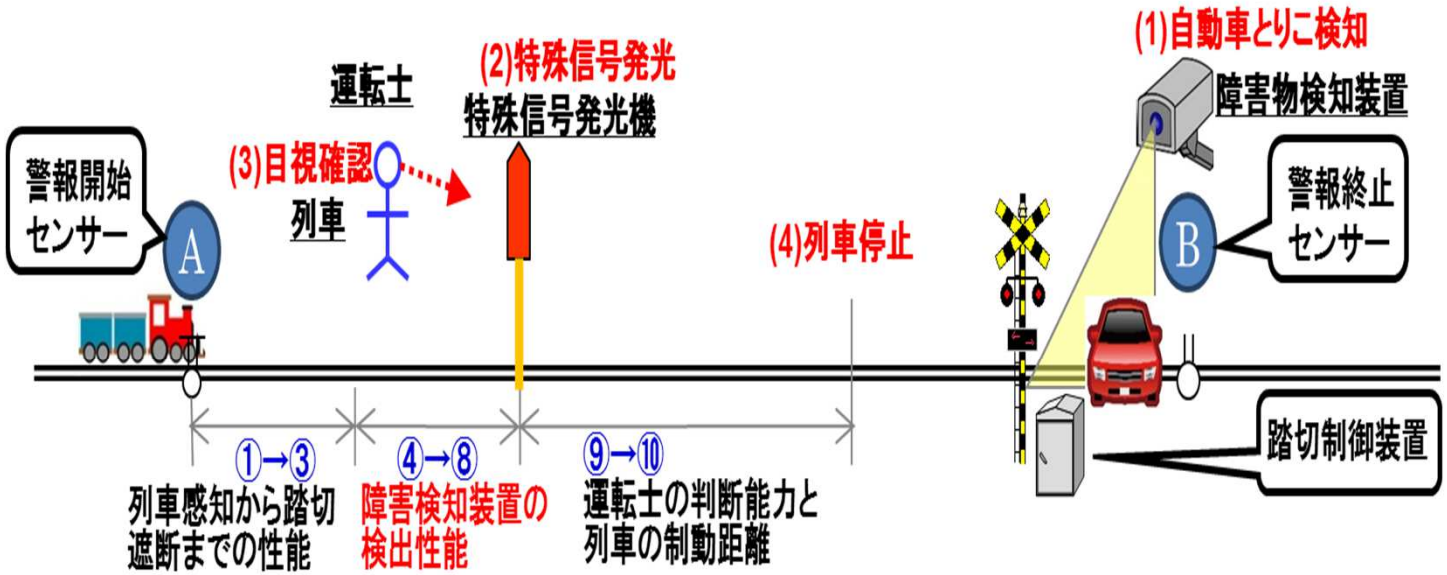
Basic STAMPの制御構造図（四つの大事な考え方）



一枚の制御構造図で、システム全体の安全制御メカニズムが見通せないといけない
→　立場の異なる人の間で共通の理解に基づいて相互レビューができる

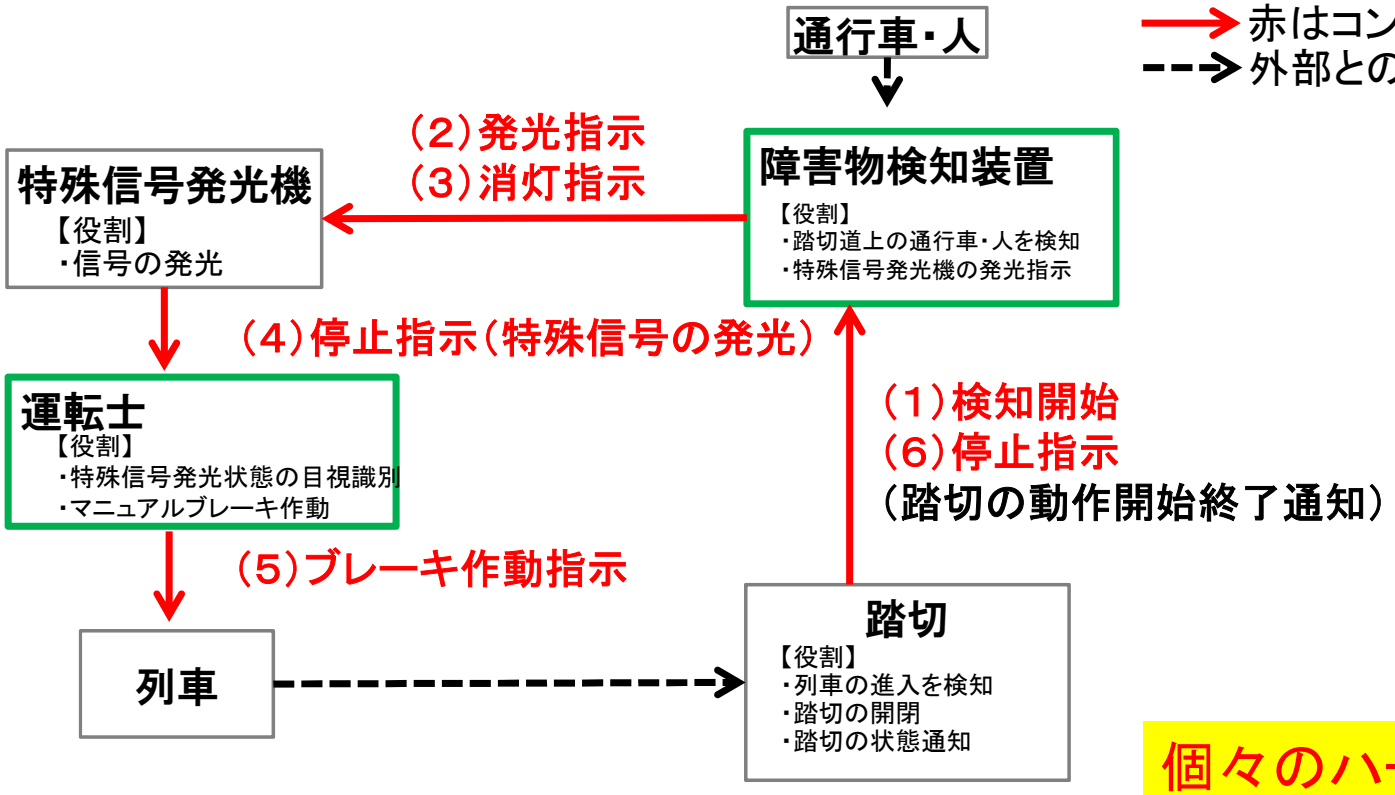
システムズ思考の事例：踏切の「とりこ」検知装置

踏切が閉まった時、踏切内に閉じ込められた人や車を検知し、特殊信号発光機で列車運転士に知らせるシステム



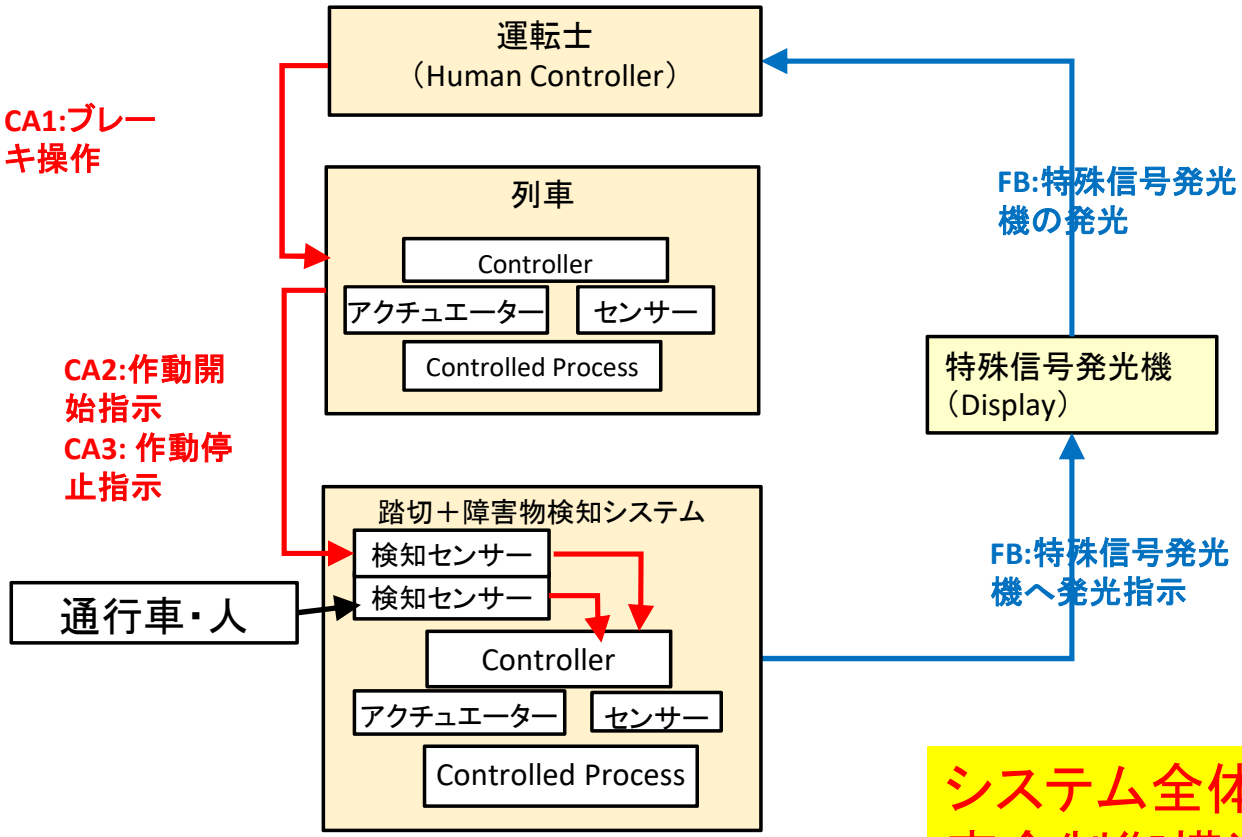
今回の分析範囲は④から⑨

検知の流れに沿った制御構造図（ハードウェア視点）



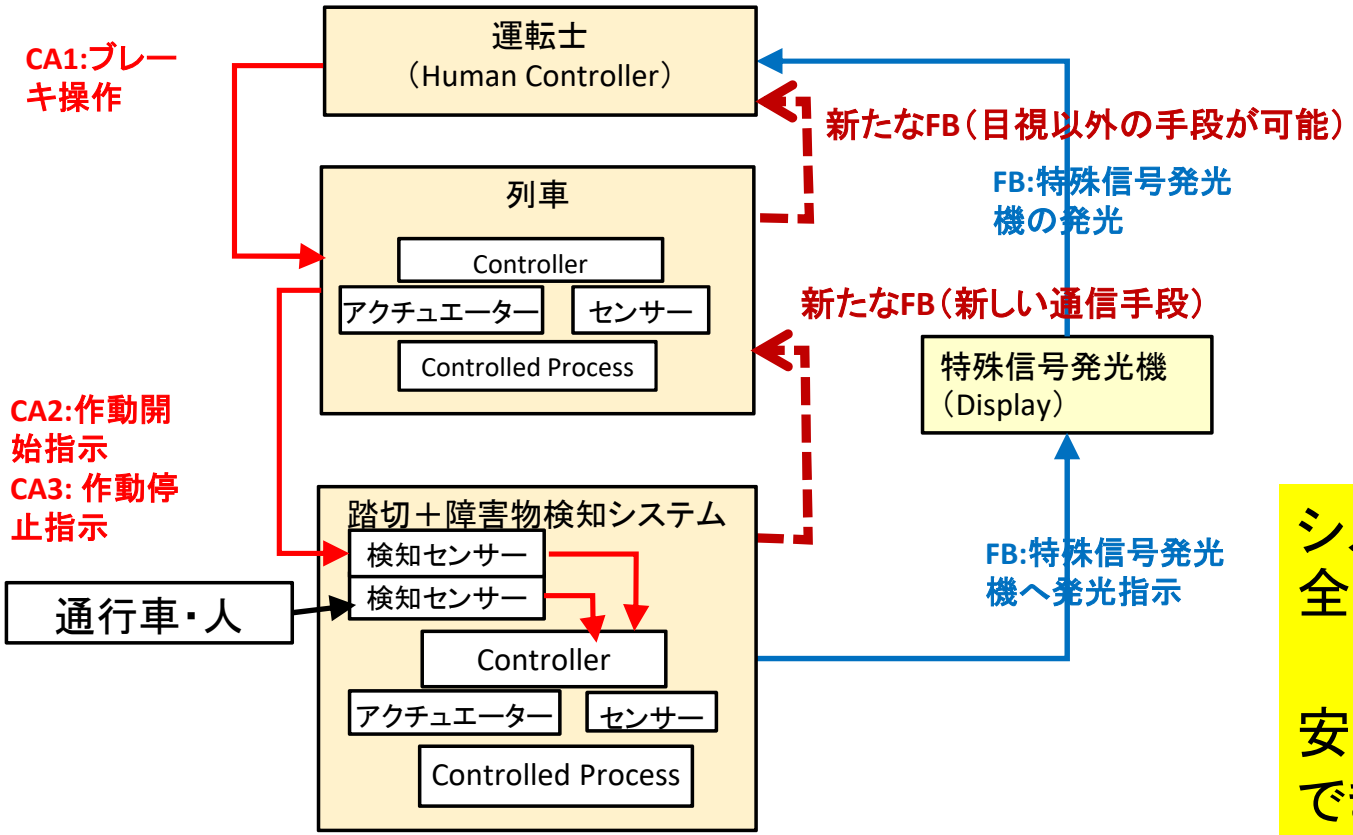
個々のハードの故障に
焦点が当たってしまう
（従来の分析法に近い）

運転士を中心にした制御構造図（運転士視点）



システム全体をみた
安全制御構造が見える

運転士を中心にした制御構造図（運転士視点）



システム全体をみた安全制御構造が見える

↓

安全要求の改善が発想できる（想定外の環境変化への対応）

STPA手順（Handbook）

(1)準備-1

分析の目的の定義
[損失（アクシデント）の定義、ハザードとシステム安全制約の定義]

(2)準備-2

制御構造図と安全コントロールアクションのモデル化

(3)Step-1

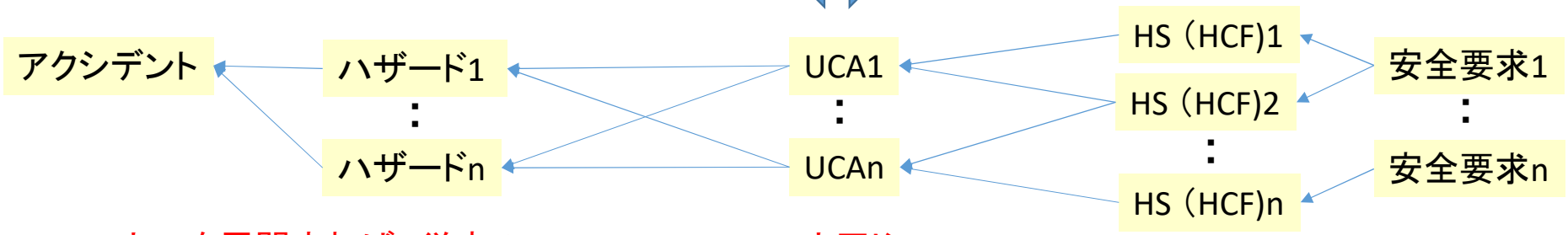
非安全コントロールアクション（UCA)の同定＋（コンポーネント安全制約への展開）

(4)Step-2

ハザード誘発シナリオ（損失シナリオ）の同定＋（コンポーネント安全制約・安全要求への展開）

HSに関わらずUCAを回避する（安全制御工学、レジリエンス工学）

HS(HCF)を全て評価し回避するのが信頼性工学



UCAとHSを展開すれば、従来のFTA、FMEA、HAZOPと同じ

後知恵で考えると、どんな手法で考えても同じ、だが、
事故前に発想できるかどうかの違いがある（プロアクティブな安全設計）

STPA手順（Handbook）

制御構造図による安全メカニズムの可視化と共有
- 目的（どんな事故を防ぐか）
- 各要素の安全責任と権限を定義
- 安全責任を達成するためのCAと、CAを作るためのFBを明示化
- 外部環境は最悪の状態を仮定

(1)準備-1
分析の目的の定義［損失（アクシデント）の定義、ハザードとシステム安全制約の定義］

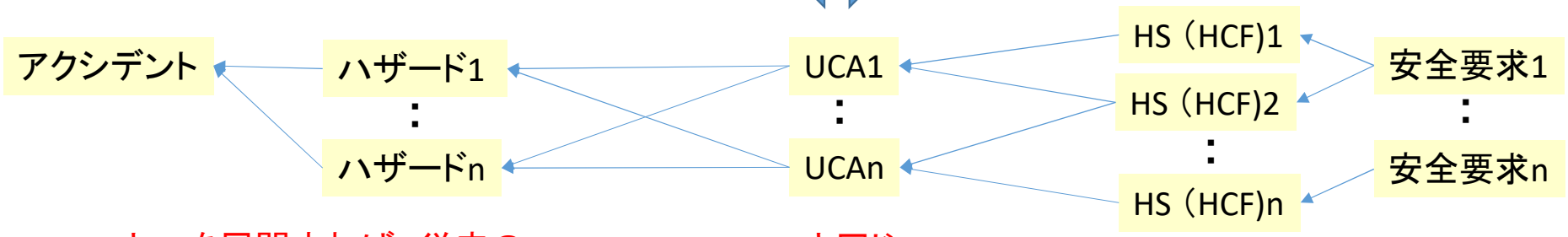
(2)準備-2
制御構造図と安全コントロールアクションのモデル化

(3)Step-1
非安全コントロールアクション（UCA)の同定＋（コンポーネント安全制約への展開）

(4)Step-2
ハザード誘発シナリオ（損失シナリオ）の同定＋（コンポーネント安全制約・安全要求への展開）

HSに関わらずUCAを回避する（安全制御工学、レジリエンス工学）

HS(HCF)を全て評価し回避するのが信頼性工学



UCAとHSを展開すれば、従来のFTA、FMEA、HAZOPと同じ

後知恵で考えると、どんな手法で考えても同じ、だが、
事故前に発想できるかどうかの違いがある（プロアクティブな安全設計）

今後の複雑システムの安全設計のまとめ

- 安全制御メカニズムの可視化（抽象化・階層化した制御構造図）
  - 一枚の絵で安全制御メカニズムを理解し、共有する
  - 異なる立場のレビューアでも、相互の誤解なく理解できる
  - 誰が誰を制御するかという責任と権限の所在が明確
  - 想定外の事故を想定する自由な議論ができる
- 安全論証に役立つ
  - どんな事故を防ぐか、その前段階のハザードをどうやって防ぐかがわかる（UCA、ハザードシナリオ（HS)、コンポーネント安全制約/要求など）
  - 前提とした外部環境の状態の明示化（最悪の状態、複数の条件の重複などの仮定）
  - 論理的な整合性（トレーサビリティ、網羅性、排他性の明示化）

◆安全設計入門改訂版

電波新聞社 2010年7月発行

改訂版の主旨

企業でのソフトウェア技術者向けの安全設計の入門書として,
安全分析の基本手法(FTA,FMEA,HAZOP),
各種機能安全規格、
STAMP/STPA による新しい安全分析法などを広く学べる。

目次

1.安全の基本
2.安全規格体系と概要
6.自動車の機能安全(ISO 26262)
7.サービスロボットの機能安全(ISO 13482)
8.システム思考で考えるこれからの安全

2019 年 3 月発刊予定（電波新聞社）

ご清聴ありがとうございました

自由なQ&Aとコメントをお願いします