事例報告：
AADLを用いたSTAMP/STPA支援
～ハザード誘発要因識別の支援～

岡本圭史，力武克彰　仙台高等専門学校
大友楓雅　仙台高等専門学校情報電子システム工学専攻

要旨

- 目標：人と機械による相補的STAMP/STPAの提案
- 今回の内容：AADLを用いたSTAMP/STPA支援事例の報告
  - STAMP/STPA：システム理論に基づく安全分析手法
  - AADL：アーキテクチャ分析設計言語
  - Error Model Annex ver.2(EMV2)：安全分析用AADL拡張
- 事例で明らかにしたい事柄
  - STAMP/STPAに必要なAADL+EMV2のモデル要素は？(下表参照)
  - AADL+EMV2で支援可能・困難なSTAMP/STPAのStepは？

Table1: ARP4761 Process Elements and Supporting AADL Error Model Constructs

		ARP4761 Process Elements				
AADL Error Model Construct		FHA	FTA	FMEA	MA	DD
Error flows	Error propagation	×	×	×	×	×
	Error source	×	×	×		
	Error path			×		
	Error sink		×	×		
Error bahavior	Error states		×		×	×
	Error transitions		×		×	×
	Error events	×	×	×		×
	Composite error model		×			×
Properties	Hzards property	×				
	OccurrenceDistribution property				×	×

HA:Functional Harzard Assessment
MA:Markov Analysis
DD:Dependence Diagram
(Reliability Block Diagram)

AADL Fault Modeling and Analysis Within an ARP4761 Safety Assessment, Julien Delange et.al, 2014

STAMP

- System-Theoretic Accident Model and Processes
- システムを構成するコンポーネント間の相互作用に着目
- STAMPの構成要素
  - 安全制約：安全が守られるために必要なルール
  - プロセスモデル：コントローラーが持つアルゴリズム
  - コントロールストラクチャー(CS)：
    コンポーネント間の機能動作を示したシステムの設計図
- STPA (System-Theoretic Process Analysis)
  - STAMPに基づくハザード分析手法
  - 複合要因や故障無しで引き起こされる要因を識別可能
    - 従来：単一・複数の故障が引き起こす要因を識別

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

STPA

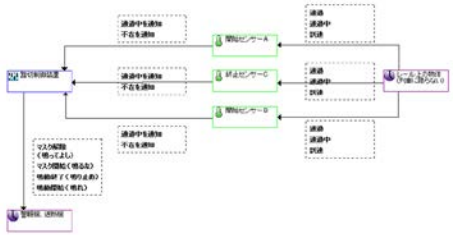
- Systems-Theoretic Process Analysis
- ハザード分析手法
  - 識別済のハザードへ至るシナリオを識別する
  - システム理論に基づくモデルを使用
- 識別するアクシデント・ハザード
  - 従来のハザード分析法：(単一・複数の)コンポーネント故障が引き起こすアクシデント(component failure accidents)を識別
  - STPA：(上記のアクシデントに加え)欠陥のあるデザインや故障していないコンポーネント間の非安全相互作用が引き起こすアクシデント(component interaction accidents)も識別
  - STPAで識別できるハザード要因は従来法で識別できるハザード要因を包含

STPAのプロセスとその課題

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

システム2	アクシデント	ハザード	安全制約
クルーズコントロール	2台の車が衝突する	自動車の前後に適切な車間距離をとっていない	自動車は定められた車間距離を破ってはならない
化学プラント	化学物質の流出によって人的被害が出る	化学物質が空気中や土壌へ流出する	化学物質が意図せずに放出されてはならない

Step1-1: アクシデント, ハザード, 安全制約の定義



Step1-2: コントロールストラクチャーの構築

コントロールアクション	与えられないとハザード	与えるとハザード	早すぎ，遅すぎ，誤順序でハザード	早すぎる停止，長すぎる適用でハザード
コマンド1	XX条件下で，コマンド1が提供されない場合，ハザードに至る(UCA1)	コマンド1の内容が誤っていた場合，処理は停止するが，ハザードには至らない	コマンド1の提供が，コマンド2よりも遅れた場合，指示が上書きされ，ハザードに至る(UCA2)	コマンド1が途中で停止した場合，ハザードには至らない
コマンド2	…	…	…	…

Step2-1:
Unsafe Control Action
(ハザードへ至るCA＋α)の識別

	1.上位からの指示や外部情報の誤り・欠落	2.CAが不適切・無効・欠落	3.動作の遅れ	4.プロセスへの入力の誤り・欠落	5.意図しない，または範囲外の外乱	6.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(踏切が閉まらず)		・踏切通過後に引き返す列車向け制御が不適切 …		センサー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)			・警報機の動作遅れ			・踏切制御装置の動作遅れ

Step2-2: Causal Factor
(UCAの原因)の識別

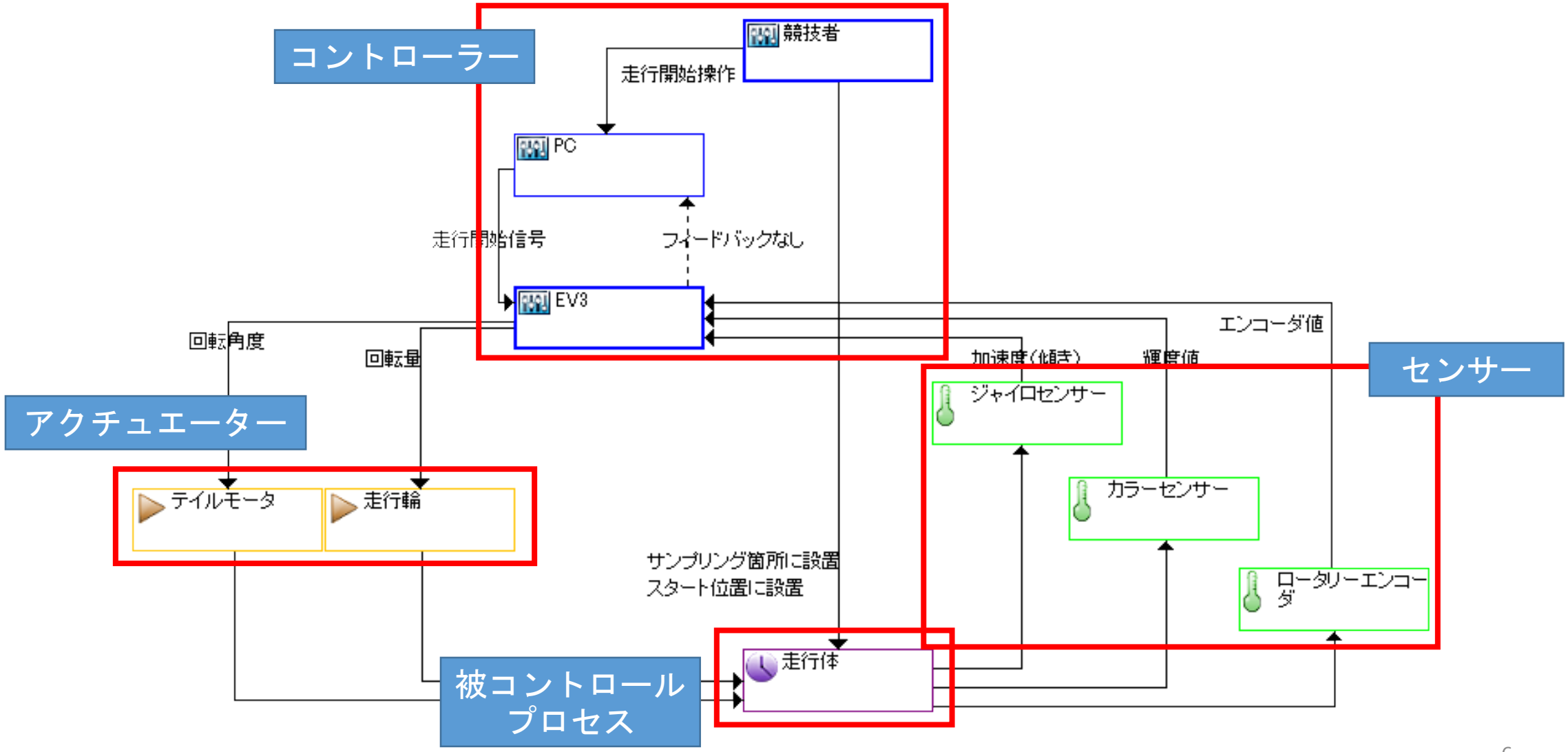
Step3:
識別された
非安全制御動作からの
安全要件と制約の作成

Step4:
潜在的にハザードを引き起こす
制御動作が
どのように起こるかの決定

素朴なSTPA試行により感じた課題
Step1における課題：
- 分析資料と開発資料の統合

Step2における課題：
- 人間が発想するまでもないことは自動化したい
- 人間の発想を支援してほしい

Step1-2:コントールストラクチャーの構築



ETロボコン走行体システムへのSTAMP/STPA適用事例の紹介, 大友楓雅他, STAMP Workshop Japan 2016

Step2-1: UCAの識別

UCAの4タイプ

制御動作	与えられないとハザード	与えられるとハザード	早すぎ、遅すぎ、誤順序でハザード	早すぎる停止、長すぎる適用でハザード
鳴動開始指示	(UCA1)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)	列車が来ないのに警報が鳴る	(UCA2)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.
鳴動停止指示	(UCAX)列車が通過しても鳴動停止指示が与えられないと，警報が鳴りっぱなしになる	(UCA3)列車が通過中に鳴動停止する	(UCA3)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)	(UCA1)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)

コントロールストラクチャーから抽出

Step2-1：第一列の制御動作がハザードへ至る「条件」(と「その制御動作が引き起こすハザード」)を記載.

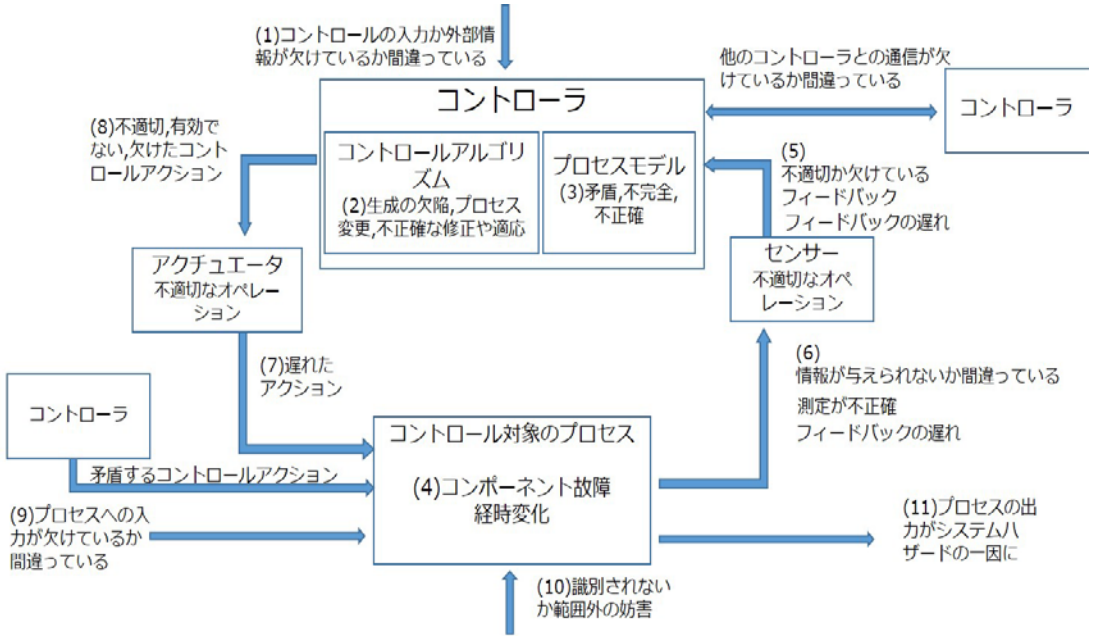
表：Step2-1の分析結果の例

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

Step2-2 ： CFの識別

- UCAへ至るシナリオの潜在的原因(CF)の識別
  - コンポーネント故障(単一ならFIAで支援可能)
  - コンポーネント間の非安全相互作用(STPA独自)
  - 複合的な(complex)人間，ソフトウェアの振る舞い
  - 設計間違い
  - 欠陥のある要件(特にソフトウェア関連)

STAMP/STPA Beginner Introduction, John Thomas, STAMP Workshop Japan (2016)



図：制御ループで間違っていることがあるもの

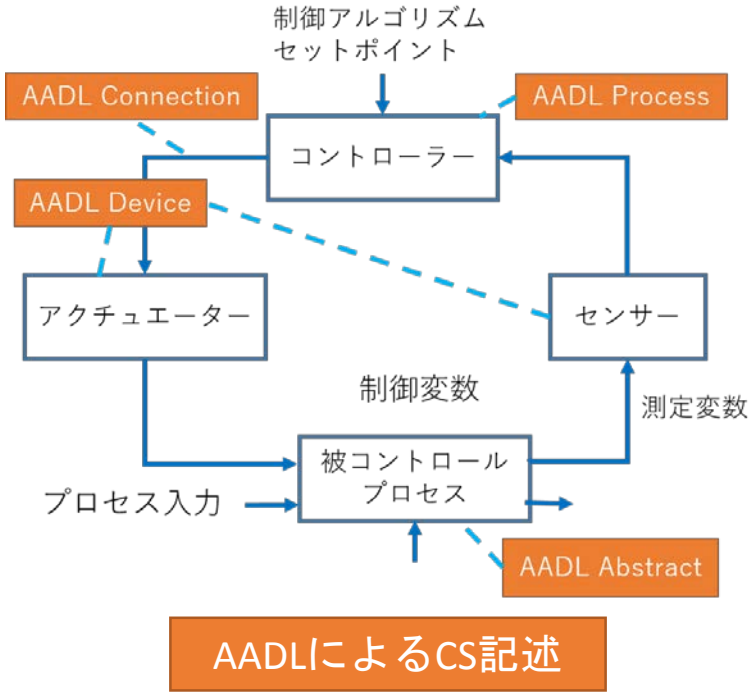
表：Step2-2の分析結果の例

	上位からの支持や外部情報の誤り・欠陥	コントロールアクションが不適切・無効・欠落	動作の遅れ	プロセスへの入力の誤り・欠落	意図しない，または範囲外の外乱	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する(踏切が閉まらず)		鳴動停止継続により次の鳴動指示と競合		落ち葉等によりセンサーAが列車到達を検知できず，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)				落ち葉等によりセンサーCが列車通過を検知できず，踏切制御装置への通知が欠落	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	

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

AADLとは？

- Architecture Analysis and Design Language
  - アーキテクチャ分析設計用言語(テキスト形式，図式)
  - コンポーネントベース
  - 主たる記述対象：組込みシステム(HW/SW)
    - SW: パッケージ，データ，サブプログラム，抽象構成要素等
    - ランタイム・アーキテクチャ：プロセス，スレッド等
    - HW: プロセッサ，メモリ，バス等
  - 厳密な意味論
- 分析用拡張アノテーションを追記して分析
  - 設計モデルと分析情報の統合
  - The Error Model Annex standard for AADL(EMV2)：安全分析用
  - OSATE2ツール：安全分析(Fault Impact Analysis, FTA等)を実施可能



AADLによるCS記述

[2] Procter, S. and Hatcliff, J. An Architecturally-Integrated, Systems-Based Hazard Analysis for Medical Applications, Formal Methods and Models for Codesign (MEMOCODE), 2014

Error Model Annex

- AADLモデルにエラーやエラー伝搬の情報を付加
  - Error type：類型化されたエラーの階層構造 ⇒STPA Step2-1の４つのタイプ
  - Error flows：コンポーネント間やコンポーネント内のエラー伝搬
  - Error behavior：エラー状態の遷移
  - Properties：ハザードの情報等
- 様々な安全分析手法を実施できる
  - FMEA, FTA等
  - AADLモデルで保守性, 可用性等も分析可能

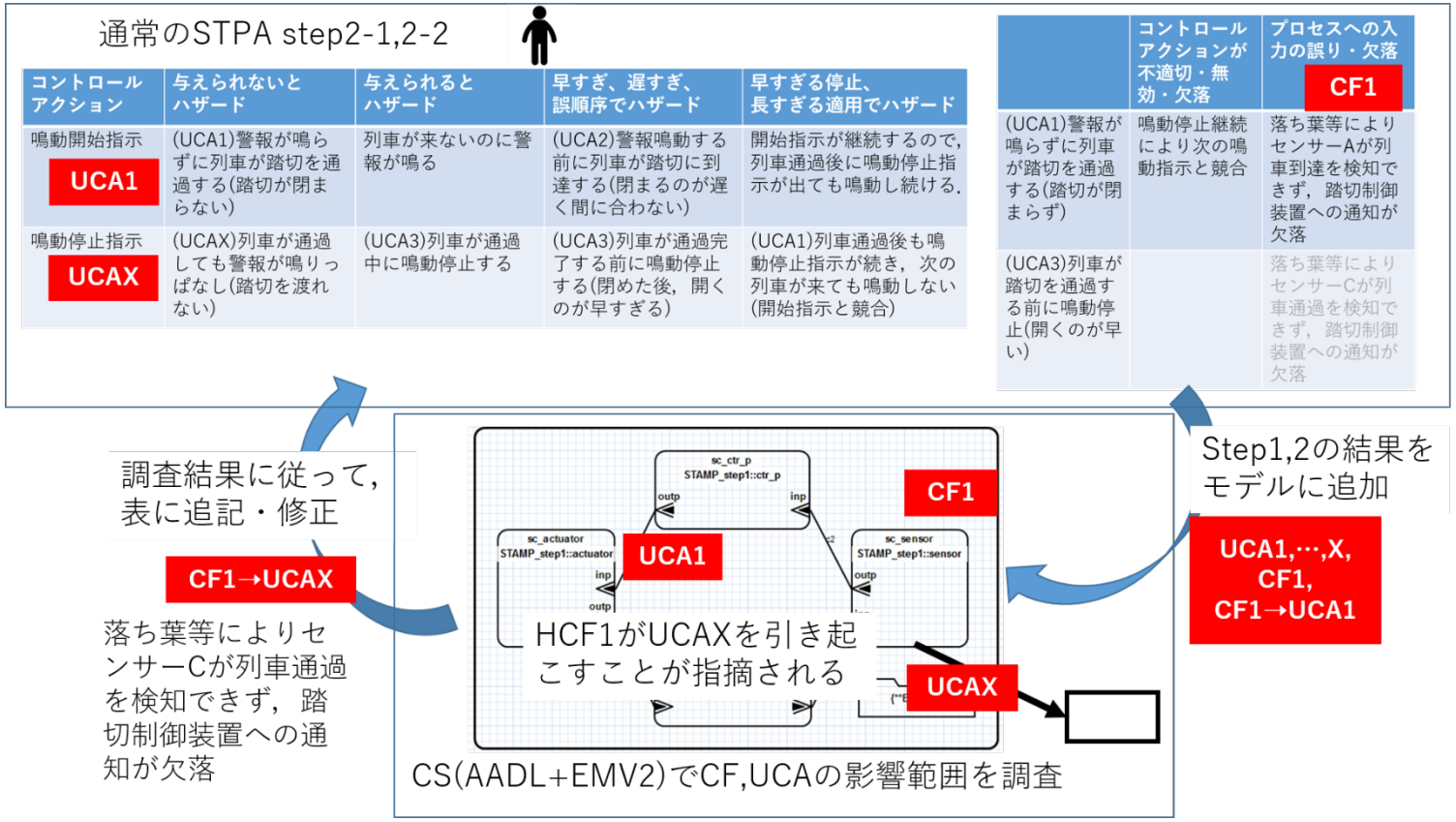
STAMP/STPA (Step2-1タイプ)	Error Model Annex (エラータイプ)	分析対象固有 エラー・タイプ (拡張として定義)
与えられないとハザード	ServiceOmission, …	ServiceOmission等の拡張ラー・タイプとして定義
与えられるとハザード	ServiceComission, …	ServiceComission等の拡張エラー・タイプとして定義
早すぎ、遅すぎ、誤順序でハザード	EarlyDelivery, …	EarlyDelivery等の拡張エラー・タイプとして定義
早すぎる停止、長すぎる適用でハザード	EarlyServiceTerminati n, …	EarlyServiceTermination等拡張エラー・タイプとして定義

エラータイプによるSTPA タイプの記述

[Feiler2011] SAE AADL Error Model Annex: An Overview, Feiler, P. (2011)
[Delange2014] AADL Fault Modeling and Analysis Within an ARP4761 Safety Assessment, Delange, J. et al. (2014)
[Feiler2016] Architecture Fault Modeling and Analysis with the Error Model Annex, Version 2, Feiler, P. et al. (2016)

STAMP/STPAとAADL [2]

- STPAの結果付きAADLモデルから分析報告書を自動作成
- 「手動分析＋自動分析⇒分析結果充実」という枠組みを提案

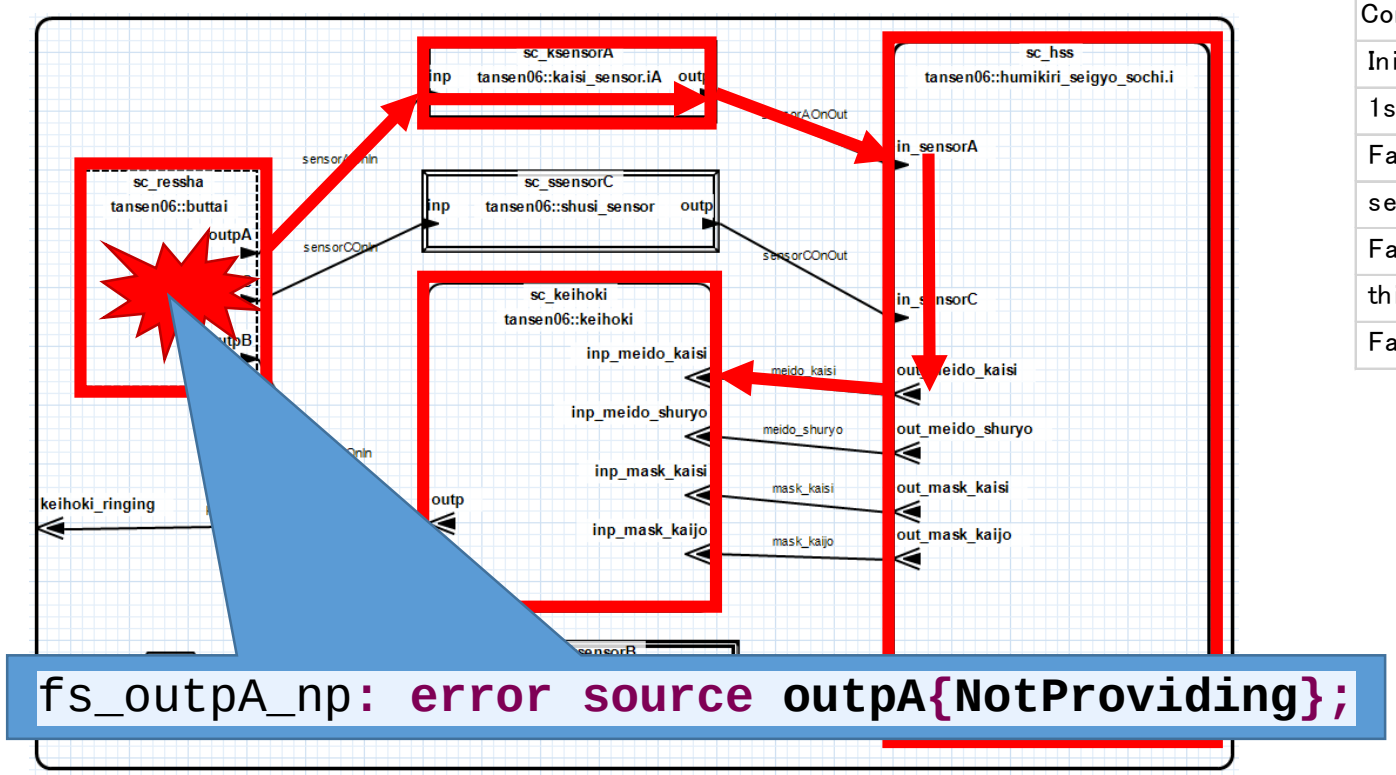


[2] Procter, S. and Hatcliff, J. An Architecturally-Integrated, Systems-Based Hazard Analysis for Medical Applications, Formal Methods and Models for Codesign (MEMOCODE), 2014

課題：[2]では自動分析法は提示されていないので，自動分析の実現法が必要である．

STAMP/STPAとAADL [3]

- エラー伝搬情報付きAADLモデルを用いて，CF影響範囲を指摘
  - 想定済CFの影響範囲指定にFault Impact Analysisを使用



Component	sc_ressha
Initial Failure Mode	{NotProviding}
1st Level Effect	{NotProviding} outpA -> sc_ksensorA:inp
Failure Mode	sc_ksensorA {NotProviding}
second Level Effect	{NotProviding} outp -> sc_hss:in_sensorA
Failure Mode	sc_hss {NotProviding}
third Level Effect	{Providing} out_mask_kaijo -> sc_keihoki:inp_mask_kaijo
Failure Mode	sc_keihoki {Providing} [Failure Effect]

Fault Impact Analysisの結果

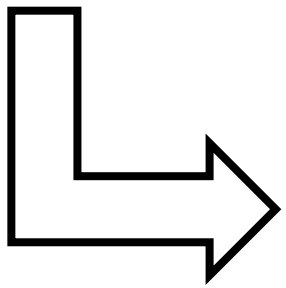
課題：Step2-2の結果はFTA, HAZOP, FMECA等による分析結果を含むが，Step2-2＝FMEAになってはいけない．

[3] Feiler, P. H. et. al., Improving Quality Using Architecture Fault Analysis with Confidence Arguments2015

今回の方針

先行研究[2]：人と機械による相補的STPAの提案
- CS(+人手で識別された原因)をAADLで記述
- 課題：実現方法は提示されていない

先行研究[3]：逆向きFIA as STPA Step2
- CS(+エラー伝搬,想定済原因候補)をAADLで記述
- FIAで原因となる単一コンポーネント故障を識別
- 課題：STPAに完全に準拠していない



- 今回の方針：[3]を用いて[2]を実現
  - STPAは人手で実施，FIAは機械支援
- CS図 := [3] + Step1の4タイプの原因候補
  - Step1 ⇒ Step2 の流れを尊重
  - 原因候補の詳細は後で考察
- 支援：[2] + [3]
  - 効果１：面倒な判定の自動化[2]
  - 効果２：STPA時の気づき漏れを指摘[3]

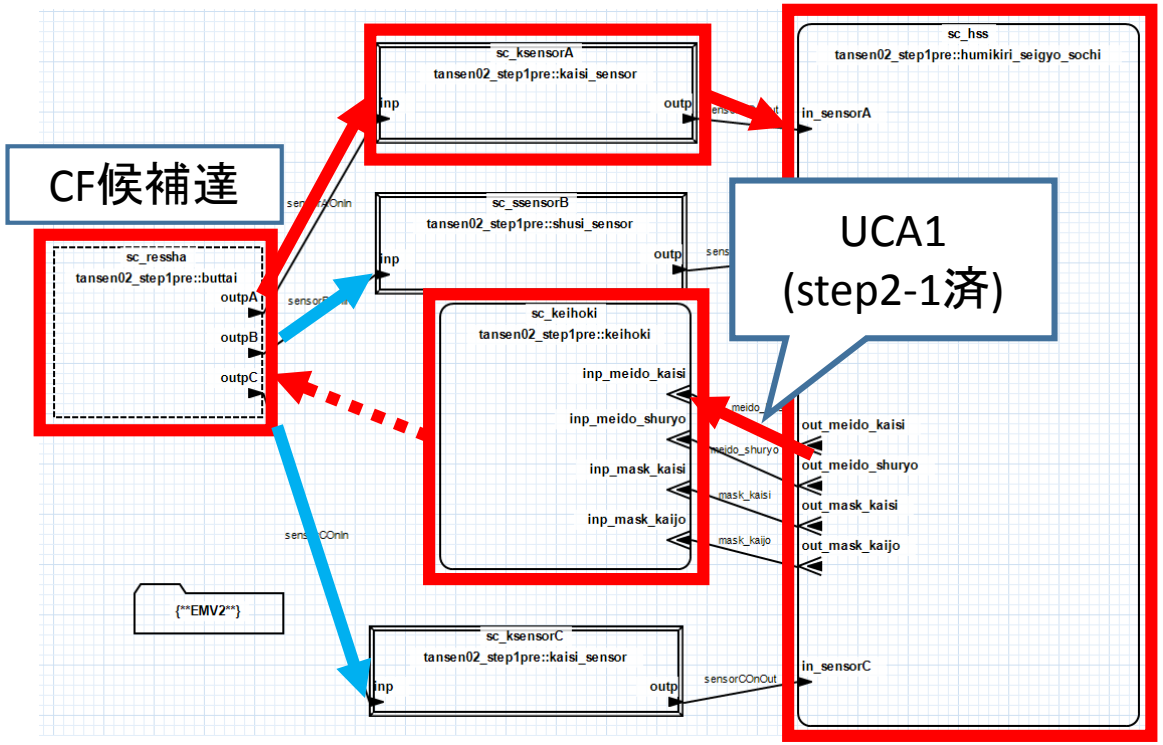
[2] Procter, S. and Hatcliff, J. An Architecturally-Integrated, Systems-Based Hazard Analysis for Medical Applications, Formal Methods and Models for Codesign (MEMOCODE), 2014
[3] Feiler, P. H. et. al., Improving Quality Using Architecture Fault Analysis with Confidence Arguments, 2015

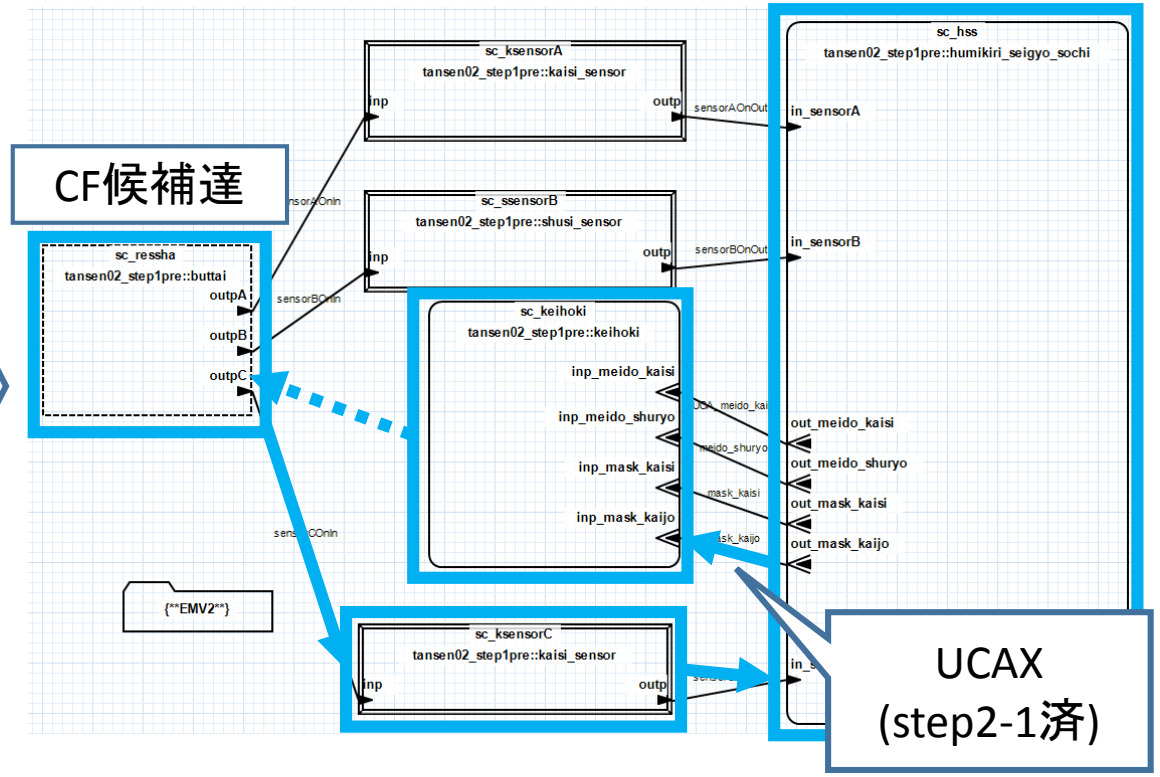
事例：AADLを用いたSTAMP/STPA支援

- 内容
  - 対象：踏切制御装置へのSTPA適用事例[4]
  - Step1-1：文献[4]の結果を利用
  - Step1-2：コントロールストラクチャーをAADL+EMV2で記述
  - Step2-1：文献[4]の結果を利用＋FIAによる支援
    - Step2-1の４タイプに基づくUCA候補の影響範囲をFIAにより指摘
  - Step2-2：FIAによる支援
    - Step2-1の４タイプに基づくCF候補の影響範囲をFIAにより指摘
- 結果
  - Step2-1の支援は効果が感じられなかった(後述)
  - CF候補の影響範囲に分析対象UCAが含まれることをFIAで指摘 [3]
  - CF候補が分析対象UCA以外のUCAを引き起こすことをFIAで指摘 [2]

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

Step2-2支援：人と機械の相補的分析([2])





UCA1のコントロールループ(赤)を分析
(CH候補達は他コントロールループ分も
含めてFIAを実施)

UCAXのコントロールループ(青)も
同時に分析

Step2-2支援：CF識別の支援([3]+)

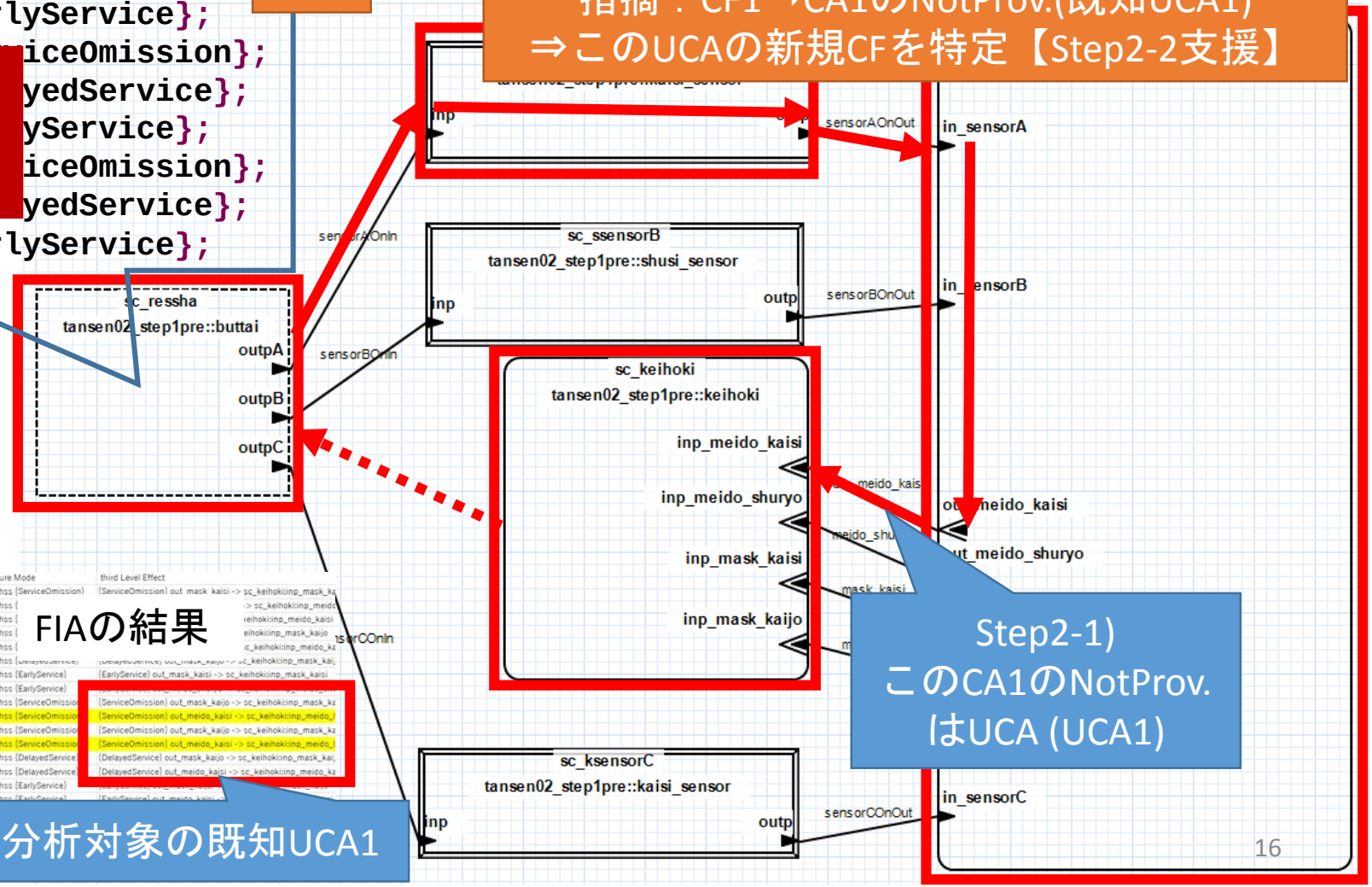
```
fs_outpA_so : error source outpA{ServiceOmission};
fs_outpA_ds : error source outpA{DelayedService};
fs_outpA_es : error source outpA{EarlyService};
fs_outpB_so : error source outpB{ServiceOmission};
fs_outpB_ds : error source outpB{DelayedService};
fs_outpB_es : error source outpB{EarlyService};
fs_outpC_so : error source outpC{ServiceOmission};
fs_outpC_ds : error source outpC{DelayedService};
fs_outpC_es : error source outpC{EarlyService};
```

Step2-1の４タイプに基づく全CF候補達をerror sourcesとしてモデルに追加

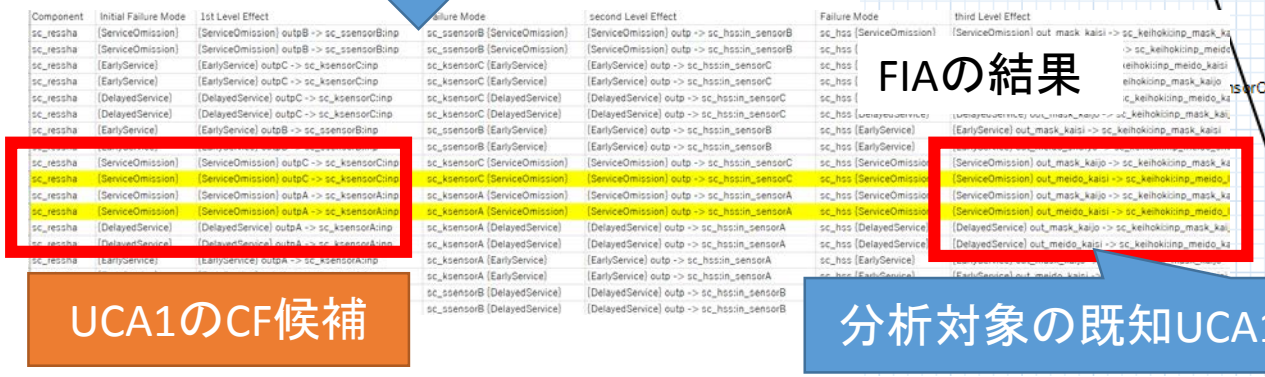
CF1

Step2-2)
指摘：CF1→CA1のNotProv.(既知UCA1)
⇒このUCAの新規CFを特定【Step2-2支援】

Step2-2)
CF候補達はUCA1を
引き起こすかをFIAで調査？



FIAの結果



UCA1のCF候補

分析対象の既知UCA1

Step2-1)
このCA1のNotProv.はUCA (UCA1)

まとめ

- 目標：人と機械による相補的STAMP/STPAの提案([2]の実現)
- 内容：AADLを用いたSTAMP/STPA支援事例の報告([2]+[3]+α)
- 事例で得た知見
  - STAMP/STPA支援に必要なAADL+EMV2のモデル要素⇒下表参照
  - Step2-1の支援は効果が感じられなかった(ハザードはモデル化されていない)
  - Step2-2の支援は有用と思われる(面倒な作業の自動化，気づき漏れの指摘)

		ARP4761 Process Elements					STPA
AADL Error Model Construct		FHA	FTA	FMEA	MA	DD	
Error flows	Error propagation	×	×	×	×	×	×
	Error source	×	×	×			×
	Error path			×			×
	Error sink		×	×			×
Error bahavior	Error states		×		×	×	?
	Error transitions		×		×	×	?
	Error events	×	×	×		×	?
	Composite error model		×			×	?
Properties	Hzards property	×					?
	OccurrenceDistribution property				×	×	

Table1: ARP4761 Process Elements and Supporting AADL Error Model Constructs +α

FHA: Functional Harzard Assessment
MA: Markov Analysis
DD: Dependence Diagram(Reliability Block Diagram)

[Delange2014] AADL Fault Modeling and Analysis Within an ARP4761 Safety Assessment, Delange, J. et.al (2014)

課題・検討項目

1. AADLモデルの詳細化[Delange2014]
   - CF詳細情報の記述(Properties::Hazard Property) ⇐[2]
   - コンポーネントの状態に依存したエラー伝搬
     (error states, error transitions, error events)
   - 複数コンポーネントの状態に依存したエラー状態(composite error model)
2. STPA Step2-1(UCA識別)の支援における課題・検討項目
   - ハザードはシステムの状態として定義される
   - UCA候補がUCAか否かはコンテキストに依存
   - FIAによる支援可能項目の検討
     - 案：分析対象が大きければ，UCA候補の影響範囲の指摘は有用？
3. STPA Step2-2(CF識別)の支援における課題
   - FIAの限界：識別可能要因は単一コンポーネントの故障のみ
     - https://github.com/osate/ErrorModelV2/issues/91

[Delange2014] AADL Fault Modeling and Analysis Within an ARP4761 Safety Assessment, Delange, J. et al. (2014)

STAMP/STPA支援用AADLモデル

- 目的：UCA候補とCF候補の影響をFault Impact Analysisで指摘
  - コントロールストラクチャー：AADLモデル
  - STPA Step2-1のタイプ：Error Model Annexのエラータイプ
- モデルのレベル(現在はレベル1) [Delange2014]
  1. Error flows(error propagations, error source, error path, error sink)
     - UCA候補： Error flows::source(Step2-1), path(Step2-2)
     - CF候補：Error flows::source(Step2-2)
     - 他のエラー伝搬： Error flows::source, path, sink
  2. Error behavior(error states, error transitions, error events)
     - コンポーネントの状態に依存して，エラー伝搬を定義できる
  3. Error behavior(composite error model)
     - 複数コンポーネントの状態をサブシステムの状態として定義できる

[Delange2014] AADL Fault Modeling and Analysis Within an ARP4761 Safety Assessment, Delange, J. et al. (2014)

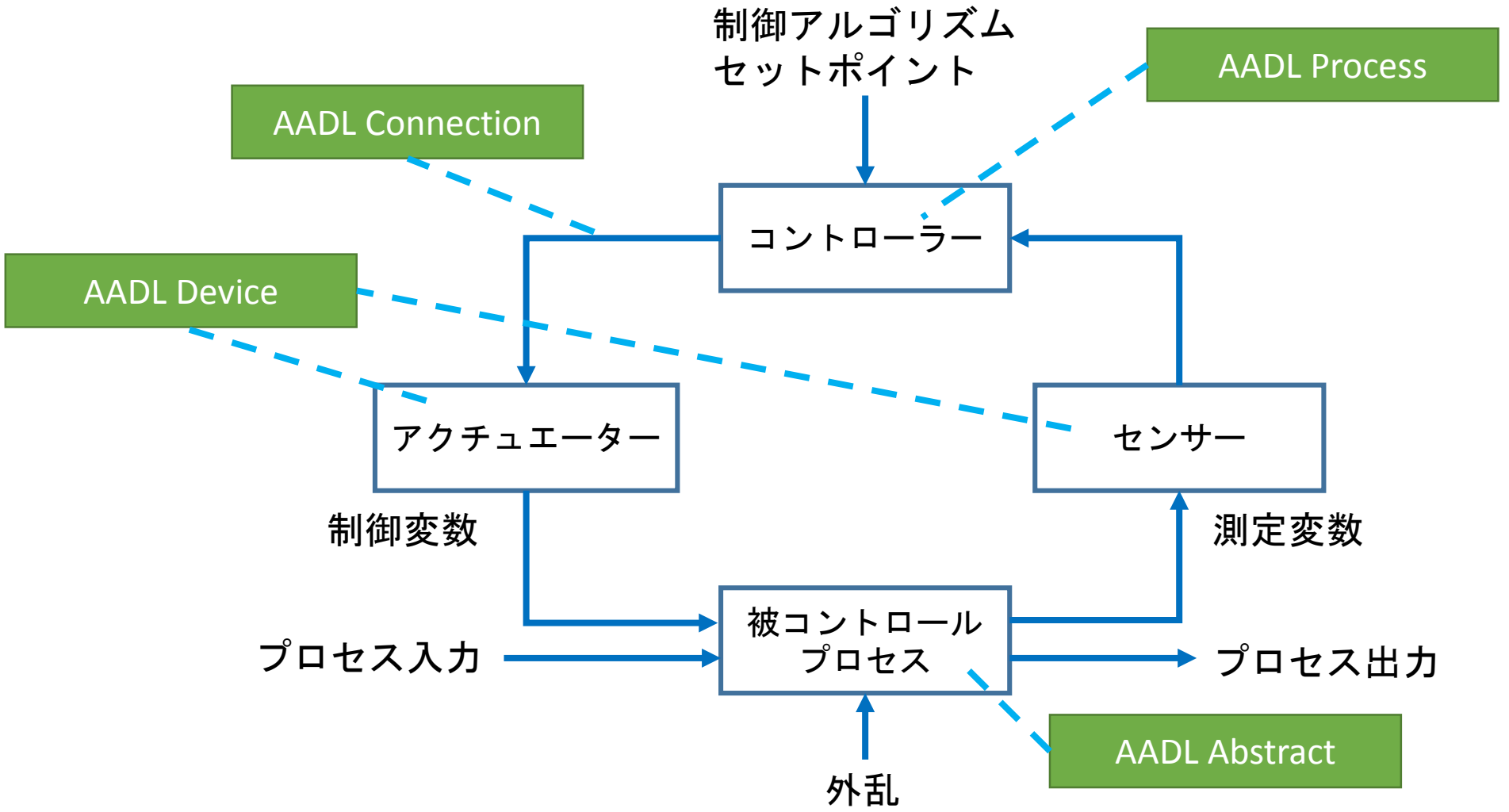
この辺りか，各stepでSTAMP/STPAの課題を入れる

- 今後普及が見込まれる手法を活用した事例の報告であり有益と感じました。
- 評価の目的に至る背景を説明いただけると、今回の組み合わせや事例がどういう問題に適しているかが明確になるように思いました。
- 計算機支援によって従来気づかなかった点は今回の事例ではどこか、今回の事例での限界はどこかといった点を明示いただけるとより有益になるように思いました。

✓支援範囲を細かくする（今回は全体を支援しているわけではない）

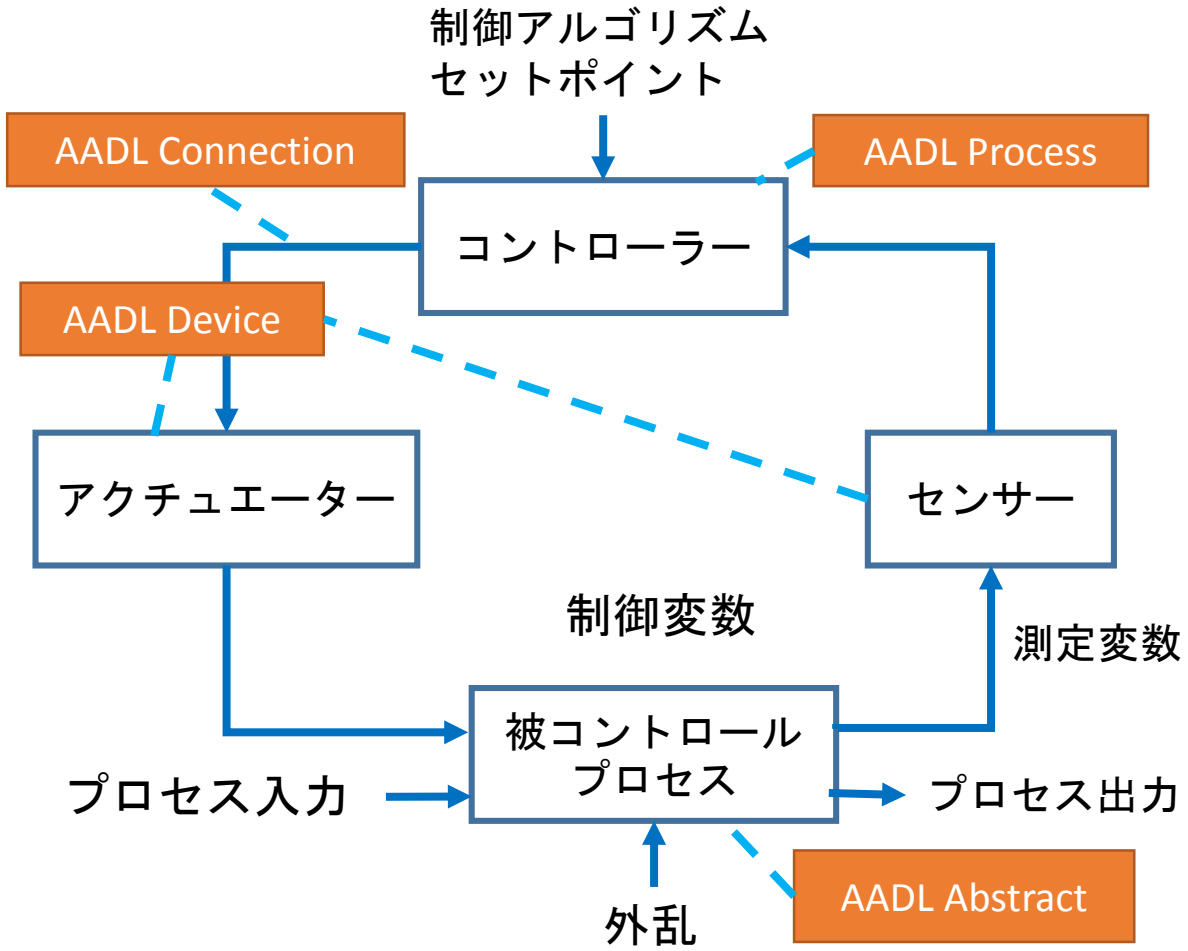
- 事例報告の書式やスペースの関係もあるだと思いますが、独自の新たな知見がどれほどあるのかよくわかりませんでした。
  - ⇒あまりないかもしれません…
  - 「Step2-2：UCAの原因の識別」に独自部分を追記
  - まとめスライドにも独自部分を追加すべき！
- ✓誤解していなければ、タイトル「AADLを用いたSTAMP/STPA支援」は広すぎるような気がします。例えば、(略語の使い方は乱暴かもしれませんが)「AADLのFIAを用いたSTAMP/STPAのCF識別支援」といったタイトルのほうが正確なように思いました。
  - ⇒タイトルにサブタイトルを追加して，タイトルをより正確にしました
  - STPAの工程解説のスライドに，今回の支援対象を追記しました.

AADLを用いたSTAMP/STPA(CS)



[2] Procter, S. and Hatcliff, J. An Architecturally-Integrated, Systems-Based Hazard Analysis for Medical Applications, Formal Methods and Models for Codesign (MEMOCODE), 2014

AADLによるSTAMP/STPA要素の記述



STAMP/STPA (Step2-1タイプ)	Error Model Annex (エラータイプ)	分析対象固有 エラー・タイプ (拡張として定義)
与えられないとハザード	ServiceOmission, …	ServiceOmission等の拡張エラー・タイプとして定義
与えられるとハザード	ServiceComission, …	ServiceComission等の拡張エラー・タイプとして定義
早すぎ、遅すぎ、誤順序でハザード	EarlyDelivery, …	EarlyDelivery等の拡張エラー・タイプとして定義
早すぎる停止、長すぎる適用でハザード	EarlyServiceTermination, …	EarlyServiceTermination の拡張エラー・タイプとして定義

[2] Procter, S. and Hatcliff, J. An Architecturally-Integrated, Systems-Based Hazard Analysis for Medical Applications, Formal Methods and Models for Codesign (MEMOCODE), 2014

AADLを用いたSTAMP/STPA(ガイドワード)

STAMP/STPA (Step2-1 タイプ)	Error Model Annex (エラータイプ)	分析対象固有エラー・タイプ (拡張として定義)
与えられないとハザード	ServiceOmission, ItemOmission, …	**ServiceOmission等の 拡張エラー・タイプとして定義**
与えられるとハザード	ServiceComission, …	**ServiceComission等の 拡張エラー・タイプとして定義**
早すぎ、遅すぎ、 誤順序でハザード	EarlyDelivery, LateDelivery, …	**EarlyDelivery等の 拡張エラー・タイプとして定義**
早すぎる停止、 長すぎる適用でハザード	EarlyServiceTermination, LateServiceTermination,	**EarlyServiceTermination等の 拡張エラー・タイプとして定義**

[Feiler2016] Architecture Fault Modeling and Analysis with the Error Model Annex, Version 2, Feiler, P. et al. (2016)

これを入れるかは微妙…⇒課題で言う？

課題：Step2-1支援

人間の気づき

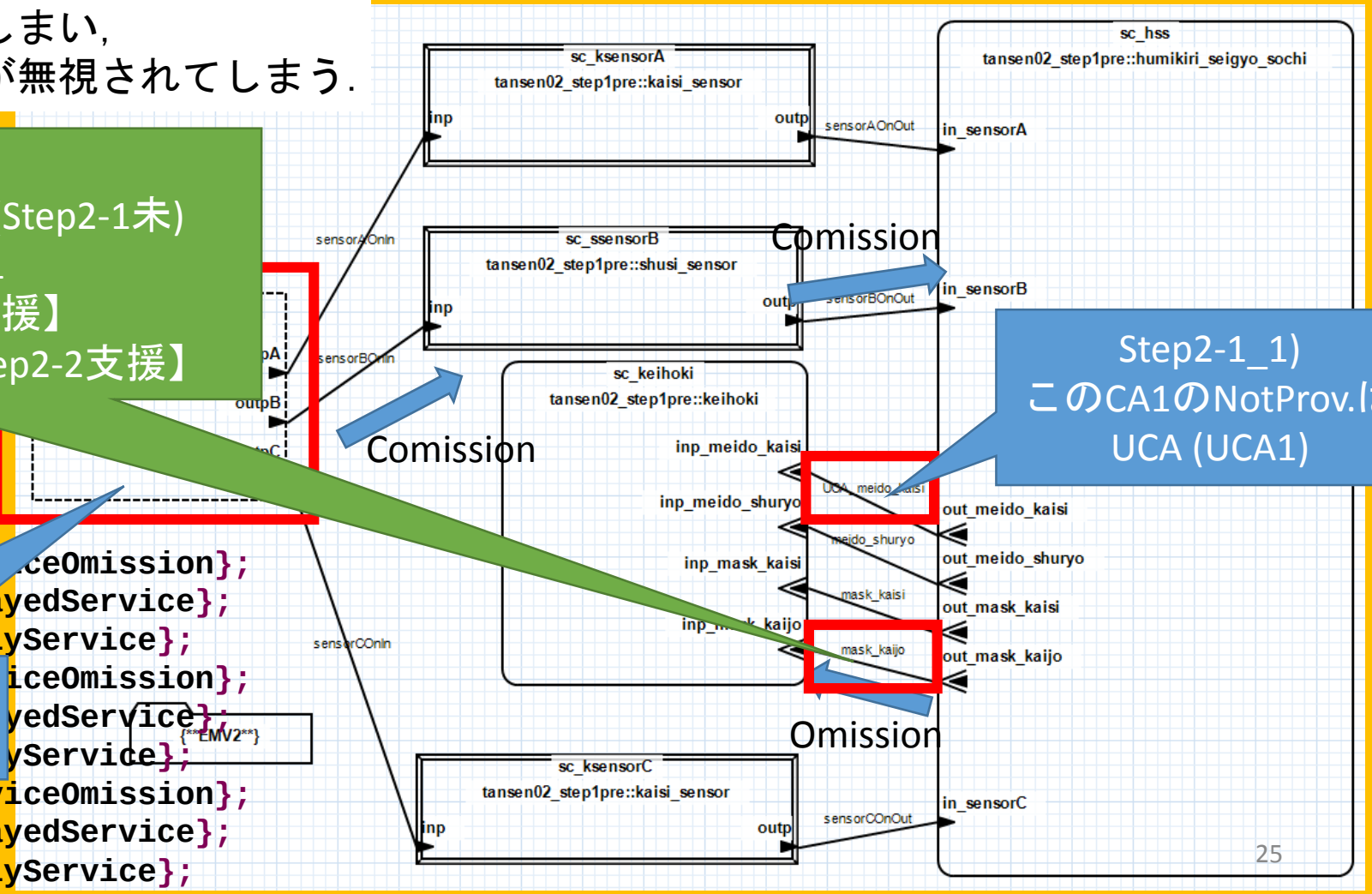
マスク開始がcomissionで入ると，マスクされ続けてしまい，
新たな鳴動開始が無視されてしまう．

限界：「ハザード＝システムの状態」なので，今回のモデルではハザードへ至るか否かを自動判定できない．

Step2-X_1)
指摘：CF1→このCAのNotProv.(Step2-1未)
⇒人間によるStep2-1
⇒UCAと識別【Step2-1支援】
＋このUCAのCF(CF1)の特定【Step2-2支援】

Step2-1_1)
このCA1のNotProv.は
UCA (UCA1)

選択したCF候補達をerror sourcesとしてモデルに追加




```
fs_outpA_so : error source outpA{ServiceOmission};
fs_outpA_ds : error source outpA{DelayedService};
fs_outpA_es : error source outpA{EarlyService};
fs_outpB_so : error source outpB{ServiceOmission};
fs_outpB_ds : error source outpB{DelayedService};
fs_outpB_es : error source outpB{EarlyService};
fs_outpC_so : error source outpC{ServiceOmission};
fs_outpC_ds : error source outpC{DelayedService};
fs_outpC_es : error source outpC{EarlyService};
```