Embedded Success dSPACE

システムアプローチによる機能安全への対応

2015.08.28
第１７回 組込みシステム技術に関するサマーワークショップ@水明館（岐阜県/下呂市）

dSPACE Japan 株式会社
藤倉 俊幸

目次

dSPACE

- 安全の考え方
- システムアプローチの必要性
- まとめ

安全とは

- 安全とは、危害のない状態
- 危害とは、死ぬ、怪我する、壊れる等

- 安全を確保するためには、どのような危害が存在するか確認する必要がある

ハザードと危険状態

- ハザードは、危害の源泉
- 危険状態は、ハザードに曝されている状態

ハザード　＋　起因事象　＝　危険状態

尖がったところ
高温になる部分
重たい物
高速に動くところ
…

エネルギーがたまっているところ

故障
天候の急変
勘違い・見間違い
接近・接触
停電
摩擦
静電気スパーク

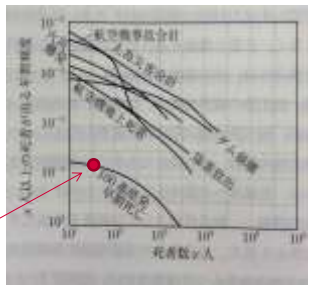
状態が変わる時

リスク

- 危害の程度とその発生確率を表した概念
- 安全性を数値化

$$r_i = \{s_i, p_i\}$$

リスク　厳しさ　その確率



安全対策の考え方

ハザード ＋ 起因事象（p_i） ＝ 危険状態 ➡ 危害（s_i）

除去
閉じ込め　　回避
閉じ込め　　軽減/緩和

- ハザードから入るか、危害から入るか

本質安全と機能安全

- 本質安全
  - ハザードそのものを除去/閉じ込めをするアプローチ
- 機能安全
  - ハザードを安全機能を追加することで回避/軽減/緩和するアプローチ

システムセーフティーとは

- 以下のハザードに対応すること

1. 要求が満たされなかった時に発生するハザード
   - 故障、バグ
2. 個々の要求は満たされているが、要求間の矛盾により発生するハザード
   - 各コンポーネントの状態の不整合
3. 要求の不足により発生するハザード

現状の26262は①のみ対応している。
システムセーフティーと言った場合は①から③まで含む?

2つのアプローチ

- 26262
  - 機能不全があったときに、安全でない状態にならない様に対策する

E/E安全関連システムの機能不全の振る舞いによって引き起こされる潜在的なハザードを扱う。「機能不全の振る舞い」(malfunctioning behavior)は、故障(failure)、またはアイテムの意図しない振る舞い(unintended behavior)である。ただし、性能限界は扱わない。

  - 機能不全が無くても、組合せ動作によって、安全でない状態にならない様に対策する
    - ただし、制御構造が対象でガイドワードによって安全でない状態を抽出する

26262フレームワーク

- 要求Rが満たされない状態　¬Rで発生する危険状態を、安全機能Sによって安全状態にする取り組み

Rが真 = 基本機能が正しく機能している(安全)

¬R = 危険

Sが真 = 安全機能が正しく機能している(安全)

¬R∧S = 安全

∴ R∨(¬R∧S) = 安全

R	S	R∨(¬R∧S)
T	T	T
T	F	T
F	T	T
F	F	F

意図しない振る舞い(unintended behavior) →

機能不全がない場合

- 現状の26262では何らかの故障や障害によって引き起こされる危険状態を回避することで安全を確保する
- 全てのコンポーネントが正常に動作していても危険な状態になる場合に対応できない
- 一般要求の段階で安全性を十分考慮する必要がある
- STAMP/STPA
  - Engineering a Safer World
    - ネットからダウンロード出来る

https://mitpress.mit.edu/sites/default/files/titles/free_download/9780262016629_Engineering_a_Safer_World.pdf

STAMP/STPAの概要

- STAMP（Systems-Theoretic Accident Model andProcess）： システム理論に基づく事故モデル
- STPA（STAMP based Process Analysis）：STAMPに基づく安全解析手法
- マサチューセッツ工科大学(MIT)のNancy.G.Leveson教授が、最新文献“Engineering a Safer World”の中で提唱
- 複数のコントローラが介在する複雑なシステムに対する安全解析の方法論
  - システムを構成するサブシステムやコンポーネントに不具合がなくとも，サブシステムやコンポーネントの組み合わせによって全体のシステムにおける不具合が発生する

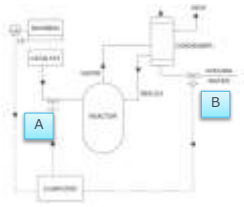
STAMP/STPAの手順

1. ハザード制御に関わるControlStructureの作成
2. 非安全なControlAction の識別によるハザードシナリオの析
   - ４つのガイドワードを用いて、危険な状態を導くコントローラの動作（非安全なコントロールアクション：UCA）を識別する
3. Control Loopの作成によるハザード要因の分析
   - Control Loop上のガイドワードを用いて、UCAの要因を識別する。特に、ソフトウェアやヒューマンに起因する要因として，コントローラの想定するプロセスモデルが，実際のプロセスの状態と矛盾することで起きる要因を識別する
4. 安全制約の識別

4つのガイドワード

1. “Not Provided”
   - 必要なコントロールアクションが供給されない
2. “Incorrectly Provided”
   - 誤った非安全なコントロールアクションが供給される
3. “Provided Too Early, Too Late, or Out of Sequence”
   - 意図しないタイミングで供給される
4. “Stopped Too Soon”
   - 途中で止まる（または必要以上に長く実施される）

モデル検査手法を使うと



R = 反応中
A = Aバルブopen中
B = Bバルブopen中
C = 冷却中

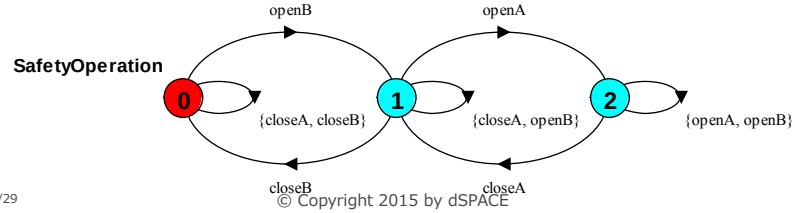
- 反応中は冷却していること
  - []($R \Rightarrow C$)

これが不足していた要求

```
fluent C = <openB, closeB> initially 0
fluent R = <openA, closeA> initially 0

assert Req = [](R -> C)
constraint Req = Req

||SafetyOperation = (Req).
```



HAZOP→SoftwareHAZOP

- HAZOP（HAZard and OPerability studies）はガイドワードを使用してハザードを特定する手法である
- ガイドワードの解釈は対象とコンテキストに依存する
- Software HAZOPは、データフロー図や状態マシン、クラス図記述に対象としたHAZOPである。これらの記述はアイテム定義における記述と類似している
- Software HAZOPのガイドワード解釈をアイテム定義における構造記述と動作記述に適用することで効率的・網羅的にハザードを抽出できる

一般HAZOP

- 初期アーキテクチャに依存しないハザード特定にHAZOPを利用する
  - 設計FMEA：部品に依存→アーキテクチャに依存
  - FTA：ハザードの原因を分析するもの、まずハザードが特定されていることが前提
  - 機能FMEA：アーキテクチャに依存しないが網羅性に不安がある、ハザード間の因果関係を分析、一般的でない(medini analyser独自)
  - HAZOP：機能FMEAと似ている。ガイドワードにより網羅性を確保、手法として確立している

一般ガイドワード

	一般的意味
No	意図された結果が達成されない、他に何も起こらない
More	定量的に多すぎる
Less	定量的に少なすぎる
As well as	意図されたことの全てが達成されるが、他に何かが起こる。 定性的に多い
Part of	意図されたことの一部のみ達成される。 定性的に少ない
Reverse	意図されたこととは論理的に逆のことが起こる
Other than	意図されたこと以外の何かが起こる
Early	時刻・相対時間に対して早すぎ
Late	時刻・相対時間に対して遅すぎ
Before	相対的なorder(離散的)やsequence(連続)が前になる
After	相対的なorder(離散的)やsequence(連続)が後になる

- 対象とコンテキストによって使用するガイドワードの選択と解釈を変える必要がある

結局ガイドワードしかないのか

- 安全要求さえ明確になれば、形式手法やシミュレーションによって実現・検証できる
- 安全要求を作るきっかけはガイドワードしかないのか?

システム思考が重要

何を目的として
何処まで
何を考えれば良いのか

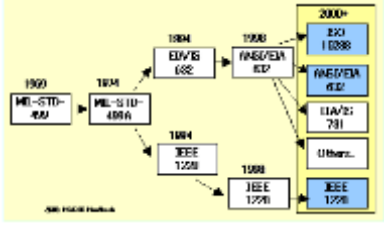
- トンネル崩落と言うこともある
  - メンテナンスの問題か
- 逃げ切った車がある
  - 屋根が丈夫、加速が良い
- 株が上がる

禍福は糾える縄の如し

http://bfaction.exblog.jp/18733173/

システムズエンジニアリング

- システムズエンジニアリングとは、大規模・複雑・高機能なシステムを実現するためのフレームワーク
  - MBSEは、文書ベースのアプローチからSysMLを中心としたモデルベースに移行した形態
- INCOSE (The International Council on Systems Engineering) 1995年に設立
- 米国国防省や米国航空宇宙局また欧州宇宙標準協会等の実践的経験に基づくベストプラクティス集を基盤としてISO/IEC 15288を2002年に制定



http://se.rdy.jp/standard.html

システムズエンジニアリングの範囲

ISO 15288

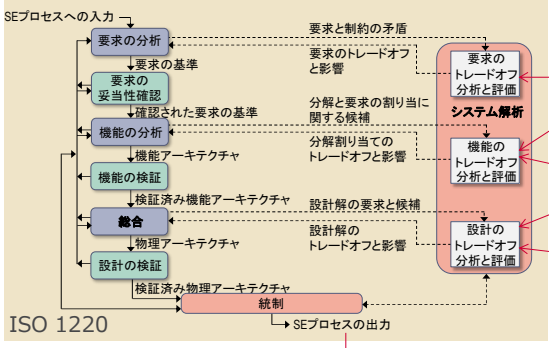
管理・組織なども含んでいる

ライフサイクルのステージ

1. 概念検討
2. 開発
3. 製造
4. 利用移行
5. 利用
6. 運用
7. 廃棄

作るもの以上の想像を働かさなければならない…

MBSEにおける安全性分析



一般要求の段階で安全性を考慮する手法

ミスユースケース

STAMP/STPA
HAZOP

モデルベースFTA/FMEA/HAZOP

MBDプロセス

まとめ

- リスクの特定ができれば、形式手法やシミュレーションによって、完全で一貫性のある実現・検証はできる
- リスクの特定が重要
  - 現状はガイドワードに依存している
  - メタモデルと事例を使ってリスクを特定するのはどうか
- システム思考によるリスク特定が必要
  - システムズエンジニアリングが定義しているライフサイクルおよびプロセスが参考になる