「EV時代の到来とともに増加する ソフトウェア起因の事故に対峙するには」

～ IoT時代に必要なリスクベースアプローチと安全アーキテクチャ ～」

Rev.1

JaSST'19 Tokai

2019年10月4日

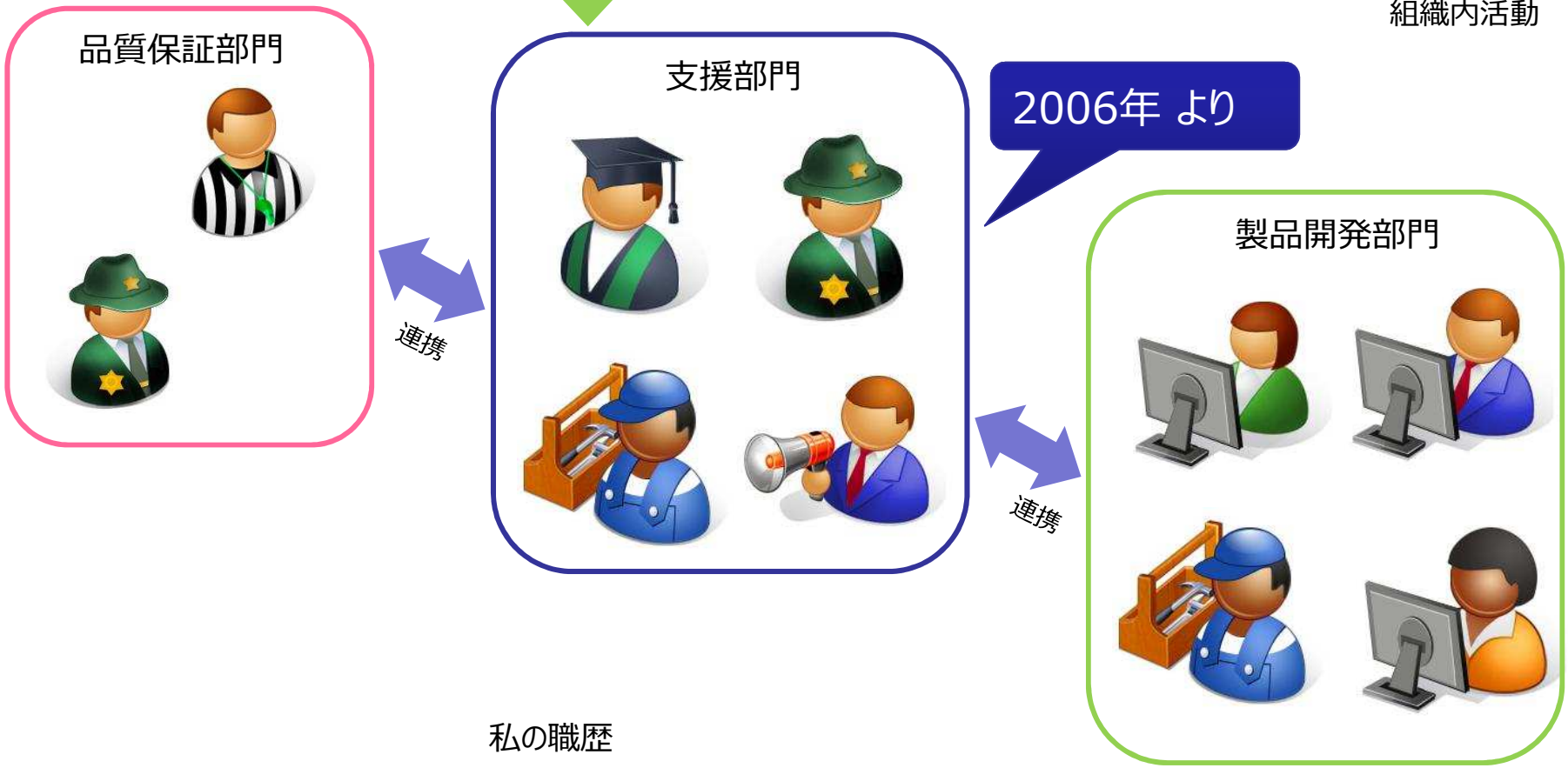
組込みソフトウェア管理者・技術者育成研究会 酒井 由夫

IEC SC62A TC62 JWG3 / JWG7 Japan Committee Member
・Discussing IEC 62304, IEC 82304-1, IEC80001 Series etc.

JEITA 電子情報技術産業協会

工業会の活動

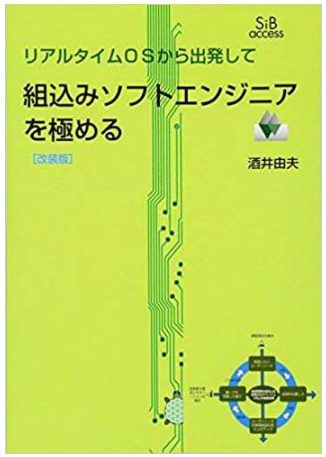
組織内活動



私の職歴

19年	13年
1987-2006 ソフトウェアエンジニア、アーキテクト、マネージャ	2006－1019 規格適合支援、ソフトウェア品質支援

著書



2006年

2010年

2016年

ブログ

組込みソフトウェア工房

特集記事

- ISO 26262 との向き合い方
- USのヘルスITの取り組み
- RaspberryPi3 を使ったリアルタイム波形描画

問題提起

そもそも何のために安全に関するソフトウェアの活動は必要なのか？

- 歴史的には事故の再発防止のために技術は発展してきた。
- FMEAしかり、FTAしかり
- 自動車業界ではソフトウェア起因の事故は頻発している？（してないのでは？）
- でも、今後 EV や自動運転技術導入後も同じか？（同じではないのでは？）

- まったく異なる業務ドメインの技術者やソフトウェアと連携してサービスを提供することになる。
  - それが原因で発生するインシデントやアクシデントが増えてくる。それをどうやって防ぐのか。

→新しい要求、基本機能、基本アーキテクチャ、新しい要求と基本機能との関係性、ハザードに至る一連の事象、安全を実現するアーキテクチャが必要になる。

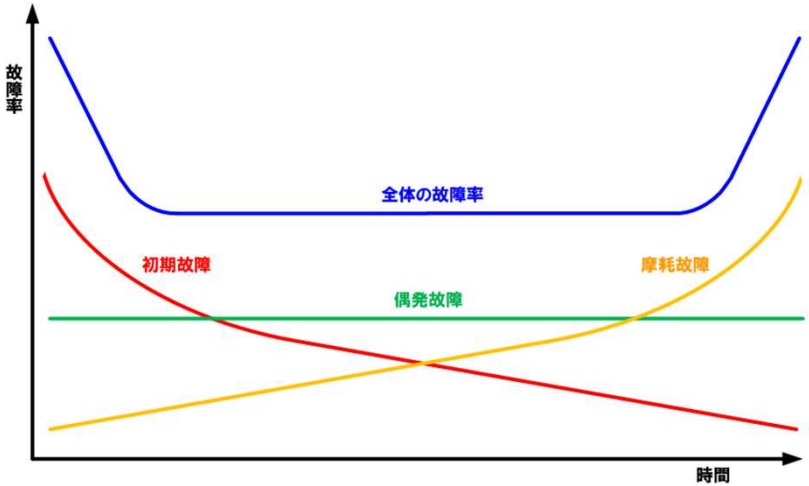
1. 国際規格との向き合い方

a. ソフトウェア起因の不具合の特徴
b. ソフトウェア品質の歴史的推移
c. 文化の違い
d. そもそも国際規格ってなに？
e. ソフトウェア系の規格を認証する意味とは？
f. ISO 26262の概要とポイント
g. ISO 26262の工業会の取り組み

ソフトウェア起因の不具合の特徴

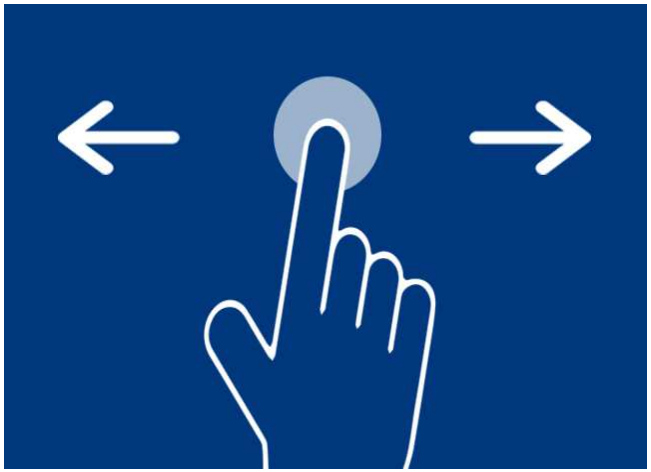
HARDWARE

ランダム故障は確率論的に発生する。



USABILITY

ユーザビリティエラーは可能性を推定する。



HIGH

しばしば
時々
わずかに
起こりそうに
ない
考えられない

LOW

SOFTWARE

ソフトウェア起因の不具合の確率や可能性を推定することは困難である。

- ソフトウェアは見えない.
- ソフトウェア起因の不具合は確率論的ではなく、決定論的に発生する。

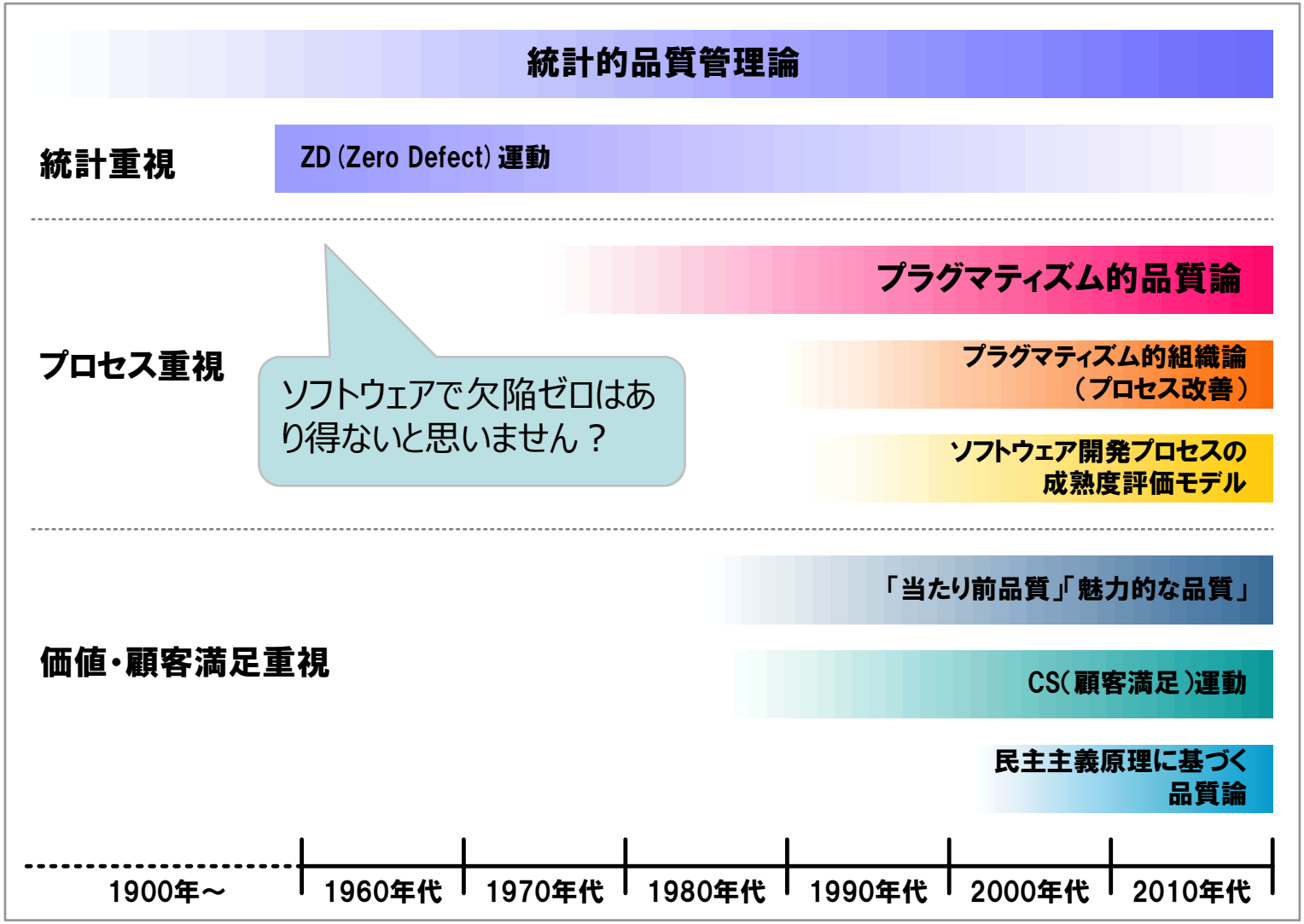
Systematic Failures/Faults（決定論的原因故障/障害）

- ハードウェアの設計に起因するもの、ソフトウェアのバグやソフトウェアに起因する問題やユーザーオペレーションが原因で発生する障害発生率の予測が難しい故障/障害
- 出荷前の検査で発見することが難しく、出荷後に故障や障害が発生してから始めて分かることが多い。
- 開発のプロセス（工程）、ライフサイクルの中で Systematic Failures/Faults の作り込みを防止し、検証や妥当性確認によって発見・除去する。

参考：IEC 61508-3:2010, ISO 26262-1:2011

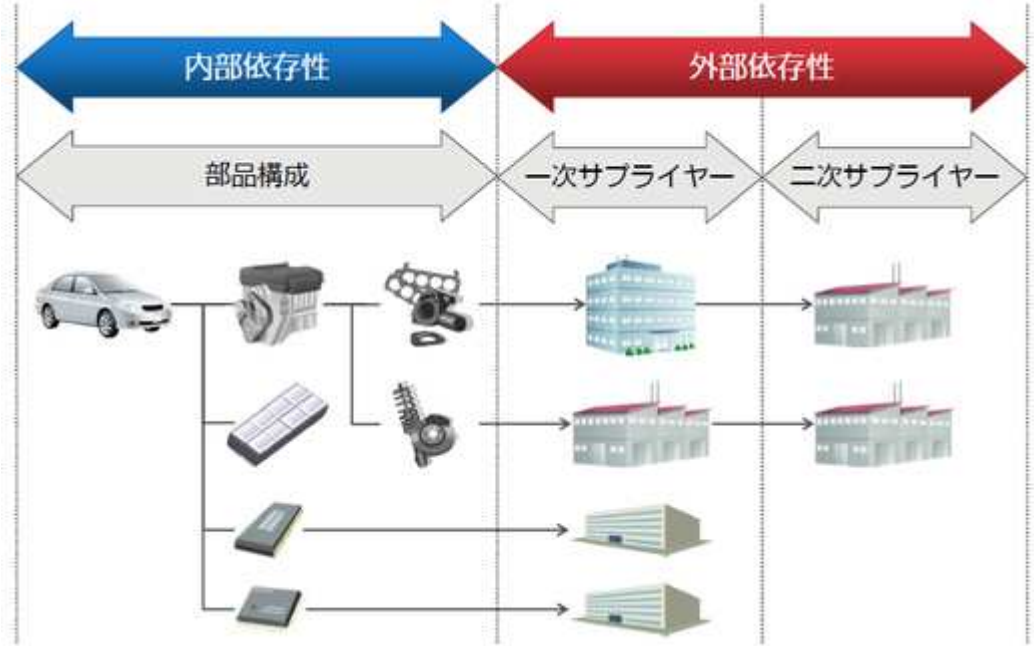
ソフトウェア品質論の歴史的推移

（広島市立大学 大場充 教授 講演「民主主義原理に基づくソフトウェア品質論入門」を参考に作成）



問題提起

- 自動車業界では、サプライチェーン上のサプライヤから供給された部品をアセンブリして商品化する製造工程を永年続けてきたため、部品の信頼性を高めることで製品の信頼性も高まると考えているのではないか？



その考え方は、数百万行を超えた規模のソフトウェアシステムには、使えなくなっている。

もしも、できているならば奇跡的であり、日本人の“品質を心配する意識”がギリギリ実現可能にしているのではないか。

出典：https://www.fujitsu.com/jp/group/fri/column/opinion/201112/2011-12-2.html

安全設計の分類

個別最適の発想

フォールト・アボイダンス

個々の構成要素の信頼性を高めることで安全性を確保しようとする設計（構成要素の故障やバグを認めない）

対照的

全体最適の発想

フェール・セーフ

個々の構成要素に故障やバグがあっても必ず安全側に落ち着くようにする設計（信頼性よりも安全性を優先する）

フォールト・トレランス

個々の構成要素の故障やバグがあっても別の手段（冗長性や多重化など）によって機能を維持しようとする設計（構成要素の故障やバグを前提とする）

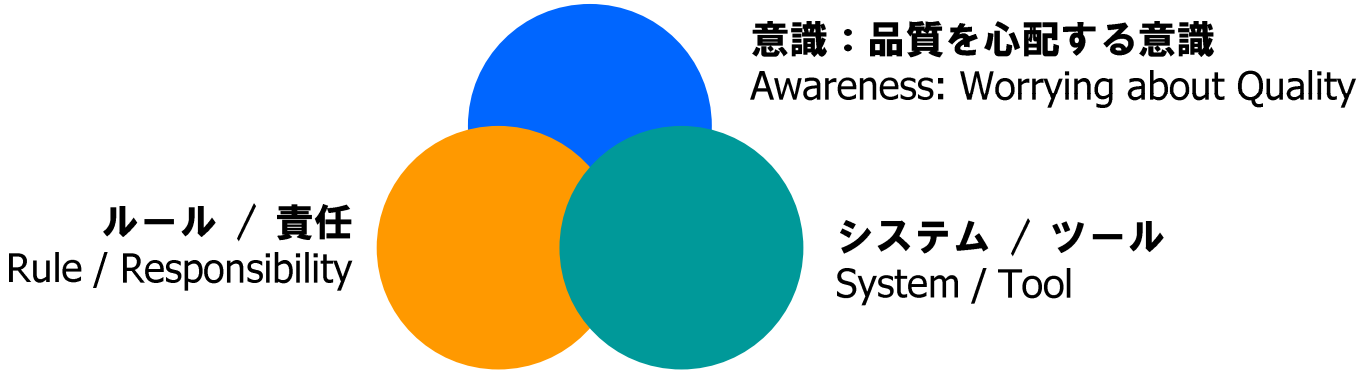
エラー・プルーフ
（フール・プルーフ）

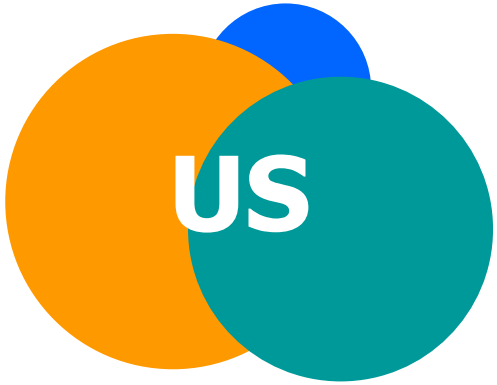
間違った操作をしようとしても危険が発生しないようにする設計。

ユーザビリティ

機器の使用者

文化の違い (Difference in Cultures)





顧客満足度 99.5～99.6%

顧客満足度 99.7～99.8%

そもそも国際規格って何？

ISO規格とは

- ISOとは International Organization for Standardization（国際標準化機構）の略称
- 電気・電子及び電気通信以外のあらゆる分野の国際規格の作成を行う国際標準化機関

IEC規格とは

- IECとは International Electrotechnical Commission（国際電気標準会議）の略称
- 電気及び電子技術分野の国際規格の作成を行う国際標準化機関

標準規格には、強制規格と任意規格があり、ISO, IEC, JIS は任意規格であり、規制に使うかどうかは、各国の規制当局が決める。

ISO/IEC規格は、各国から代表（国際エキスパート）が新規提案、コメント募集、議論を重ね、最終的には投票によって国際規格とするかどうかを決定する。規格の見直しの期間も定められている。→国際規格を神の啓示のようにあがめるのは間違い。
規格を取りまとめている国内委員会に参加すれば、意見を言うこともできる。

ソフトウェア系の規格を認証する意味とは？

自動車
メーカ

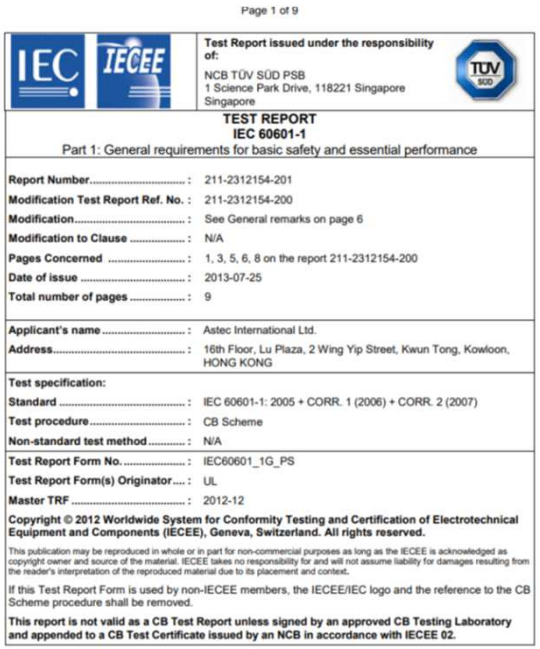
ソフトウェアエンジニア

第三者認証機関による試験

プライベート認証

IEC IECEE CB SCHEME

認定されたテスト機関による試験



Page 1 of 9

Test Report issued under the responsibility of:
NCB TÜV SÜD PSB
1 Science Park Drive, 118221 Singapore
Singapore

TEST REPORT
IEC 60601-1
Part 1: General requirements for basic safety and essential performance

Report Number	211-2312154-201
Modification Test Report Ref. No.	211-2312154-200
Modification	See General remarks on page 6
Modification to Clause	N/A
Pages Concerned	1, 3, 5, 6, 8 on the report 211-2312154-200
Date of issue	2013-07-25
Total number of pages	9
Applicant's name	Astec International Ltd.
Address	16th Floor, Lu Plaza, 2 Wing Yip Street, Kwun Tong, Kowloon, HONG KONG
Test specification:	
Standard	IEC 60601-1: 2005 + CORR. 1 (2006) + CORR. 2 (2007)
Test procedure	CB Scheme
Non-standard test method	N/A
Test Report Form No.	IEC60601_1G_PS
Test Report Form(s) Originator	UL
Master TRF	2012-12

Copyright © 2012 Worldwide System for Conformity Testing and Certification of Electrotechnical Equipment and Components (IECEE), Geneva, Switzerland. All rights reserved.

This publication may be reproduced in whole or in part for non-commercial purposes as long as the IECEE is acknowledged as copyright owner and source of the material. IECEE takes no responsibility for and will not assume liability for damages resulting from the reader's interpretation of the reproduced material due to its placement and context.

If this Test Report Form is used by non-IECEE members, the IECEE/IEC logo and the reference to the CB Scheme procedure shall be removed.

This report is not valid as a CB Test Report unless signed by an approved CB Testing Laboratory and appended to a CB Test Certificate issued by an NCB in accordance with IECEE 02.

CBスキーム認証

DEPARTMENT OF HEALTH & HUMAN SERVICES • USA
FDA

規制当局

認証を受けることの意味

良い面

- 自社の取り組みについて第三者に認められたことになる。
- 取り組みを普及促進、継続するためのモチベーション・インセンティブとなる。
- 業界全体の底上げにつながる。

悪い面

- そもそも、ソフトウェア系の規格要求への適合の可否の判定が妥当かどうかが怪しい。
- 規格に適合することが目的となり、本質（何のための活動か？）が忘れられる。
- 規格認証ビジネスが活発になり、認証コストが利益を圧迫する。

【大事なこと】
- 規格要求事項の一つ一つが「何のためにあり」「目的（例えば安全）にどう役立つのかを常に考えておくこと。
- 目的（安全）に役立つという確信がなければ、モチベーション高く活動を続けることはできない。

ISO 26262 の概要図とポイント

Part 1. 用語集

Part 2. 機能安全の管理
- 2.5 全体的な安全管理
- 2.6 プロジェクト依存の安全管理
- 2.7 生産,運用,サービス及び廃棄に関する安全管理

Part 3. コンセプトフェーズ
- 3.5 アイテム定義
- 3.6 ハザード分析及びリスクアセスメント
- 3.7 機能安全コンセプト

Part 4. システムレベルにおける製品開発
- 4.5 システムレベルにおける製品開発での一般的なトピックス
- 4.6 技術安全コンセプト
- 4.7 システム及びアイテム統合並びにテスト
- 4.8 安全妥当性確認

Part 7. 生産,運用,サービス及び廃棄
- 7.5 生産,運用,サービス及び廃棄の計画
- 7.6 生産
- 7.7 運用,サービス及び廃棄

Part 12. モーターサイクルへの適応
- 12.5 モーターサイクルへの適応の一般的なトピックス
- 12.6 安全文化
- 12.7 確証方策
- 12.8 ハザード分析及びリスクアセスメント
- 12.9 車両統合及びテスト
- 12.10 安全妥当性確認

Part 5. ハードウェアレベルにおける製品開発
- 5.5 ハードウェアレベルにおける製品開発の一般的なトピックス
- 5.6 ハードウェア安全要求の仕様
- 5.7 ハードウェア設計
- 5.8 ハードウェアアーキテクチャメトリックの評価
- 5.9 ランダムハードウェア故障による安全目標侵害の評価
- 5.10 ハードウェア統合及び検証

Part 6. ソフトウェアレベルにおける製品開発
- 6.5 ソフトウェアレベルにおける製品開発の一般的なトピックス
- 6.6 ソフトウェア安全要求の仕様
- 6.7 ソフトウェアアーキテクチャ設計
- 6.8 ソフトウェアユニット設計及び実装
- 6.9 ソフトウェアユニット検証
- 6.10 ソフトウェア統合及び検証
- 6.11 組込みソフトウェアのテスト

Part 8. 支援プロセス
- 8.5 分散開発でのインタフェース
- 8.6 安全要求の仕様及び管理
- 8.7 構成管理
- 8.8 変更管理
- 8.9 検証
- 8.10 文書管理
- 8.11 ソフトウェアツールの使用への信頼
- 8.12 ソフトウェアコンポーネントの認定
- 8.13 ハードウェアエレメントの評価
- 8.14 使用実績による論証
- 8.15 ISO 26262の適用範囲外のアプリケーションとのインタフェース
- 8.16 ISO 262626に準拠して開発していない安全関連システムの統合

Part 9. 自動車用安全度水準(ASIL)指向及び安全指向の分析
- 9.5 ASILテーラリングのための要求のデコンポジション
- 9.6 エレメントの共存に関する基準
- 9.7 従属故障の分析
- 9.8 安全分析

Part 10. ISO 26262ガイドライン

Part 11. 半導体へのISO 26262の適用の指針

3.6 及び 12.8 の
ハザード分析及びリスクアセスメント
12.10 安全妥当性確認

が EVや自動運転が実用化されると重要になると予想される。

構成管理、変更管理を含むソフトウェアのプロセスアプローチは近代のソフトウェア開発では、避けることのできない管理手法である。

安全を実現するために必要な具体的な要求がなんなのかを押さえる。

安全を実現するためのソフトウェアアーキテクチャが、ソフトウェアの変更によって、崩れていないかどうかを確認する必要がある。

ツール使用の信頼性やソフトウェアコンポーネントの認定は全体の中の一部にすぎない。

出典：一般財団法人 日本自動車研究所（JARI） WEBサイトより
http://www.jari.or.jp/tabid/112/Default.aspx

ISO 26262に対する工業会の取組み

出典：一般財団法人 日本自動車研究所（JARI） WEBサイトより
http://www.jari.or.jp/tabid/112/Default.aspx

2. そもそも何のための活動？

1. セーフティ・クリティカルシステムを取り巻く環境
2. ソフトウェアに対する安全要求がある分野
3. プロセスアプローチだけで安全は確保できるか？

そもそも（機能安全って）何のための活動なの？

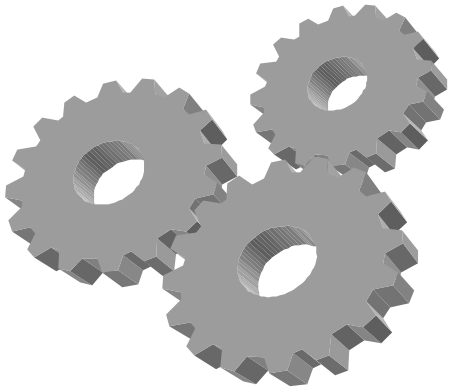
安全の技術は事故の再発防止のため

- 歴史的には事故の再発防止のために技術は発展してきた。
- FMEAしかり、FTAしかり、Thearc−25しかり
- 自動車業界では、何か現時点で起こっているのか？
- 日本では、技術者の気付き（Awareness）で未然に防いでいたのではないか。

今後の EV でも同じように、Awareness事故を防ぐことができるのか？

Topics

FTA/FMEAの歴史的背景



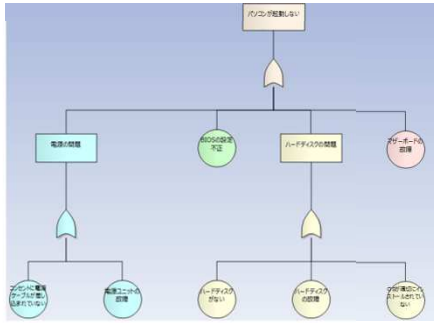
FTA (Fault Tree Analysis) とは

- フォルトツリー解析（Fault Tree Analysis：略してFTA）とは、故障・事故の分析手法である。
- 国際規格
  - IEC 61025:1990 Fault tree analysis(FTA)
  - JIS Z 8115:2000
    - フォールトの木解析と定義されているが、故障の木解析と呼ぶこともある。
  - JIS C 5750-4-4ディペンダビリティ マネジメント
    - 第 4-4 部：システム信頼性のための解析技法―故障の木解析（FTA）
- FTAは、望ましくない事象に対しその要因を探る、トップダウンの解析手法を特徴とする。これは、類似のFMEA（故障モード影響解析）とは逆のアプローチになる。

FTA(Fault Tree Analysis)の歴史

現在

原子力、化学プラント、交通システムなどに幅広く利用されている。



1965年

ボーイング社によって補完され完成度が高められた。

1961年

米国のBELL研究所のＨ・Ａ・ワトソンらによってミニットマンミサイルの信頼性評価・安全性解析を目的に考案された。

ミニットマンミサイル

■旧ミサイル
旧大陸間弾道ミサイル、アトラス・タイタンは液体燃料ロケットで、発射直前に酸化剤の注入に必要性があり即応性に問題があった。
- 燃料漏れ事故が多かった

ハードウェア
起因の事故

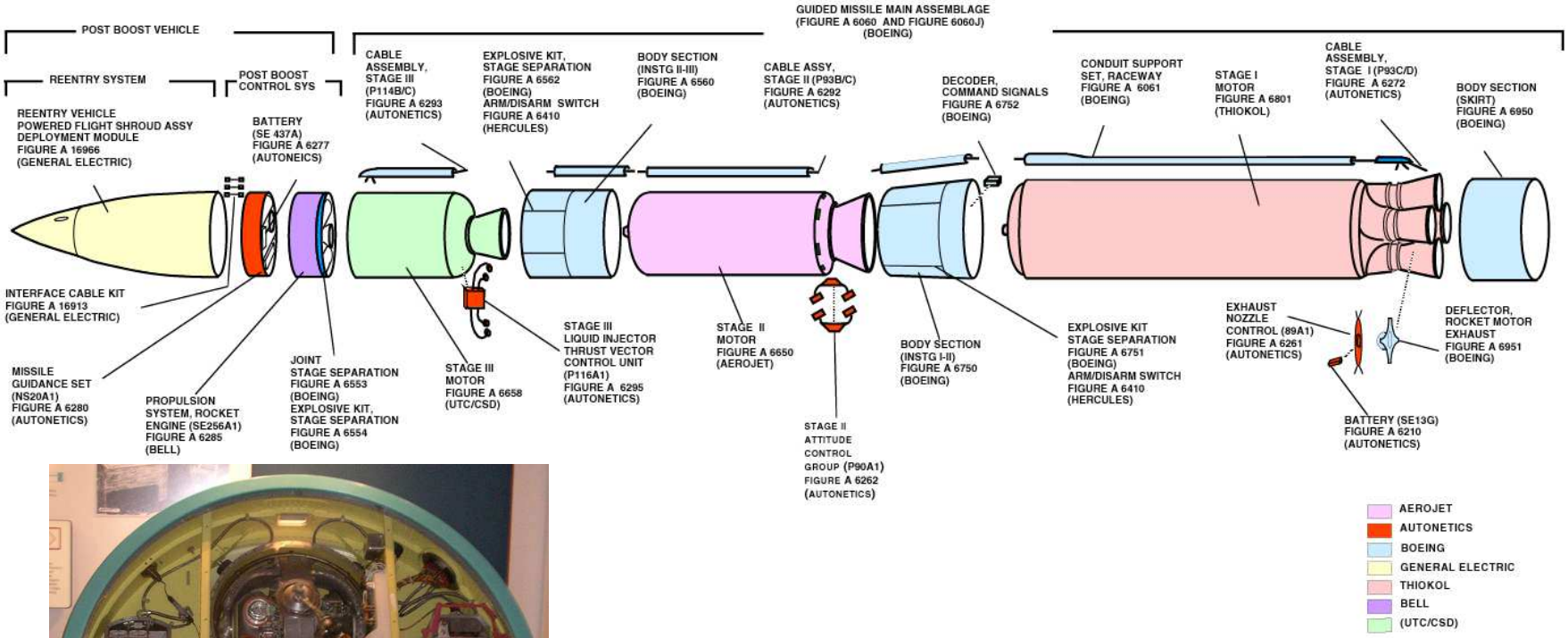
■新ミサイルの欠点
燃焼制御が難しく、振動がひどい。
電子装置が振動に耐えられず故障する。
↓
そのためFTAが考案された

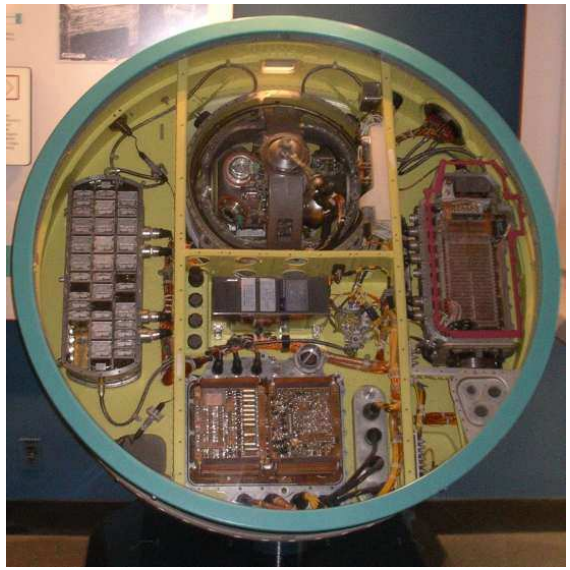
■新ミサイルの特長
ミニットマンミサイルは即応性を高めるため固体燃料を使っている。燃焼制御技術の開発が必要だったが、小型軽量かつ安価だった。
- ボーイング社が開発
- 3段式
- 1基6億円(700万ドル)
- 1965年までに800基配備
- メンテナンス不要
- 即時発射可能

■技術的課題の克服
4基ジンバルノズルによる3軸制御姿勢制御、振動にも故障しないロバートノイスが発明した7mm四方のシリコン基板に電子回路を組み込んだ集積回路による高性能コンピュータ制御により小型化に成功。

https://commons.wikimedia.org/wiki/File:Minuteman_III_diagram.png http://blogs.yahoo.co.jp/bdcxs228/33339822.html 参照

ミニットマンミサイルの構造





ミニットマンの電子誘導装置。これだけ液体燃料ロケットの大型トランジスタ回路誘導装置に比べて小型軽量、高性能化したのは集積回路のおかげ。

FMEAとは

- FMEA（Failure Mode and Effect Analysis）は、故障・不具合の防止を目的とした、潜在的な故障・不具合の体系的な分析方法である。
- 国際規格
  - IEC 60812:2006 システム信頼性の解析技法－故障モード影響解析(FMEA)の手順
  - JIS C 5750-4-4第 4-4 部：システム信頼性のための解析技法―故障の木解析（FTA）
- FMEA(意味：故障モードとその影響の解析）は、「設計の不完全や潜在的な欠点を見出すために構成要素の故障モードとその上位アイテムへの影響を解析する技法」である。
- フォルトツリー解析(Fault Tree Analysis)FTAがトップダウン手法であるのに対し、FMEAはボトムアップ手法という違いがある。

"Procedure for performing a failure mode effect and criticality analysis, November 9, 1949, United States Military Procedure, MIL-P-1629"

FMEA（Failure Mode and Effect Analysis)の歴史

米国フォード社の「ピント」が高速道路で突然エンストして停車していたところ、約50km/hの速度で走ってきた後続車に追突されて炎上し、運転者が死亡、同乗者が重度の火傷を負う事故が発生した。

1970年頃

フォードモーター社の Pinto のトラブルをきっかけにして、FMEAは自動車業界に安全性、生産、設計分野での改良の方法として取り入れられた。

ハードウェア起因の事故

1960年代

アポロ計画などのミッションのためのロケット工学のテストでNASAに採用され、1960年代に月に初めて人間が降り立ったことでその有効性が認められた。

1949年

“故障モードの影響及び分析の手順”というタイトルのマニュアルとして米陸軍マニュアルが発行された。

- システムや設備の故障の影響を決定するための信頼性及び評価のテクニックとして使われた。
- 故障はミッションの成功及び人員や設備の安全への影響の度合いによって分類された。

そもそも何のための活動？

- 答え

1. 事故（インシデント/アクシデント）を未然に防ぐため。

- 現状で、事故（インシデント/アクシデント）が多く発生していないなら、活動にモチベーションが上がらない。
  - 特に Awareness で動く日本のエンジニアにとってインセンティブにならない。

これまで自動車業界でソフトウェア起因の事故が多発してこなかったのは、「走る」「止まる」「曲がる」の基本機能、基本性能が侵食されるケースがなかったからではないか。それは EV時代でも同じか？

2. 発生してしまった事故（インシデント/アクシデント）の再発を防止するため。

- Intended Use（意図する使用）や使用環境が多様であればあるほど、ソフトウェア起因の事故は大小様々発生する。

医療機器では体温計からMRI（スモールMEからラージME）まで、診断装置から生命維持装置（クラスⅠからクラスⅣ）までさまざまな機器が存在し、ソフトウェア起因の多種多様な事故が発生している。

自動車業界の取り組みの過去/現在/未来

- **過去/現在**
  - インシデント/アクシデントが発生していない状況で、規格に適合していることが求められる。
    - 規格適合ビジネスが盛んになる。
    - エンジニアは活動の効果を実感することができず、モチベーションが下がり、活動が形骸化する。
- **未来**
  - EV時代には、限られたメーカーだけでなく、新規参入を含めさまざまな企業が自動車ドメインに参入する。
  - 自動車がネットワークに接続し、自動運転が普及することで、ソフトウェアが「走る」「止まる」「曲がる」の基本機能・基本性能を制御するようになる。
  - いままで想定もしなかったインシデント/アクシデントが発生し、危害に至る。
    - 安全なソフトウェアを開発し、維持する活動は、事故を未然に防ぐ、また、再発防止の活動となる。
    - エンドユーザと対峙しないソフトウェアエンジニアの関わりも増えることになり、セーフティを実現するためのマネジメントの重要性が増す。(Awarenessだけでは事故を防げなくなる)

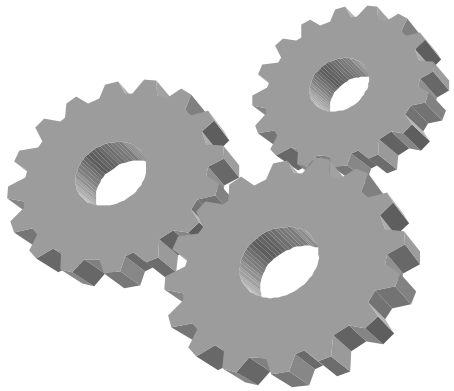
3. ケーススタディ

1. 緊急操舵回避システム
2. システムの複雑化とリスクコントロール
3. エレベータモデル
4. 電気メスモデル

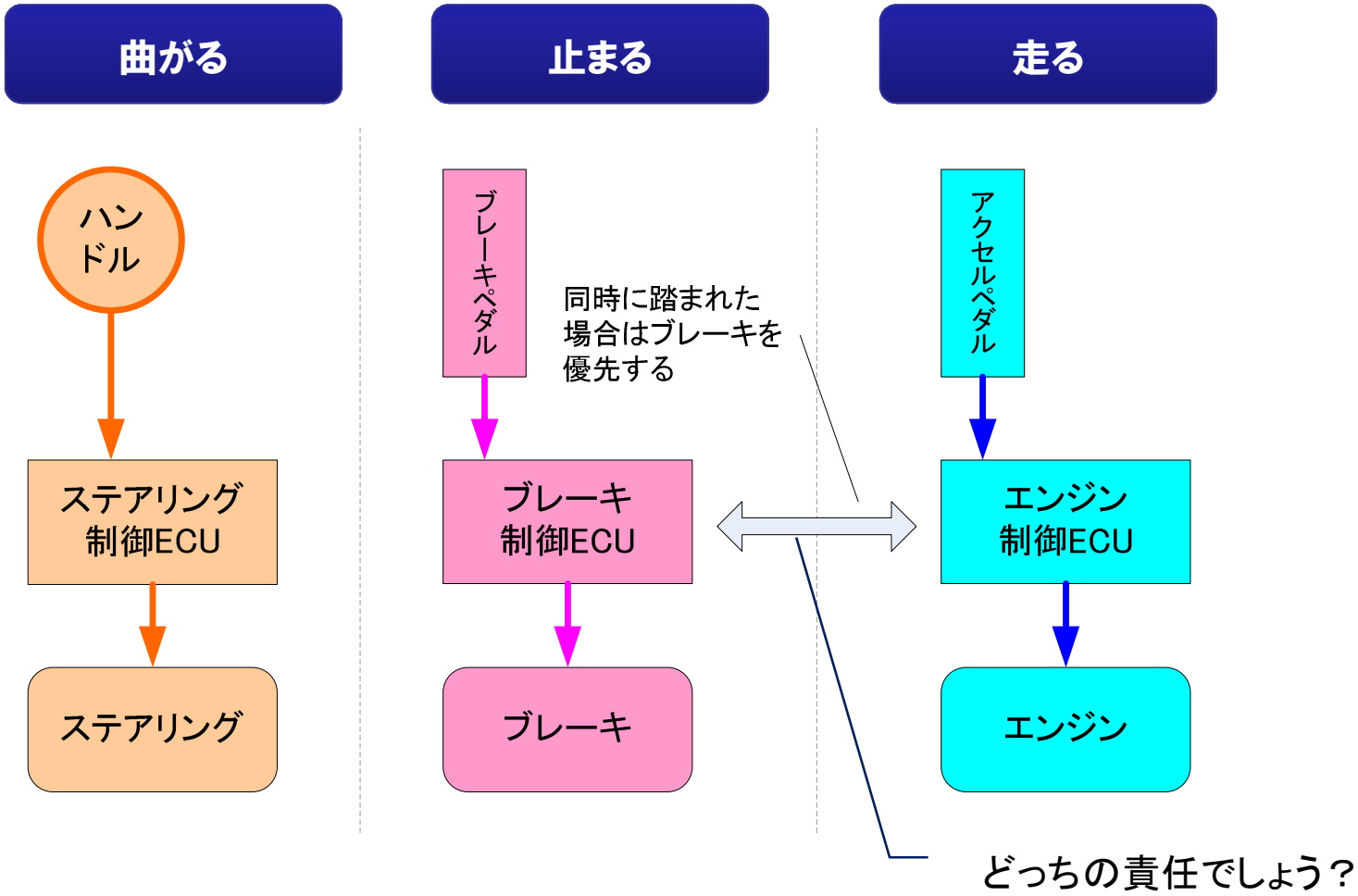
Topics 3-1.

Case Study 1

緊急操舵回避支援システム

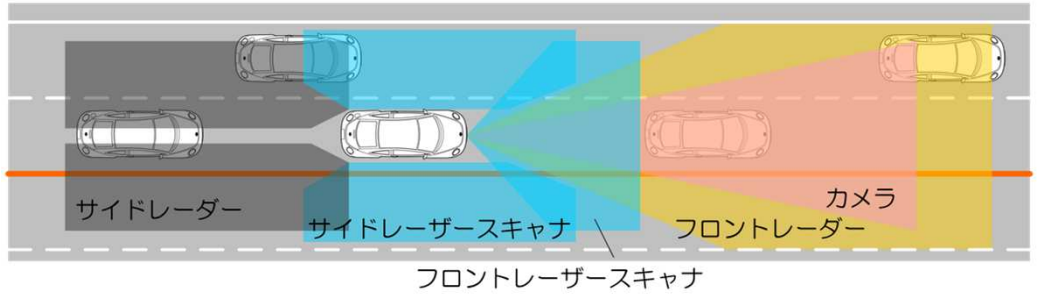


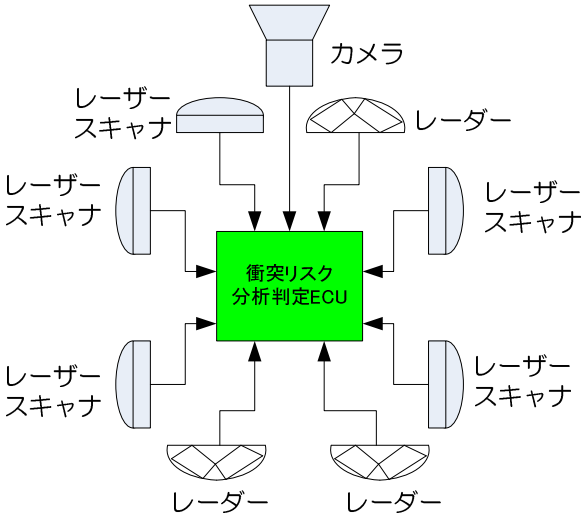
自動車のアーキテクチャの基本



転載禁止

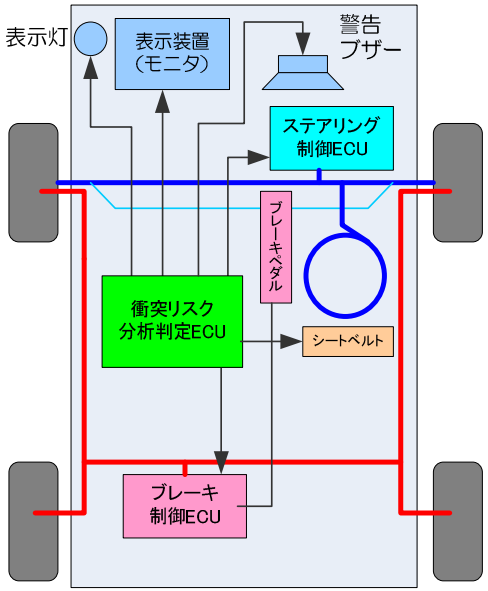
システムは複雑化し協調して Intended Use（意図した使用目的）を満たすようになる



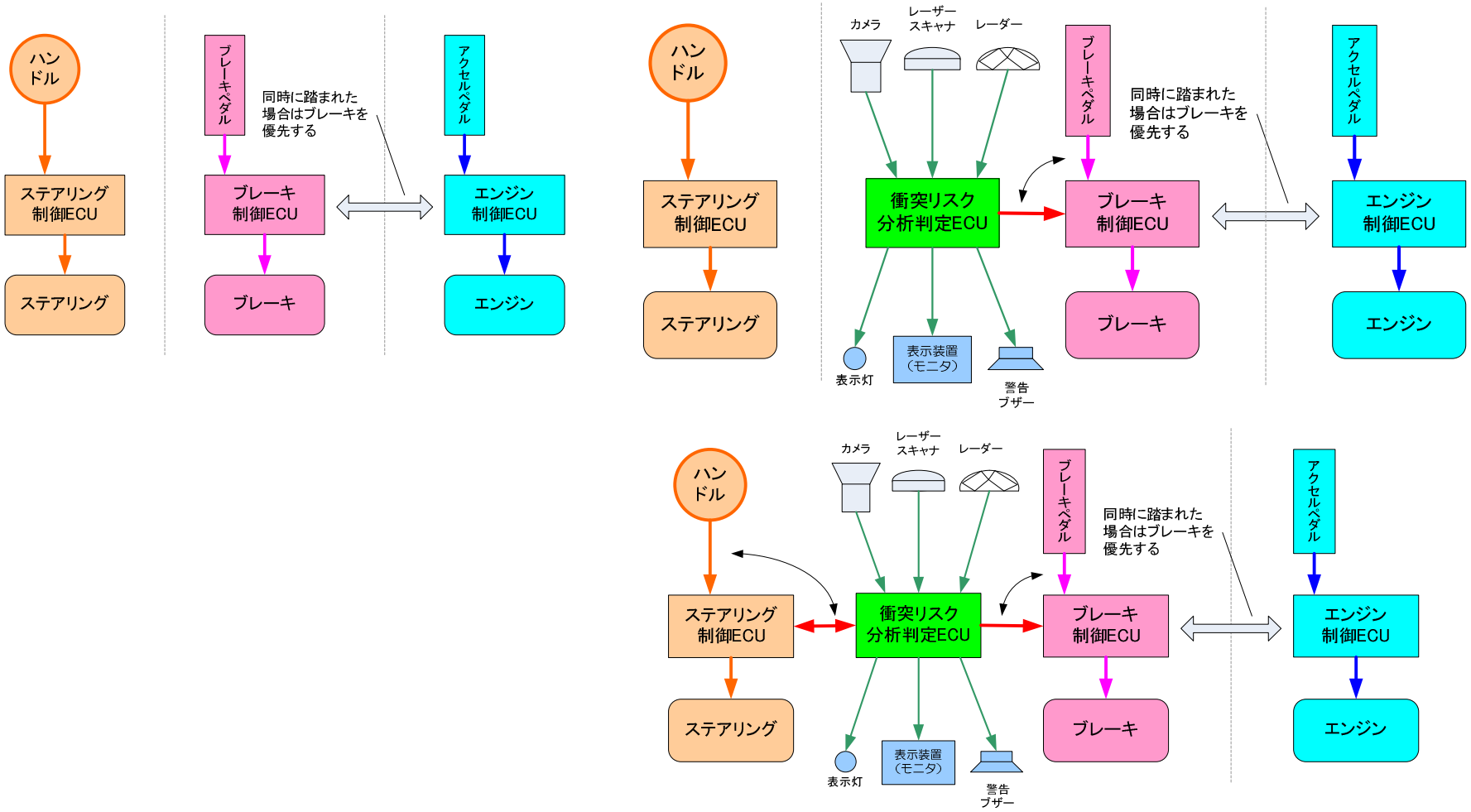


- 【前方衝突回避支援コンセプト】
  - 対象物を検知・識別する。
  - ドライバーに警告を発する。
  - 緩やかな減速を開始。
  - 衝突に備えてシートベルトをきつく締める。
  - コンマ数秒後、システムが衝突回避の限界に近づきドライバーが自力で衝突を回避できないと判断すると緊急ブレーキを作動しはじめ、衝突回避動作を支援する。

- 【緊急操舵支援システム】
  - 対象物を検知・識別する。
  - ドライバーに警告を発する。
  - 物理的に回避可能な限界に近づきドライバーの操作は不十分と判断する。
  - ECUは緊急ブレーキと緊急ステアリングのどちらで対応するかを一瞬で決定する。
  - ステアリングが最適な措置の場合ハンドルを切ることが可能な方向をドライバーにモニタで知らせる。
  - その直後に必要な操作（緊急ブレーキの作動か、緊急操舵か、それらの両方）を正確に実行する。

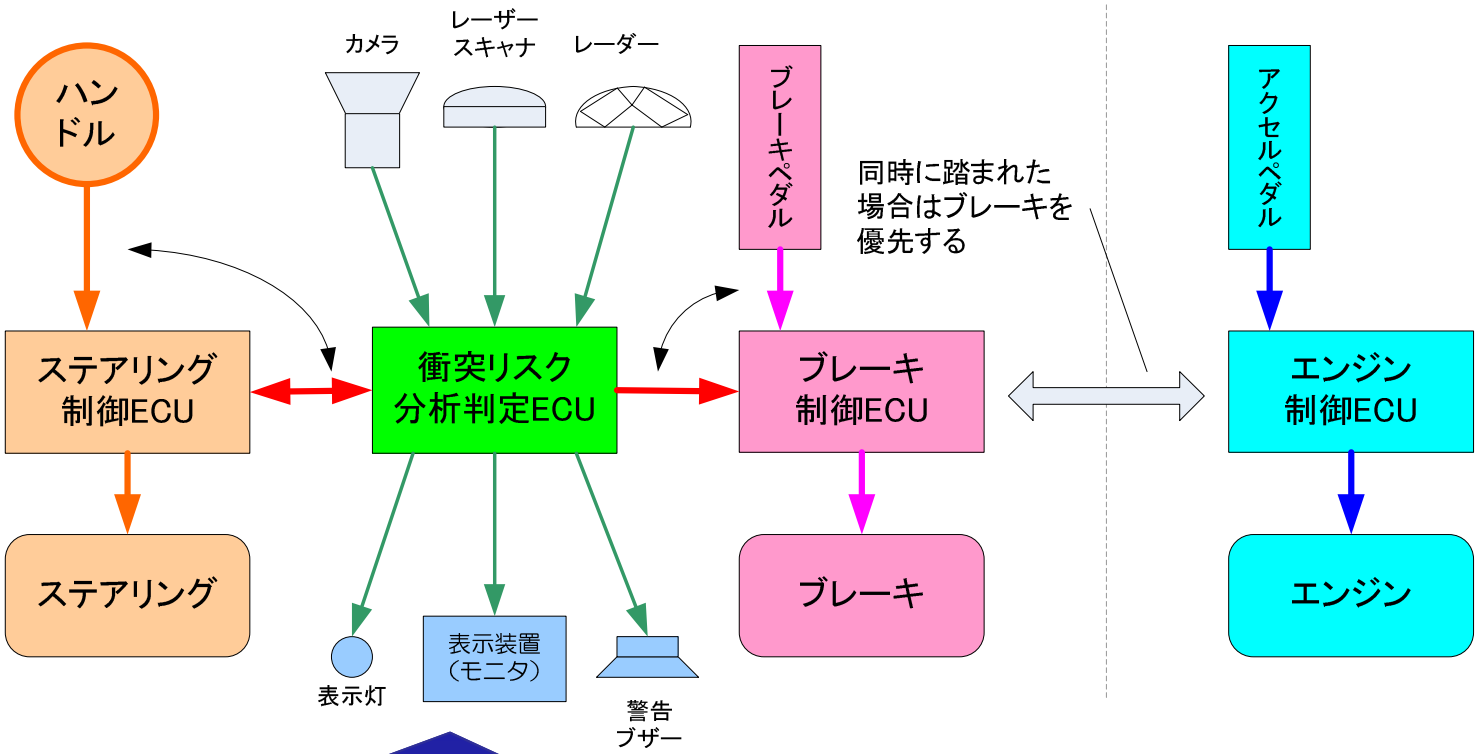


自動車のアーキテクチャの変化



転載禁止

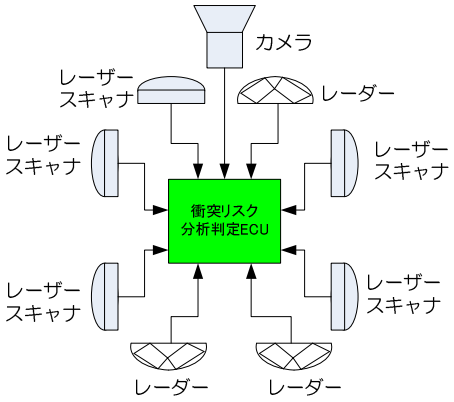
独立していた機能が結合していく



衝突リスク分析判定ECUを介して「曲がる」と「止まる」が結合してしまった。

転載禁止

衝突回避システムの安全対策案



- 対象物を認識していないときは「曲がる」と「止まる」の関連を絶つ。
  - 対象物がないのにあると判定することを確実に防止する。
    - 「対象物がないのにあると判定する」をトップ事象にしてフォルトツリー分析を実施する。
- 衝突判定に使う要素の故障を検出し、故障がある場合は運転手に知らせる。
  - 故障検出センサの故障に注意
  - 故障がある場合は確実に伝える。（ユーザビリティ）
- 衝突回避のシナリオテスト
  - 衝突回避のシナリオをどれだけ想定できるかが鍵となる。
  - ソフトウェアのアーキテクチャを複雑化は絶対に防ぐ。

日本では交通事故による死亡者数は減少しており、2009年からは年間5000人を切っている。

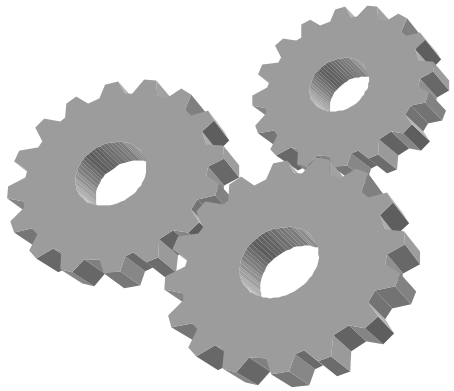
しかし、世界規模で見ると、運転手の不注意による自動車事故の死亡者数は年間150万人もいる。

ソフトウェアによるリスクコントロールの落とし穴

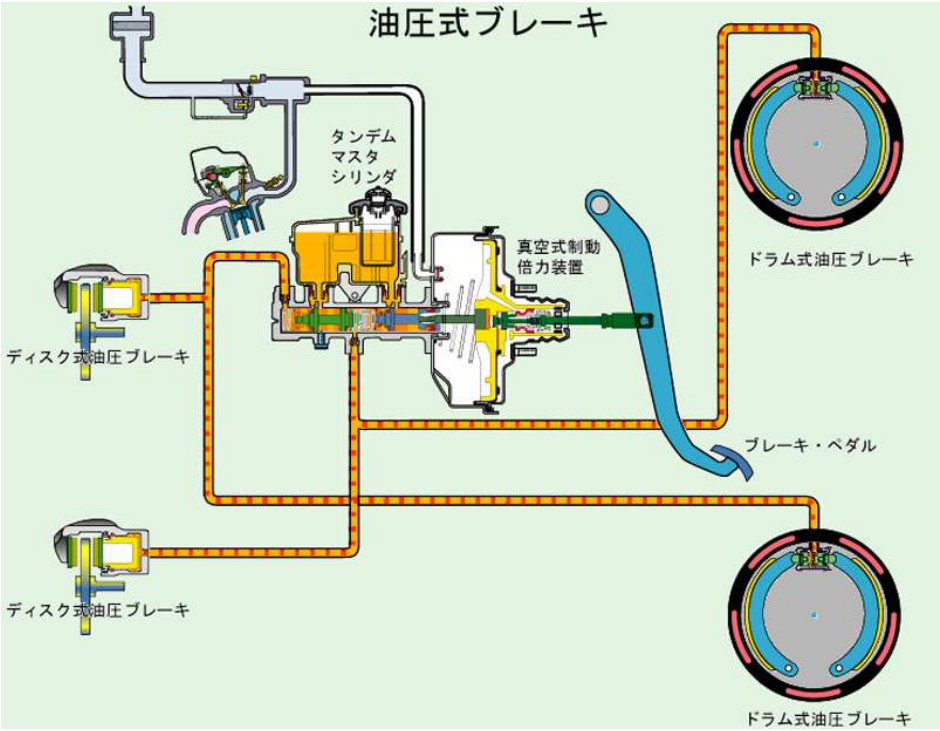
- フィールドで発生した不具合の対策をソフトウェアの追加・変更で実施すると、セーフティアーキテクチャが崩れる危険性があるので注意が必要
- ハザード分析によりハザード項目を追加し続けると、何の対策をしているのか開発者自身が分からなくなってくる。
  - リスクコントロール手段の要求と検証の実施はトレースを取ることになる（トレーサビリティ・アナリシス）が、複合的な分かりにくい不具合を埋め込んでしまう危険がある。
- そういう時は FTA により、危険事象がどのようにブロックされるのかを可視化する。
  - セーフティ・アーキテクチャが有効であることを確認する。
  - 重要度の高いソフトウェアアイテムを重点的に検証する。

Topics 3-2.

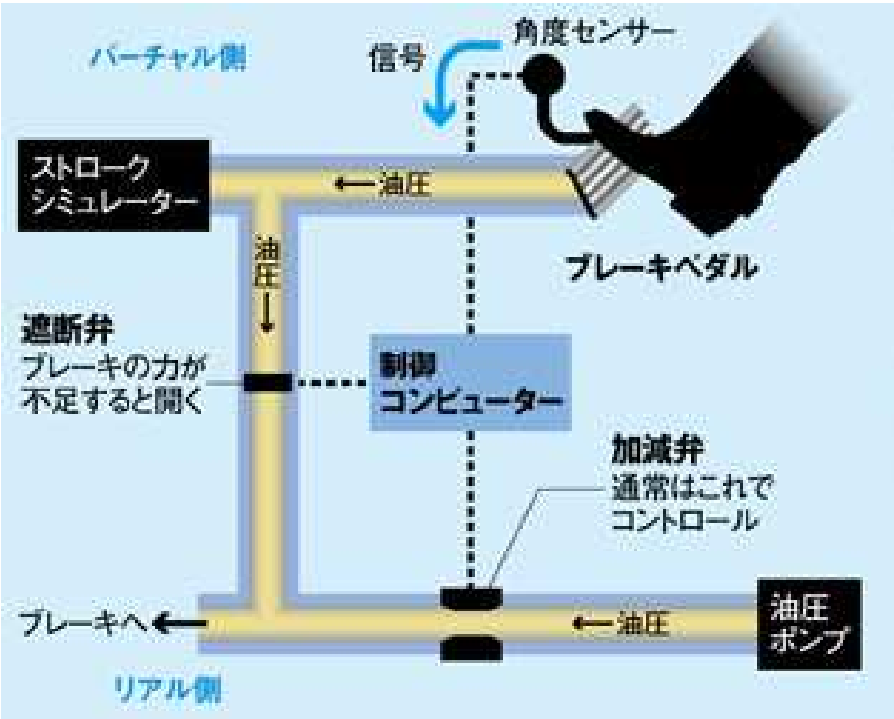
システムの複雑化とリスクコントロール



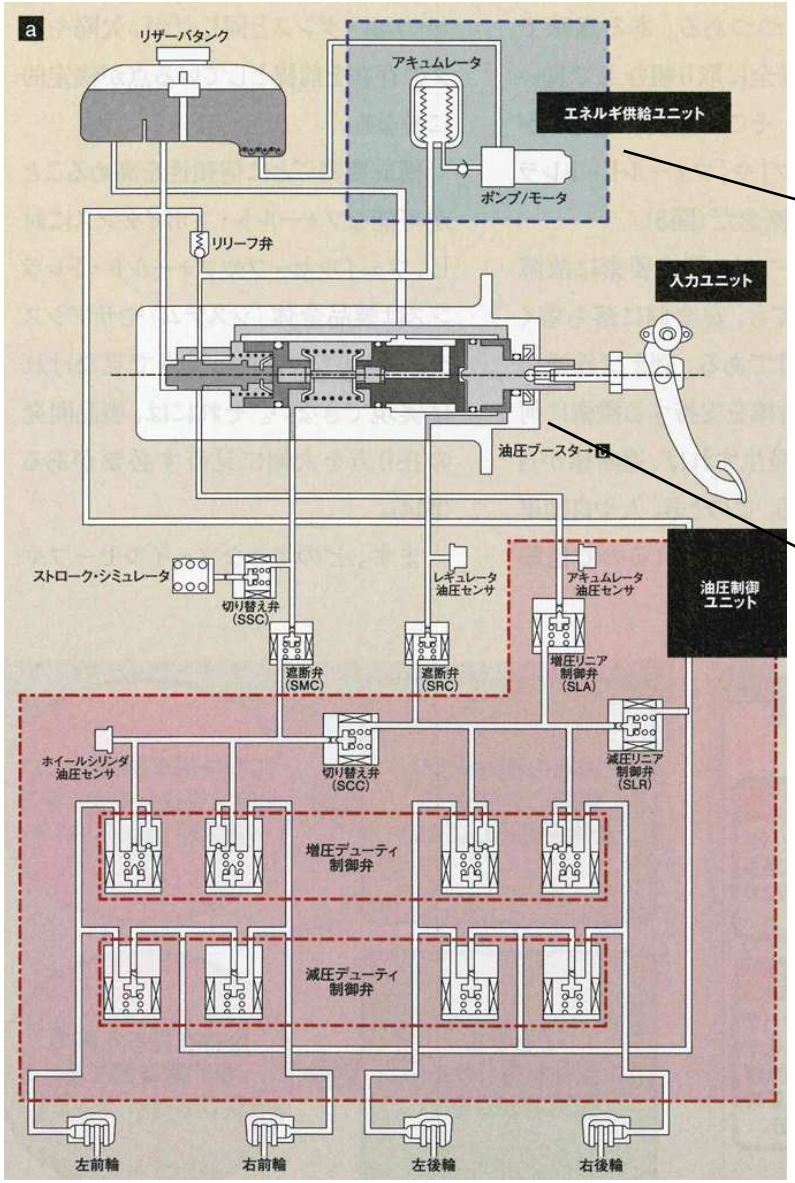
油圧式ブレーキと電子制御のブレーキシステム



単純な構造の油圧式ブレーキ



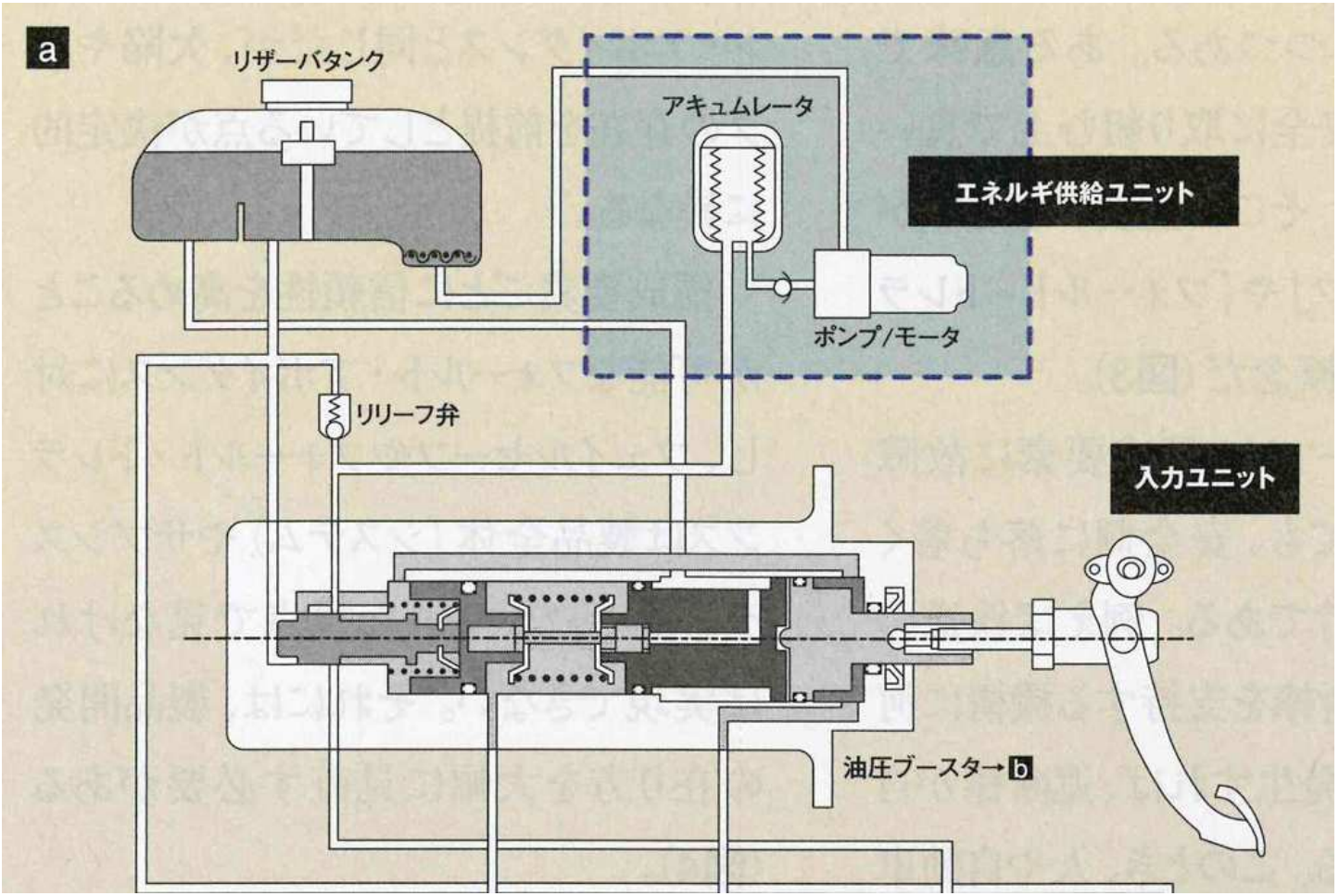
電子制御の油圧ブレーキ

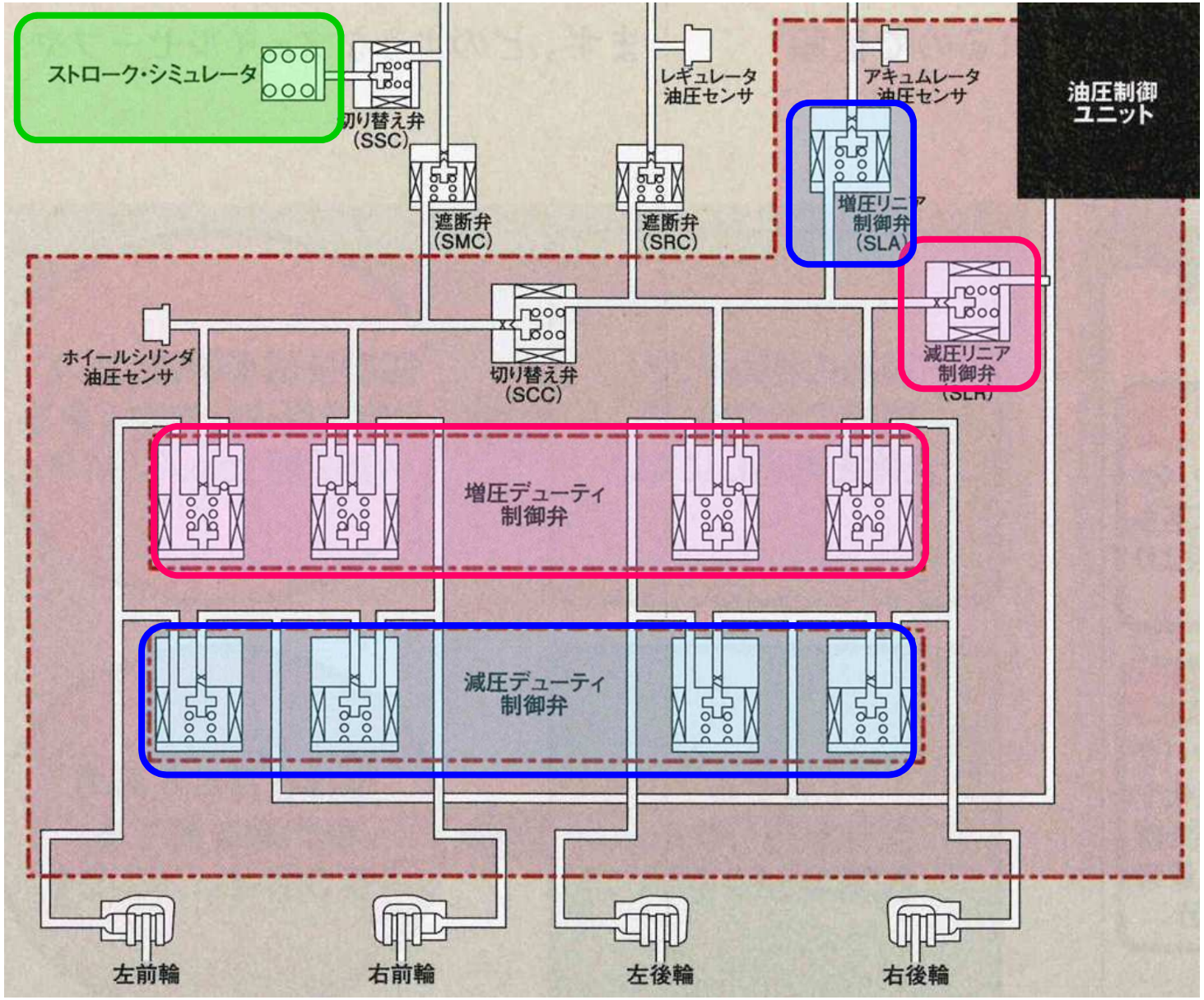

a
リザーバタンク
アキュムレータ
エネルギ供給ユニット
ポンプ/モータ
リリーフ弁
入力ユニット
油圧ブースタ→b
ストローク・シミュレータ
切り替え弁
(SSC)
レギュレータ
油圧センサ
アキュムレータ
油圧センサ
油圧制御
ユニット
遮断弁
(SMC)
遮断弁
(SRC)
増圧リニア
制御弁
(SLA)
ホイールシリンダ
油圧センサ
切り替え弁
(SCC)
減圧リニア
制御弁
(SLR)
増圧デューティ
制御弁
減圧デューティ
制御弁
左前輪
右前輪
左後輪
右後輪

(b)
油圧ポンプ
モータ
アキュムレータ

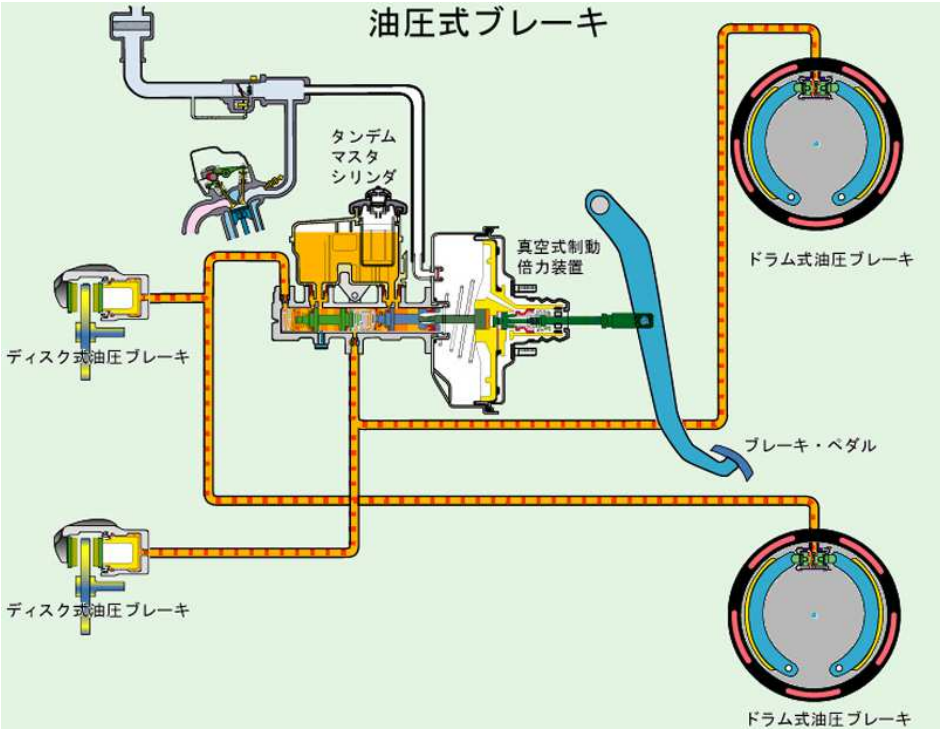
(a)
ECU
油圧ブースタ

ブレーキシステムの上部

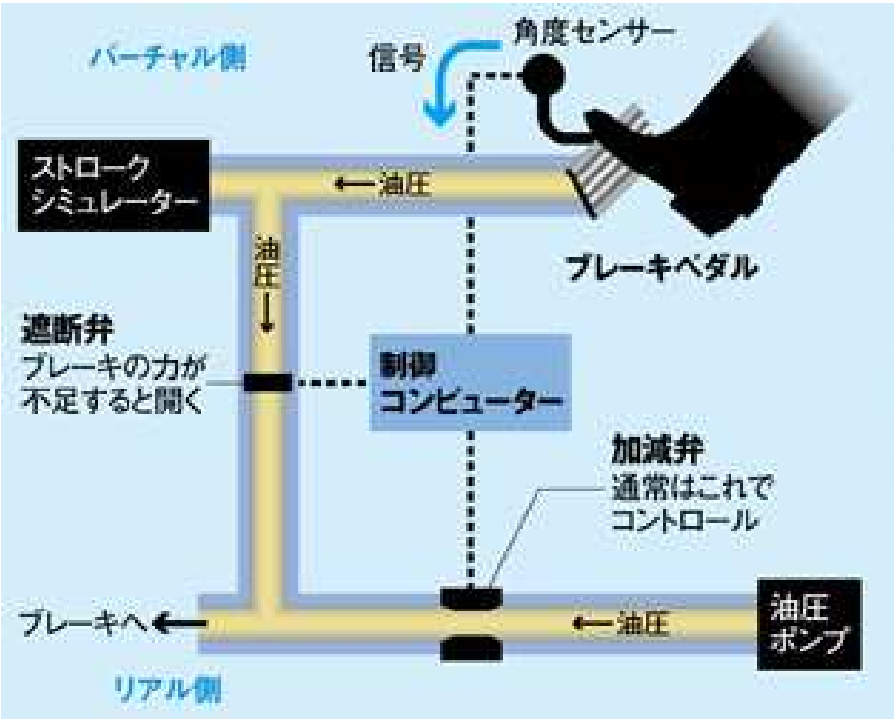



ストローク・シミュレータ
切り替え弁
(SSC)
レギュレータ
油圧センサ
アキュムレータ
油圧センサ
油圧制御
ユニット
遮断弁
(SMC)
遮断弁
(SRC)
増圧リニア
制御弁
(SLA)
ホイールシリンダ
油圧センサ
切り替え弁
(SCC)
減圧リニア
制御弁
増圧デューティ
制御弁
減圧デューティ
制御弁
左前輪
右前輪
左後輪
右後輪

油圧式ブレーキと電子制御のブレーキシステム

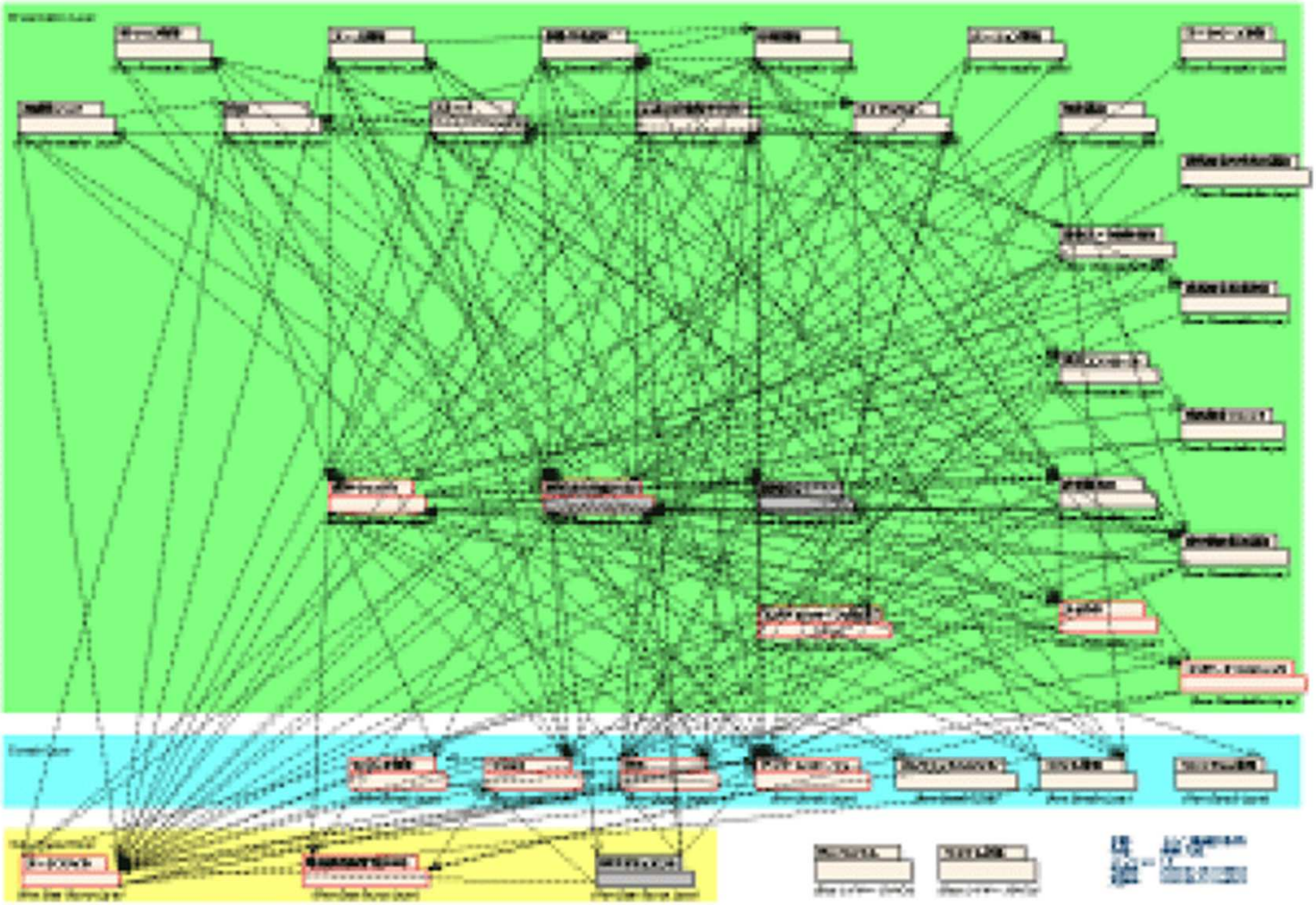


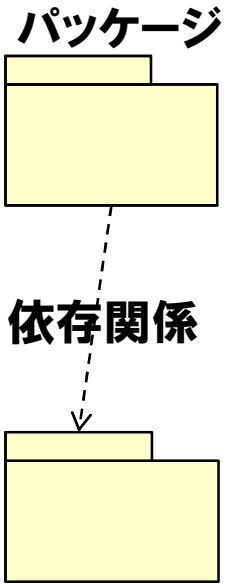
単純な構造の油圧式ブレーキ



電子制御の油圧ブレーキ

数十万行クラスのソフトウェアの例



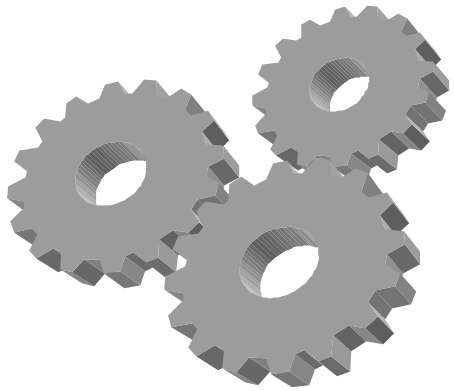


転載禁止

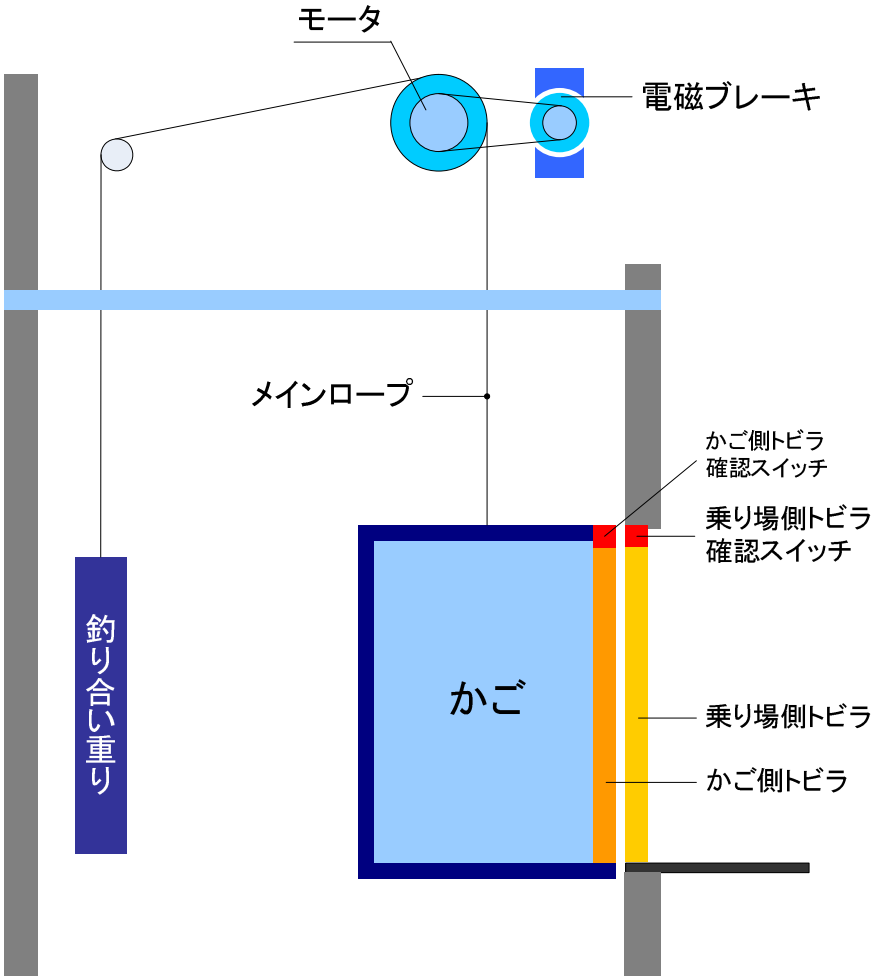
Topics 3-3.

Case Study 2

エレベータモデル



エレベータの安全装置の例



- **トビラが開いていれば、モータは動かないはず！**

エレベータのかご及び乗り場側のトビラには、閉まっていることを検知するスイッチ(センサ)取り付けてある。電磁ブレーキの解除や駆動モータを制御するプログラムは、これらのスイッチがすべて「入」の状態にならない限り、エレベータを動かさない。

ミスを誘発しやすい安全処理

エレベータ制御メイン

```
void main()
{
    :
    :
    /* かごを上げる */
    driveMotor ( UP ):
    :
    :
}
```

モータ制御関数

```
void driveMotor （ 命令 ）
{
    if ( 命令 == STOP) {
        モータを止める;
        電磁ブレーキON ;
    }

    else if ( 命令 == UP ) {
        status = getCageStatus();
        if ( status != open ) {
            電磁ブレーキを解除 ;
            モータを正転させる;
        }
    }

    else if ( 命令 == DOWN ) {
        status = getCageStatus();
        if ( status != open ) {
            電磁ブレーキを解除 ;
            モータを逆転させる;
        }
    }
}
```

ドア開ならモータを動かさない完全処理

安全処理が分散しているため、安全性が網羅されているかどうかわかりにくし、漏れが生じやすい

転載禁止

driveMotor (Check→OK→Drive)

エレベータ制御メイン

```
void main()
{
    :
    /* かごを上げる */
    driveMotor ( UP ):
    :
}
```

モータ制御関数

```
int driveMotor( 命令 )
{
    status = checkCageStatus();
    if ( status == open ) {
        モータを止める;
        電磁ブレーキON;
        return ERROR;
    }
    else {
        driveMotorAtferCheck( 命令);
    }
}
```

ドア開ならモータを動かさない安全処理

```
void checkAbnomalRun()
{
    戸開走行を監視し止める;
}
```

エレベータメイン制御から直接モータを駆動する関数を隠蔽する

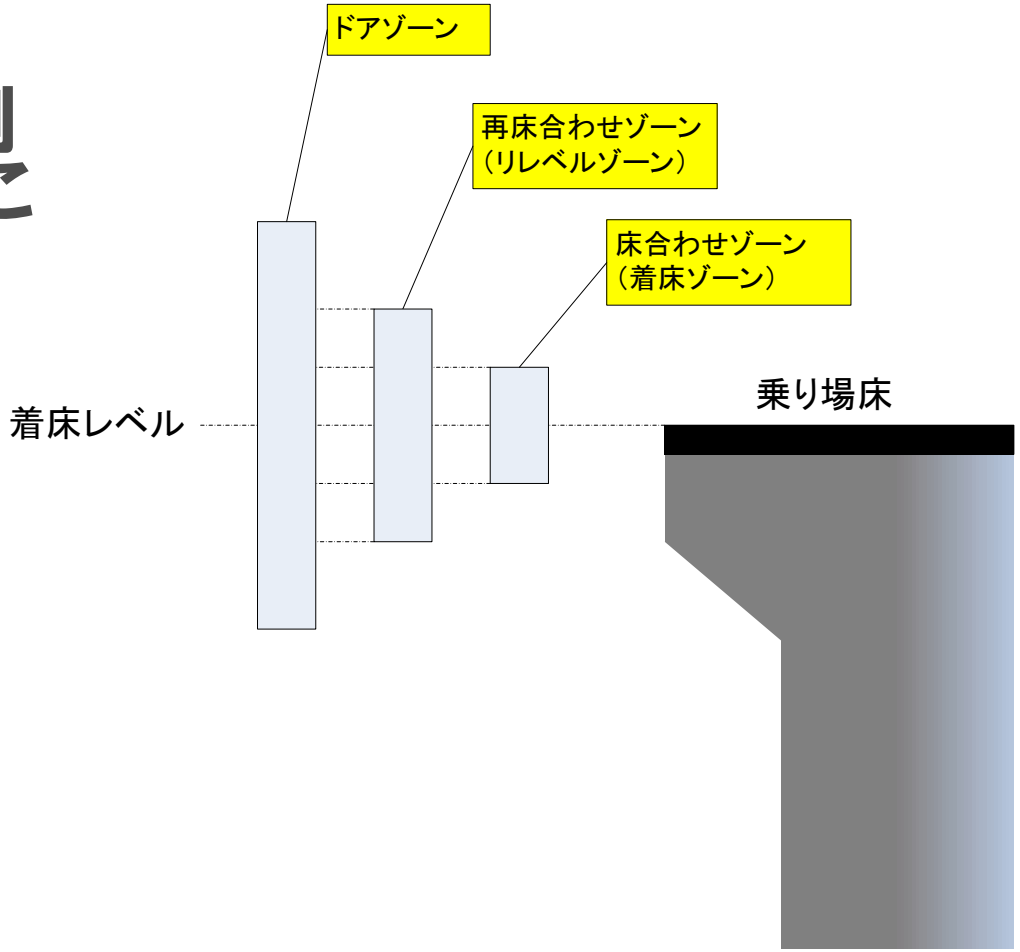
チェック後のモータ制御関数

```
void driveMotorAfterCheck ( 命令 )
{
    if ( 命令 == STOP) {
        モータを止める;
    }
    else if ( 命令 == UP ) {
        モータを正転させる;
    }
    else if ( 命令 == DOWN ) {
        モータを逆転させる;
    }
}
```

転載禁止

さらに例外処理がミスを誘発する

- トビラが開いていても例外的にモータを動かすことがある！



カゴと乗り場側の床に段差が生じた場合に、その差を自動的に解消する制御を行う場合に限って、トビラが開いたままでもゆっくりとカゴを動かす。ただし、そのような制御が可能か範囲は再床合わせゾーンに限られる。

driveMotor (Check→OK→Drive)

エレベータ制御メイン

```
void main()
{
    :
    /* かごを上げる */
    driveMotor ( UP ):
    :
}
```

モータ制御関数

```
int driveMotor( 命令 )
{
    status = checkCageStatus();
    if ( status == open ) {
        モータを止める;
        電磁ブレーキON;
        return ERROR;
    }
    else {
        driveMotorAtferCheck( 命令);
    }
}
```

ドア開ならモータを動かさない安全処理

調整のためのモータ制御関数

```
void driveMotorSlowly （ 命令 ）
{
    if ( 命令 == SLOWUP) {
        モータをゆっくり正転させる;
        :
```

再床合わせゾーンならゆっくりモータを動かす例外処理

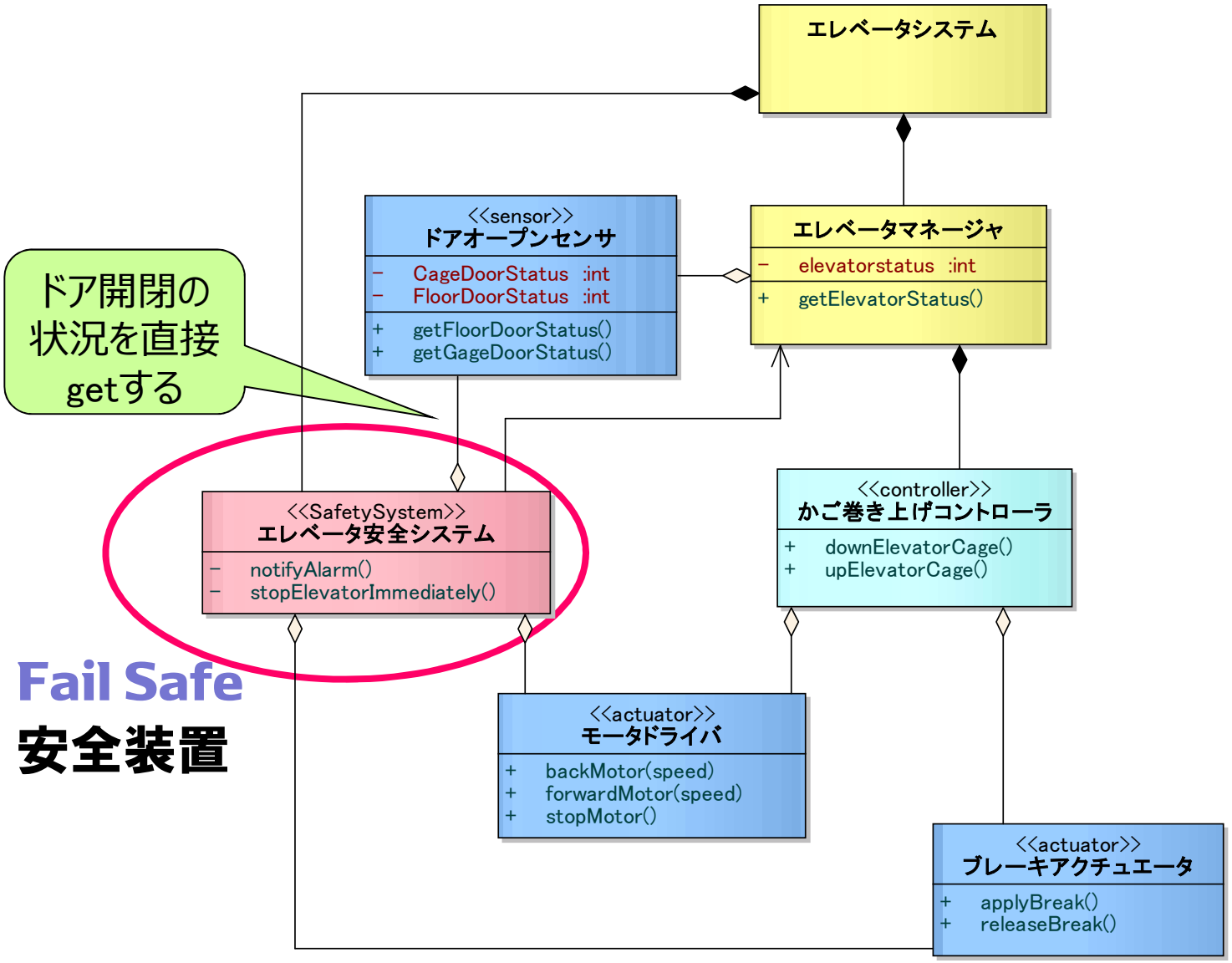
```
void checkAbnomalRun()
{
    例外を除き戸開走行を監視し止める;
}
```

エレベータメイン制御から直接モータを駆動する関数を隠蔽する

チェック後のモータ制御関数

```
void driveMotorAfterCheck （ 命令 ）
{
    if ( 命令 == STOP) {
        モータを止める;
    }
    else if ( 命令 == UP ) {
        モータを正転させる;
    }
    else if ( 命令 == DOWN ) {
        モータを逆転させる;
    }
}
```

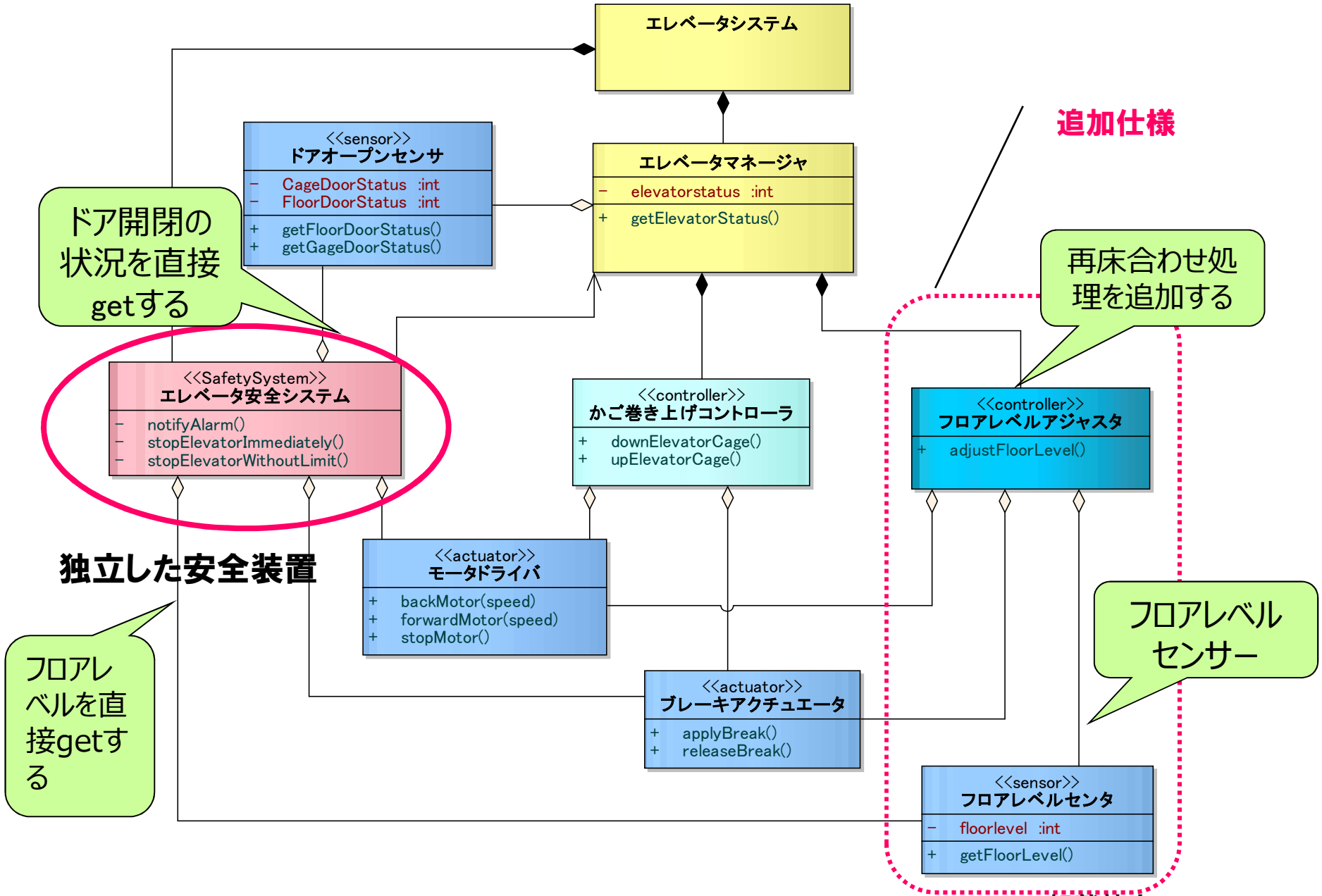
転載禁止



Fail Safe

安全装置

転載禁止


エレベータシステム
追加仕様
<<sensor>>
ドアオープンセンサ
− CageDoorStatus :int
− FloorDoorStatus :int
+ getFloorDoorStatus()
+ getGageDoorStatus()
エレベータマネージャ
− elevatorstatus :int
+ getElevatorStatus()
ドア開閉の状況を直接getする
再床合わせ処理を追加する
<<SafetySystem>>
エレベータ安全システム
− notifyAlarm()
− stopElevatorImmediately()
− stopElevatorWithoutLimit()
<<controller>>
かご巻き上げコントローラ
+ downElevatorCage()
+ upElevatorCage()
<<controller>>
フロアレベルアジャスタ
+ adjustFloorLevel()
独立した安全装置
<<actuator>>
モータドライバ
+ backMotor(speed)
+ forwardMotor(speed)
+ stopMotor()
フロアレベルセンサー
フロアレベルを直接getする
<<actuator>>
ブレーキアクチュエータ
+ applyBreak()
+ releaseBreak()
<<sensor>>
フロアレベルセンタ
− floorlevel :int
+ getFloorLevel()

転載禁止

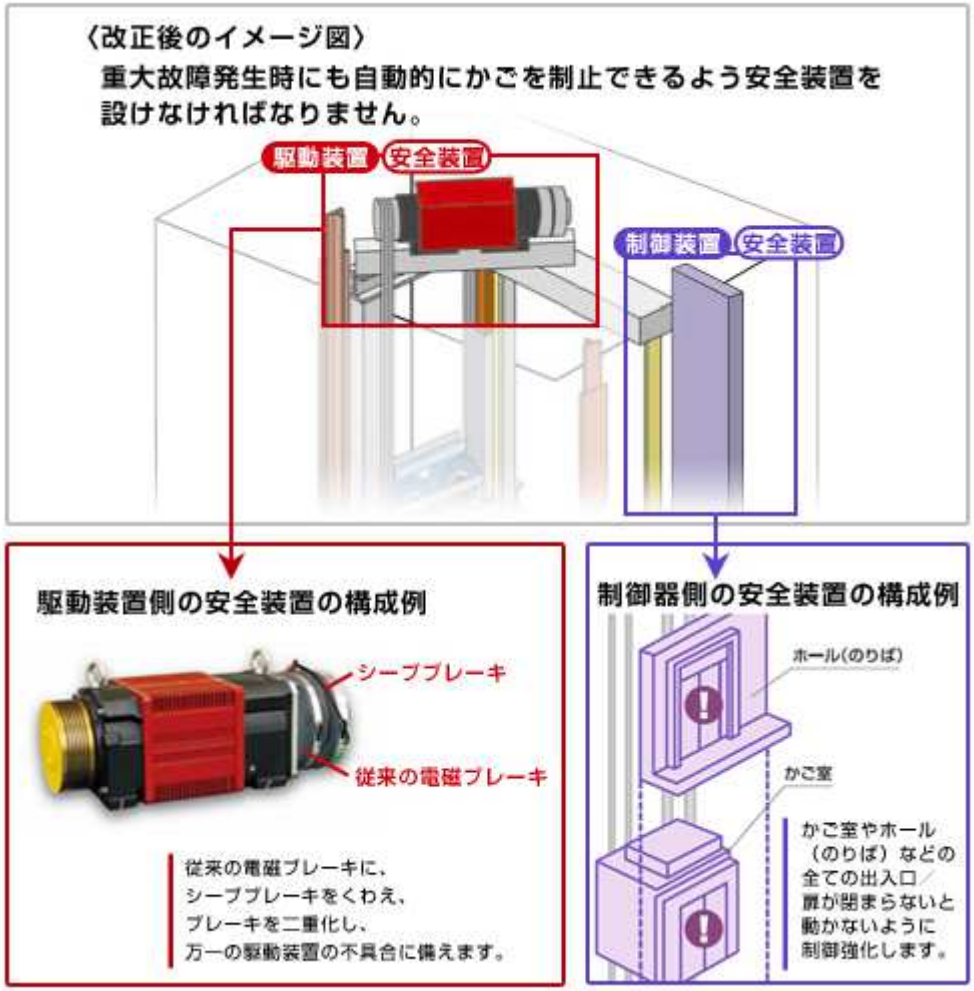
メカとエレキによるインターロック機構

- 扉が完全に閉まって通電がないとエレベータが動かないインターロック機構

実現の方法	
フォールト・アボイダンス Fault Avoidance **信頼性を高める**	
フェール・セーフ Fail Safe **安全側に倒す**	○
フォールト・トレランス Fault Tolerance **冗長設計を施す**	
エラー・プルーフ **（フール・プルーフ）** Error Proof / Fool Proof **間違いにくい操作性**	

独立した安全用ブレーキの設置

- エレベーター新安全基準とは
- 平成18年6月に港区で発生した死亡事故および、平成17年7月の「千葉県北西部地震」で、多くの「閉じ込め」が発生した事象を受けて国土交通省は建築基準法の一部改正を行いました。



ハードとソフトの安全設計の違い

- **ハードウェアによるリスクコントロール手段（安全装置）**
  - 統計確率的な故障のしかたをする。(Random Failure)
  - 変更が容易ではない。
  - システムにおける安全機能の独立性が明白。
  - コストダウンで取り除かれてしまう可能性がある。

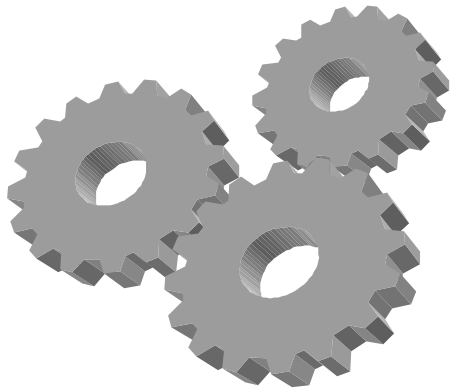
- **ソフトウェアによるリスクコントロール手段**
  - 決定論的な問題発生のしかたをする。(Systematic Failure)
  - 変更が容易なため、デグレードの危険がある。
  - システムにおける安全機能の独立性を示すことが難しい。

ハードとソフトのリスクコントロールは相互に補完すべきだが、ハードではブロックできないリスクもある。

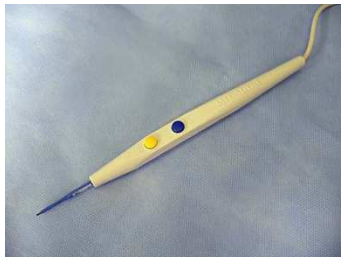
Topics 3-4.

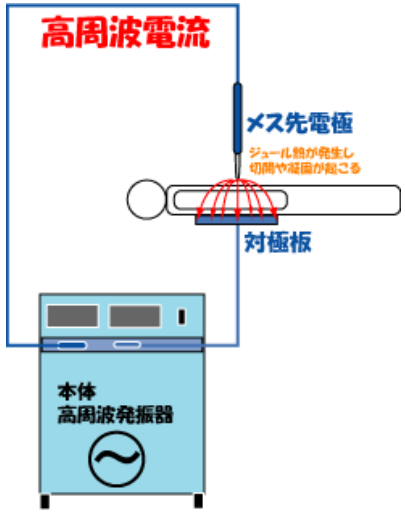
Case Study 3

電気メスモデル



電気メスはソフトウェアで切開と凝固を切り替える

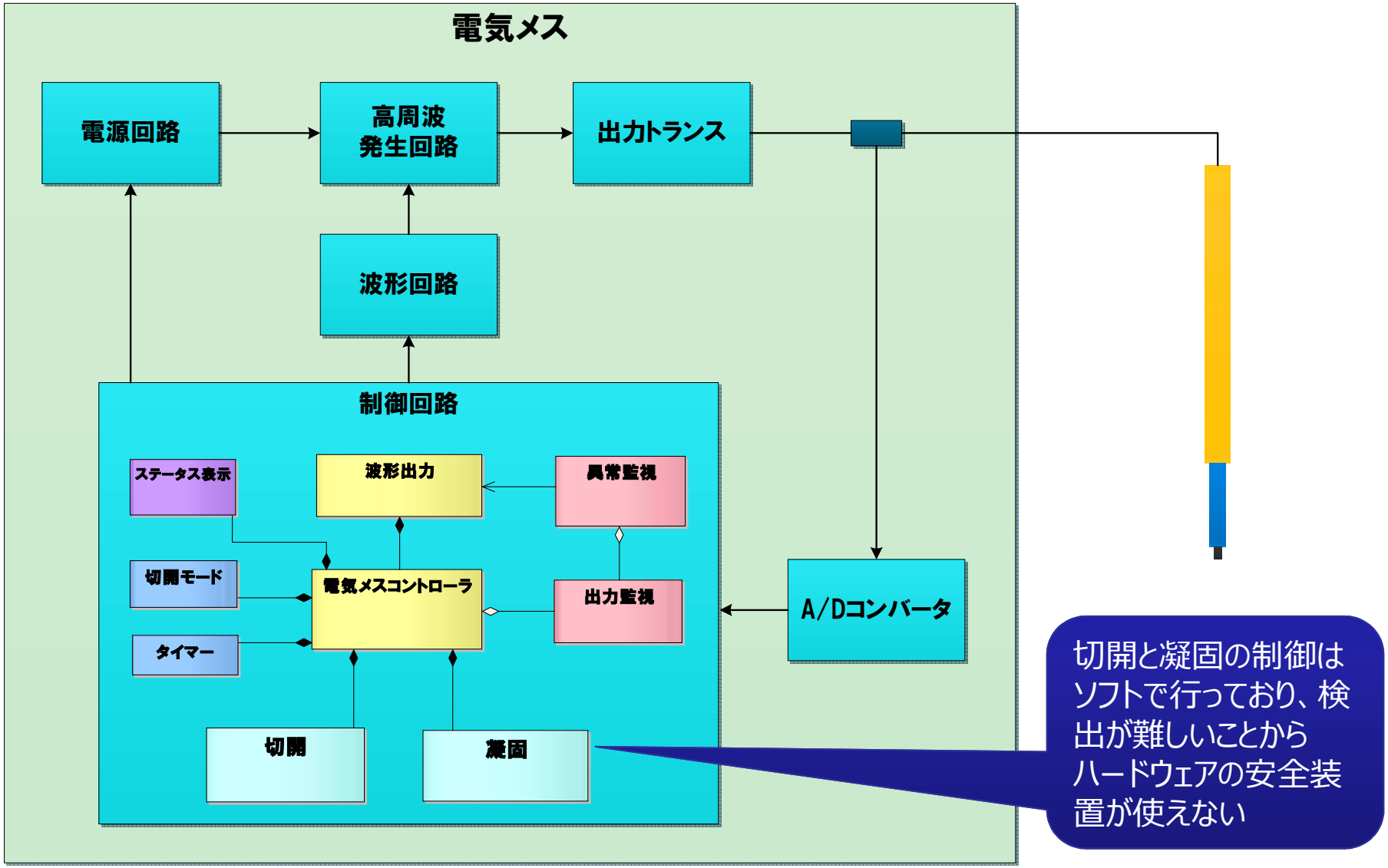




切開と凝固	原理	モード	特長
切開 (cut)	高周波を連続して発生させる。 高周波エネルギーによって人体組織は数百度に熱せられるため、体内の水分が瞬時に気化して蒸発する。 その蒸発によって組織が一緒に吹き飛び切れたように見える。	**Pure**	出血がある。
		Blend	高周波を断続的に出しながら凝固させながら組織を切るように調整したモード
凝固 (coagulation)	やや弱めの高周波を断続的に発生させるモード。 温度が数百度まで上がらないように100℃程度に調整し、組織を凝固（止血）させる。		

転載禁止

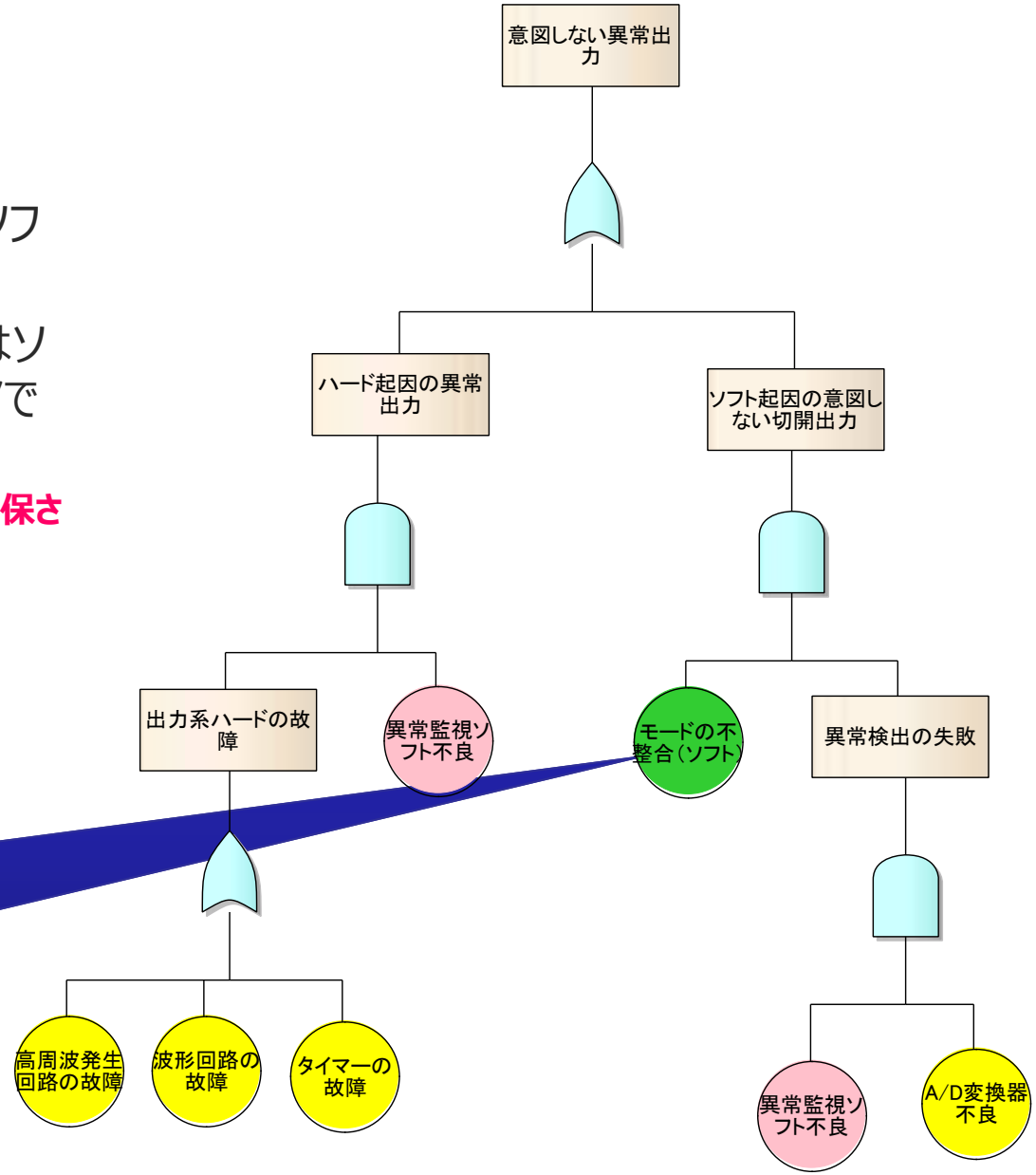
ハードウェアの安全装置が使えない



転載禁止

FTAの分析

- 異常出力の監視を異常監視ソフトが担っている。
- 切開と凝固のモードの不整合はソフトウェアの問題で、ハードウェアでバックアップすることができない。
  - **ソフトウェアの信頼性で安全が担保される。**



切開と凝固のモードの不整合は致命的なリスクであり、ソフトウェアの検証と妥当性確認で安全を担保する。

転載禁止

FMEAの分析(ハザード分析)

ハザード	潜在的原因	半定量的な確率レベル	定性的な重大さレベル	リスクコントロール手段		対策後の残留リスク評価	関連するソフトウェアアイテム	ソフトウェア安全クラス
				ハードウェア	ソフトウェア			
意図しない切開モードの出力	モード切り替えソフトに誤りがある	NA (Systematic Failure)	重大な (Critical)	出力波形をA/D変換してCPUへフィードバックする	a) 出力波形の監視し、異常検出時に出力を止める。 b) 関連するソフトウェアアイテムの独立性の検証 c) アイテムごとの信頼性検証 d) 負荷試験を含むテスト計画の立案と実施	受容できる (Acceptable)	電気メスコントローラ 切開 凝固 切開モード ステータス表示 出力監視 異常管理 波形出力	Class C Class C Class B Class C Class B Class C Class C Class C

a) Fail Safe
b) Fault Avoidance
c) Fault Avoidance
d) Fault Avoidance

Fault Avoidance だけでは安心できない

転載禁止

ソフトウェアによるリスクコントロールの落とし穴

- フィールドで発生した不具合の対策をソフトウェアの追加・変更で実施すると、セーフティアーキテクチャが崩れる危険性があるので注意が必要
- ハザード分析によりハザード項目を追加し続けると、何の対策をしているのか開発者自身が分からなくなってくる。
  - リスクコントロール手段の要求と検証の実施はトレースを取ることになる（トレーサビリティ・アナリシス）が、複合的な分かりにくい不具合を埋め込んでしまう危険がある。
- そういう時は FTA により、危険事象がどのようにブロックされるのかを可視化する。
  - セーフティ・アーキテクチャが有効であることを確認する。
  - 重要度の高いソフトウェアアイテムを重点的に検証する。

4. まとめ（安全を実現するには）

a. 安全の実現に有効なサイクル
b. 注意すべき点
c. 業務ドメイン別 リスクマネジメントの未来
d. 医療機器ドメインの昔と今の違い
e. ネットワークが絡むリスク
f. 自動運転ソフトウェアの意図しない動きは誰のせい？
g. まとめ

安全の実現に有効なサイクル

1. 要求分析（QFD） Intended Use の特

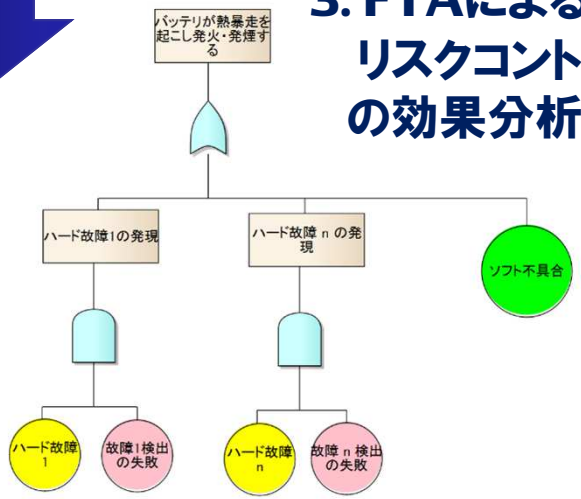
市場要求			重要度 ◎(高) ○(中) △(低)			ソフトウェア関連モジュール	
1次	2次	3次	お客様要求度	ビジネス目標	実現難易度	名称	固定/変動
利用者の満足度が高い	画面レイアウトが見やすい	画面の明るさを調整できる	△	△	△	User Interface	変動
		各情報の表示レイアウトが見やすい	○	○	△	User Interface	変動
	有益な表示内容	故障情報が表示される	◎	◎	○	User Interface	変動
						DataManager1・HardwareDriver	固定
						Sensor	変動
		接続機器の状態が表示される	◎	◎	○	DataManager1・HardwareDriver	固定
						User Interface	変動
保守が容易	実機、接続機器の診断機能を有する	故障記録機能を有する	◎	○	○	Sensor	変動
						DataManager1・HardwareDriver	固定
		接続機器の自動試験の機能を有する	○	◎	◎	User Interface	変動
						DataManager1・HardwareDriver	固定
						Diagnostician	変動

2. ハザード分析(FMEA)リスクの洗い出し

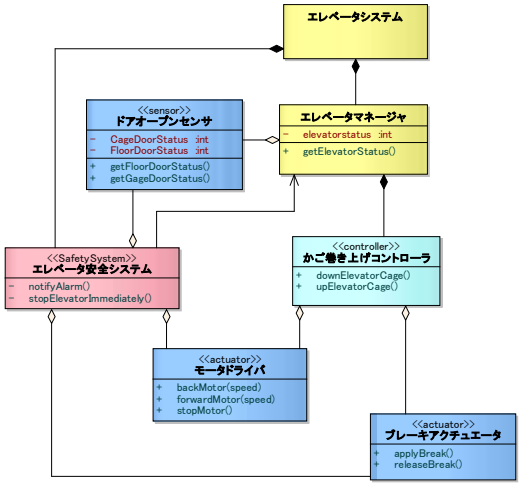
ハザード	潜在的原因	半定量的な確率レベル	定性的な重大さレベル	リスクコントロール手段			対策後の残留リスク評価
				ハードウェアによる対策	ソフトウェアによる対策	表記（ラベリング）による対策	
カフの圧迫により300mmHg以上の過大な圧力が加わり測定部位でうっ血が起こる。	アプリケーションソフトウェアの不具合。リアルタイムOS(OTSソフトウェアの不具合)	考えられない	重大な	300mmHg以上の圧を安全サブシステムにて検知した場合、強制大気圧開放を実施する。	300mmHg以上の圧を測定系のサブシステムにて検知した場合、測定を中止し、急速排気を行い、エラーメッセージを表示する。	取扱説明書のトラブルシューティングにエラーメッセージの説明を記載する。	受容できる
15mmHg 以上の圧が180秒以上連続してカフにかかり続け測定部位でうっ血が起こる。	アプリケーションソフトウェアの不具合。リアルタイムOS(OTSソフトウェアの不具合)	考えられない	重大な	15mmHg 以上の圧を180秒以上連続して安全サブシステムにて検知した場合、強制大気圧開放を実施する。	15mmHg 以上の圧を180秒以上連続して検知した場合、測定を中止し、急速排気を行い、エラーメッセージを表示する。	取扱説明書のトラブルシューティングにエラーメッセージの説明を記載する。	受容できる

Safety Analysis

3. FTAによる リスクコントロールの効果分析



4. Safetyアーキテクチャ設計



転載禁止

注意すべき点

- システムの複雑化、多様化はセーフティの実現には危険信号
- サプライヤ１社で実現できないリスクが伴う機能の安全性の確保はリスクアセスメントを実施し、分析に基づき行動する。
- エンドユーザの利益を最優先に考える。（安全にとってセクショナリズムは最大の敵）
- 安全は一発で実現しようとはしてはダメ。繰り返しプロセスを回す。
- セーフティアーキテクチャを可視化しキープする。

エンドユーザーの安全を常に意識し、安全を実現するアーキテクチャを保持すること。
（ハード、ソフトの実現方法は問わない）

業務ドメイン別　リスクマネジメントの未来

自動車

Intended Use が多様化し、アーキテクチャが複雑化する

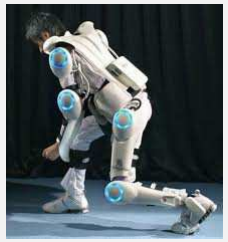
機器がネットワークにつながりこれまでにはなかったリスクが増える。

一体一の関係ではなくなる。

Intended Use が定まらず、リスクが特定しにくい。

医療機器

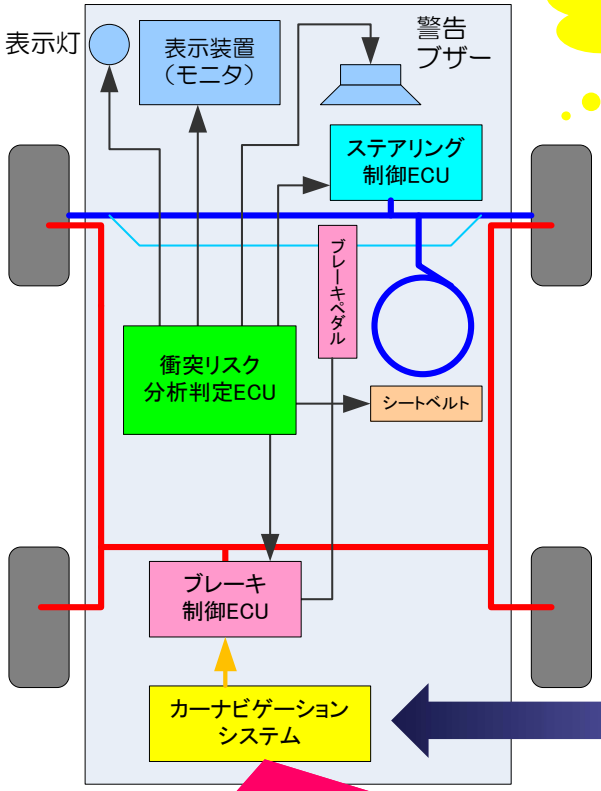
ロボット



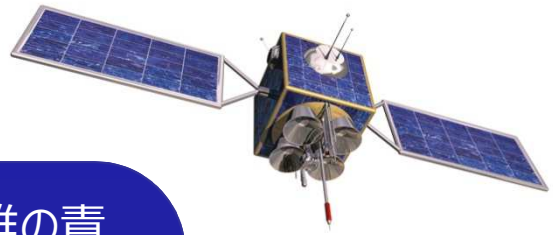
医療機器ドメインの昔と今の違い

30年前(1980年代)	現在(2010年代)
ソフトウェアへの過信	ソフトウェアは完全ではないと認識されている
安全性と信頼性の混同	一部の分野で安全性と信頼性が混同されている
モジュール分割の技術が進んでいなかった。	**リアルタイムOS, オブジェクト指向設計あり。**
防御的設計の欠如 （アーキテクチャを可視化できない）	リスクアセスメント、リスクコントロール要求あり （アーキテクチャを可視化することが可能）
不具合発生時のトレース機能がない	不具合発生時のトレース機能を用意
非現実的なリスクアセスメント	**今でもある**
レガシーソフト、OTSの安易な利用	**レガシーソフト、OTSへの要求がある**
ユーザーの事故報告義務がない	ユーザーの事故報告義務がある
ソフトに対する当局の規制と監督がない	ソフトに対する当局に規制と監督がある
ユーザビリティのリスク分析要求なし	**ユーザビリティのリスク分析要求あり**
開発プロセス要求、V&V要求なし	**開発プロセス要求、V&V要求あり**
ネットワーク接続なし	**ネットワーク接続による不具合が増加**

自動運転ソフトウェアの意図しない動きは誰のせい？



Intended Use が変わる



- 不具合の原因は誰の責任？
- 不具合を起こしたサプライヤを責めることで問題は解決するのか。
- 事故の再発を防止するという視点と、リスクを低減するアーキエクチャが必要。

低リスクアイテムが高リスクアイテムに変わる

責務が不明確な複合システムではリスク分析と対策が不可欠

まとめ

- 国際規格は神の啓示ではありません。各国の国際エキスパート達の議論と投票の共同作業によってできた標準です。規格要求の一つ一つがエンドユーザの安全のためになぜ役に立つのかを常に考えながら、規格適合の活動を進めましょう。
- ソフトウェア起因の不具合の特徴を理解した上で、プロセスアプローチの必要性を認識しましょう。
- 大規模複雑化したソフトウェアシステムにおいては、安全を実現するためのアーキテクチャが重要です。危害を防止するためのソフトウェアアーキテクチャ/ハードウェアアーキエクチャを認識し、仕様変更等によってアーキテクチャが崩れていないかどうかを確認しましょう。
- 自動運転などがもたらす相互運用性（Interoperability）の要求は基本機能、基本性能の独立性を崩し、想定し得なかった事故や原因が分かりにくい不具合を誘発します。そのような場合は、リスクベースアプローチにより、個別最適の発想ではなく、全体最適の発想で問題を解決しましょう。

参考文献 (1)

- **国際規格/ガイダンス**
  - ISO/IEC GUIDE 51:1999 Safety aspects — Guidelines for their inclusion in standards (JIS Z 8051:2004 安全側面-規格への導入指針)
  - IEC 61508：2000 Functional safety of electrical/electronic/programmable electronic safety － related systems（JIS C 0508－4：1999 電気・電子・プログラマブル電子安全関連系の機能安全）
  - ISO 26262:2010 - Road vehicles - Functional safety

参考文献 (2)

- **書籍**
  - 『安全設計の基本概念』」, 向殿政男 監修, 日本規格協会, 2007年
  - 『ソフトウェア品質保証の考え方と実際』, 保田 勝通
  - 『セーフウェア 安全・安心なシステムとソフトウェアを目指して』, ナンシー・レブソン, 翔泳社
  - 『リアルタイムOSから出発して組込みソフトエンジニアを極める』, 酒井由夫, エスアイビーアクセス
  - 『リコールを起こさないソフトウェアのつくり方』, 酒井由夫, 技術評論社
  - 『FMEA・FTA実践法』, 鈴木 順二郎, 日科技連

- **文献/調査研究資料**
  - 医療用ソフトウェアに関する研究会-中間報告書, 平成25年3月28日, 経済産業省, Ⅱ医療用ソフトウェアの関連調査
  - 広島市立大学 大場充 教授 講演「民主主義原理に基づくソフトウェア品質論入門」
  - 第15回 品質機能展開シンポジウム 特別講演 GD3 コンサルティング代表/JMAC GD3 センター長 吉村達彦 氏発表資料
  - 日経ものづくり 2010年8月号 の特集記事『ソフトが揺さぶる製品安全』
  - 日経ものづくり 2006年10月号 [特集] シンドラーの波紋
  - SESSAME e-Learning シリーズ D-01 ソフトウェア安全分析・設計, 2013, 酒井由夫, http://www.sessame.jp/e-learning/DomainSpecificContents/index.htm

参考文献 (3)

- **WEBサイト**
  - 社会人MBA-技術者編 シックスシグマ(工程FMEA) , http://tech-d.blogspot.jp/2008/01/1fmea.html
  - 話題沸騰ポット GOMA-1015型　要求仕様書, SESSAME WEBサイト , http://www.sessame.jp
  - 新安全基準対応機能, http://www.toshiba-elevator.co.jp/elv/common/contents/products/new/new_safety.jsp
  - 意外と知らない電気メスの話, http://or-nurse.seesaa.net/article/21976211.html
  - 日産ニュースリリース,『日産自動車、「緊急操舵回避支援システム」を開発』, http://www.nissan-global.com/JP/NEWS/2012/_STORY/121017-01-j.html

- **Wikipedia**
  - http://ja.wikipedia.org/故障率曲線
  - http://ja.wikipedia.org/フォルトツリー解析
  - http://ja.wikipedia.org/FMEA
  - http://ja.wikipedia.org/HACCP

SEESAME コンテンツのご利用に際して

- 本著作物の著作権は作成者または作成者の所属する組織が所有し、著作権法によって保護されています
- SESSAMEは本著作物に関して著作者から著作物の利用※を許諾されています
- 本著作物はSESSAMEが利用者個人に対して使用許諾を与え、使用を認めています
- SESSAMEから使用許諾を与えられた個人以外の方で本著作物を使用したい場合は query@sessame.jp までお問い合わせください

※ SESSAMEが著作者から許諾されている権利
著作物の複製・上演・演奏・公衆送信及び送信可能化・口述・展示・上映及び 頒布・貸与・翻訳・翻案・二次的著作物の利用

- ドキュメント中には Microsoft社, Adobe社等が著作権を所有しているクリップアートが含まれています