（安全仕様化WG）

AI・IoT時代を見据えた
セーフティ技術の知見を体系化する取り組み

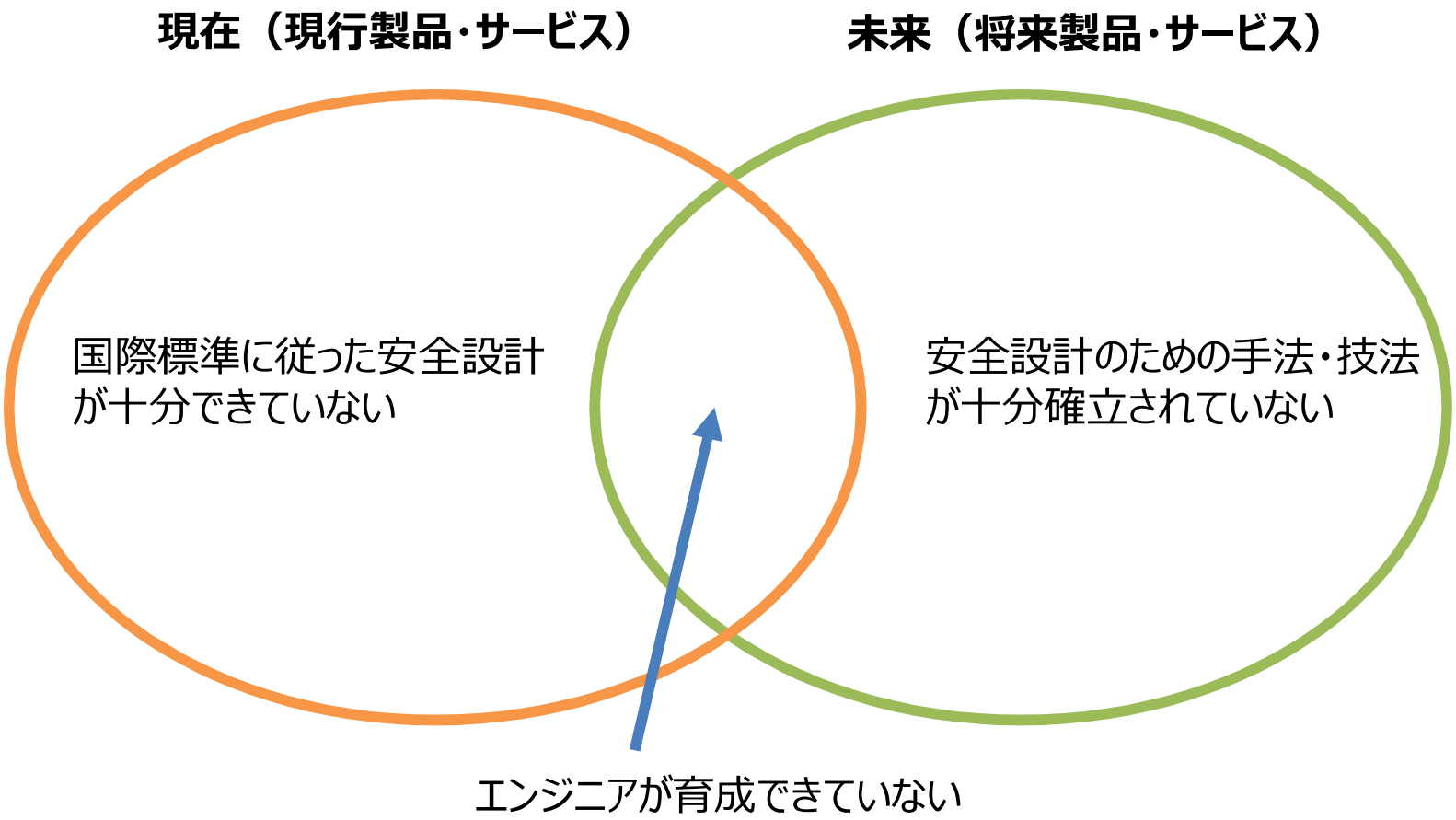
2019年5月20日

(株)東芝　余宮　尚志
hisashi.yomiya@toshiba.co.jp
hisashi.yomiya@gmail.com

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

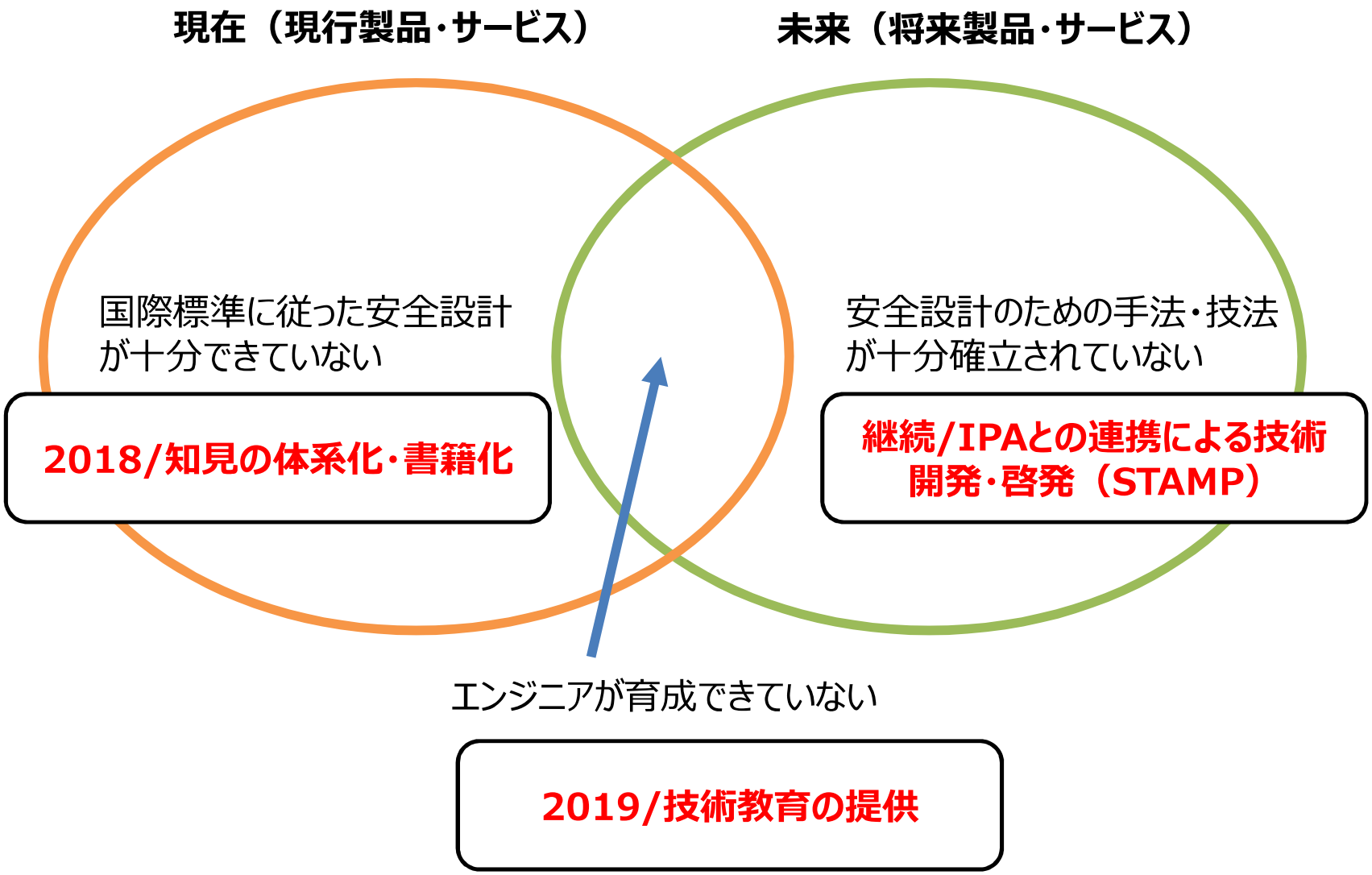
© Japan Embedded Systems Technology Association

AI・IoT時代における安全設計の主な課題



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

安全仕様化WGのアプローチ



発表内容

1. 安全仕様化WGの活動紹介

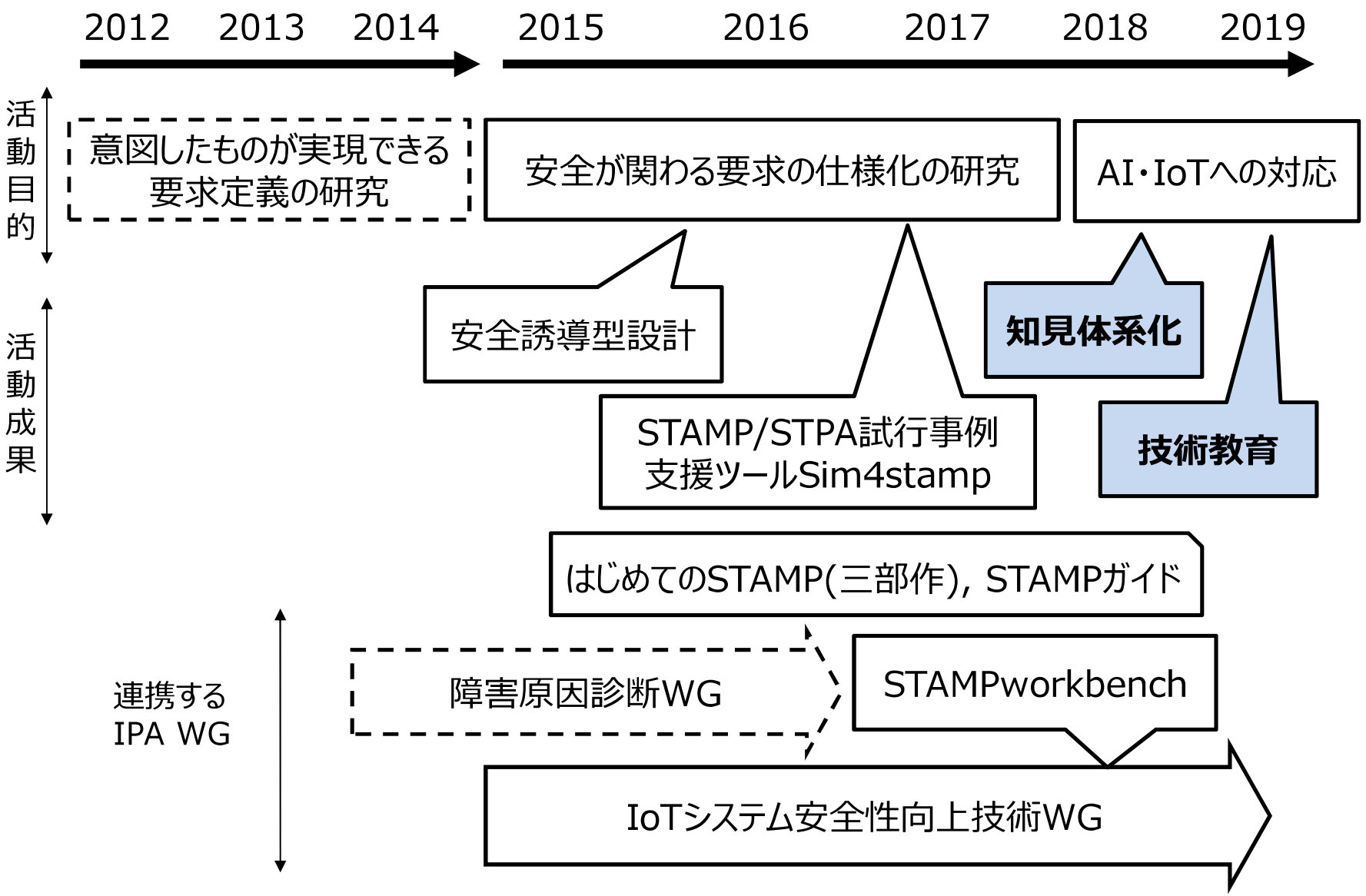
2. 知見の体系化・書籍化の取組み

安全仕様化WGの活動紹介

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

活動推移


2012
2013
2014
2015
2016
2017
2018
2019
活動目的
活動成果
意図したものが実現できる要求定義の研究
安全が関わる要求の仕様化の研究
AI・IoTへの対応
安全誘導型設計
STAMP/STPA試行事例
支援ツールSim4stamp
知見体系化
技術教育
はじめてのSTAMP(三部作), STAMPガイド
連携する
IPA WG
障害原因診断WG
STAMPworkbench
IoTシステム安全性向上技術WG

2018年度活動紹介

① 知見の体系化・書籍化

本日の主な発表

② IPA連携によるSTAMP/STPAの事例開発、啓発

③ ET(WEST, 横浜, 名古屋)での講演

④ ET(横浜)でのパネル展示

⑤ Sim4stamp(STAMPシミュレータ)の改良

②IPA連携によるSTAMP/STPA事例開発・啓発

		2015	2016	2017	2018
成果物	目標	知る ・STAMP紹介、基本的手法解説 ・既存設計手法との連携解説	分かる ・STAMP事例解説、適用ノウハウ紹介 ・システム俯瞰の必要性を警告	できる ・STAMP事例解説、場面に応じた解説 ・新技術:FRAM, Safety2.0解説	定着 ・有効な活用方法と、効果的な事例紹介 ・新技術:Securityとレジリエンス
	結果	はじめてのSTAMP入門編 10,229ダウンロード	はじめてのSTAMP実践編 2,171ダウンロード	はじめてのSTAMP活用編 2,750ダウンロード	STAMPガイドブック ツールダウンロード2,100件
			ツール仕様検討	ツール開発	ツール公開
普及活動	目標	STAMP認知度向上	STAMP認知度向上	STAMP認知度向上	STAMP認知度向上 理解度向上
	結果	ETにて紹介 ETWestにて紹介	ETにて紹介 ETWestにて紹介	ETにて紹介 ETWestにて紹介 ET名古屋にて紹介	ETWestにて紹介 ET出展せず ET名古屋出展せず
			第1回STMP-WS 参加者：**117名**	第2回STAMP-WS 参加者：**180名**	第3回STAMP-WS 参加者：**280名**

STAMPはAI・IoTを見据えた次世代の新しい安全分析手法
6名の安全仕様化WGメンバがIPA委員として連携活動
技術の確実な開発・定着・啓発が進んだ

③ET横浜での講演



JG-1

11月14日（水）13:30-14:30
会議センター［211＋212］

システムズ理論によるソフトウェアの安全設計・現状と今後

自動車、列車、ロボットからプラントまで現在の工学システムはソフトウェアで安全が保たれており、そのソフトウェアはますます複雑化している。現状の安全設計法や安全規格の限界も見えており、システムズ理論による新しい安全解析法STAMP/STPAにも注目が集まっている。本講演では、ソフトウェア集約型のシステムの安全設計に焦点を当てて、その現状と今後の動向を紹介する。

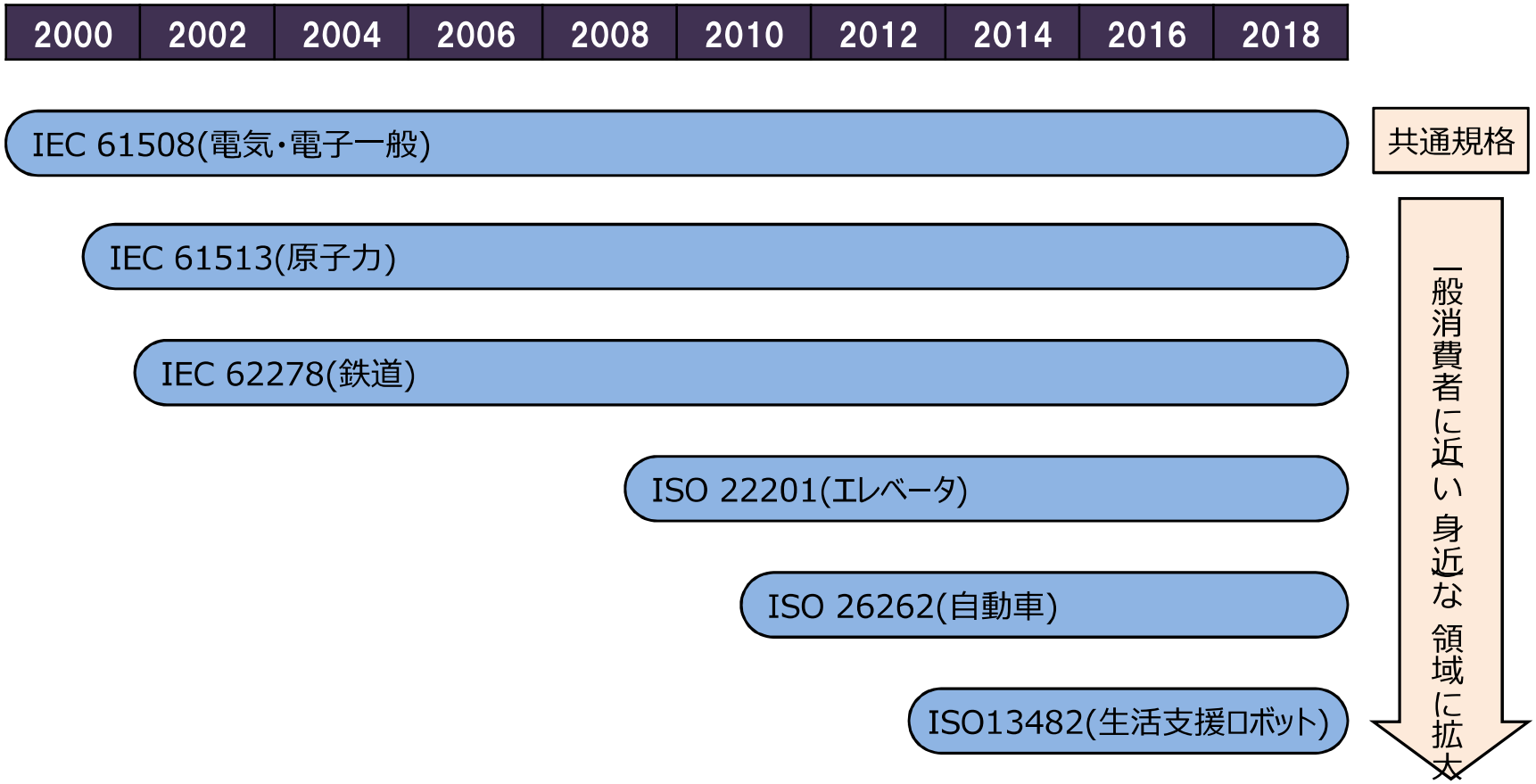
知見の体系化・書籍化で進めている内容と、STAMP解説を講演
130名の講演申込み、実聴講者80名以上と盛況
聴講者によるアンケート結果も良好(WGメンバ新規申込みもあり)

知見の体系化・書籍化の取組み

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

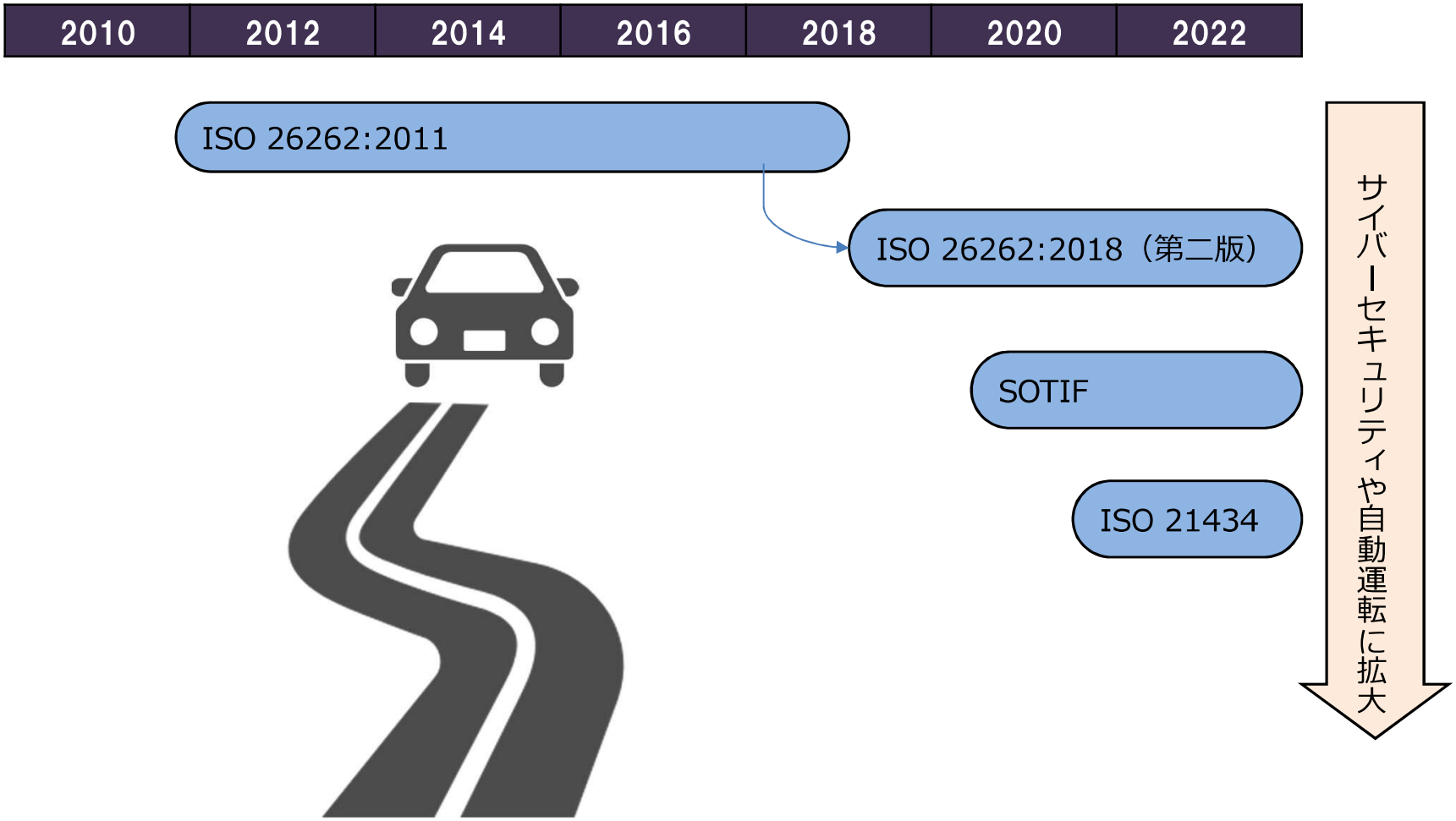
セーフティの重要性：機能安全規格の広がり



国際標準に従った安全設計が急務となっている

© 2019 Toshiba Corporation

自動車業界におけるセーフティ：技術範囲の広がり



AI・IoT時代で「安全設計の範囲」が広がってきている

© 2019 Toshiba Corporation

セーフティ技術を取り巻く環境と課題



- 現在
  - システムの大規模化・複雑化
  - **国際規格に従った安全設計**
- 未来
  - AI・IoT時代の到来
  - より大規模化・複雑化するシステム・ソフトウェア
  - 国際安全規格の限界、及びその対応
  - 体験したことのない未知の製品・サービスでの危険・脅威増
- 現在～未来　共通
  - **エンジニアの不足、スキル不足**
  - **安価な情報習得機会がない**（中小企業での対応が難しい）

課題に対する現市場でのソリューション

- 情報源
  - 体系化した書籍、文献が極めて少ない
  - 数少ない書籍は、専門家向けではない
  - 最新規格に対応していない、規格そのままの内容
  - ロボット安全(規格解説)など特定分野の書籍はそもそも存在しない



- 教育の機会
  - 教育自体が少なく、不親切で、高額である



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

安全仕様化WGの2018年度の成果

- 既存の知見を体系化し、書籍化を実現
- 「**システム技術に基づく安全設計ガイド**」
  （2019年9月上旬に発売予定）





Why in JASA ?
「会員企業の積極的な参加により一企業では難しいさまざまな課題に取り組んでいます。」…JASAのHPより
JASAの取組む「人材育成」にも非常に関連性が強い

15

「書籍:安全設計ガイド」の特色

- **予備知識なく**読める内容
- 現行の**国際安全規格を幅広く**網羅
- 各分野の専門家による解説
- 豊富な具体事例

企業の実務家、大学教員
など日本を代表する専門家
8名が執筆に参加

- 規格の裏にある本質的な考え方まで解説
- AI・IoT時代を見据えた新しい安全分析の概念も解説
- **ソフトウェア技術**の重要性を指摘(豊富なソフトウェア安全設計の技術解説)

- 十分なボリュームと手に取りやすい価格

260ページ前後、2,600円(予価)



「書籍:安全設計ガイド」の紹介　1/4

- 第1章「安全の基本」

  安全の基本、安全設計の原理、安全論証の考え方、安全設計の基本戦略を解説

- 第2章「安全規格体系と概要」

  安全規格と標準化、基本規格であるISO/IEC Guide 51やISO 12100の解説。規格の適用範囲や規格の限界にも言及

- 第3章「リスクアセスメント」

  リスクアセスメントの原理原則、プロセスを解説。FMEA, FTA, HAZOPなど既存の安全分析手法の紹介と豊富な事例

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

「書籍:安全設計ガイド」の紹介　2/4

- 第4章「機械系安全規格から見た安全設計の基本」
  機械系安全規格ISO 13849に基づいたリスク低減方策として、3ステップメソッドを中心に解説

- 第5章「機能安全設計の基本/IEC 61508」
  電気・電子・プログラマブル電子の機能安全規格IEC 61508の要求事項に基づいて、安全関連系の設計・開発方法について解説
  IEC 61508:2010に対応

- 第6章「自動車の機能安全/ISO 26262」
  自動車の機能安全規格ISO 26262について、規格の概略や全体構成、安全ライフサイクル、規格の主要なパートにおける考え方を中心に解説
  ISO 26262:2018に対応

日本初

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

「書籍:安全設計ガイド」の紹介　3/4

- 第7章「生活支援ロボットの安全規格/ISO 13482」

  生活支援ロボットの安全規格であるISO 13482について、規格の構成や安全設計の流れについて、簡単な事例を用いながら、概略を説明

  ISO 13482:2014に対応

日本初

- 第8章「システム思考で考えるこれからの安全」

  システム理論に基づく安全分析手法（STAMP/STPA）について、背景にあるシステム思考の考え方、分析手順をいくつかの具体例を通して解説

IPA連携成果

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

「書籍:安全設計ガイド」の紹介　4/4

- 第9章「ソフトウェアエンジニアのための安全設計」

  安全設計で近年重要度を増すソフトウェアについて、(1)ウォータフォールとアジャイル開発プロセス、(2)モデルベース開発、(3)モデル検査、(4)コーディングガイド、(5)ソフトウェアFMEAという5つの要素技術を解説

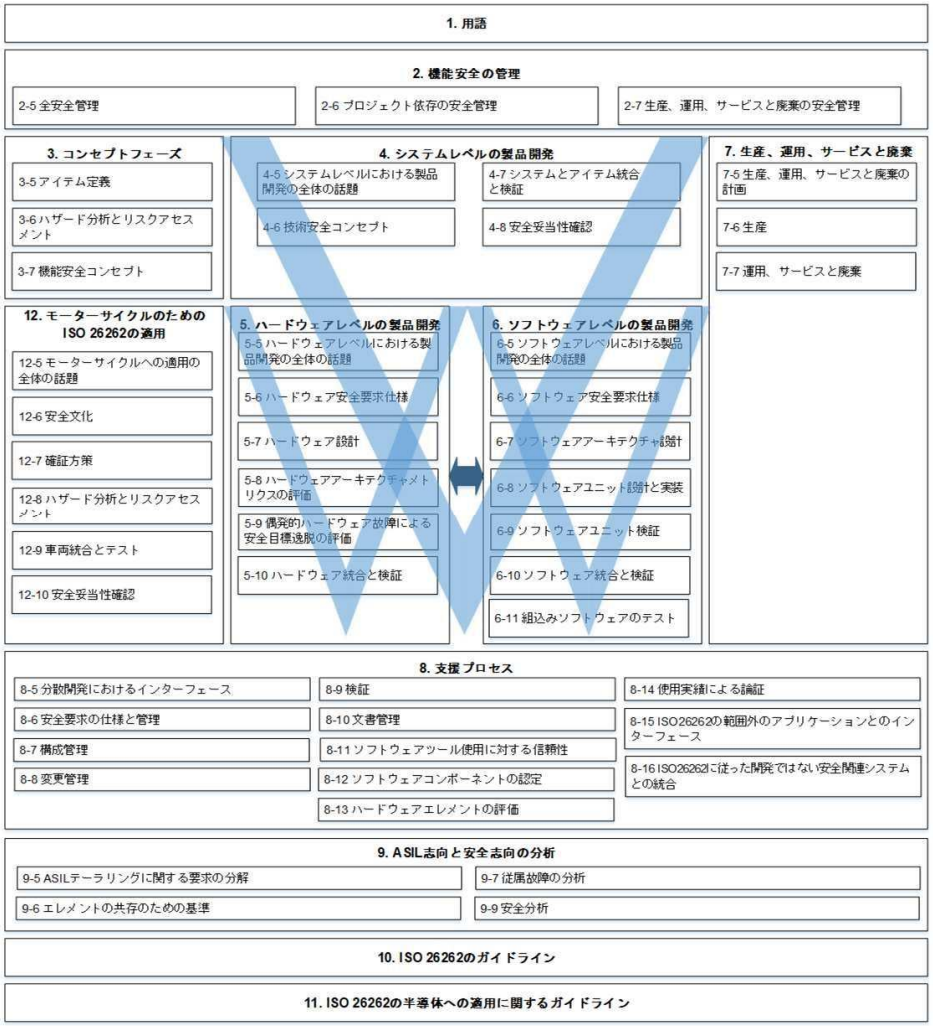
日本初

- 豊富なコラム
  - ディペンダビリティと安全性
  - リスク管理
  - ハードウェア故障とソフトウェア故障
  - サイバーセキュリティと安全
  - 性能限界や誤操作、誤使用をカバーする規格―SOTIF
  - ソフトウェア障害発生に関する課題 etc

10以上の豊富なコラムで、周辺技術、最新トピックスを網羅

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

技術トピック紹介: 第6章



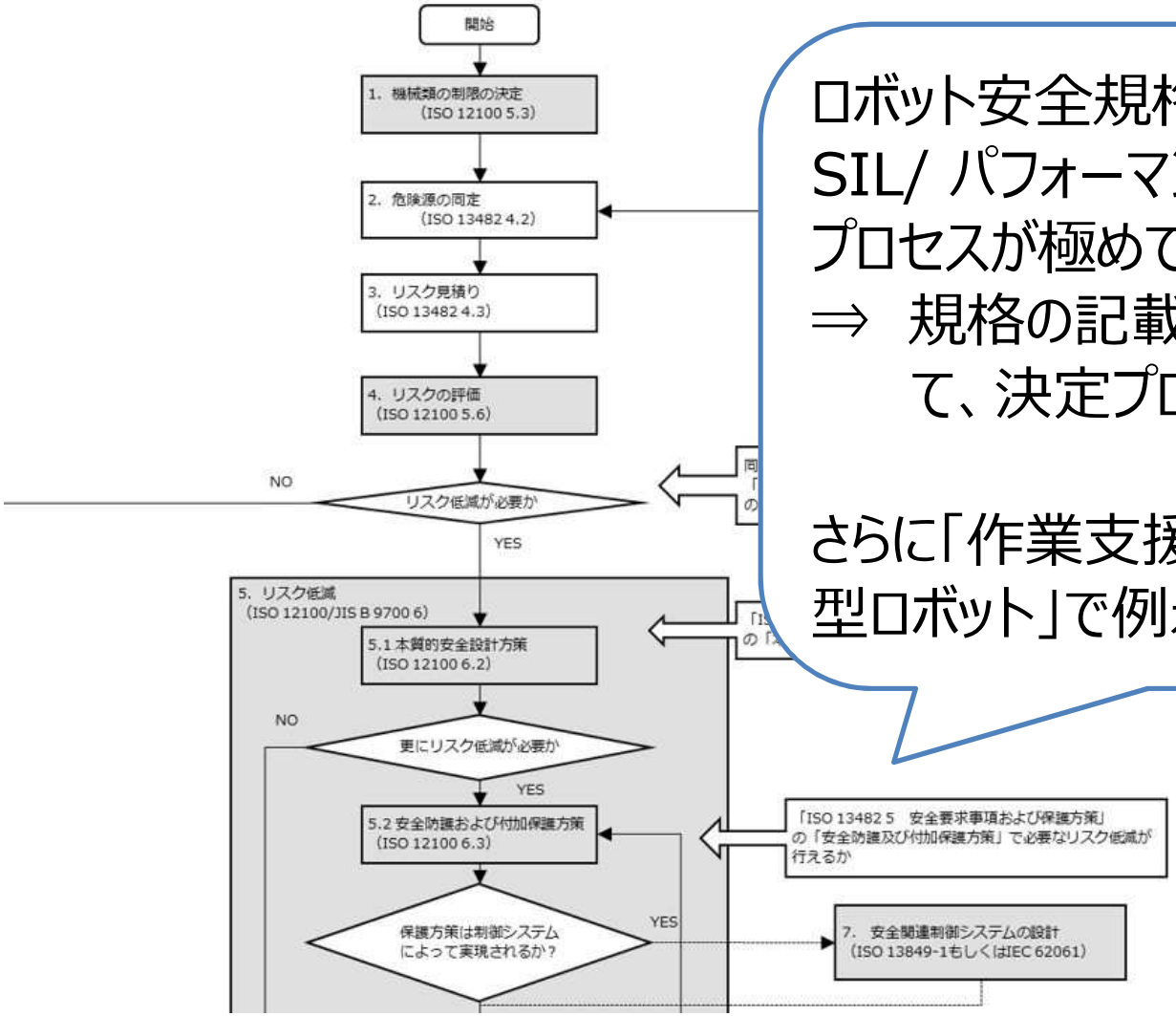
ISO 26262:2018 の概観

全800ページを超える膨大で難解な規格

主要な章(Part)について、初心者向けの簡単な概要を解説して理解を促した

21

技術トピック紹介: 第7章



ロボット安全規格では、安全度水準SIL/ パフォーマンスレベルPLの決定プロセスが極めて不明瞭

⇒ 規格の記載内容に解釈を加えて、決定プロセスを具体化

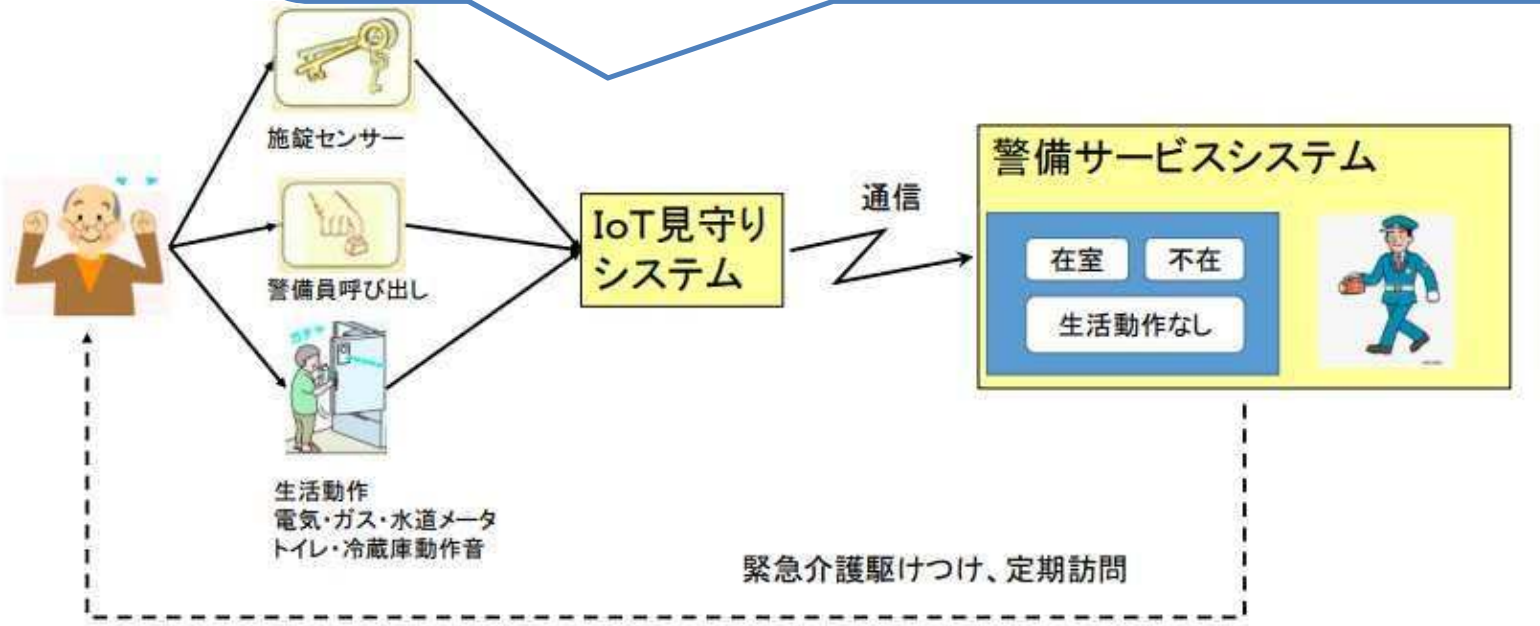
さらに「作業支援を目的とした装着型ロボット」で例示

ISO 13489安全設計の手順(一部)

技術トピック紹介: 第8章

AI・IoT時代に必要な新しい安全分析手法「STAMP」

概念の解説に留まらず、
「電源インターロック管理システム」、「鉄道踏切における安全監視装置“とりこ検知”」、「高齢者見守りサービス」を例示



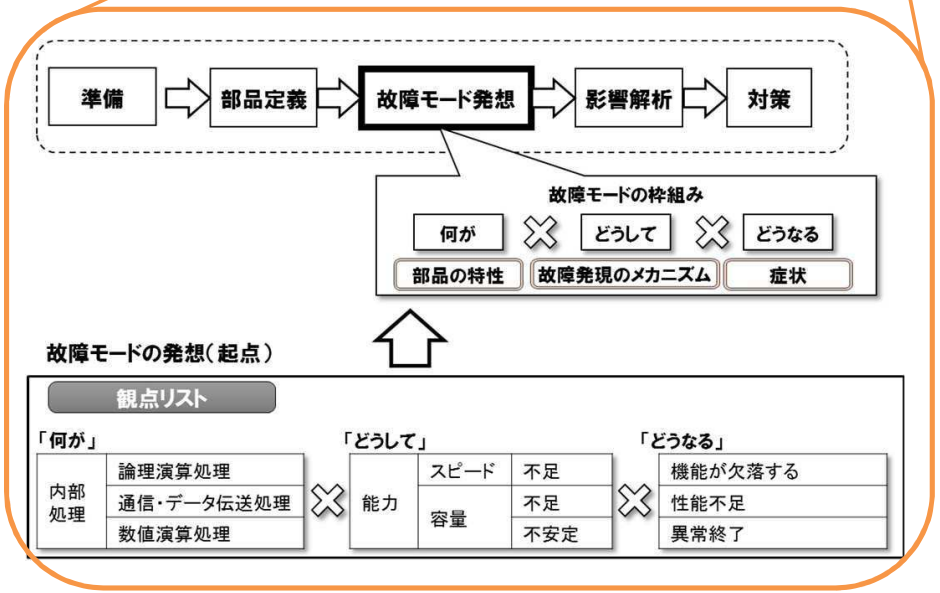
高齢者見守りシステムの概略

技術トピック紹介: 第9章

	ウォータフォール・アジャイル開発プロセス	モデルベース開発	モデル検査	コーディングガイド	ソフトウェアFMEA
要求分析	○	○			○
基本設計	○	○			◎
詳細設計	○	○			◎
実装(コーディング)	○	○		◎	
単体テスト	○	○	○		○
結合・総合テスト	○	○	◎		○
システムテスト	○	○	◎		

ソフトウェア安全設計に重要なコア技術を網羅的にピックアップし、詳細に解説

モデル検査は「踏切制御システム」で例示



ソフトウェアFMEA概略

まとめと2019年度の活動計画

- まとめ
  - AI・IoT時代のセーフティ技術に前提となる知見の体系化・書籍化を進めてきた(2019年9月に書籍発売決定)
  - 外部への情報発信、及びAI・IoT時代におけるコア技術の開発・啓発をIPAと連携して取り組んできた

- 2019年度からの活動計画
  - JASA内外へ技術教育を提供
  - コア技術の調査・開発・普及を促進(継続)
  - AIのセーフティ技術、情報セキュリティと協調するセーフティ技術の研究

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

組込み機器開発におけるセキュリティの考え方

～セキュリティ設計の2つの視点～

2019年5月20日

安全性向上化委員会/情報セキュリティWG
牧野 進二

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

0. 2018年度 活動報告

1. STAMP/STPA-SECの実証
    1. IPA様が推進しているSTAMP/STPA-SECの実例を使った検討結果と活用方法のまとめ
2. セキュリティ設計の視点と考え方
    1. システム全体における視点と考え方
    2. 製品ライフサイクルにおける対応の考え方
3. 簡易PKI認証システムの実証実験結果
    1. Block chainを使ったPKI認証の実証実験結果報告

1. STAMP/STPA-SECの実証

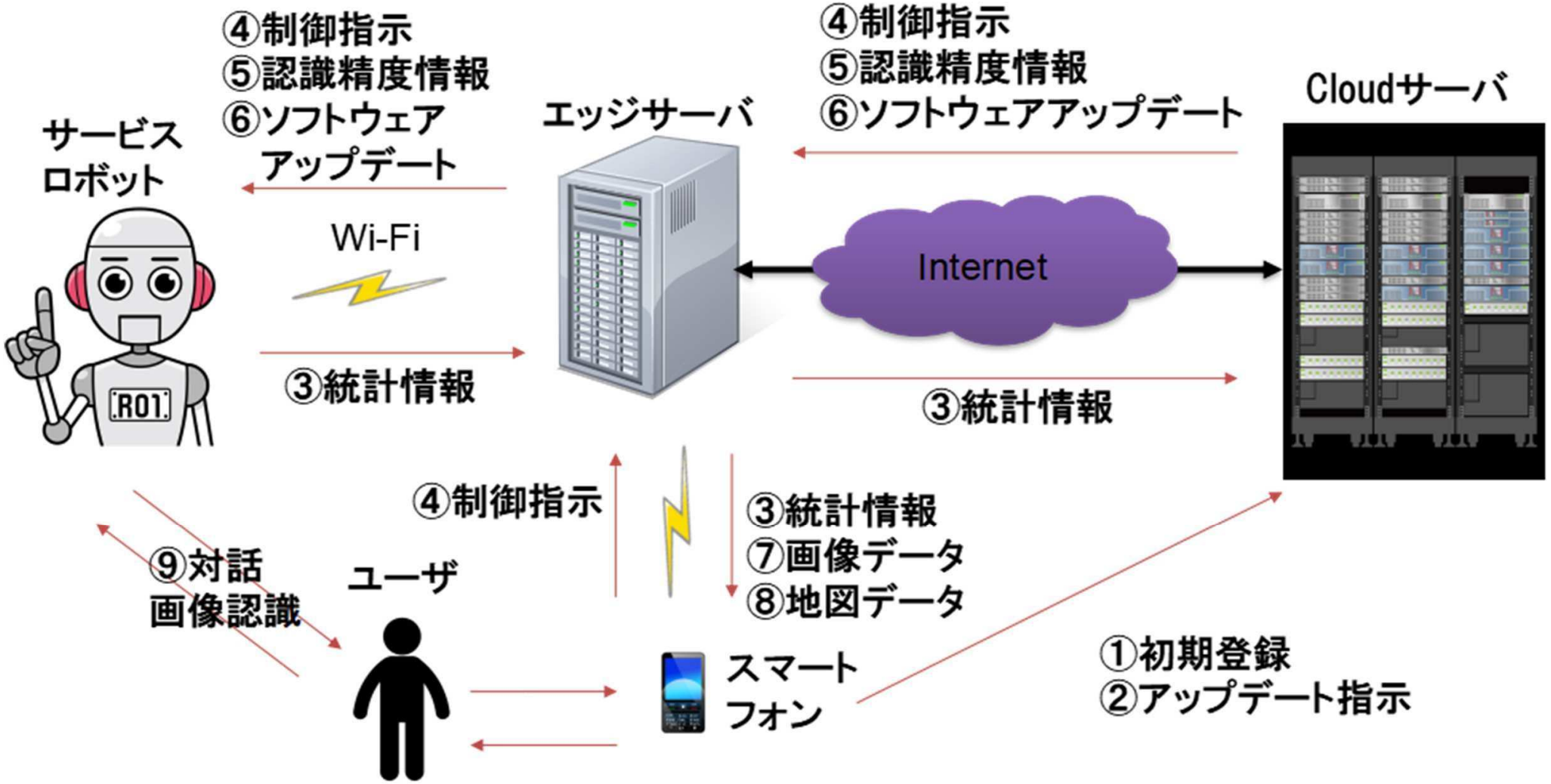
1. STAMP/STPA-SECの実証

■ 17年度に続きSTAMP/STPA-SECに関する検討を実施。

- 実例となるシステム構成を検討し、セキュリティ設計視点で実証を行った。

1. STAMP/STPA-SECの実証

■ システム構成案



1. STAMP/STPA-SECの実証

■ サービス仕様を定義して、STAMP/STPA-SECの検討を実施。

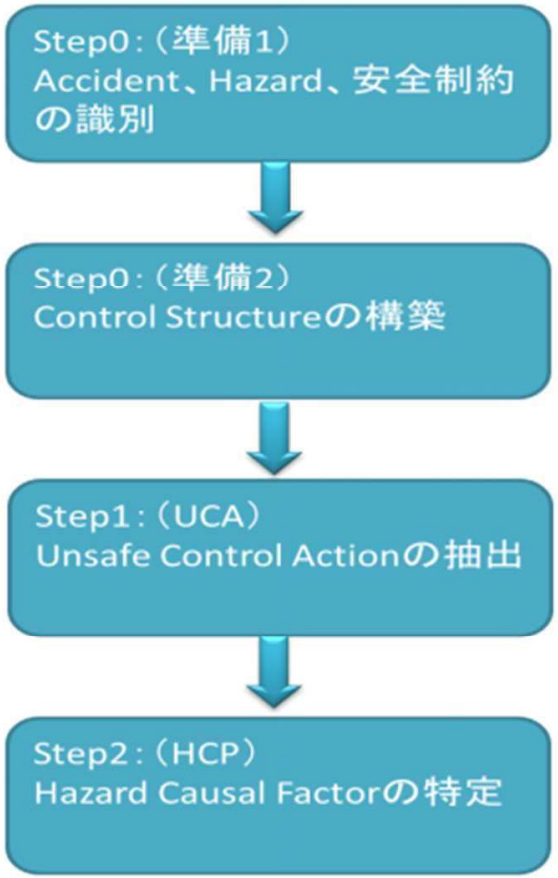
- サービスロボットは、ユーザと会話を行い、ユーザ指示にようる行動をする。ユーザからの指示や行動情報の統計を行い、ユーザの特性に合わせた自律行動をする。

機器名	機能
サービスロボット	画像認識、音声認識、各種センサー、GPSを搭載し、センサー情報、位置情報からモータ制御を行う。 ユーザからの指示は、スマートフォン経由、音声認識にて行われる。スマートフォン経由は、遠隔操作もできる。
エッジサーバ	サービスロボットからの統計情報を周期的に収集し、学習を行い認識精度の向上を行う。
Cloudサーバ	エッジサーバに溜まった統計情報を定期収集(1週間単位)に収集し、ユーザの行動認識を行い、ユーザに新たなサービス提供を行う。
スマートフォン	ユーザがサービスロボットに指示するのに利用される。遠隔操作もできる。

1. STAMP/STPA-SECの実証

■ STAMP/STPA-SECの手順通りに検証

■解析手順



■解析内容

対象システムにおいて、分析対象となる①Accident（望ましくない事象）、②Hazard（Accidentが潜在している具体的な状態）を定義し、Hazardを制御するためのシステム上の安全制約を識別する。

システムにおいて、安全制約の実現に関するコンポーネント（サブシステム、機器、組織等）、および、**コンポーネント間の相互作用（コントローラによる指示、フィードバックデータ）**を分析し、Control Structureを構築する。

Control Structure Diagramから安全制約の実行に必要なコントローラによる指示（Control Action）を識別し、**4つのガイドワード**を適用して、ハザードにつながる非安全なControl Action（UCA）を抽出する。

Step1で抽出したUCA毎に、関係するコントローラと制御対象プロセスを識別して、Control Loop Diagramを作成し、**ガイドワード**を適用して、ハザード誘発要因（HCF）を特定する。
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特定する。

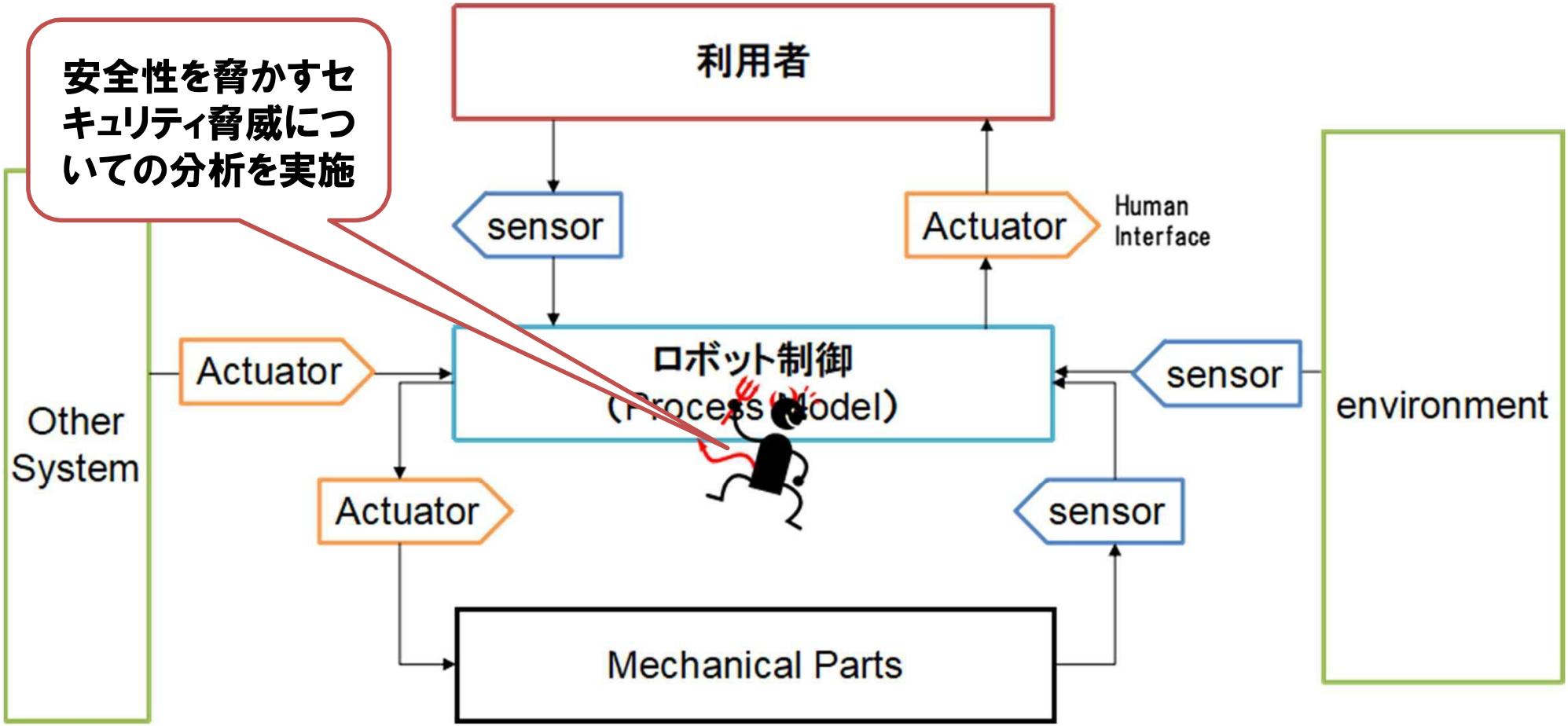
1. STAMP/STPA-SECの実証

■ Step2 Hazard Causal Factorの特定

- この段階でセキュリティ面の「STRIDE」を使ったガイドワードを使って安全性に対する脅威を分析することになる。。
  - Spoofing（なりすまし）
  - Tampering（改ざん）
  - Repudiation（否認）
  - Information Disclosure（情報漏えい）
  - Denial of Service（サービス拒否）
  - Elevation of Privilege（特権の昇格）

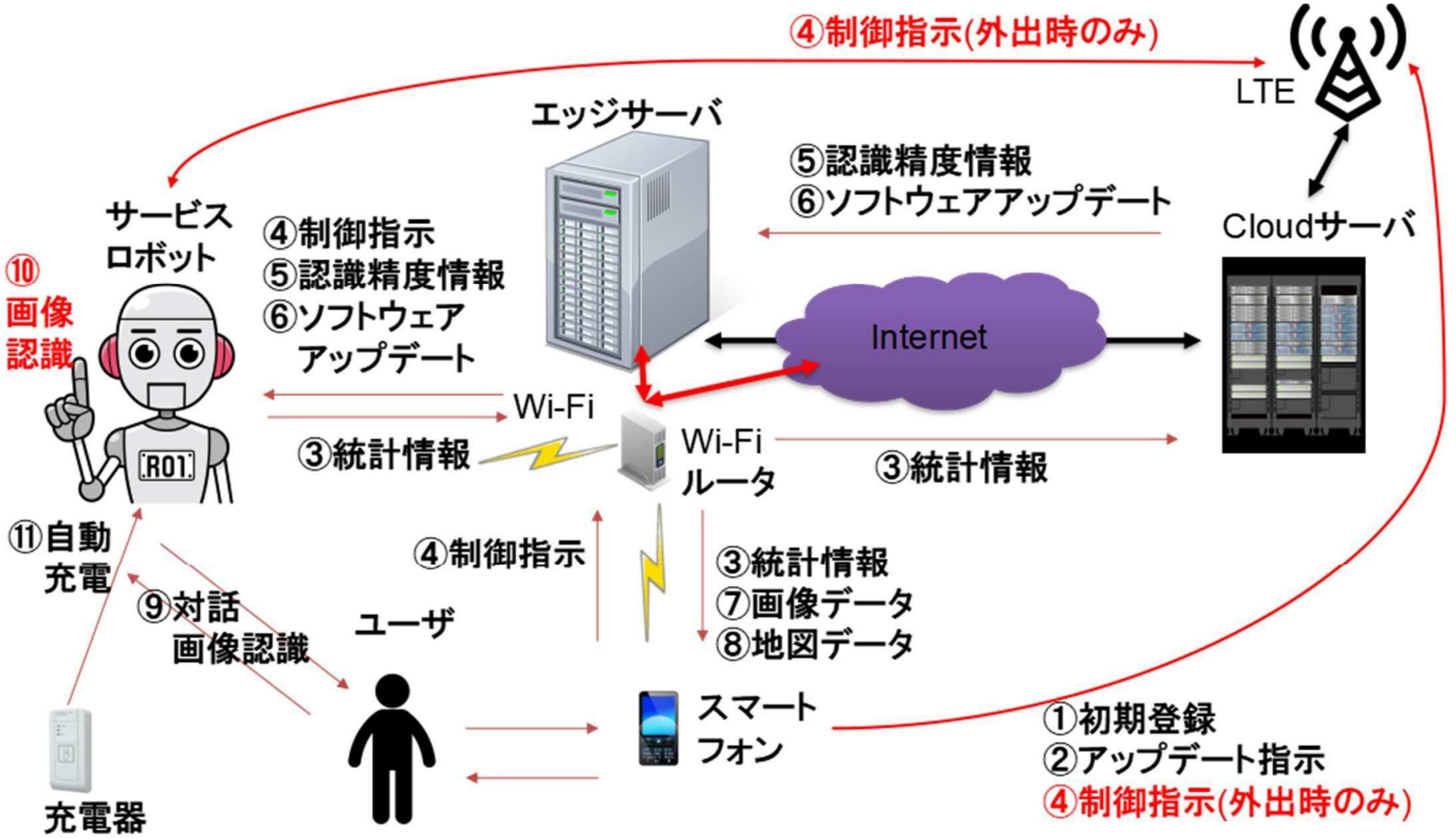
1. STAMP/STPA-SECの実証

■ Control Structureを用いて分析



1. STAMP/STPA-SECの実証

■ システム構成の見直し



1. STAMP/STPA-SECの実証

■ 仕様案の見直し

- 不足部分に対する仕様の見直しを実施。

機器名	機能
サービスロボット	画像認識、音声認識、各種センサー、GPSを搭載し、センサー情報、位置情報からモータ制御をする。 ロボットの用途は、対話と監視をするためのもの。 ユーザからの指示は、スマートフォン経由、音声認識にて行われる。スマートフォン経由は、遠隔操作(LTE経由)もできる。
エッジサーバ	サービスロボットからの統計情報を周期的に収集し、学習を行い認識精度の向上を行う。 ロボットの電池残量監視を行い、電池残量が10%以下になるとロボットに充電指示をする。
Cloudサーバ	エッジサーバに溜まった統計情報を定期収集(1週間単位)に収集し、ユーザの行動認識を行い、ユーザに新たなサービス提供を行う。
スマートフォン	ユーザがサービスロボットに指示するのに利用される。遠隔操作(LTE経由)もできる。 ロボットの状態を画像や位置情報でモニタリングできる。電池残量やロボットに行ってもらいたい事を制御指示として送ることで、ロボットを育てていく。

1. STAMP/STPA-SECの実証

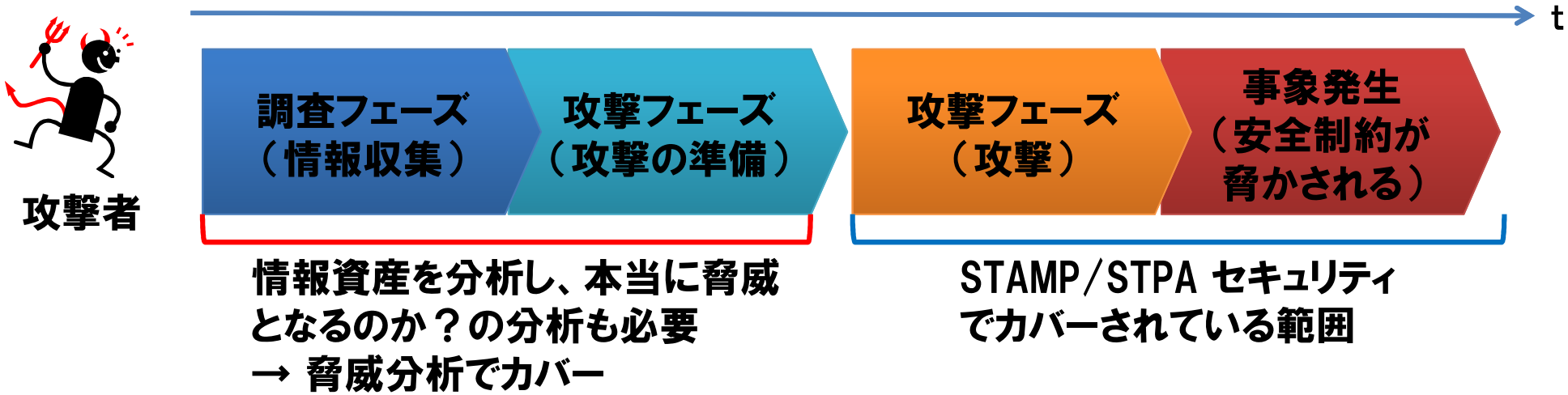
■ ２つの視点（まとめ）

- STAMP/STPA-SECを使うことで、システム全体を見渡した上流設計での安全性を脅かす脅威についての分析が有用であることが分かった。
  - ー システム改善や仕様の変更などに有用に使えることが分かった。
- 上流設計における改善点を下流設計にて証明することが重要であることが分かった。

1. STAMP/STPA-SECの実証

■ 課題となる点

- 攻撃者の行動を想定した場合、攻撃フェーズに移行する前の段階の分析についての分析視点も必要であることが分かった。



2. セキュリティ設計の視点と考え方

2. セキュリティ設計の視点と考え方

■　システム全体における視点と考え方

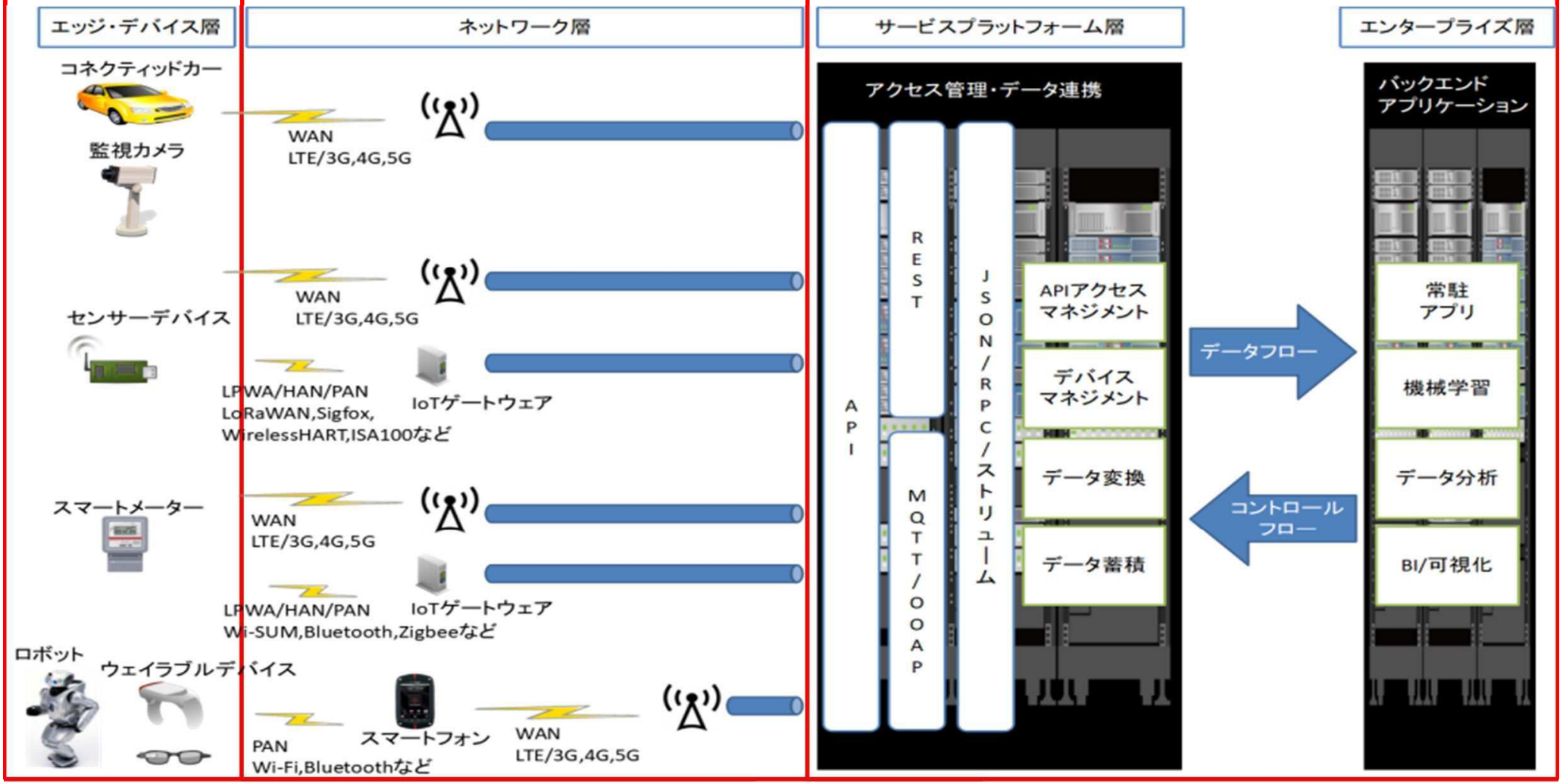
- IoT機器における組込み製品に対する全体的な視点をもって設計を行うことが必要となる。
- 組込み製品がシステム全体に及ぼす影響を踏まえたセキュリティ対策や設計が必要になる。
- スタンドアローンであった組込み製品は、IoTでの活用が必要になったことで、広いスキルが求められる。
  - ー　システム設計のスキル
  - ー　ネットワーク設計のスキル
  - ー　OSS活用などの開発プロセス改善のスキル

組込み製品開発スキル以外のスキルが求められる。

2. セキュリティ設計の視点と考え方

電気通信事業者の約款や契約内容などを確認する。

Cloud事業者の約款やサービス提供会社の契約内容などを確認する必要がある。



2. セキュリティ設計の視点と考え方

■ IoTシステムにおいては、組込み製品開発での「エッジ・デバイス層」のセキュリティ対策を重視する必要がある。

- 組込み製品の対策が不十分であるとシステム全体の脆弱性につながる恐れがある。
- 組込み機器の視点においては、他の機器との連携した動作の「考慮漏れ」や組込み製品のセキュリティ機能の「誤った使い方」が脆弱性につながる。
  - ー スマートカードや携帯電話のUSIMなど一部の重要なコンポーネントのみでは不十分で、システム全体から考えた対策を考慮する必要がある。

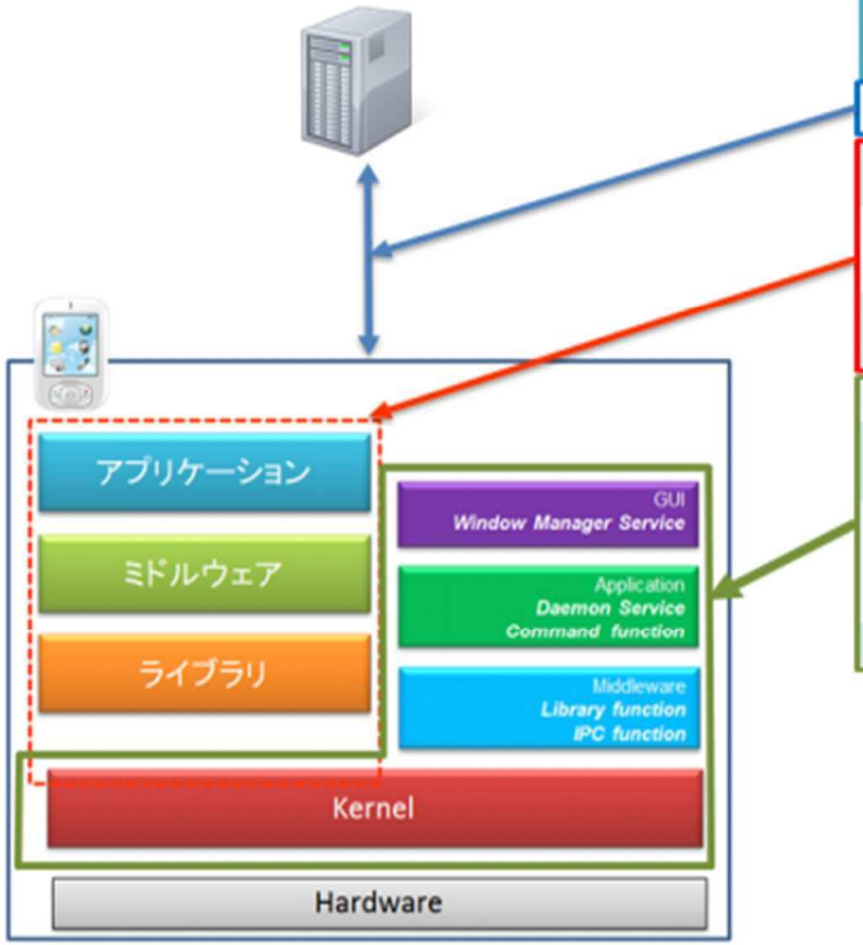
2. セキュリティ設計の視点と考え方

■　組込み製品視点でのセキュリティ対策の視点（案）

脆弱性の観点	概要
「記憶媒体」の脆弱性	不揮発メモリなどの記憶媒体に「ファームウェア」や「ファイルシステム」、暗号鍵、証明書などの「機徴情報」を平文で保存されているなど → 窃取、改ざんなどされ、更なる攻撃の糸口となる可能性がある。
基板上の通信路における脆弱性	IoTデバイスとして組み立てる際には多数のデバイス間でデータの送受がなされ、平文で行っている場合がある。回路上の信号線に直接接続して、これらのデータをモニタリングすることで、「機徴情報」や「制御パラメータ」などが読み取られ、更なる攻撃の糸口となる可能性がある。
デバッグ端子の脆弱性	プロセッサ/マイコンが備える保護機能を正しく利用していなかったり、デバッグ機能が無効化されていなかったりすることで、ファームウェアやメモリ状態などの読み書き可能となり、「機徴情報の読み取り」や「ファームウェア改ざん」など、更なる攻撃の糸口となる可能性がある。
メンテナンスコンソールの脆弱性	メンテナンスコンソールが有効なまま出荷されており、シリアル通信端子等を介して、特権的な操作を行うことができる場合がある。無認証で操作できることも少なくない。 認証が設定されていても、他の脆弱性と組み合わせて認証自体が無効化されることもある。
外部インタフェースの脆弱性	IoTデバイスが持つ汎用インターフェース（USBやBluetooth、Wi-Fiなど）が攻撃を受ける可能性がある。例えば、USBでは、USBメモリけいゆうでの不正アクセスファイル読み込みが挙げられる。 Bluetoothでは暗号鍵を窃取して機器同士の通信を解析したり、なりすまして不正接続することも考えられる。
ファームウェア解析で判明した脆弱性	前途の攻撃によりプログラムのファームウェアの抽出に至った場合、ファームウェアを解析することで、IoTデバイスが外部から受信したコマンド処理するコードに脆弱性（サービス拒否や任意コード実行など）が存在することが明らかになることがある。

2. セキュリティ設計の視点と考え方

■ 組込み機器の対策事例

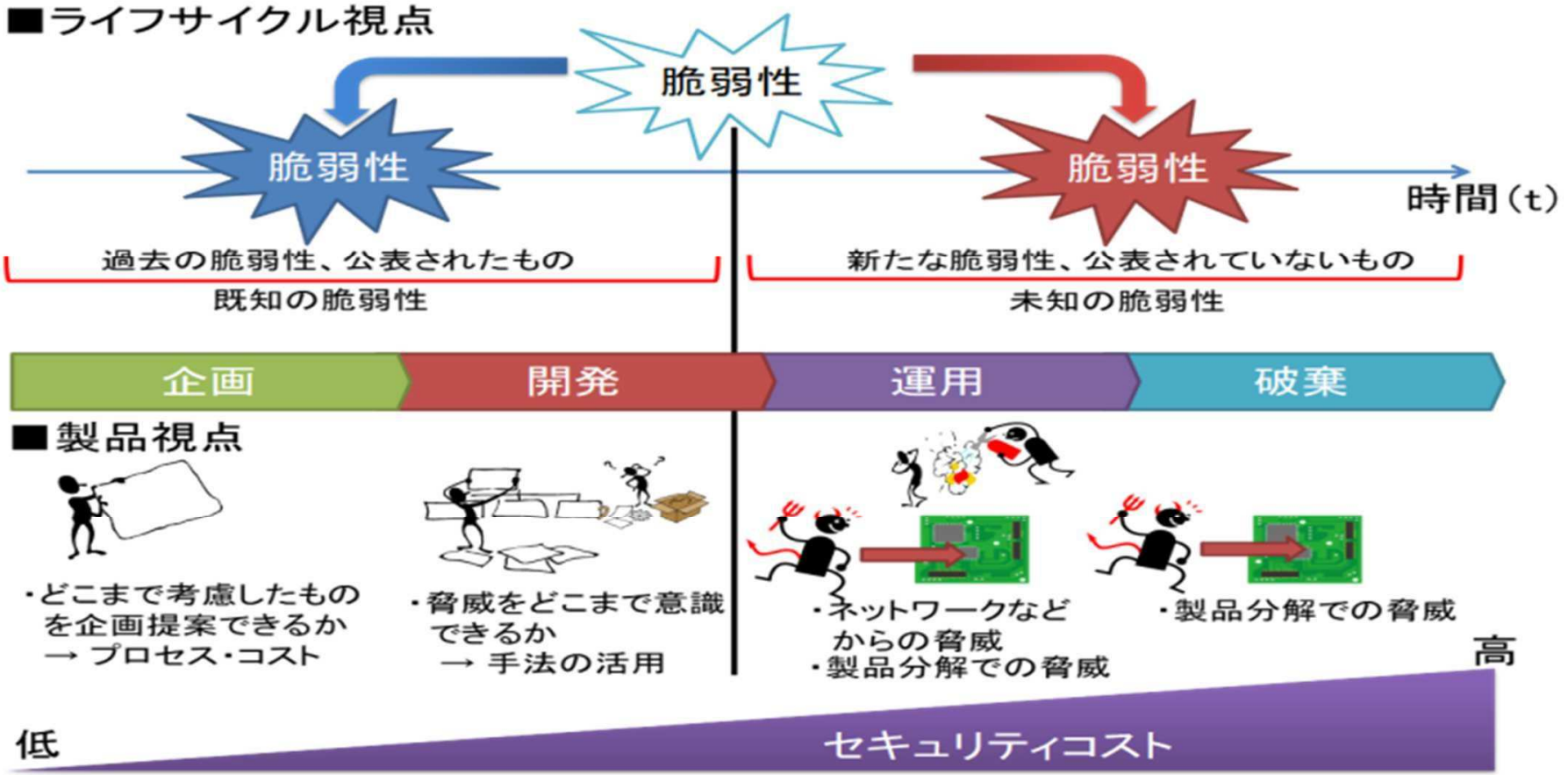


対策	説明	対策者	
		機器側	サーバ側
通信の暗号化	独自暗号でサーバと通信	アプリ	アプリ
デバッガ検知	デバッガを検知すると動作をとめる	ドライバ/アプリ	–
アプリケーション暗号化	インストールした以外の機器で動作していることを検知すると動作を止める	アプリ	–
暗号ロジックと暗号鍵の頻繁な変更	攻撃者に対して解析に十分な時間を与えない	アプリ	アプリ
パッチの迅速な適用	特権に昇格できるような脆弱性を塞ぐ	全般	–
Root化の検知	Root化を検知するとシステムロックをかける	全般	–
JTAGの無効化	秘密鍵を知らないとJTAGを利用できなくする。若しくは、JTAGを物理的に使えなくする	全般(ハード)	–
ストレージの暗号化	別の機器につないでも解析できない	全般	–

2. セキュリティ設計の視点と考え方

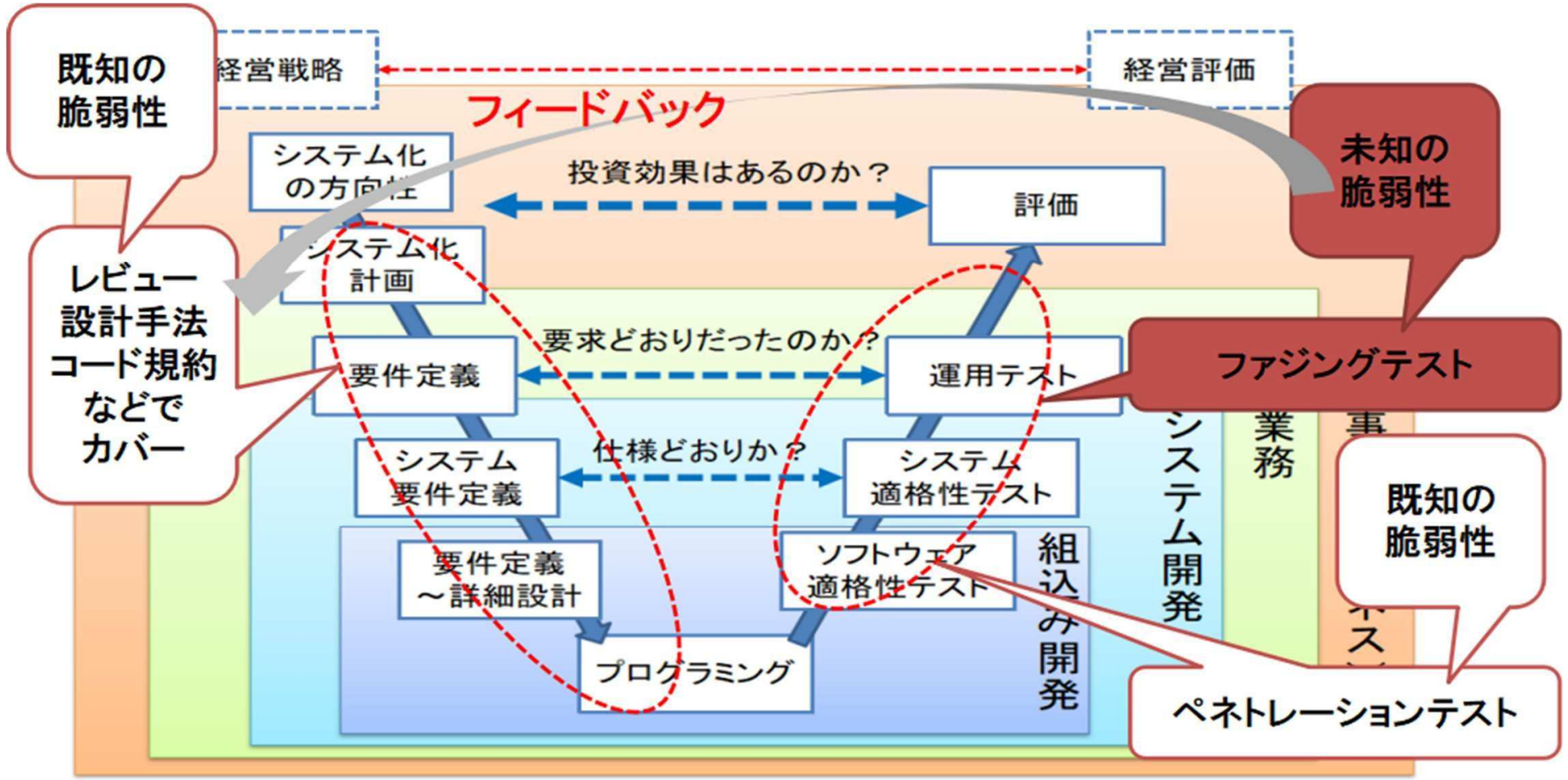
■ 製品ライフサイクルの考慮

- セキュリティ脅威は、時間とともに変化する。セキュリティ対策は、運用面の考慮をした上で、システム設計を実施する。



2. セキュリティ設計の視点と考え方

■ 開発プロセスでは、未知の脅威に対するプロセス改善を定義する必要がある。



2. セキュリティ設計の視点と考え方

■ 未知の脅威に対しては、セキュリティ関連の規定書にセキュリティテストを重視することと定義されていされている。

- テスト結果のフィードバックが上流での設計の対策向上につなげられる。

分野	文書	概要
情報セキュリティ全般	NIST SP800	セキュリティテストの1つとして実施を奨励している。
	重要生活機器連携セキュリティ協議会（CCDS）	未知の脆弱性検査手法として、実施を推奨している。
自動車	NHTSA	セキュリティ評価の手法として実施が奨励されいる。
	J3061	同上

2. セキュリティ設計の視点と考え方

■ 認証プログラムにおいてもテスト実施が定義されている。

分野	認証プログラム	概要
産業機器、医療機器	UL CAP (アメリカ)	ネットワークデバイスのソフトウェアに対するテストとして実施が定義されている
制御機器	EDSA (国際学会)	認証要求項目の堅牢制テストして実施が定義されている。

2. セキュリティ設計の視点と考え方

■ 昨今、組込み製品開発においては、スクラッチから作ることが少なくなってきた。

- OSS（オープンソースソフトウェア）に代表されるようにソフトウェア開発においては、部品として利用することが多くなってきている。
- OSSなど利用する部品に対する信頼度は、下流プロセスでの評価ではなく、利用する段階で評価を実施することで脆弱性対策につながる。
  - ー OSSなど外部ソフトウェアはブラックボックス。利用にあたっては、適切な評価・診断を実施することが必要。

2. セキュリティ設計の視点と考え方

■ ２つの視点（まとめ）

- IoTシステムは、システム全体の視点が必要である点が分かった。
  - 特にエッジ・デバイスとなる組込み機器の対策に考慮漏れがあるとシステム全体の脆弱性につながるので対策は不可欠となる。
- セキュリティ脅威は、時間とともに進化する。設計段階の未知の脅威に対する設計考慮が必要となる。
  - 上流設計での結果を下流設計で対策漏れがないことを証明し、フィードバックして改善する。
  - 未知の脅威に対しては、ファジングテストが有用である。
    - ✓ OSSを使う場合は、テストと診断をした上で上流設計での考慮をする必要がある。

3. 簡易PKI認証システムの実証実験結果

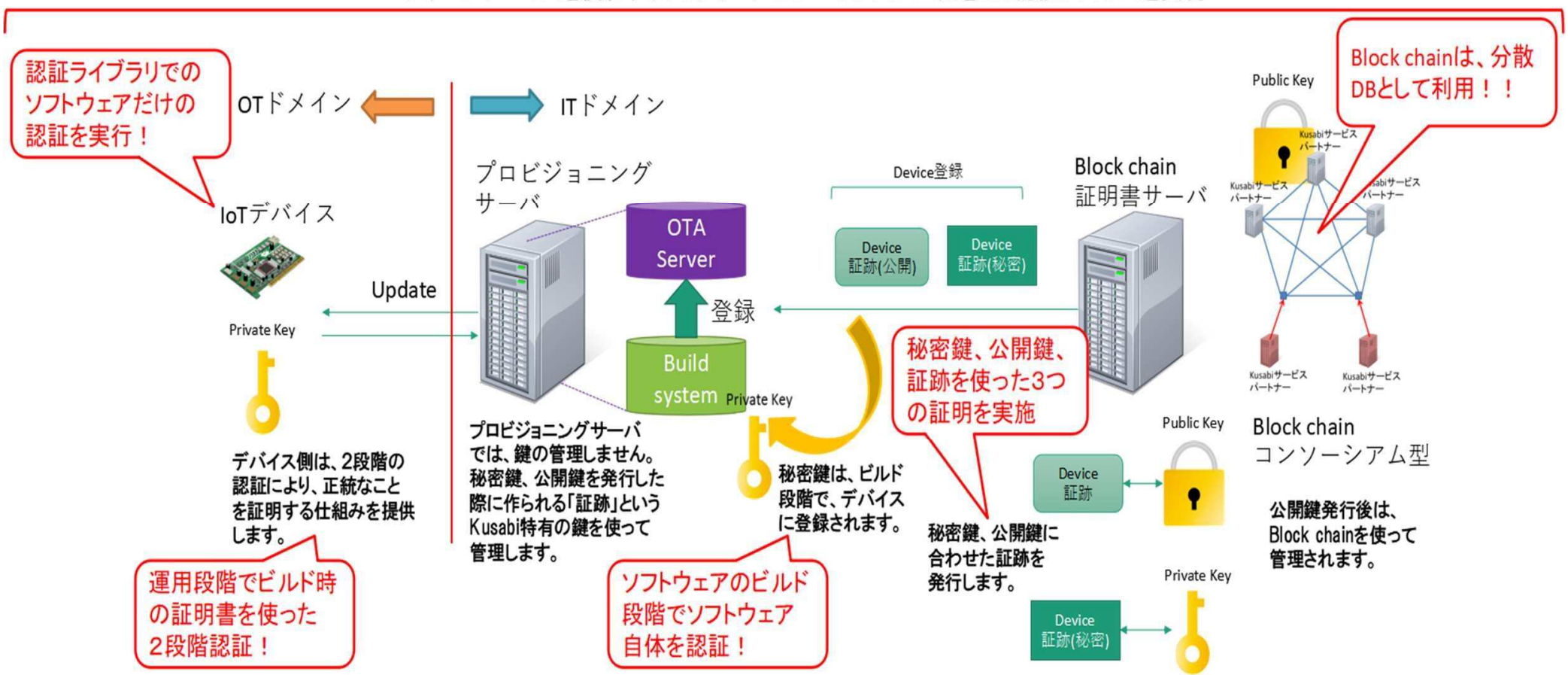
3. 簡易PKI認証システムの実証実験結果

■ 目的としては、安価な組込み製品における運用をする場合に、Trust zone/TPMなどの高価なハードウェアを使わないIoT機器の管理・監視ができるシステムを検討した。

- 既存の認証局（中央管理）のコストを下げる。
  - 分散管理を使うことで多数での監視を目的とした。
- ハードウェアデバイスを使わないで、ソフトウェアだけを使った認証方法を検討。
- 脅威に脅かされたときに検知し、自動的に正常な状態に戻すことを検討。
- 開発段階での認証を行える構成を検討。

3. 簡易PKI認証システムの実証実験結果

■ Block chainを活用したシステムの検討し、IoTデバイスの管理・監視ができるシステムを構築・実験を実施。



3. 簡易PKI認証システムの実証実験結果

■ Block chain利用メリット1

- 中央管理型のシステムでは管理者の一任となるが、Block chainを使うことで、分散データベース運用と参加者の合意形成ができ、真正性が向上できる点。

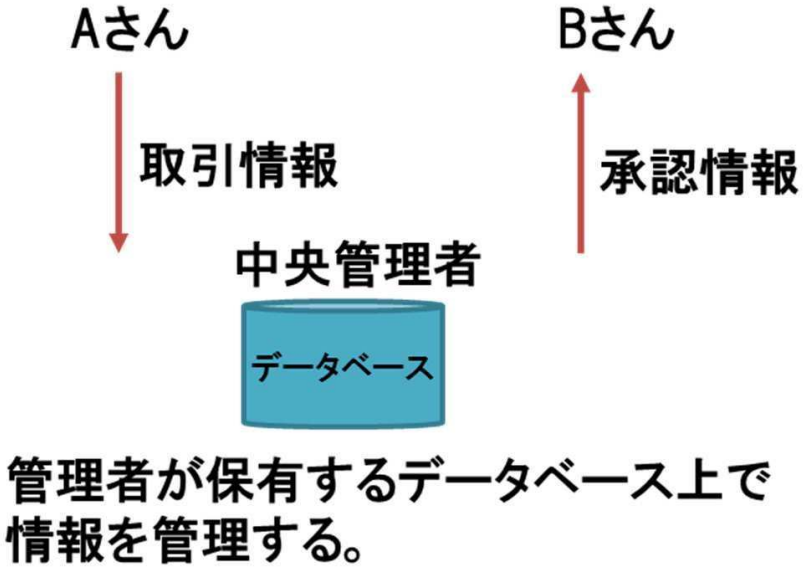


図 中央管理型

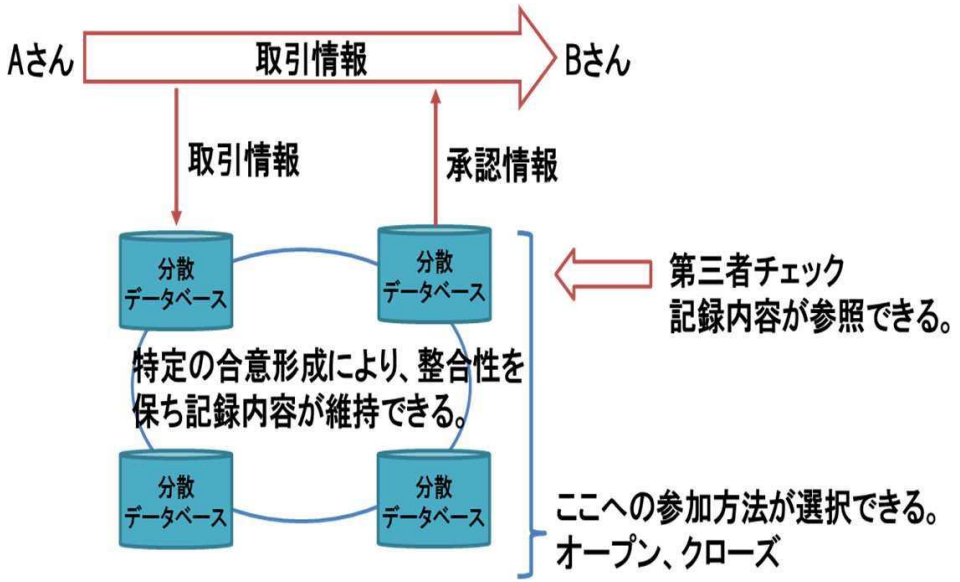
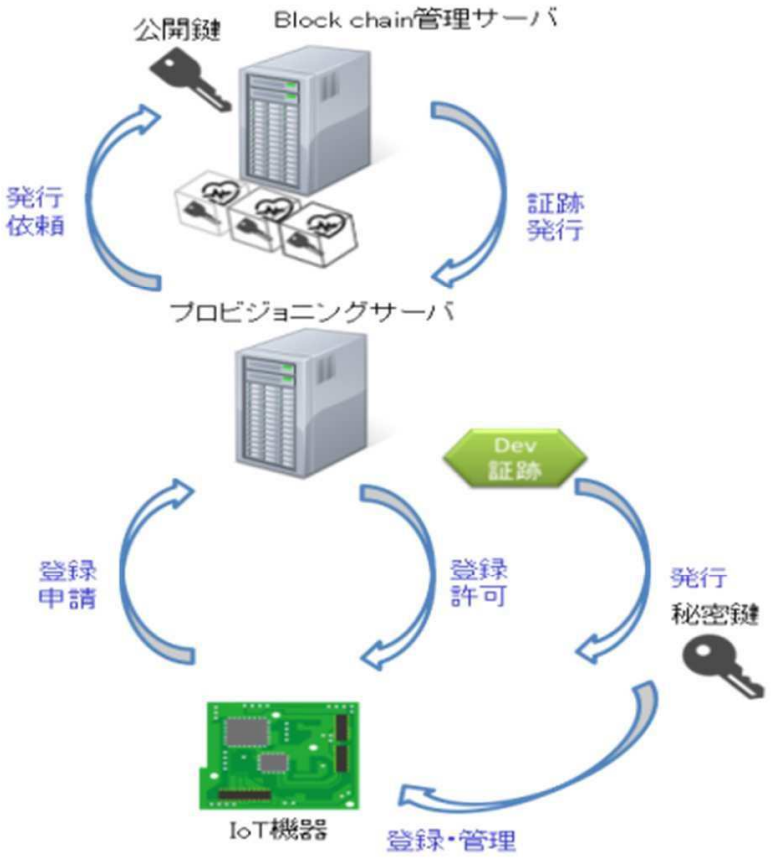


図 分散管理型

3. 簡易PKI認証システムの実証実験結果

■ Block chain利用メリット2

- 既存の認証局では人手を介した認証となるが、Block chainを使うことで自動的な認証が行え、有効期限を管理できる。



Block chain方式では、機器を管理するプロビジョニングサーバの管理で機器を管理することで、人手を介さないで、証明書の発行ができるようになる。
→ 青字の部分が自動化

IoT機器が運用される運用段階で証明書の発行ができるようになる。

Block chain特性を使うことで、追跡性の恩恵を受けられ、機器の状態管理、異常検知が行える。

証明書自体に有効期限がなく、機器が運用されている間は、自動化、分散管理ができるようになる。

従来の証明書管理では、サーバ証明のみとなるが、今回のシステムでは、サーバクライアント両方の認証が自動的に行える。

証明書管理後、通信は何でも使える。
→ データの取引情報をBlock chainにて管理できるので、改ざん検知、不正アクセス等が取引情報から確認できる。

パブリック型ではなく、プライベート型/コンソーシアム型を選択することで、管理主体者の制限、ネットワークの安定性が望める。

3. 簡易PKI認証システムの実証実験結果

■ 市販の安価なIoTデバイス（Raspberry PiやUpボードなど）を使って、「認証、監視、OTA（On The Air）」の検証を実施。

- 既存の認証局（中央管理）システムと同じ効果が得られることが証明できた。
- IoTデバイスを、ソフトウェアのみで認証、管理できることを証明できた。
  - 開発段階から認証でき、運用フェーズで認証された状態で管理・監視できることが証明できた。
- 総務省「IoTの普及に対応した電気通信設備の技術基準等に関する制度整備」など省庁のガイドラインに合わせた運用ができていることも証明できた。

3. 簡易PKI認証システムの実証実験結果

■ ２つの視点（まとめ）

- 組込み製品における認証・監視を安価に実現できることで、IoTシステムの運用を低コストにできることが分かった。
- セキュリティ・バイ・デザインに代表されるように製品自体の設計は大事であるが、運用面を考えた全体システムを導入することで、脆弱性対策になることが分かった。

18年度 成果まとめ

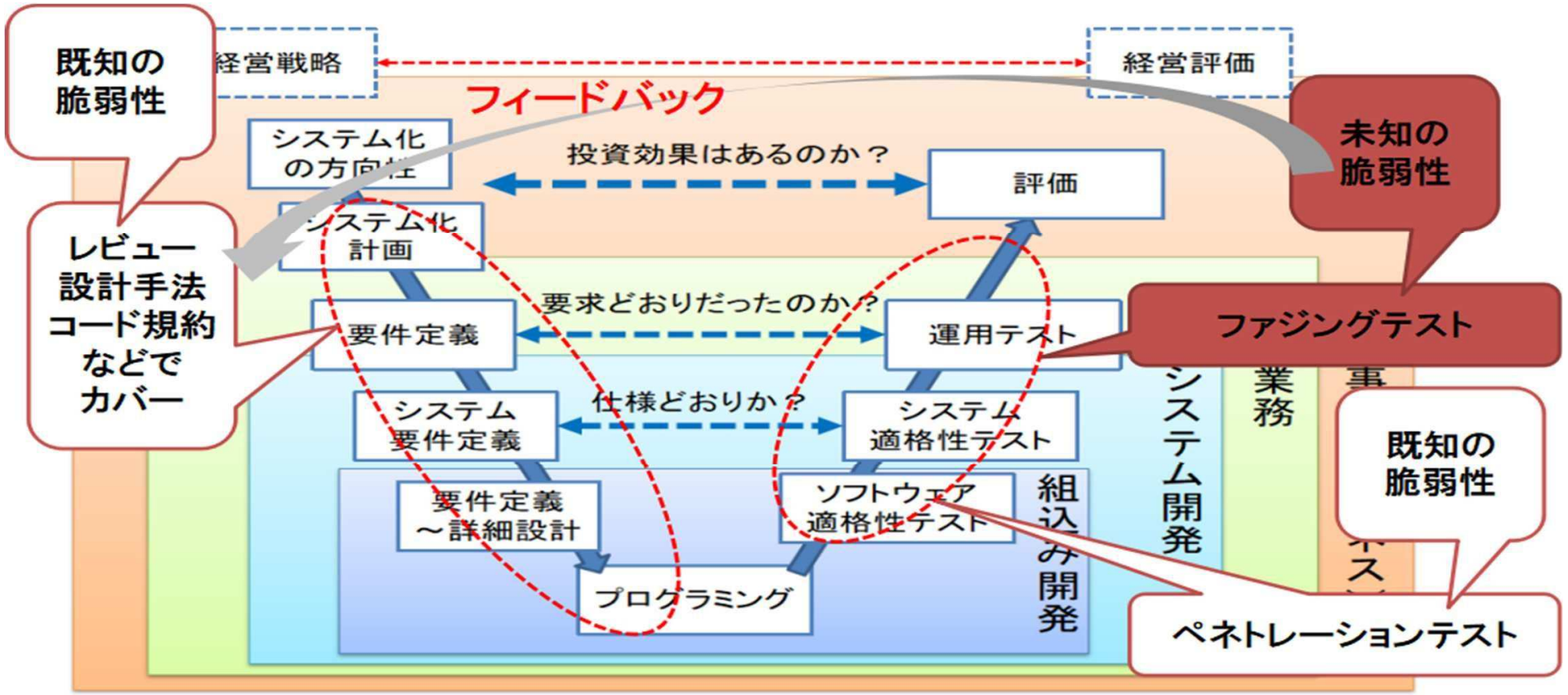
4. 18年度 成果まとめ

- STAMP/STPA-SECの分析を活用することでセキュリティ対策に有用である。
  - 課題点もあるが、設計に適用することが有用である。
- セキュリティ設計は、システム全体分析、製品開発プロセスにおける評価・証明が必要となる。
  - 開発プロセスの定義をセキュリティ設計として考える必要がある。
- セキュリティにおいては、認証・監視をするようなシステム導入をすることで、運用での脆弱性対策になる。

19年度 活動計画

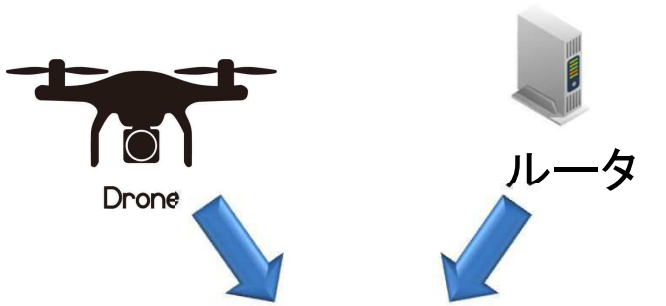
5. 19年度活動計画

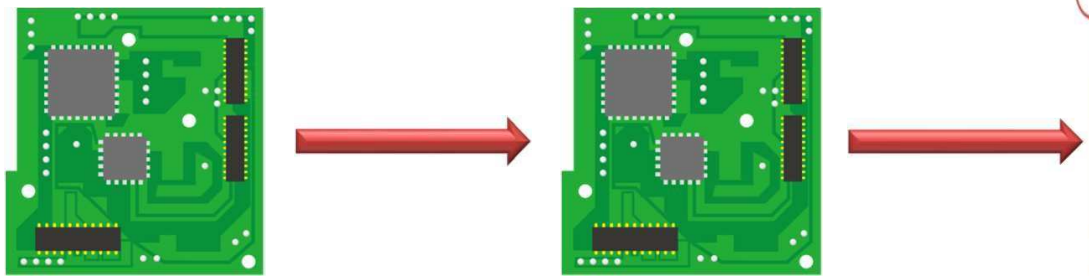
■ 実際のデバイスを使ったセキュリティに対する分析・検討を行い、セキュリティ設計のプロセス定義を実施する。



5. 19年度活動計画

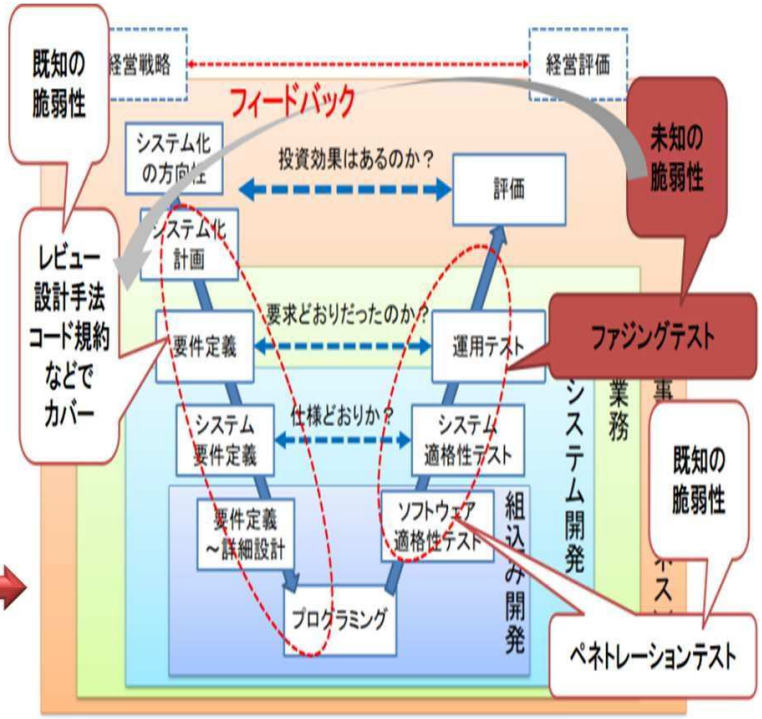
■ イメージ





ハードウェアの
フィジカルな面
での分析検討
（基板回路も
分析対象）

ソフトウェアを
脆弱性診断して
分析検討
ファジング、脆弱性
診断ツールを選定適用

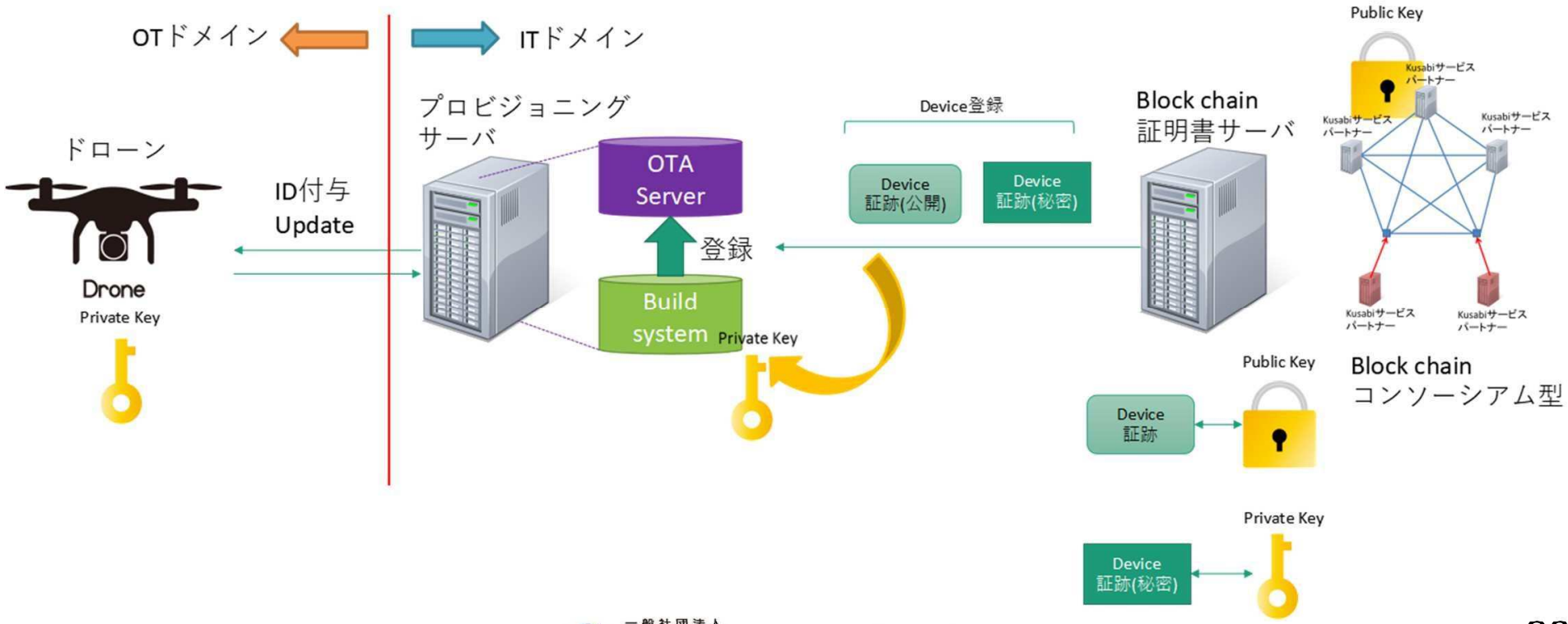


開発プロセスにフィードバックして
セキュリティ設計の定義を行う。

5. 19年度活動計画

■ 簡易PKIシステムの応用事例

- ドローンWGとの連携を図り、ドローンの認証・監視に応用することができること実証実験を行う。
  - ー システムの自動番号付与できる機能を使って、ナンバープレートとして利用できるか？の実証実験を実施予定。



【講演タイトル】組込み機器開発におけるセキュリティの考え方

2019/5/20 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会（以下、JASA）が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

アジャイル研究会活動報告

2019年5月20日

JASA中部　アジャイル研究会

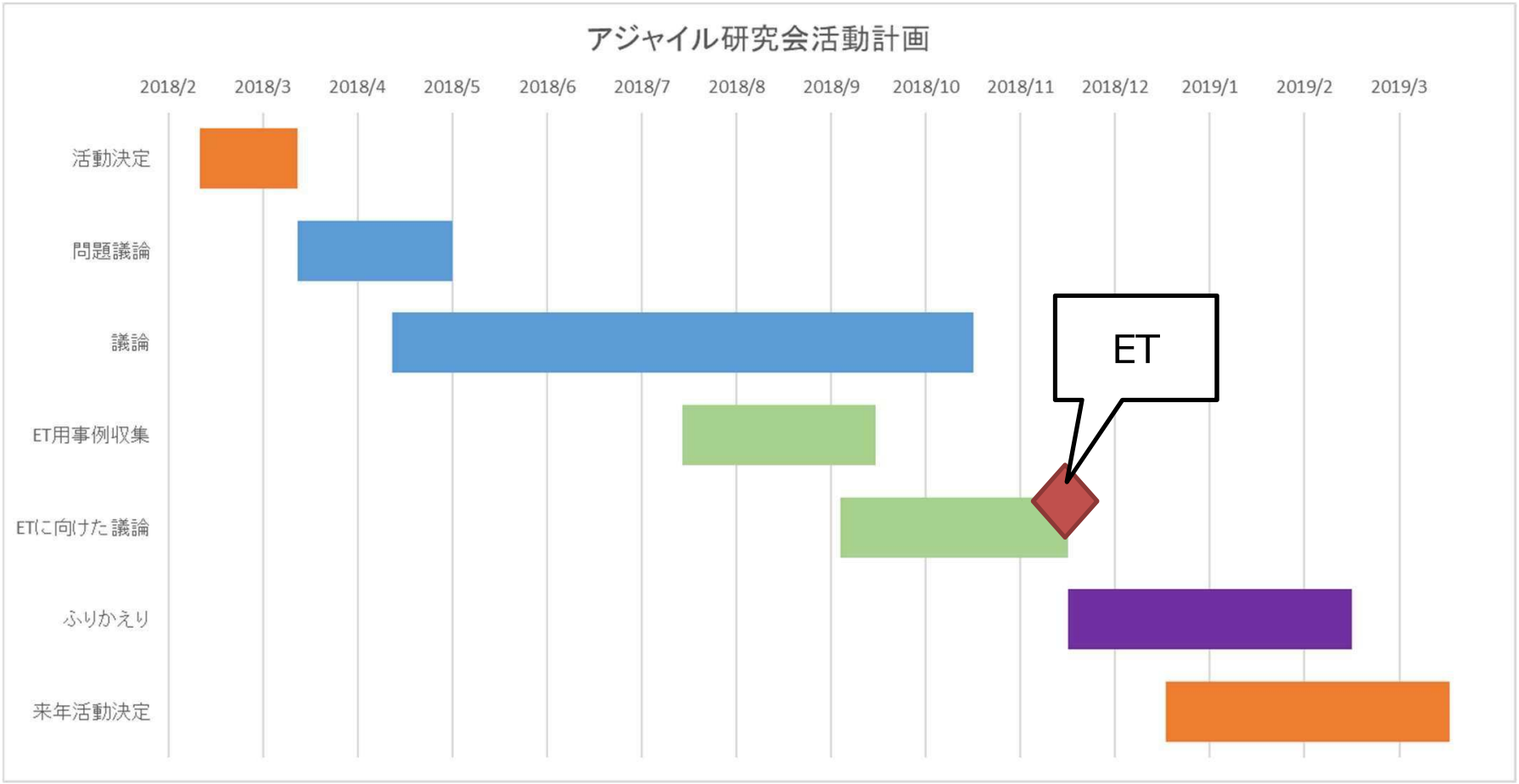
秋谷　勤

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

Agenda

1. スケジュール
2. 活動内容まとめ
3. アンケート結果

スケジュール

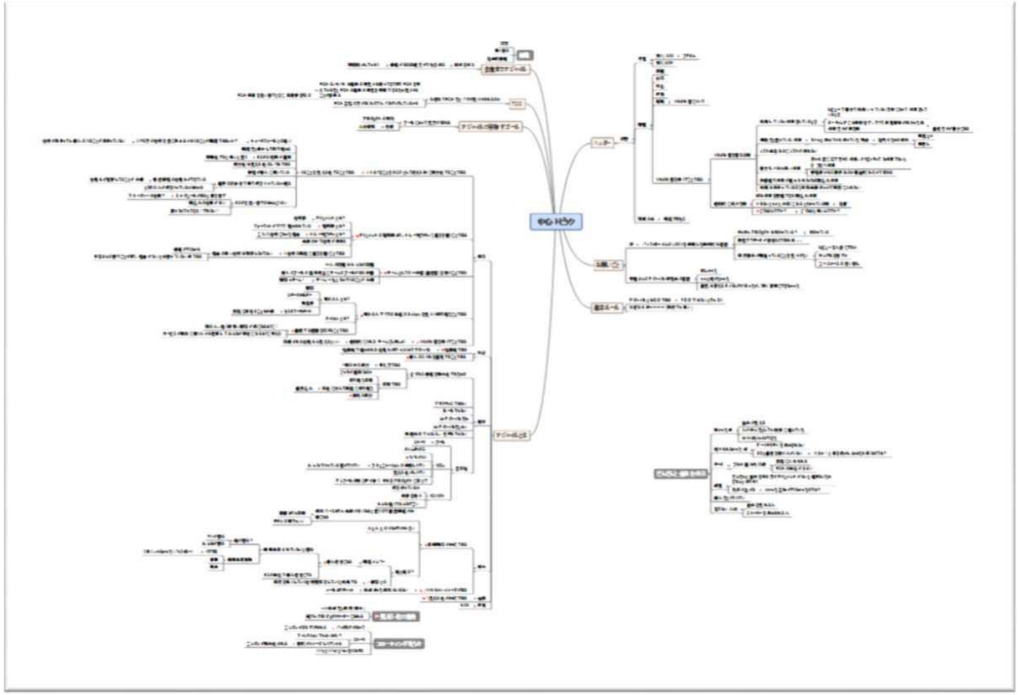

アジャイル研究会活動計画
2018/2
2018/3
2018/4
2018/5
2018/6
2018/7
2018/8
2018/9
2018/10
2018/11
2018/12
2019/1
2019/2
2019/3
活動決定
問題議論
議論
ET用事例収集
ETに向けた議論
ふりかえり
来年活動決定
ET

活動内容まとめ

アジャイルで変える意識と開発方法

アジャイルをやってみての感想

- 今年は「アジャイルとは」を考え続ける年としました。
- 過去のアジャイル研究会で出た話を振り返り、もう一度アジャイルな考えをまとめました。



アジャイルで変わること



■ だんだんとアジャイルな意識になってくる。
- ユーザー視点の考え方
  - ー 価値がある機能
  - ー 価値がある仕組み
- 頭だけで考えるのではなく見えるものでの議論
  - ー 動くコードだけではない
  - ー 話す土台となるもの

■ 全員が全体を見るようになる
- チームとして動く
- 個人個人で動き始める

■ 責任の押し付け合いが無くなる
- 前向きな思い

実例

アジャイルな考え方を行うメンバーが中心でプロジェクトを進めた結果

プロジェクト背景

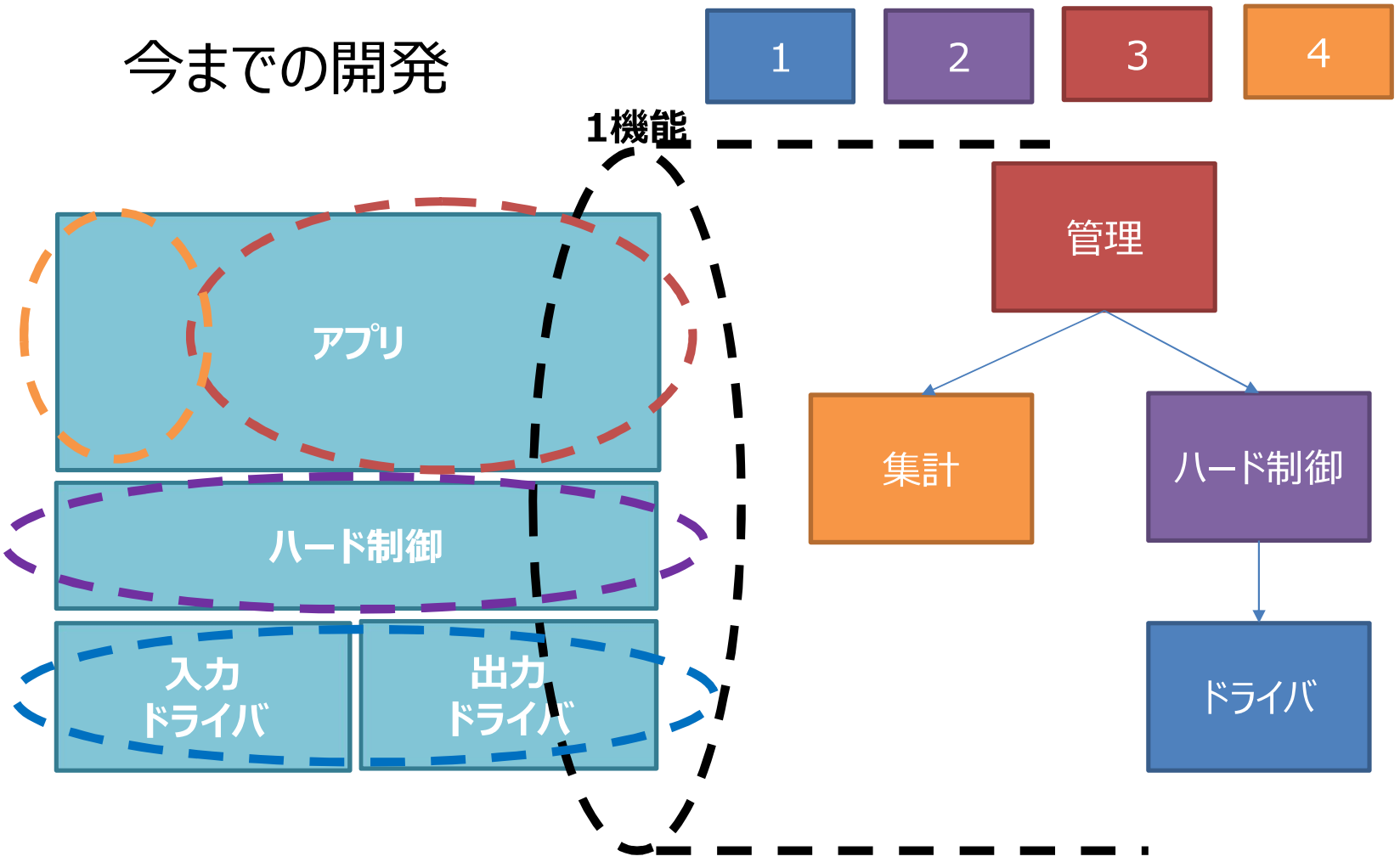


■ チーム構成
- 同じ拠点で7人
  - 10年以上　1人
  - 5年以上　1人
  - 5年未満　5人

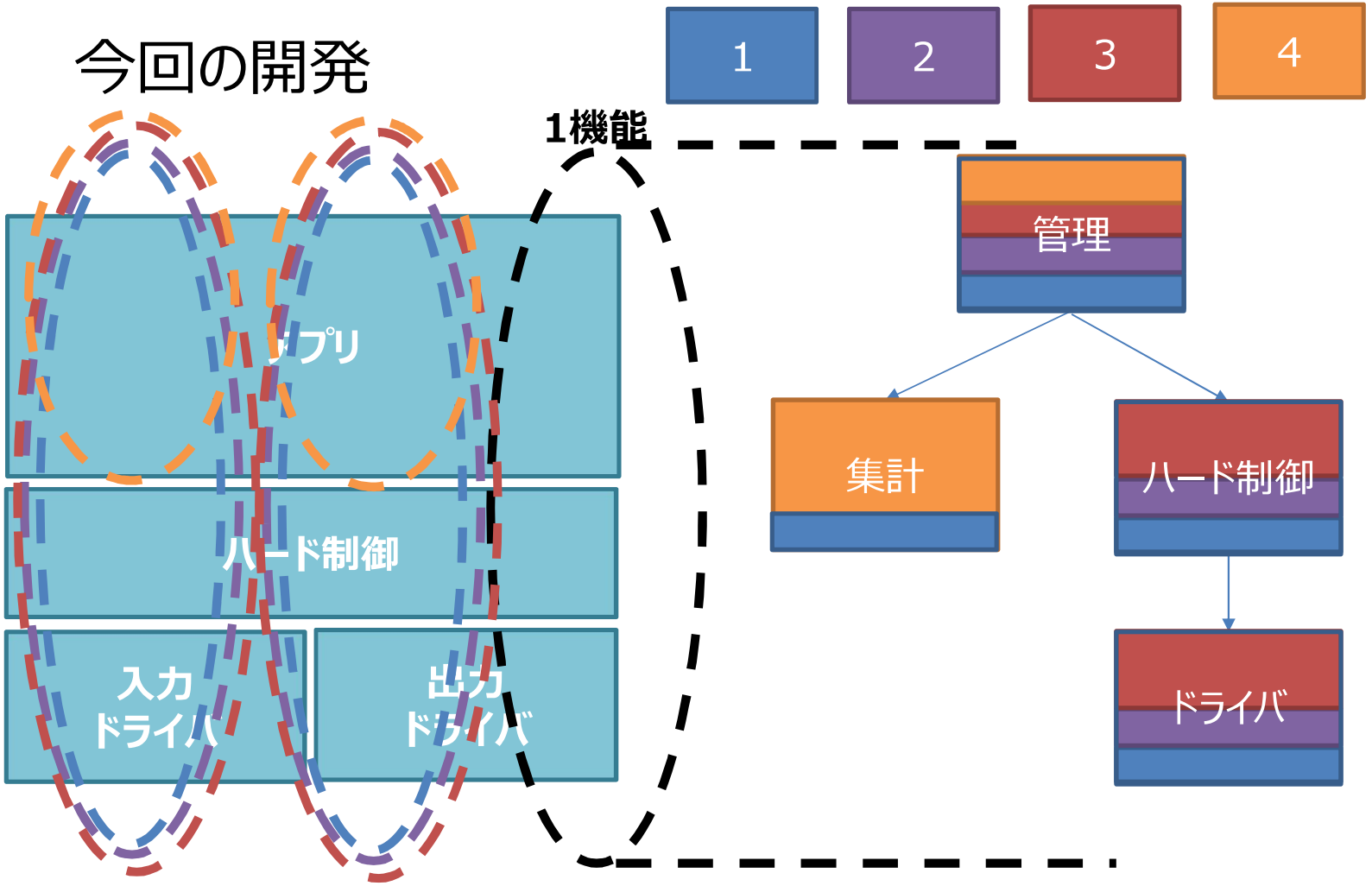
■ アジャイル？
- 完全なタイムボックスではない
  - 同じ周期ではない
- 1部自動テスト
- テストは先に考える

■ 問題点
- ふりかえりが効果的でない

大きく変えた点

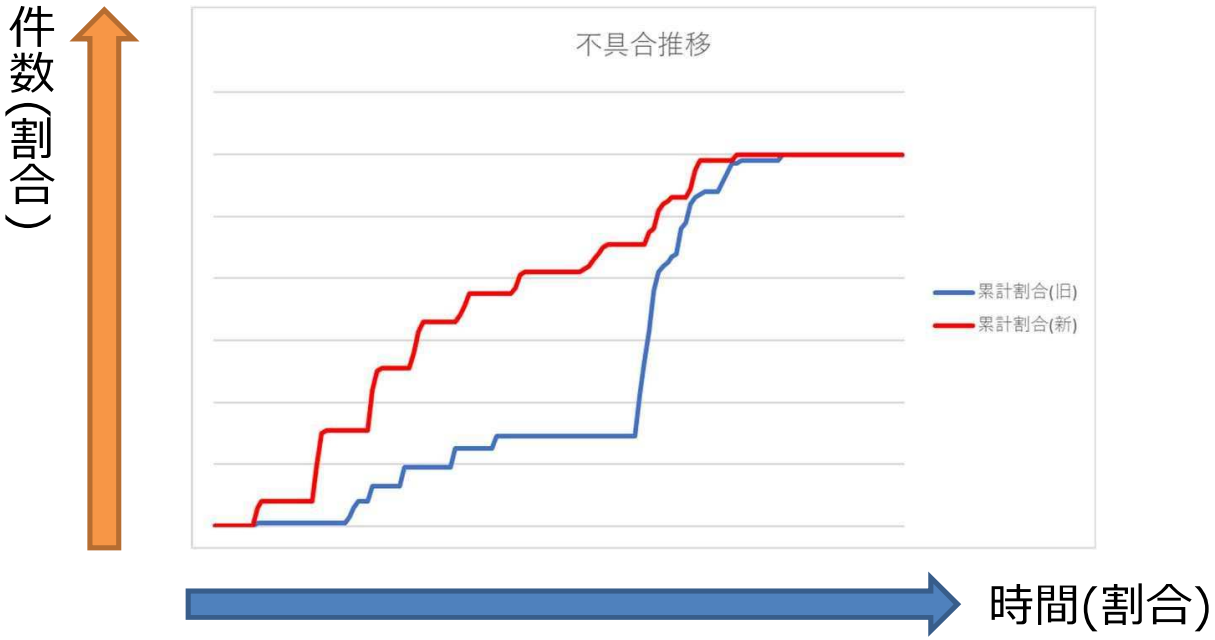


大きく変えた点



以前行った似た業務との比較データ

- コミット回数(チーム)　約20倍
  - 一人で抱え込んでいる時間が少ない
- バグ発見ピーク　前倒し
  - 早い段階で有効なテストができる

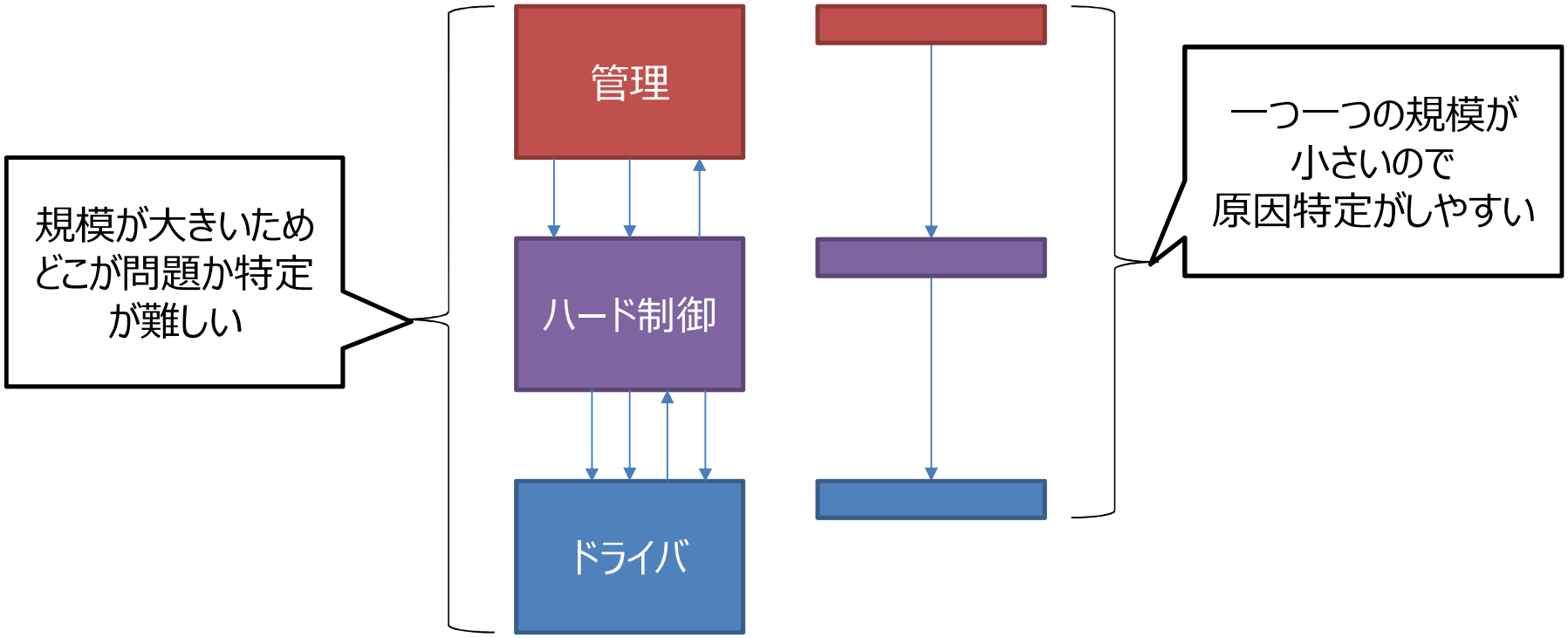


一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association
アジャイル研究会

以前行った似た業務との比較データ

■ バグ修正速度　約20%アップ

- 結合が小規模のため原因の特定が早くなった効果



以前行った似た業務との比較データ



■ 残業時間　30%削減
- バグ修正が早い
- バク発見ピークの前倒しもあり、
  納期間近のトラブルが少ない

■ 作業時間　25%削減
- プロジェクト中でのふりかえりが効果的に働いた
- 作業時間の平均化により慌てず作業ができた

良かった点　※要因は別である可能性があります



- チーム一人だけが残業するということがなくなった
- 人が休んだとしても勝手に進んでいた
- はやく動くものができた
- 結合テストが段階的に行われていくので
  原因(Not根本原因)の特定がしやすい
- チームとしての活動が行えた
  - マネージャー以外もお互いがやっていることがわかった
  - いつもよりコミュニケーションが活発だった
    - 多くの人と
    - 多くの量を
- 顧客も含め楽しかった、いっぱい笑った

悪かった点

■　開発方法の変更によるもの特になし

特になし！



でも、続かなかった・・・

続けられなかった要因



■ 難易度が高い

- 全体を見る
- 顧客から“良い物”の定義を引き出す
  - ストーリーを決める
- アーキテクチャ設計
- タスクの完了条件の定義

■ 同じように開発ができるように改善中

- 視点のガイドワードを作成し、”良い物“を聞き出せるように
- タスクの完了条件を決めれるよう、検証基準を常に意識する

ET＆ET NAGOYA
アンケート結果

アジャイル研究会で独自にとったアンケート結果をご紹介します

【アジャイル研究会活動報告】

2019/5/20 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

組込みIoTモデリングWG活動紹介
～IoT時代のビジネスモデルは、”モデル”で考えよう！～

2019年5月20日

組込みIoTモデリングWG
渡辺　博之

目次

- WGの紹介
- これまでの活動振り返り（～2018年5月）
- STEP1の活動報告（2018年6月～2018年11月）
- STEP2の活動報告（2018年12月～2019年5月）
- 今後の活動予定

組込みIoTモデリングWGとは？

- 目的
  - IoT時代に必要とされる手法やモデルを明らかにし、その活用を促すとともに共有資産の創出を目指す
- 活動内容
  - モデルや手法の整理を進めつつ、並行してそれらを実際のIoTサービスに適用し、その有効性を検証する

①メインWG
- IoT時代に必要とされる手法やモデルの定義および整理を行う

②サブWG
- 実際のIoTサービスをモデリングし、①の成果を検証する

- 進め方
  - ① 本活動のスコープは広範囲に渡るため、メンバーはJASA会員に限定せず、関係する諸団体・知見者まで広げる
  - ② サブWGは、旧UMTP組込みモデリング部会のメンバーを中心に、モデリング主体の活動を進める

WGメンバー

ＷＧ主査	渡辺 博之	株式会社エクスモーション
ＷＧ副主査	芳村 美紀	株式会社エクスモーション
メインWGメンバー	熊谷 恒治	株式会社チェンジビジョン
メインWGメンバー	有馬 仁志	有馬マネジメントデザイン株式会社
メインWGメンバー	白坂 成功	慶應義塾大学
メインWGメンバー	光井 隆浩	株式会社東芝
メインWGメンバー	井上 樹	株式会社豆蔵
メインWGメンバー	田丸 喜一郎	独立行政法人情報処理推進機構
メインWGメンバー	齋藤 毅	独立行政法人情報処理推進機構
メインWGメンバー	川野 喜一	（一社）モデルベース思考研究所
メインWGメンバー	砂口 洋毅	NEDO
サブWGメンバー	庄司順和	株式会社エクスモーション
サブWGメンバー	大森淳夫	パイオニア株式会社
サブWGメンバー	松嶺恭守	理想科学工業株式会社
サブWGメンバー	鷲澤純平	株式会社クレスコ
サブWGメンバー	町田修宏	パイオニア株式会社
サブWGメンバー	石滝智洋	オリンパス株式会社
サブWGメンバー	清水鉄矢	オリンパス株式会社
サブWGメンバー	福島祐子	日本ユニシス株式会社
サブWGメンバー	新家了訪	株式会社テイジイエル
サブWGメンバー	川縁幸平	株式会社ゼネテック
サブWGメンバー	越智典子	個人事業主
サブWGメンバー	石井基樹	JVCケンウッド
サブWGメンバー	難波秀之	NEC通信システム

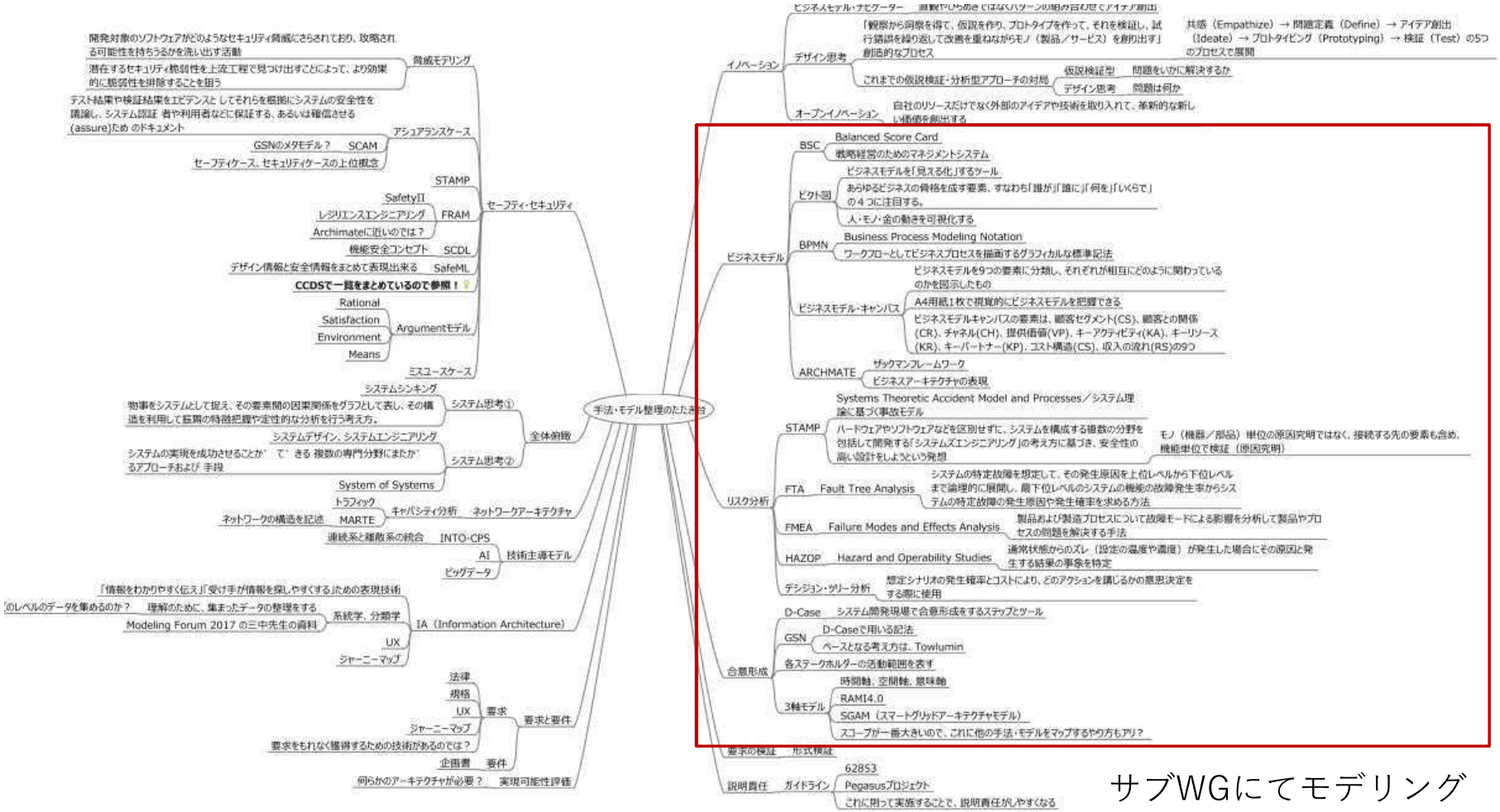
WGの活動内容

1. メインWGにて、モデリングや手法の洗い出しを実施し、さまざまなものをピックアップし、使い方や順序などを整理中
   - イノベーション
   - ビジネスモデル
   - リスク分析
   - 合意形成
   - 説明責任
   - セキュリティ/セーフティ
   - プライバシー/アシュアランス
2. サブWGにて、具体的なIoTサービスを題材に、WGでピックアップされたモデリング・手法を施行してみる
   - IoTサービスは、ビジネスとテクノロジーの融合であり、まずはビジネスに対するモデリング（＝ビジネスモデルへのモデル活用）から着手
3. モデリングの結果を踏まえ、各モデル・手法の使い方、活用方法等、実践からのプラクティスの導出を図る

目次

- WGの紹介
- これまでの活動振り返り（～2018年5月）
- STEP1の活動報告（2018年6月～2018年11月）
- STEP2の活動報告（2018年12月～2019年5月）
- 今後の活動予定

モデリングと手法のピックアップ

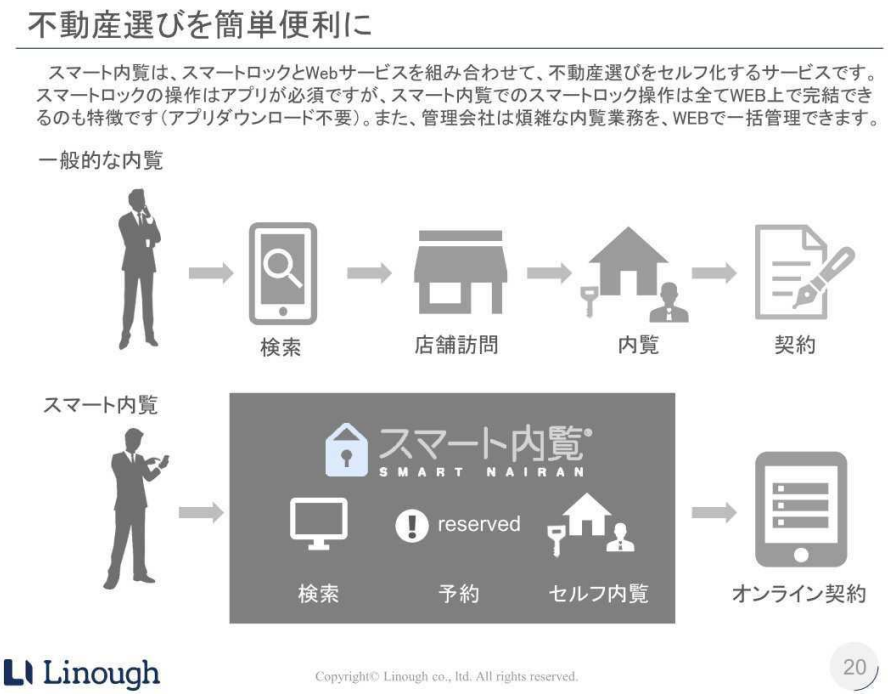


サブWGにてモデリングにトライ

IoTサービスのモデリング

1. 具体的なIoTサービスを題材に、メインWGでピックアップしたモデリング・手法を施行してみる
   - 今回は「スマート内覧」を題材として選択
2. モデリングの結果を踏まえ、各モデル・手法の使い方、効果的な作成順序等、実践を通してモデリングに関するプラクティス導出を図る
   - いくつかのモデルについて実施（詳細は次ページ以降）

題材：スマート内覧



当WGでは、「スマート内覧」を提供している株式会社ライナフ社からの使用許諾を得て、当WGでの題材として活用させていただいています。

「スマート内覧」とは、スマートフォンや携帯電話で開錠できるため、鍵の受け渡しや解錠・施錠に関する煩雑な手続きが不要になるとともに、内覧者にとっても、時間制約がなく自分のペースで内覧が可能になる等、新しい内覧を可能にするサービスです。

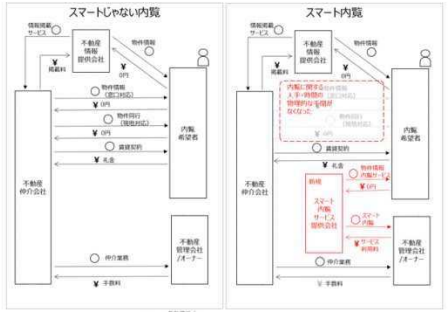


成果物一覧

カテゴリ	モデル名	概要
ビジネス	ピクト図解	ヒト・モノ・カネの間の関係性を時系列で整理し、ビジネスモデルを見える化する図解方法
	IoTビジネス キャンバス	ビジネス・モデル・キャンバスとIoTキャンバスを合成し、IoTビジネスにおける価値・コストというビジネス側面と、それを実現するソリューション側面の双方を一つのチャートで表現したもの
	ユーザストーリー	システムの具体的な流れをストーリーとして記載したもの 具体的な要求を把握するためや、検証時のテストシナリオの元になる
	アクティビティ図	SysML/UMLで使われる、ステークホルダやシステム間の時系列でのやり取りを記載したもの
合意形成	D-case	システムのディペンダビリティ（信頼性）を関係者間で共有し互いに分かり合い、社会の人々にわかってもらい、 説明責任を果たすための手法とツール

得られた知見（各モデルの役割と効果的な作成手順）

①ピクト図で、ビジネスの外側を可視化



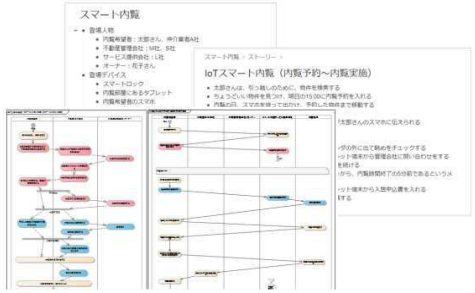
ステークホルダーとその役割を明確化するとともに、人・モノ・カネの流れを可視化
意外なところからお金が発生する等の発見がある（特に社会性のあるビジネス等）

③IoTビジネスキャンバスで、ビジネスの内側と、それに必要な技術的要素を可視化

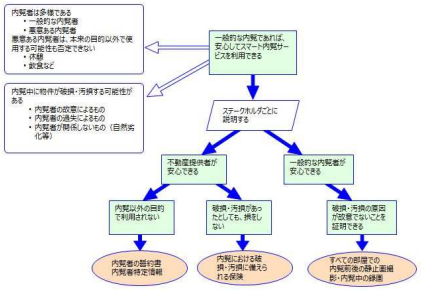


ビジネスを実現するために必要な要素を、予め用意されているフォーマットに則って抽出
周辺がビジネス、中央が技術で、特にデータに着目することで、新たなサービスを発見可能

②アクティビティ図で、詳細に見たいやり取りを時系列で可視化



④ビジネス上のリスクや、サービス・製品の課題に対する対応を可視化する



11

活動の振り返りと課題

- IoT時代のビジネスを、モデルで可視化することの有用性を確認
  - IoT時代のビジネスは、範囲が広く、 利害関係者が多く、 内容（技術、マネタイズ）が複雑

  ⇒　モデルで“可視化”して議論することが重要

  ⇒　STAMP/STPAについては、時間がなく未実施（課題2）

- ただし、今回のモデルは、いずれもビジネスのアイディアが生み出された後の段階（＝ビジネス領域）で活用されるもの
  - アイディアを生み出す領域（＝デザイン領域）のメジャーな手法はデザイン思考だが、そこにモデルは出てこない…
  - 新しいビジネスを生み出すためのモデルはあるのか？それは有用か？

  ⇒　デザイン領域で使えるモデルの模索（課題1）

目次

- WGの紹介
- これまでの活動振り返り（～2018年5月）
- STEP1の活動報告（2018年6月～2018年11月）
- STEP2の活動報告（2018年12月～2019年4月）
- 今後の活動予定

活動内容

- 2018年4月までの活動で明らかになった課題への対応
  - 課題①
    - ビジネスのアイディアを生み出す「デザイン領域」で活用できるモデルの調査とその試行
  - 課題②
    - 「ビジネス領域」におけるSTAMP/STPAの試行
- これらの課題を検証するため、以下の2ステップで活動を推進
  - STEP1
    - デザイン領域で、新たなアイディアを創出するためのモデルを調査
    - これまでの「スマート内覧」を題材に、①と②を含めて新たにモデリングをやり直し、その有用性を検証
  - STEP2
    - ①の有用性を正しく検証するために、既存のビジネスではなく、全く新しいビジネスアイディアを創出する段階から、これらのモデルを使って考える活動を実践

STEP1での成果物一覧

- デザイン領域のモデルとして「ジョブ理論」を試行
- IoTビジネスキャンバスの代わりに、本来のビジネスモデルキャンバスとIoTキャンバスを個々に実施

カテゴリ	モデル名	概要
デザイン	ジョブ理論	顧客が商品・サービスを購入する際の背景にあるメカニズム
ビジネス	ビジネスモデルキャンバス	ビジネスの構造を考えるためのツール
	ピクト図解	ヒト・モノ・カネの間の関係性を整理し、ビジネスモデルを可視化する記法
	IoTキャンバス	IoTビジネスに関する初期のアイディアを創造するためのツール
リスク分析	STAMP/STPA	「相互作用する機能単位でハザード要因を考える」という安全性解析手法

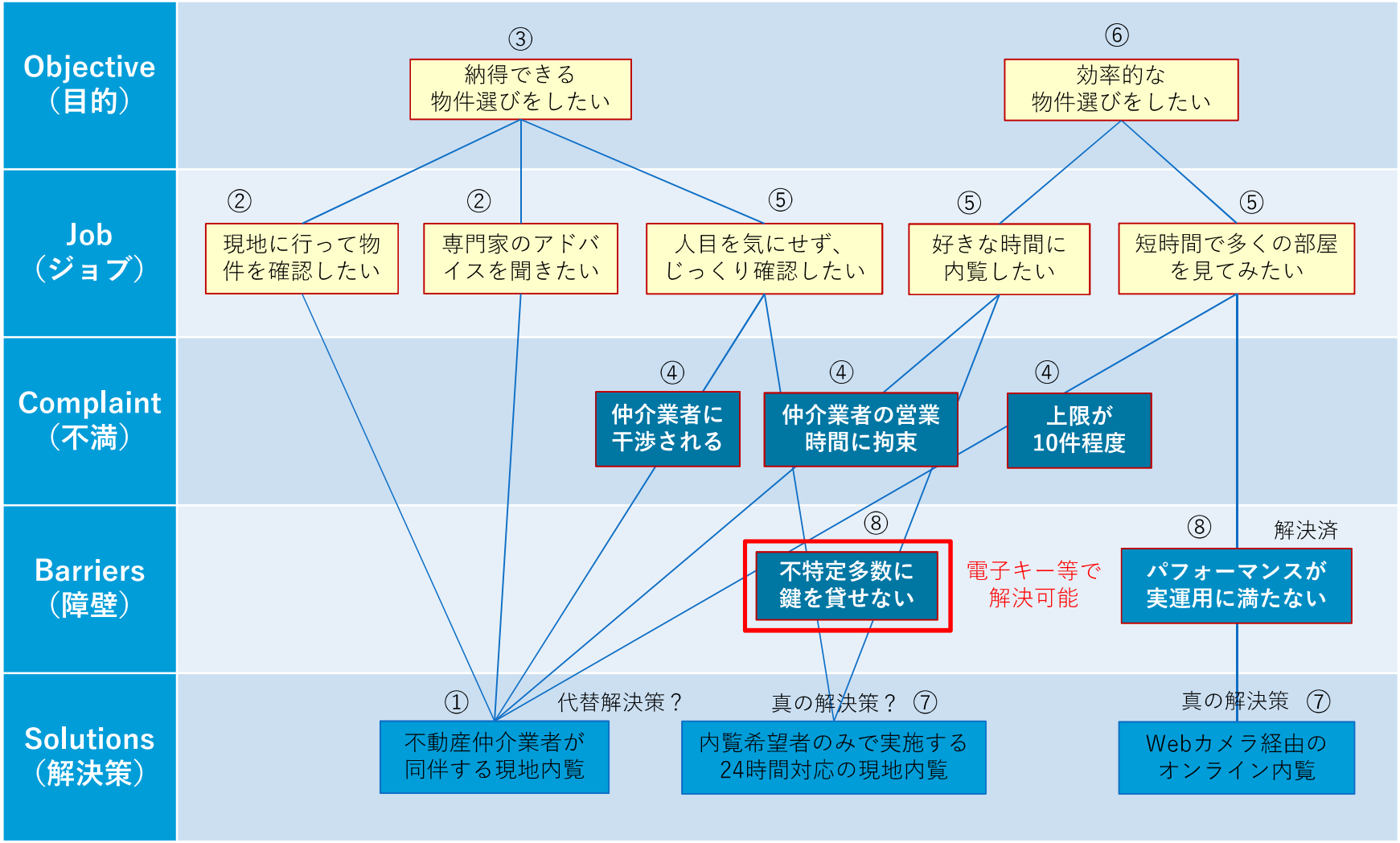
ジョブ理論

- 顧客が商品・サービスを購入する際の背景にあるメカニズム
  - 「課題」と、それに見合った「価値提案」を行き当たりばったりではなくロジカルに導出するための考え方
- 「ジョブ」⇒　片付けるべき用事（＝課題）
  - 「私たちが商品を買うということは、基本的になんらかの“ジョブ”（＝課題）を片付けるために、何かを“雇用する”（＝価値提案を受け入れる）ということである」
  - 解決策が存在しないか、現状では満たされていない「新たなジョブ」を見つけることが、ビジネスアイディアを発見する糸口
- 但し、ジョブ理論自体は、モデルではなく考え方のガイドライン

  ⇒　理解性や使い勝手を上げるために、モデル化にトライ

JOB理論モデル

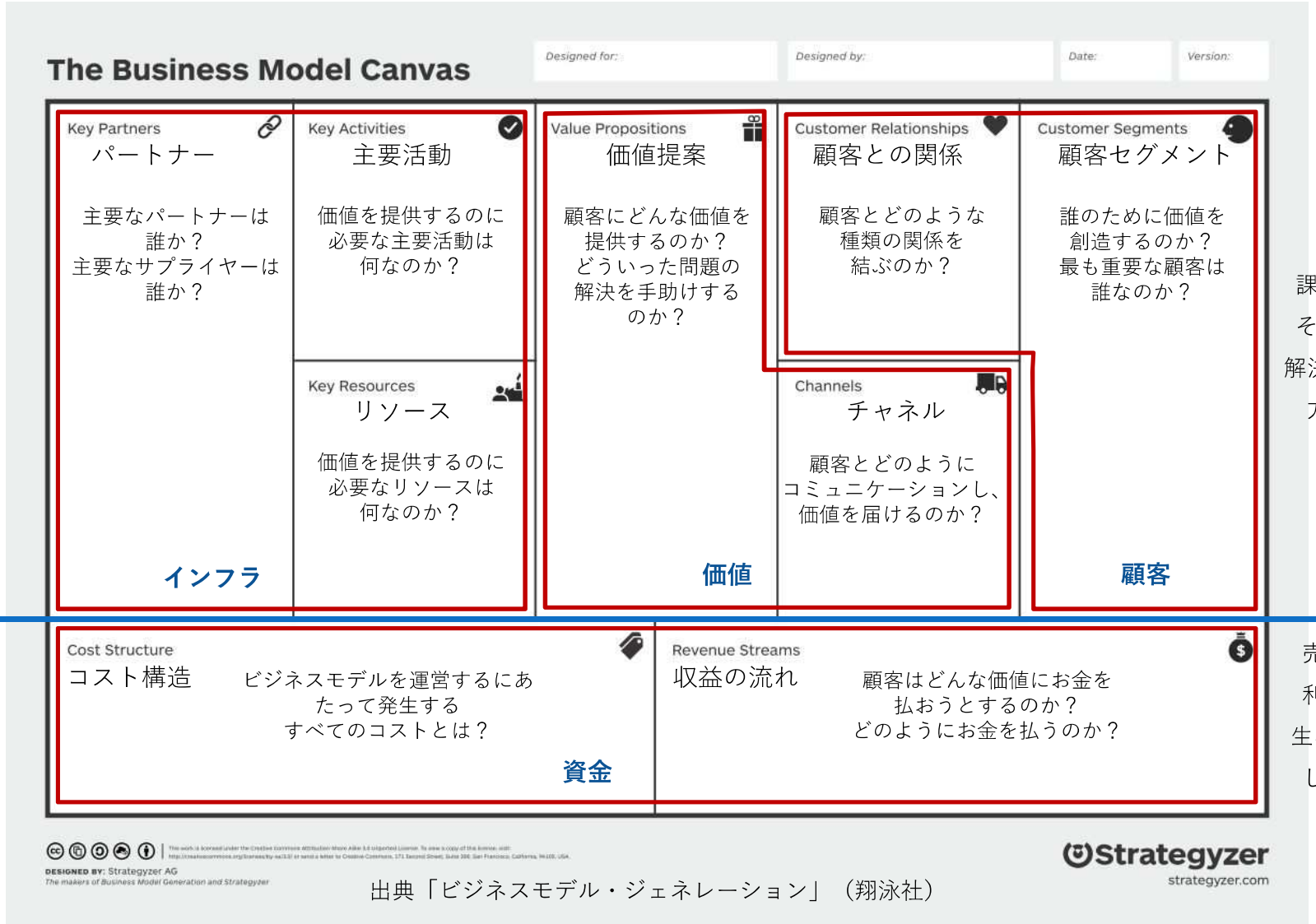
①～⑧までの手順に沿うことで、
ジョブの発見と解決策の導出を支援



ビジネスモデルキャンバス

- ビジネスの構造を考えるためのツール
- ビジネスモデルを考える際に必要となる４つの領域と９つの要素から構成される
  - ビジネスを俯瞰、検証し、検討の抜け漏れを防ぐ

ビジネスモデルキャンバス

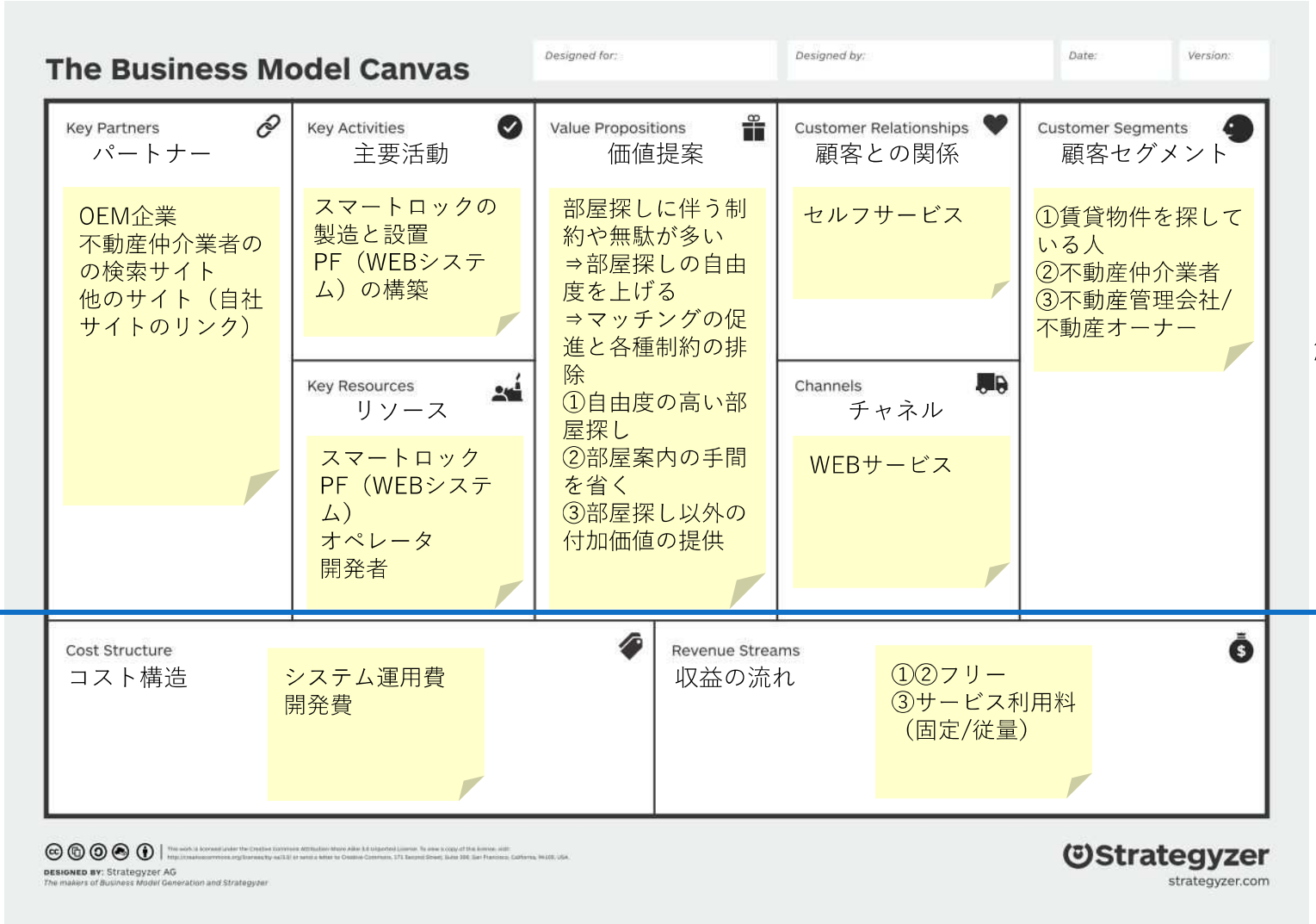


出典「ビジネスモデル・ジェネレーション」（翔泳社）

19

ビジネスモデルキャンバス

顧客毎に異なる価値を提供できることや、
スマートロックの提供以外に、プラットフォーム構築の重要性がわかる



課題と
それを
解決する
方法

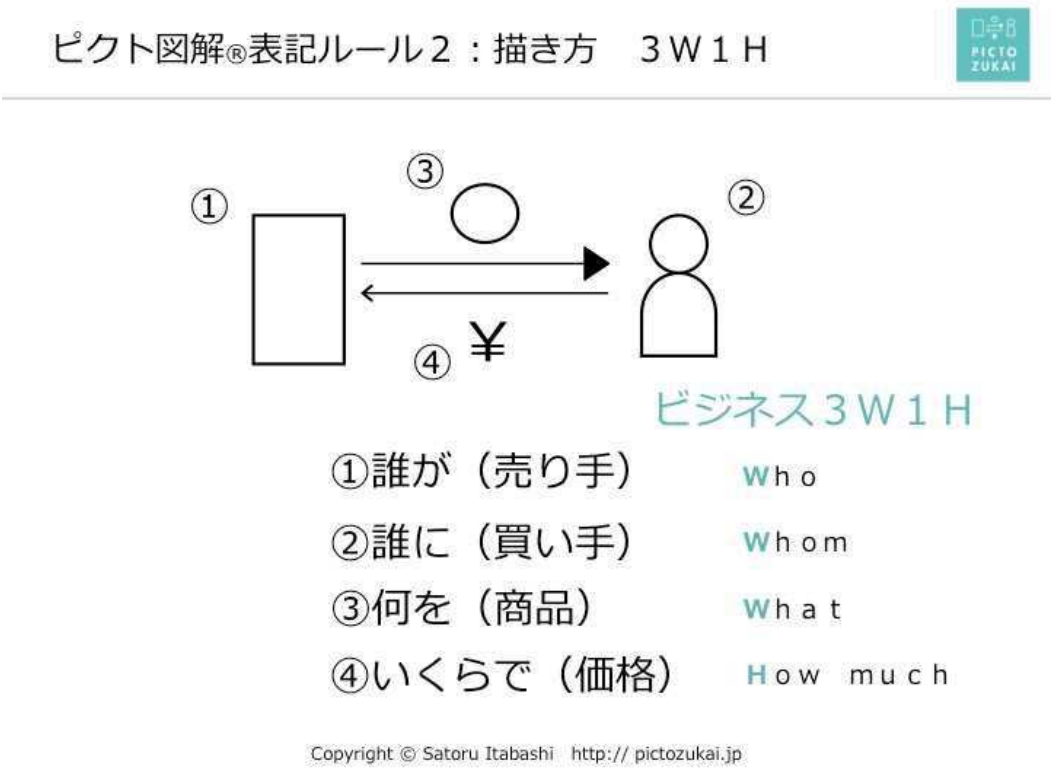
売上や
利益を
生み出す
しくみ

20

ピクト図解

- ビジネスを外部から考えるためのツール
- ビジネスに必須な「ヒト」「モノ」「カネ」に着目したモデル

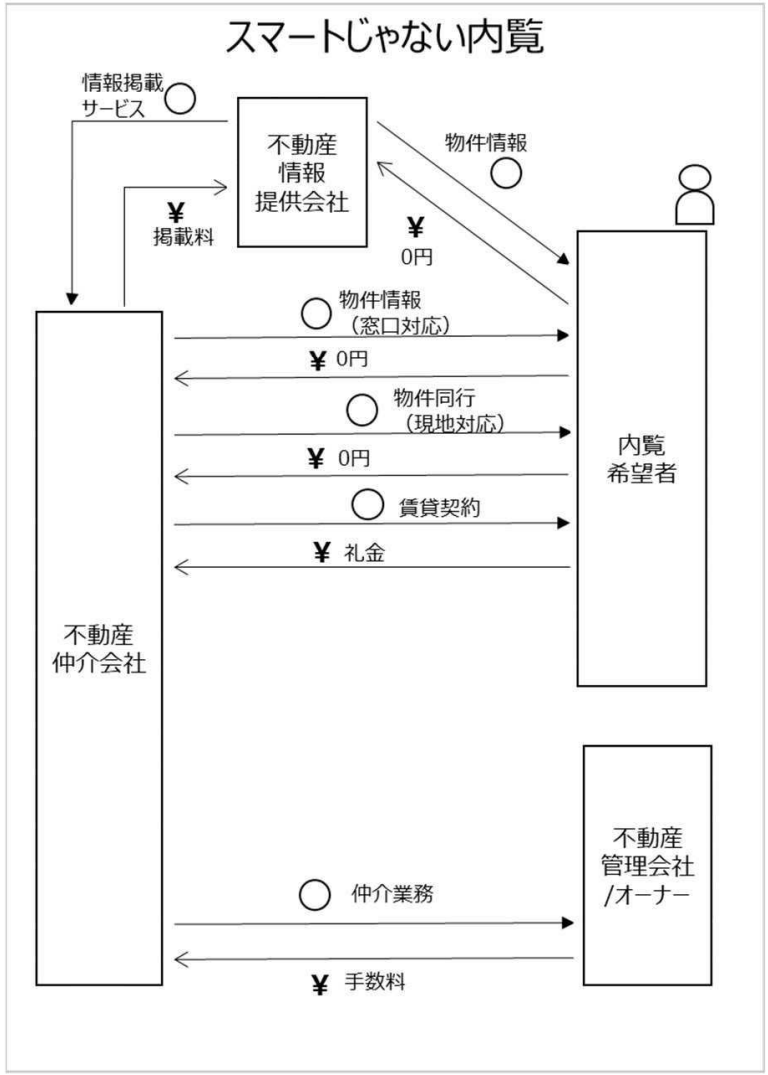
ピクト図解

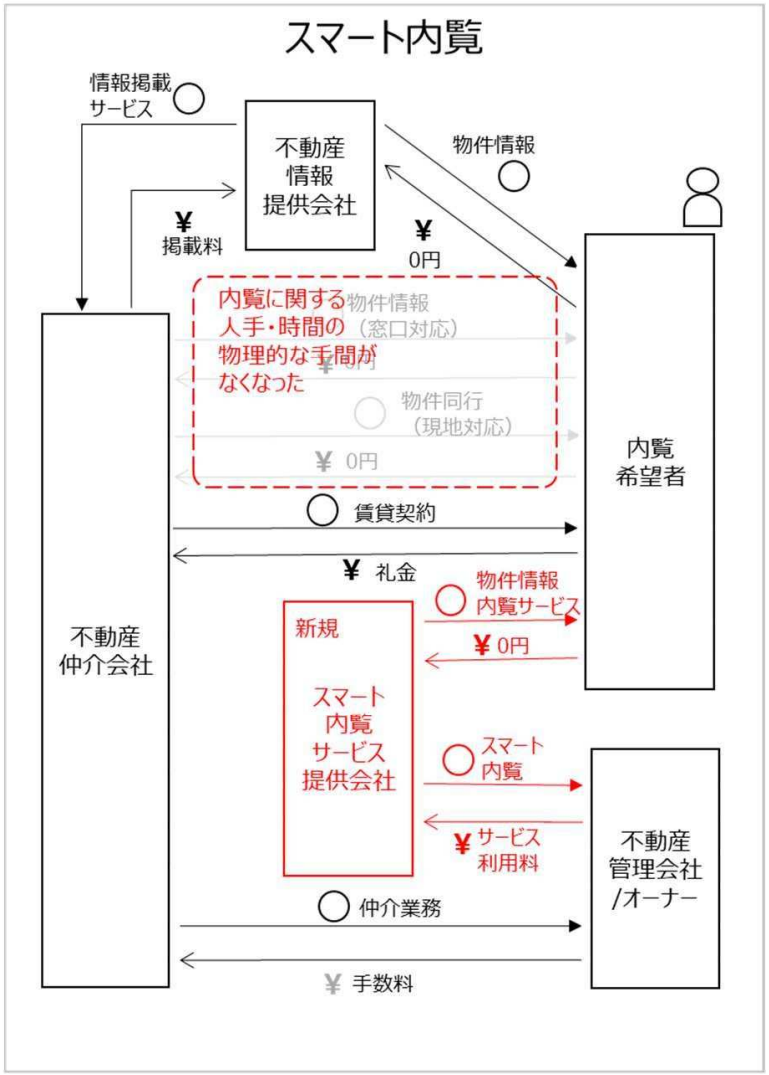


出典 PICTO ZUKAI
http://pictozukai.jp/philosophy/

ピクト図解

「スマート内覧」に変わることで、人手・時間がかかる物理的な手間を減らせる
少子高齢化により働き手が減る将来に対して、有効なサービスであることが明確に



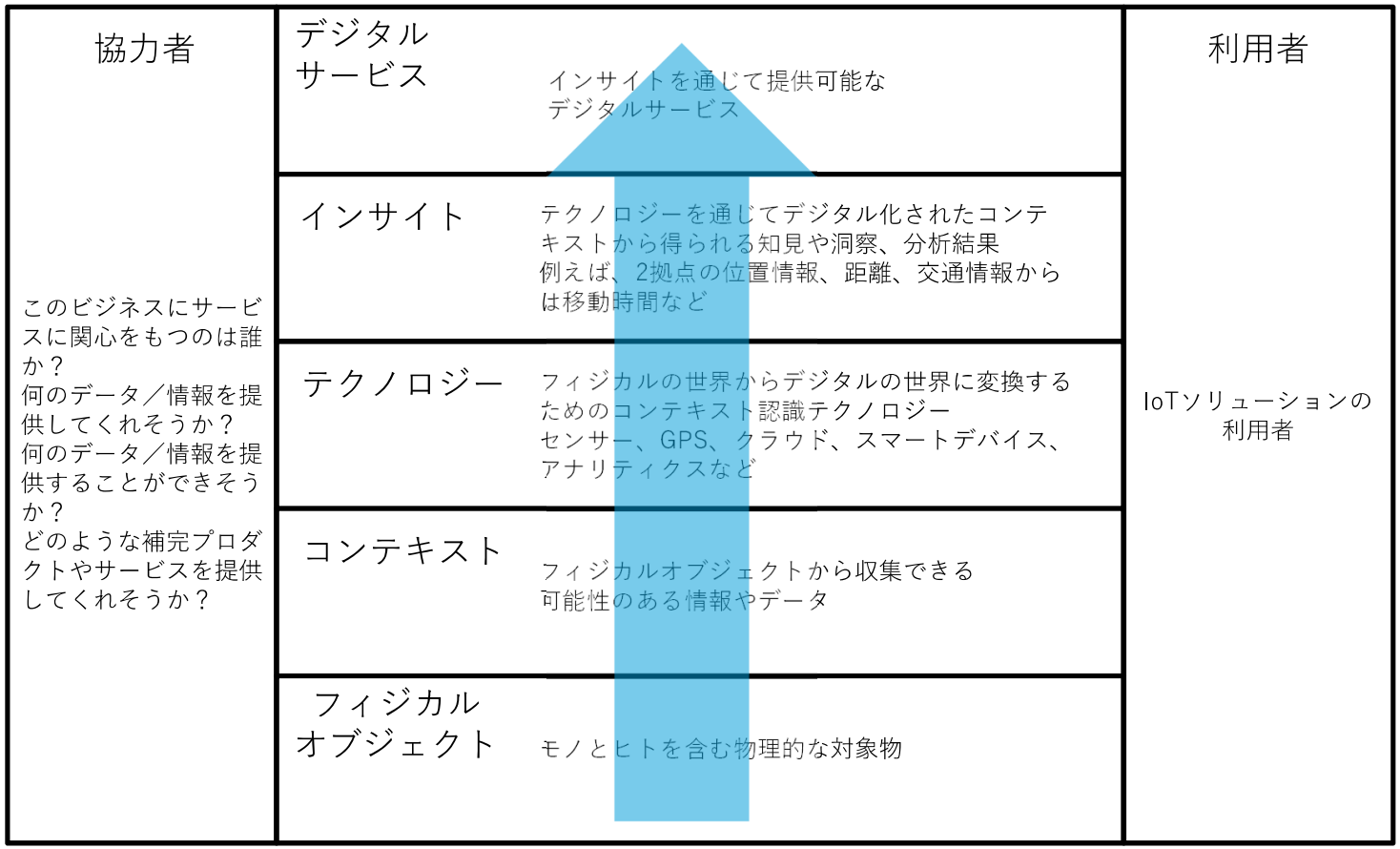


23

IoTキャンバス

- IoTビジネスに関する初期のアイディアを創造するためのツール
- “ソリューション”から “価値提案”を導出する
- つまり、価値提案をテクノロジー視点からボトムアップに導出するアプローチ
  ＝エンジニアでも取り組みやすい！

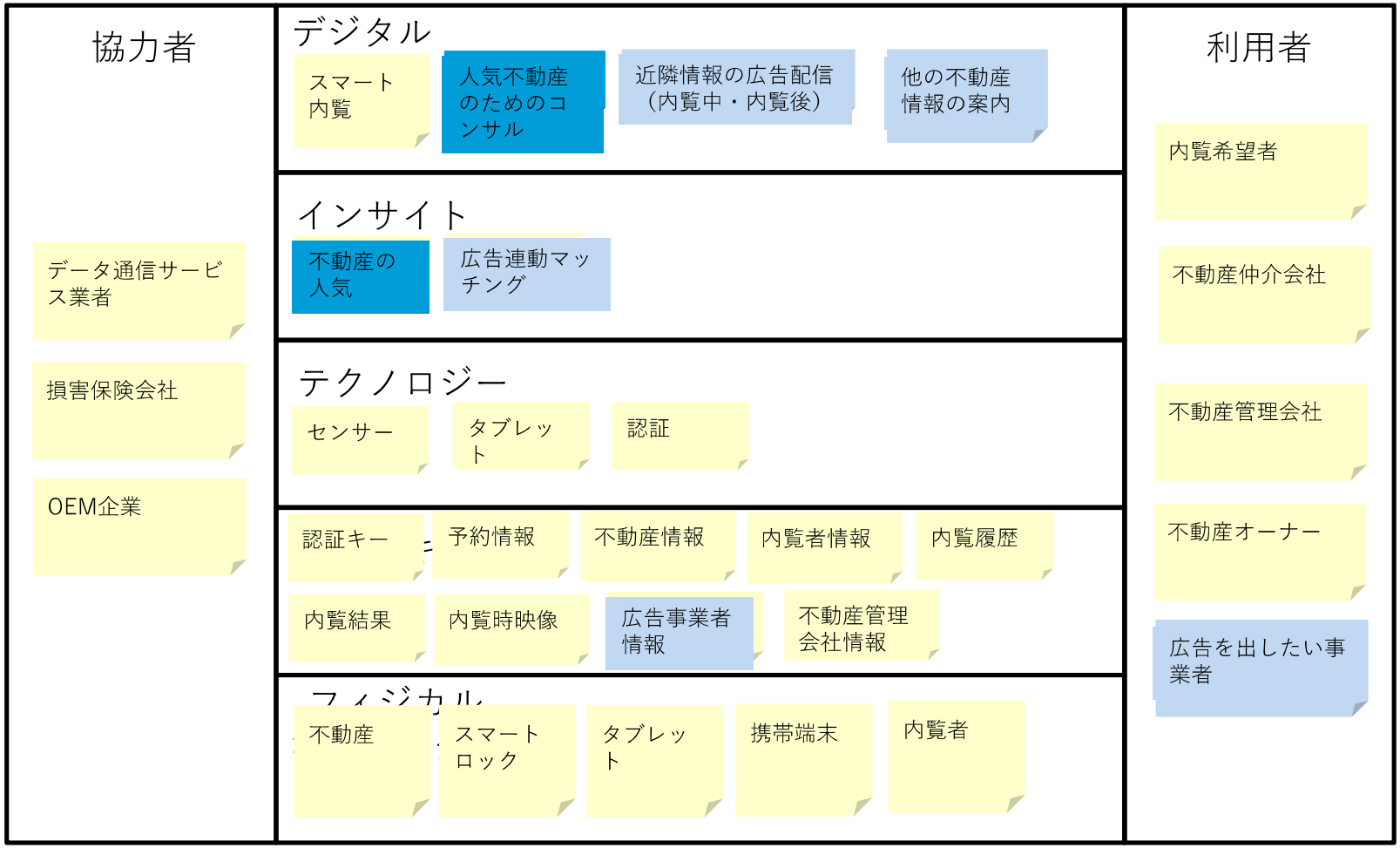
IoTキャンバス



出典 「本格化するIoTのビジネスモデルをデザインする方法
－IoTキャンバスとは？」
https://bizzine.jp/article/detail/1295

IoTキャンバス

入力情報を契機としたサービスを想起しやすい
但し、事前の課題設定が必要



STAMP/STPA

- システムを対象とした、“相互作用する機能単位”で
  ハザード要因を考える安全性解析手法
- 構成要素単体では問題なくても、それらが相互作用する
  ことで発生する問題をガイドワードを使って網羅的に導出
- 構成要素の粒度に応じた構築が可能なので、粒度が粗い
  ビジネスモデルの段階でも適用が可能

STAMP/STPA（1/2）

ビジネスレベルにおける損失
- ・機会の損失　⇒内覧機会の損失 等
- ・経済的損失　⇒ステークホルダーの倒産 等
- ・評判の損失　⇒内覧サービス・ステークホルダーの評判の悪化 等

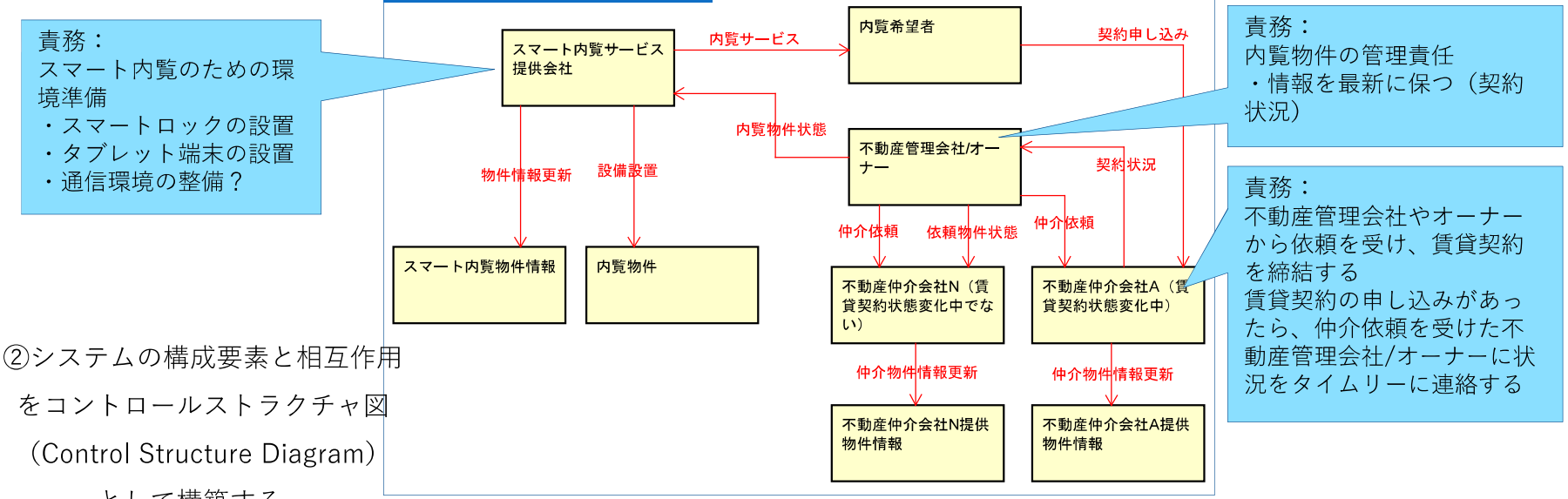
①発生する損失に関するハザードとハザードを制御するための安全制約を識別する。

損失：内覧機会の損失（⇒賃貸契約に結び付かない）

ハザード　：内覧サービスが提供されない

安全制約　：内覧サービスは提供されなければならない
：内覧サービスが提供されていないときには、内覧サービスを提供する対策が取られなければならない

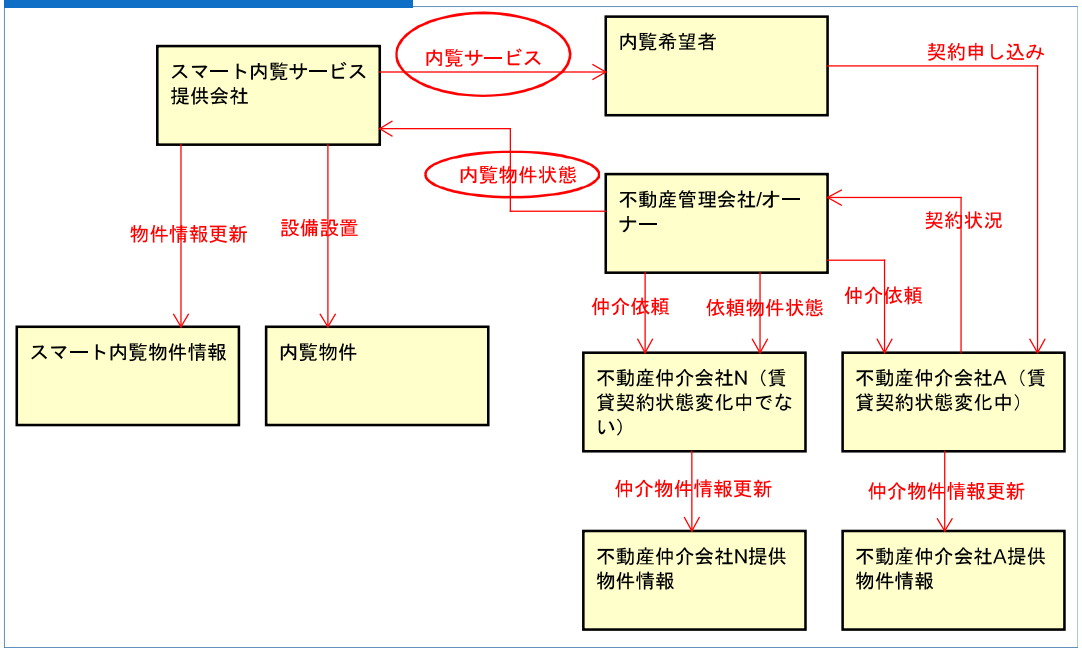
コントロールストラクチャ　要素はピクト図のステークホルダーから抽出



②システムの構成要素と相互作用をコントロールストラクチャ図（Control Structure Diagram）として構築する。

STAMP/STPA（2/2）

コントロールストラクチャ



ビジネスモデルの段階で、
論理的に
リスク（ハザード要因）
の導出が可能

③安全制約の実行に必要なコマンドに対してガイドワードを適用して、ハザードを起こし得る非安全なコントロールアクション（UCA：Unsafe Control Action）を抽出する。

ガイドワード

No	CA	From	To	Not Providing	Too early / Too late
1	内覧サービス	スマート内覧 サービス提供会社	内覧希望者	(UCA1-N-1) 部屋に設備が設置されていない	(UCA1-T-1) 他で契約済みの物件の内覧サービスを提供する (UCA1-T-2) まだ退去していない部屋の内覧サービスを提供する
5	内覧物件状態	不動産管理会社/ オーナー	スマート内覧 サービス提供会社	(UCA5-N-1) 連絡を怠った	(UCA5-T-1) 内覧物件の状態変化を連絡するのが遅い (UCA5-T-2) 稼働時間の違いにより、情報更新がタイムリーにできない

ビジネスモデルの検討ステップと各モデルのカバー領域

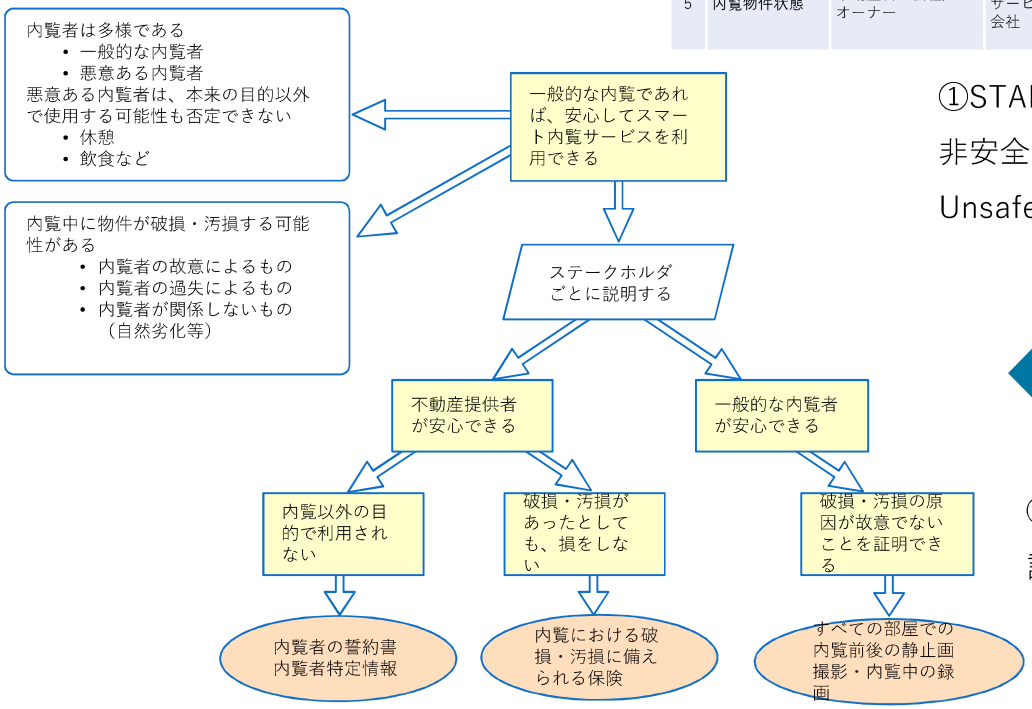
ビジネスモデルキャンバスは、全体を網羅
それ以外のモデルは、特定ステップの深耕

	ビジネスモデルキャンバス	ジョブ理論	ピクト図解	IoTキャンバス
①課題設定 “誰のどんな課題を解決したいのか？”	○	◎		
②価値提供 “そのためには、誰に何を提供すればよいのか？”	○	◎		
③ソリューション “それは、どのように実現するのか？”	○			◎
④マネタイズ “それは、事業として成立するか？”	○		◎	

リスク分析とその対処方法

STAMP/STPAでリスクを分析し、
各リスクに対してどう対処するかをD-Caseで検討？

No	CA	From	To	Not Providing	Too early / Too late
1	内覧サービス	スマート内覧サービス提供会社	内覧希望者	(UCA1-N-1) 部屋に設備が設置されていない	(UCA1-T-1) 他で契約済みの物件の内覧サービスを提供する (UCA1-T-2) まだ退去していない部屋の内覧サービスを提供する
5	内覧物件状態	不動産管理会社/オーナー	スマート内覧サービス提供会社	(UCA5-N-1) 連絡を怠った	(UCA5-T-1) 内覧物件の状態変化を連絡するのが遅い (UCA5-T-2) 稼働時間の違いにより、情報更新がタイムリーにできない



①STAMP/STPAで、ハザードを起こし得る非安全なコントロールアクション（UCA：Unsafe Control Action）を抽出する。

②D-Caseで、各リスクに対してどう対応するかを説明して、ステークホルダ間の合意を形成する。

目次

- WGの紹介
- これまでの活動振り返り（～2018年5月）
- STEP1の活動報告（2018年6月～2018年11月）
- STEP2の活動報告（2018年12月～2019年4月）
- 今後の活動予定

活動内容

- STEP2として、STEP1までに得られたモデルの活用方法、有用性を、全く新しいビジネスアイディアを創出する段階から適用して、その効果を検証する

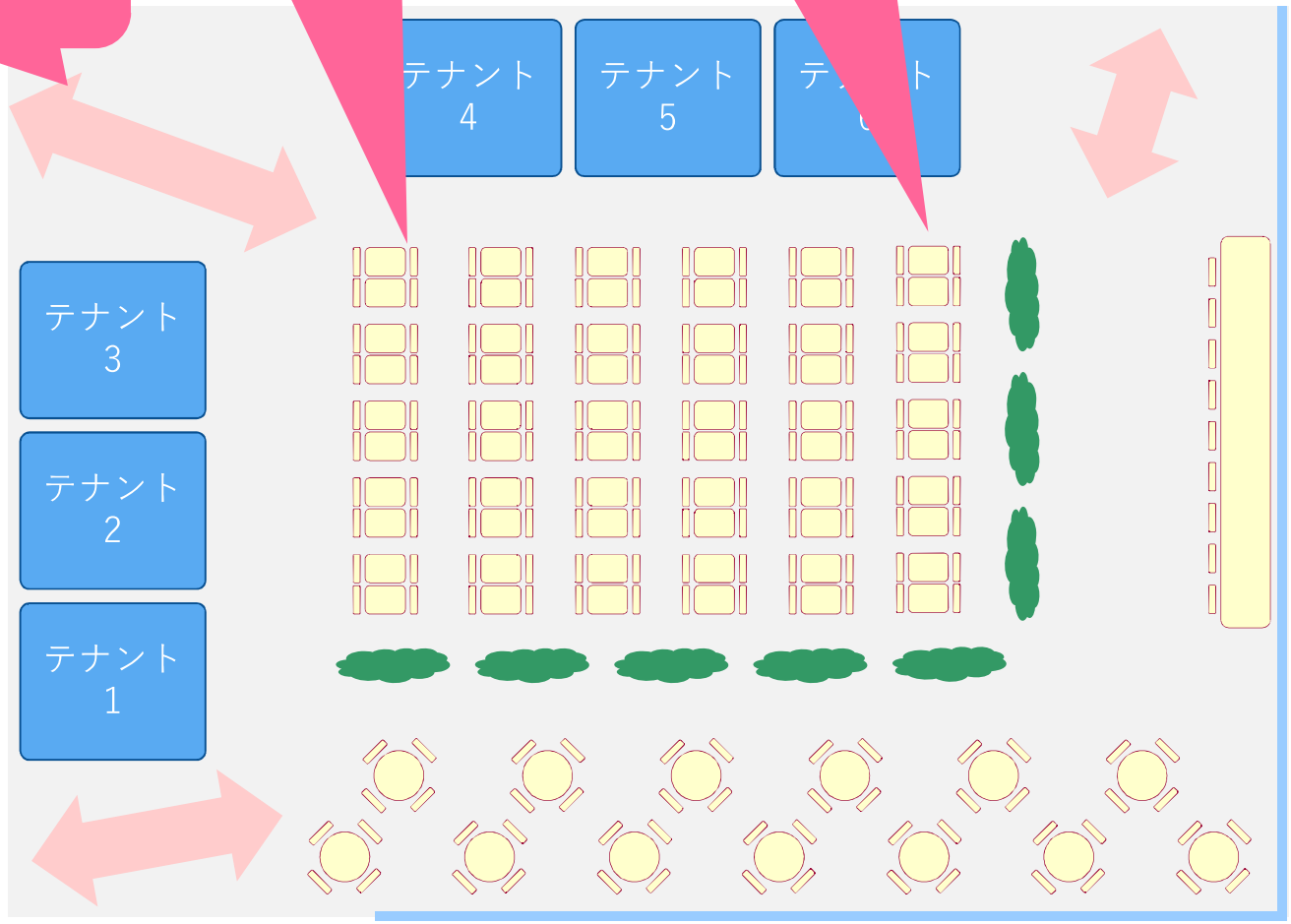
- 新しいビジネスアイディアについては、フリーディスカッションする中でメンバー全員がストレスに感じている「フードコートの席取り問題」を解決するためのビジネスを検討していくことに決定

『フードコートの席取り問題』とは

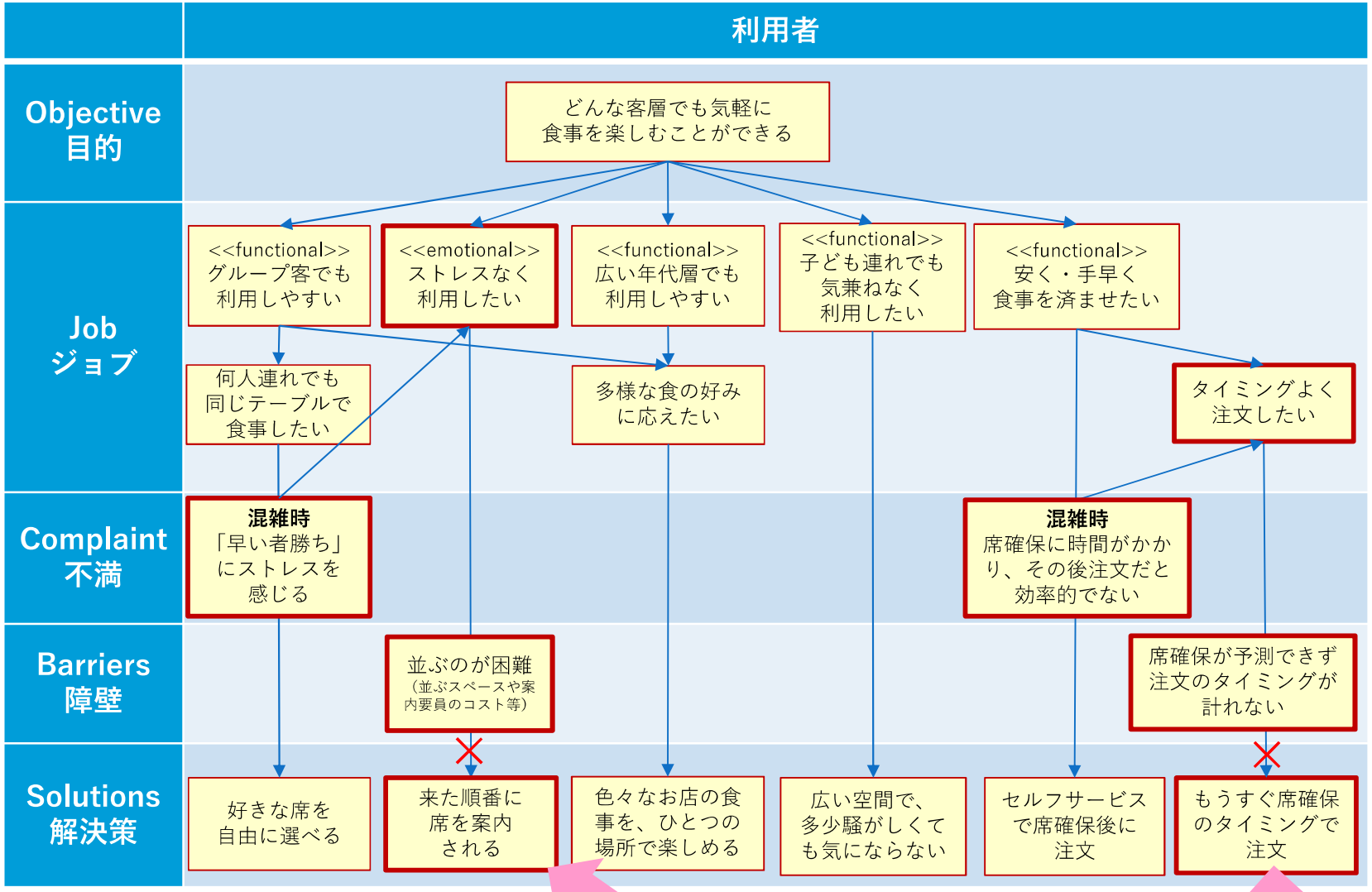
問題１.
複数個所から出入りできるため、席取りの順番待ちができない

問題２.
次にどこが空くか分からない（運に左右され、不公平）

問題３.
どれくらい待つか分からず、注文ができない

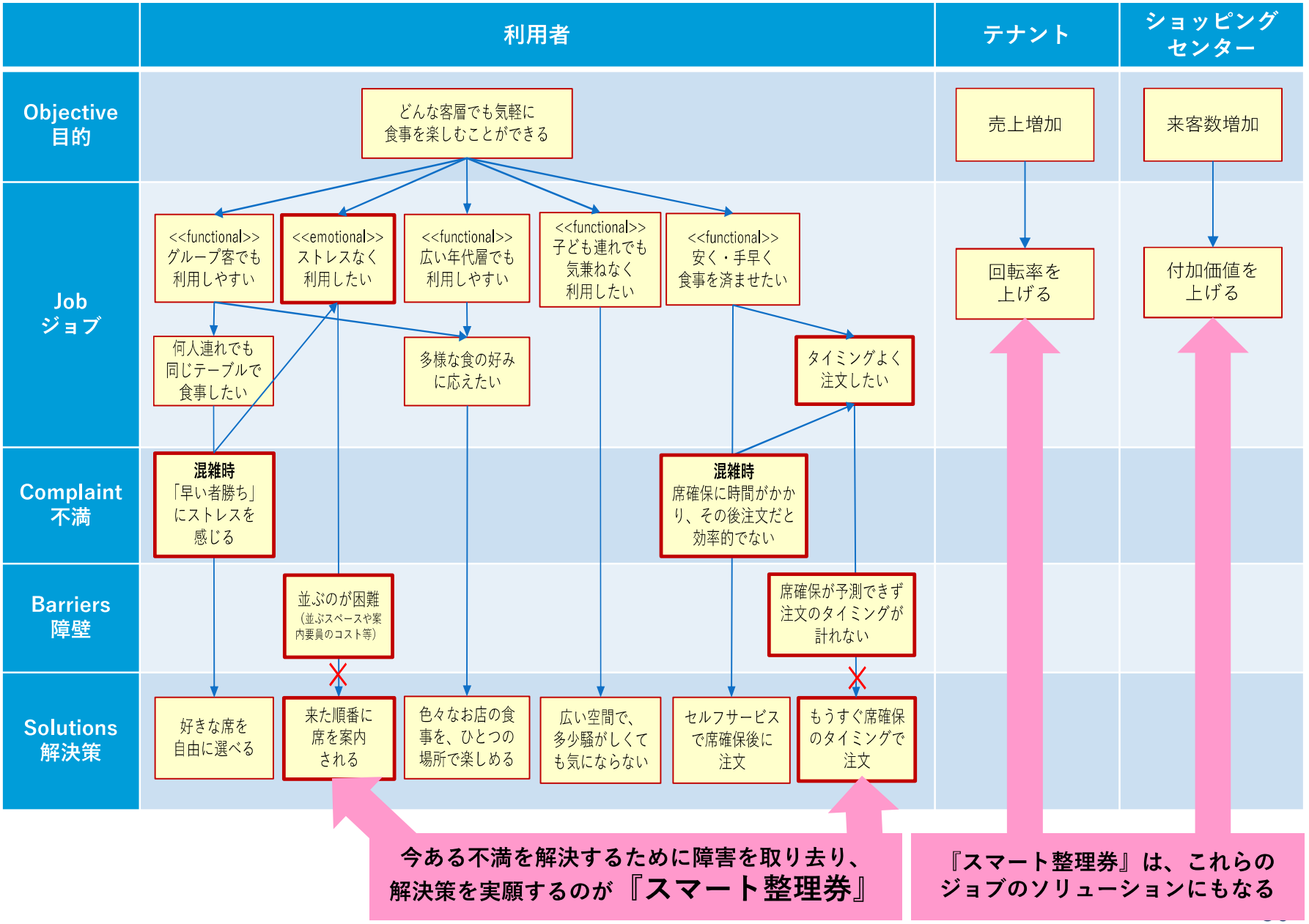


JOB理論モデル


利用者
Objective 目的
どんな客層でも気軽に食事を楽しむことができる
Job ジョブ
<<functional>> グループ客でも利用しやすい
<<emotional>> ストレスなく利用したい
<<functional>> 広い年代層でも利用しやすい
<<functional>> 子ども連れでも気兼ねなく利用したい
<<functional>> 安く・手早く食事を済ませたい
何人連れでも同じテーブルで食事したい
多様な食の好みに応えたい
タイミングよく注文したい
Complaint 不満
混雑時「早い者勝ち」にストレスを感じる
混雑時席確保に時間がかかり、その後注文だと効率的でない
Barriers 障壁
並ぶのが困難（並ぶスペースや案内要員のコスト等）
席確保が予測できず注文のタイミングが計れない
Solutions 解決策
好きな席を自由に選べる
来た順番に席を案内される
色々なお店の食事を、ひとつの場所で楽しめる
広い空間で、多少騒がしくても気にならない
セルフサービスで席確保後に注文
もうすぐ席確保のタイミングで注文
今ある不満を解決するために障害を取り去り、
解決策を実願するのが『スマート整理券』

35

JOB理論モデル


利用者
テナント
ショッピングセンター
Objective 目的
どんな客層でも気軽に食事を楽しむことができる
売上増加
来客数増加
Job ジョブ
<<functional>> グループ客でも利用しやすい
<<emotional>> ストレスなく利用したい
<<functional>> 広い年代層でも利用しやすい
<<functional>> 子ども連れでも気兼ねなく利用したい
<<functional>> 安く・手早く食事を済ませたい
回転率を上げる
付加価値を上げる
何人連れでも同じテーブルで食事したい
多様な食の好みに応えたい
タイミングよく注文したい
Complaint 不満
混雑時 「早い者勝ち」にストレスを感じる
混雑時 席確保に時間がかかり、その後注文だと効率的でない
Barriers 障壁
並ぶのが困難 (並ぶスペースや案内要員のコスト等)
席確保が予測できず注文のタイミングが計れない
Solutions 解決策
好きな席を自由に選べる
来た順番に席を案内される
色々なお店の食事を、ひとつの場所で楽しめる
広い空間で、多少騒がしくても気にならない
セルフサービスで席確保後に注文
もうすぐ席確保のタイミングで注文
今ある不満を解決するために障害を取り去り、解決策を実願するのが『スマート整理券』
『スマート整理券』は、これらのジョブのソリューションにもなる

『スマート整理券』ユーザストーリー

① 入り口でスマート整理券サービスにエントリする
- スマートフォンのアプリで、人数・テーブル数・希望エリア等を入力
- エントリすると、「スマート整理券」が電子配布される

② アプリのメニューから注文を行う
- スマート整理券に注文が紐づけされる

③ 席が空くのを待つ

④ 席が空くと、アプリを通じて連絡が来る

⑤ 案内された席に座る
- 着席は、スマートフォンが席近くに来ることで判断する

⑥ 注文した料理ができた連絡が来て、受け取る
- ここで決裁する

⑦ 食事する

⑧ 食事が終わったら、食器を返却する

⑨ フードコートから出る
- “席開放”は、スマートフォンがフードコートエリアから外れる事で自動的に判断する

『スマート整理券』ビジネスモデルキャンバス

Key Partner
パートナー

サービス運用会社
システム運用会社（クラウド）
システム開発会社
機材提供会社

Key Activities
主要活動

スマート整理券の製造
バックエンドのシステム開発
座席使用の仕組み開発

Key Resourses
リソース

スマホ＋専用アプリ
テナント用端末

Value Propositions
価値提案

①の顧客に対して、席確保のストレスを軽減
②の顧客に対して、上記不満を解決することでフードコートの付加価値アップ
③の顧客に対して、注文のオペレーションを効率化できる、回転率が上がって売上が上がる

Customer Relationships
顧客との関係

セルフサービス

Channels
チャンネル

スマート整理券

Customer Segmentation
顧客セグメント

①食事したい人
②ショッピングセンター
③テナント（飲食店）

Cost Structure
コスト構造

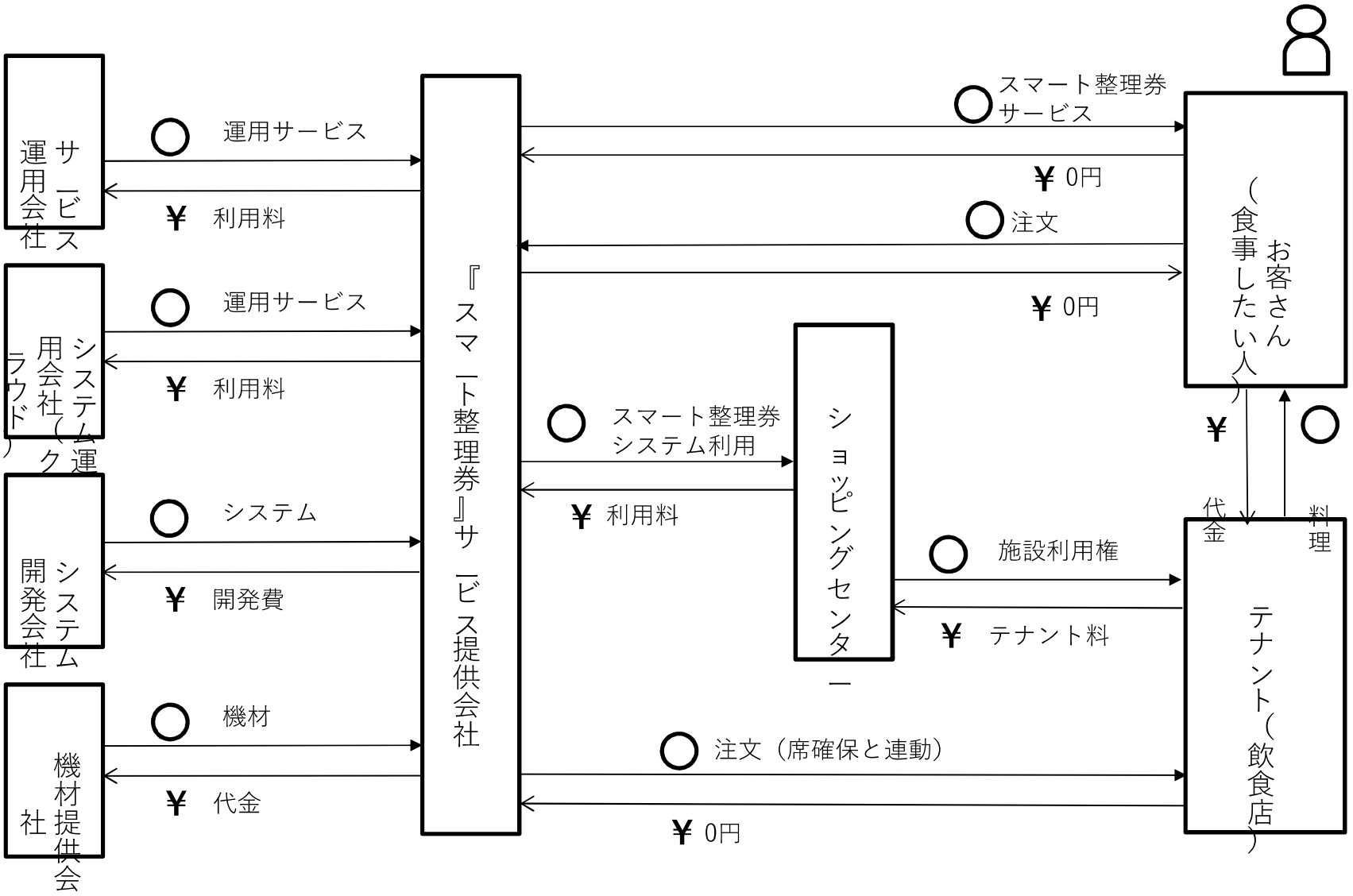
システム運用費
システム開発費
機材費用

Revenue Streams
収益の流れ

①③フリー
②サービス利用料（固定/従量）

38

『スマート整理券』ピクト図



『スマート整理券』STAMP/STPA（STEP0）

STEP0

- 前提条件の決定
- アクシデントハザード安全制約表
- コンポーネント抽出
- コントロールストラクチャ作成

STEP1

前提条件表 / 前提条件表

ID	
Pre-1	キャッシュレス決裁を対象とする（ToBeを先に考え、現状への対応は後で考える）
Pre-2	空席がある場合は、席を指定できる
Pre-3	空席がない場合は、予約できる（希望（席数・エリア）は出せるが、席の指定はできない）
Pre-4	注文と席の予約は一緒に実施する
Pre-5	席の開放は、サービスが自動で検知（端末を返却、または、席から遠く離れる）

アクシデントハザード安全制約表 / アクシデントハザード安全制約表

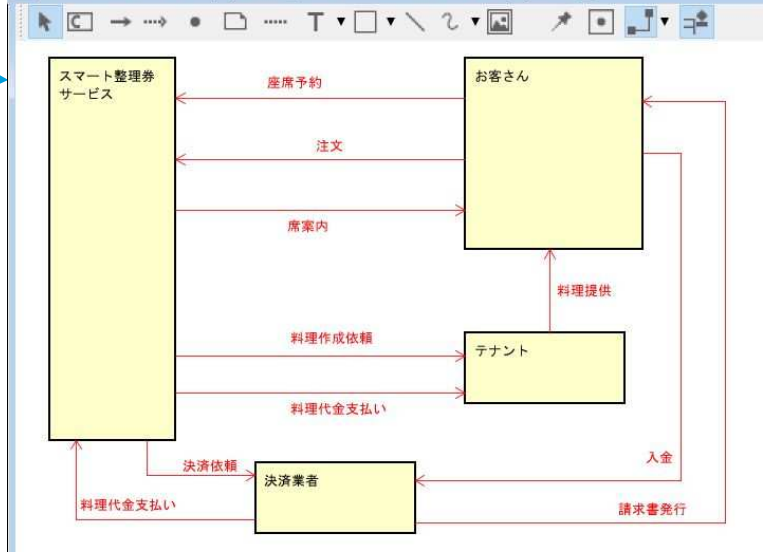
アクシデントID	アクシデント	ハザードID	ハザード
A1	ショッピングモールの評判が落ちる	H1	注文したのに、注文されていなかった
A1	ショッピングモールの評判が落ちる	H2	席が空いていないのに、空いていると言われた
A1	ショッピングモールの評判が落ちる	H3	席が空いているのに、案内されない
A1	ショッピングモールの評判が落ちる	H4	利用者が不満に感じる
A2			

コンポーネント抽出表 / コンポーネント抽出表

同期中のCS図名： コントロールストラクチャー図0

対象	登場人物	責務	コントロールアクション
☑	スマート整理券サービス	スマート整理券サービスを提供し、お客さまの予約順番通りの席案内と、席に着いてから料理提供がなされるようテナントへの料理依頼のタイミングを計り、快適な喫食環境を実現する	席案内 (To: お客さん) 料理作成依頼 (To: テナント) 料理代金支払い (To: テナント) 決済依頼 (To: 決済業者)
☑	お客さん	自分のスマホやショッピングモールから貸し出される端末を使い、フードコートを利用する	座席予約 (To: スマート整理券サービス) 注文 (To: スマート整理券サービス) 入金 (To: 決済業者)
☑	テナント	フードコートに出店し、お客さまにタイミングよく料理を提供する	料理提供 (To: お客さん)
☑	決済業者	カード会社や決済代行業者で、お客さまには月次で請求し、スマート整理券サービスからは都度清算する	料理代金支払い (To: スマート整理券サービス) 請求書発行 (To: お客さん)

コントロールストラクチャー図0 / コントロールストラクチャー図

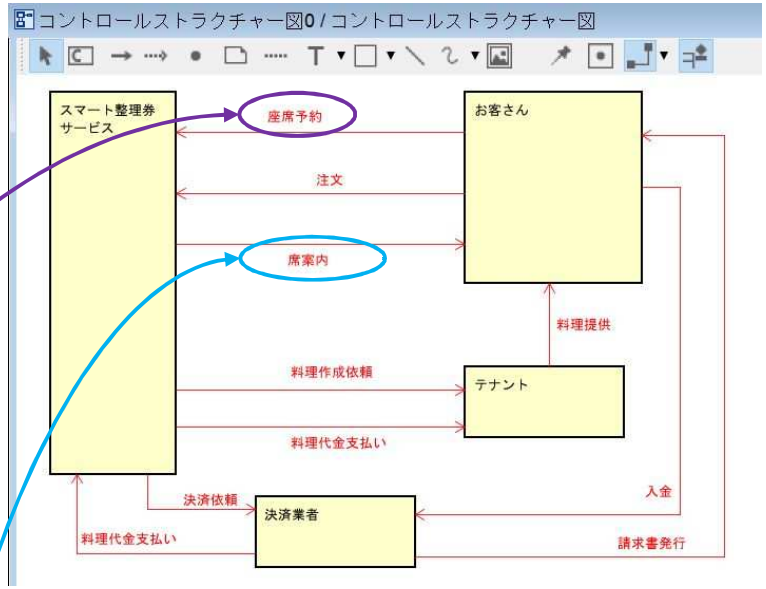


『スマート整理券』STAMP/STPA（STEP1）

STEP0

STEP1

- UCA（Unsafe Control Acton）の抽出



UCA表0 / UCA表

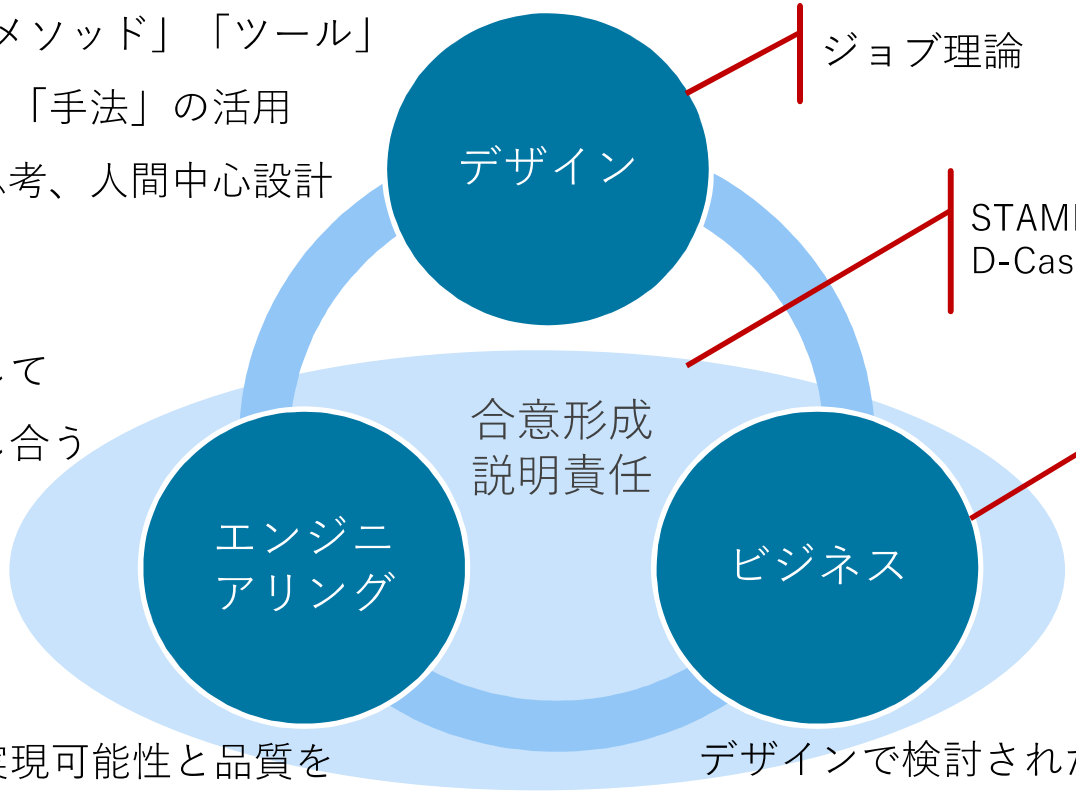
No	CA	From	To	CA提供条件	Not Providing	Providing causes hazard	Too early / Too late
1	座席予約	お客さん	スマート整理券サービス		(UCA1-N-1) お客さんが予約したいのに予約できない (UCA1-N-2) 席が空いているのに、指定できない		
2	席案内	スマート整理券サービス	お客さん		(UCA2-N-1) 座席が空いているのに、案内しない	(UCA2-P-1) 人数が足りない席を案内する (UCA2-P-2) 希望と異なる席を案内する (UCA2-P-3) 空いていない席を案内する	(UCA2-T-1) 料理が完成しているのに、席案内していない

41

IoTサービスを検討する際の、各モデルとその活用領域

さまざまな「メソッド」「ツール」
「技法」「手法」の活用
デザイン思考、人間中心設計

モデルを通して
人と人が理解し合う



ジョブ理論

STAMP/STPA
D-Case

ビジネスモデル
キャンバス
IoTキャンバス
ピクト図解

実現可能性と品質を
技術主導型モデルを使って検討

デザインで検討されたアイディアを
ビジネスモデリングにより可視化・深耕

ビジネスをモデル化する際のポイント

- デザインやビジネスの領域でも、対象をきちんと可視化することは有効
  - アイデアだけで議論していると発散しがち
  - お互いイメージしているものが、実は違っていた、ということが明確になる
  - 開発者の視点は「どう作るか」に行きがちだが、モデルを活用することにより、「どう価値づくりするか＝デザイン」や「どう儲けるか＝ビジネス」に視野を広げることができる
- 1つのモデルではなく、複数のモデルの使い分けが重要
  - モデルがスポットを当てるポイント（＝ビュー）が、それぞれ異なる
  - 複数のビューでモデリングすることにより、複雑なシステムを単純化して考えられる
  - 多様なステークホルダーにあわせて、説明するモデルがあったほうが良い

目次

- WGの紹介
- これまでの活動振り返り（～2018年5月）
- STEP1の活動報告（2018年6月～2018年11月）
- STEP2の活動報告（2018年12月～2019年5月）
- 今後の活動予定

今後の予定

- ET2019での冊子配布に向けて活動します
  - スマート整理券を題材としたモデルの作成
    - STAMP/STPA（洗練）、D-case（新規）
  - モデリングプロセスの検討
    - 各モデルのつながり
    - モデル作成の順番



ついに公開！レガシーコードをリサイクル！
～リバースモデリングツール RExSTM for Cのご紹介～

2019年

状態遷移設計研究WG

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

状態遷移設計研究WGとは

http://www.jasa.or.jp/top/activity/state_transition.html



詳細は、JASAのWEBページに掲載！
毎月開催されるWGのアジェンダなどをWEB上に公開しています。

© Japan Embedded Systems Technology Association 2016

状態遷移設計研究WGの背景と方針

■ 組込みソフトウェア開発の傾向

- 派生開発による短納期・高品質の要望
- レガシーコードの肥大化・複雑化→メンテナンス性低下

→ 設計資料なし、担当者もすでにいない、
修正したら予想外の問題が出る…

■ 研究テーマ

- 「状態遷移表のリバースモデリングへの適用」
- レガシーコードから状態遷移表をリバースモデリングする

■ ツールREXSTM for Cの普及啓蒙、「状態遷移表のリバースモデリングへの適応」の研究

状態遷移モデルを見つけるツールの研究・開発

→ REXSTMの開発

© Japan Embedded Systems Technology Association

状態遷移表とは？



関数	条件	スロットの状態			
		全て回転中	1つ停止	2つ停止	3つ停止
Entry4	L、M、Rがすべて一致	状態遷移：1つ停止	状態遷移：2つ停止	状態遷移：3つ停止 総額にBET金額の5倍を加算	状態遷移：全て回転中
	L、M、Rの2つが一致	状態遷移：1つ停止	状態遷移：2つ停止	状態遷移：3つ停止 総額にBET金額加算	状態遷移：全て回転中
	else	状態遷移：1つ停止	状態遷移：2つ停止	状態遷移：3つ停止	状態遷移：全て回転中

状態変数

状態

イベント

遷移・処理

© Japan Embedded Systems Technology Association

状態遷移表



関数	条件	flag_b1_c			
		0	1	2	3
Entry4	L、M、Rがすべて一致	flag_b1_c = 1	flag_b1_c = 2	flag_b1_c = 3 Total += Bet*5	flag_b1_c = 0
	L、M、Rの2つが一致	flag_b1_c = 1	flag_b1_c = 2	flag_b1_c = 3 Total += Bet	flag_b1_c = 0
	else	flag_b1_c = 1	flag_b1_c = 2	flag_b1_c = 3	flag_b1_c = 0

状態変数

状態

イベント

遷移・処理

© Japan Embedded Systems Technology Association

サンプル検証1（意味不明→仕様明確化）



状態変数名が意味不明

状態を0～3の直値で指定

関数	条件	flag_b1_c			
		0	1	2	3
Entry4	L、M、Rがすべて一致	flag_b1_c = 1	flag_b1_c = 2	flag_b1_c = 3 Total += Bet*5	flag_b1_c = 0
	L、M、Rの2つが一致	flag_b1_c = 1	flag_b1_c = 2	flag_b1_c = 3 Total += Bet	flag_b1_c = 0
	else	flag_b1_c = 1	flag_b1_c = 2	flag_b1_c = 3	flag_b1_c = 0

条件は 状態2 のときのみ有効

© Japan Embedded Systems Technology Association

リバースモデリングの作業手順

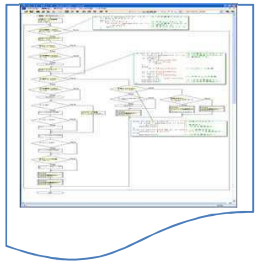
①レガシーコードの整形

- コメント削除、＃ifdefを設定
- 解析範囲の決定



②構文ツリーの作成

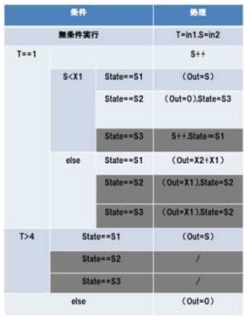
- フローチャートの生成
- 構文ツリー図の作成



③条件処理表（仮の状態遷移表）の作成

- 分岐条件を階層化
- 条件・処理の対応を記載した条件処理表の作成

条件			処理
無条件実行			T=in1.S=in2
T==1			S++
	S<X1	State==S1	(Out=S)
		State==S2	(Out=0),State=S3
		else	S++,State=S1
	else	State==S1	(Out=X2+X2)
		else	(Out=X1),State=S2
T>4	State==S1		(Out=S)
else			(Out=0)



④状態変数の抽出

- 状態変数の抽出
- 状態遷移表の編集

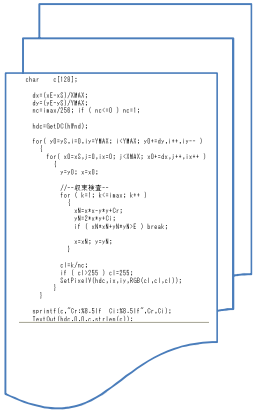
		State		
		S1	S2	S3
無条件実行		T=in1.S=in2	T=in1.S=in2	T=in1.S=in2
T==1		S++	S++	S++
	S<X1	(Out=S)	(Out=0),State=S3	S++,State=S1
	else	(Out=X2+X1)	(Out=X1),State=S2	(Out=X1),State=S2
T>4		(Out=S)	/	/
else		(Out=0)	(Out=0)	(Out=0)

状態遷移表の作成 完了！

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association
© Japan Embedded Systems Technology Association

リバースモデリングの作業手順

ソースコード前処理



①レガシーコードの整形

・コメント削除、＃ifdefを設定
・解析範囲の決定

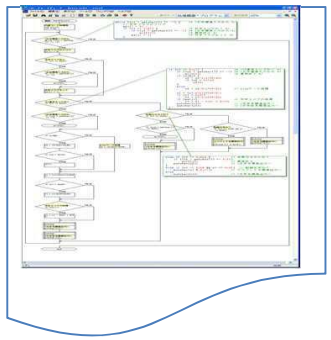
コメントはいらない！





②構文ツリーの作成

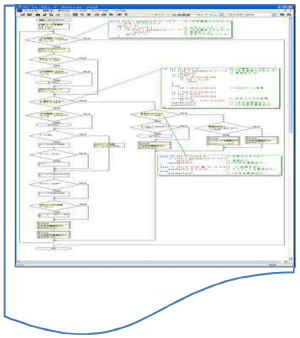
・フローチャートの生成
・構文ツリー図の作成



© Japan Embedded Systems Technology Association

リバースモデリングの作業手順

条件処理表の作成



③条件処理表（仮の状態遷移表）の作成

・分岐条件を階層化
・条件・処理の対応を記載した
　条件処理表の作成

条件			処理
無条件実行			T=in1,S=in2
T==1			S++
	S<X1	State==S1	(Out=S)
		State==S2	(Out=0),State=S3
		else	S++,State＝S1
	else	State==S1	(Out=X2+X2)
		else	(Out=X1),State=S2
T>4	Sate==S1		(Out=S)
else			(Out=0)

・条件分岐をif－else構造でモデル化
・switch-caseもif－else構造
・条件処理表の作成

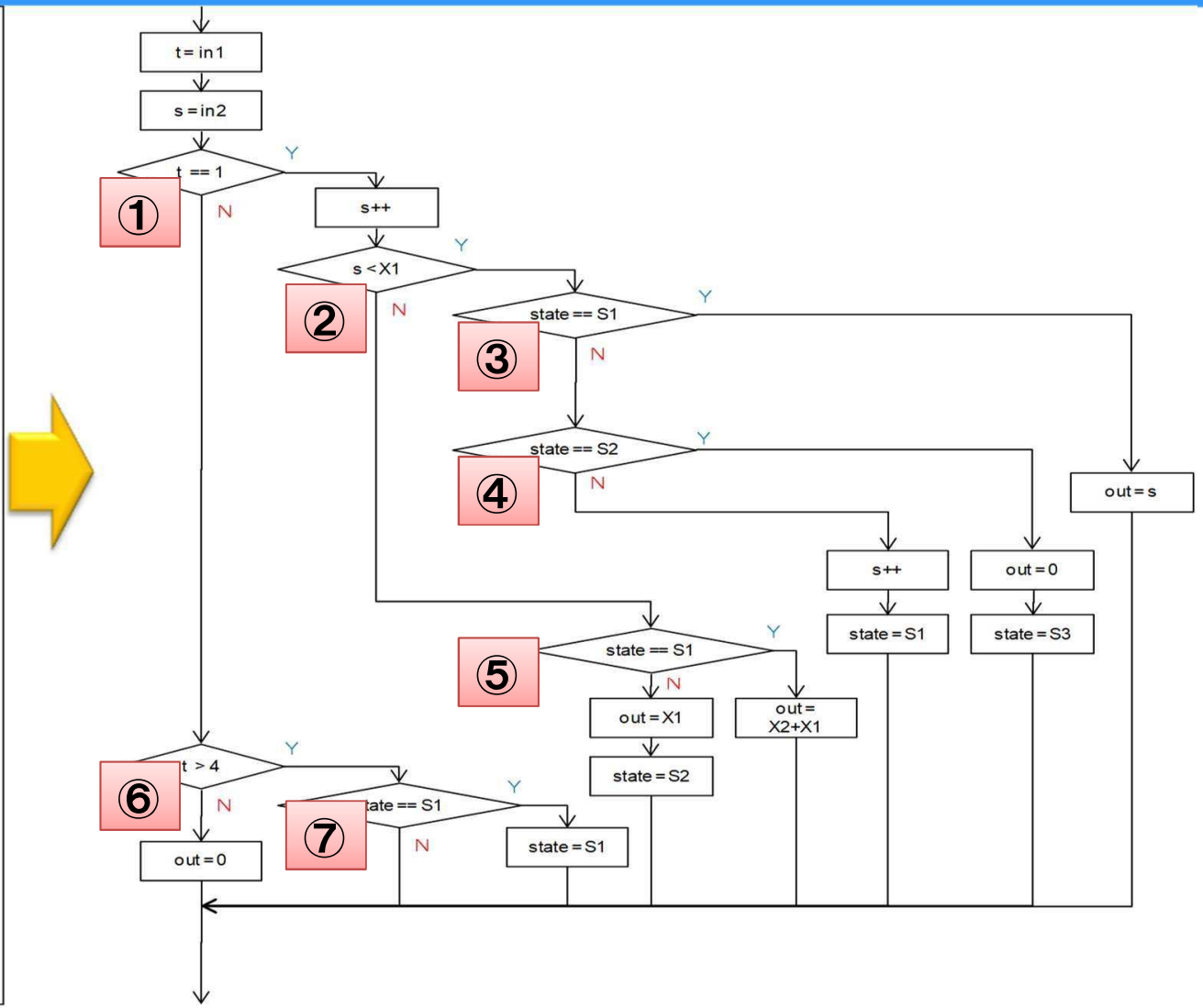
© Japan Embedded Systems Technology Association

リバースモデリングの作業手順

```
extern int in1,in2
int state;
#define S1  1
#define S2  2
#define S2  3
#define X1 5
#define X2 T1*2
int out

void task(void)
{
   int t,x;

   t = in1;
   s = in2;
   if ( t == 1) {
      s++;
      if (s < X1) {
         if(  state == S1 ){
            out = s;
         else if( state == S2 ){
            out = 0;
            state = S3
         else{
             s++;
            state = S1;
         }
      } else {
         if(  state == S1 ){
            out = X2+X1;
         else{
            out = X1;
            state = S2
      }
   }
   else  if(  t > 4 ){
        if(  state == S1 ){
            out = s;
        }
   }
   else{
       out = 0;
    }
}
```



© Japan Embedded Systems Technology Association

リバースモデリングの作業手順

条件処理表の作成

条件			処理
無条件実行			T=in1,S=in2
T==1 ①			S++
	S<X1 ②	flg==S1 ③	(Out=S)
		flg==S2 ④	(Out=0), flg=S3
		else	S++flg,=S1
	else	flg==S1 ⑤	(Out=X2+X1)
		else	(Out=X1), flg=S2
T>4 ⑥	⑦	flg==S1	(Out=S)
else			(Out=0)

ソースの分岐条件から条件を左、処理を右、に置いた仮の状態遷移表を作成

フローチャートの①～⑦の構成と、条件処理表の①～⑦の構成は同じになる

© Japan Embedded Systems Technology Association

リバースモデリングの作業手順

状態変数の抽出　　状態遷移表の作成

条件			処理
無条件実行			T=in1,S=in2
T==1			S++
	S<X1	State==S1	(Out=S)
		State==S2	(Out=0),State=S3
		State==S3	S++,State=S1
	else	State==S1	(Out=X2+X1)
		State==S2	(Out=X1),State=S2
		State==S3	(Out=X1),State=S2
T>4	State==S1		(Out=S)
	State==S2		/
	State==S3		/
else			(Out=0)

④状態変数の抽出

- 状態変数の抽出
- 状態遷移表の編集

		State		
		S1	S2	S3
無条件実行		T=in1,S=in2	T=in1,S=in2	T=in1,S=in2
T==1		S++	S++	S++
	S<X1	(Out=S)	(Out=0),State=S3	S++,State=S1
	else	(Out=X2+X1)	(Out=X1),State=S2	(Out=X1),State=S2
T>4		(Out=S)	/	/
else		(Out=0)	(Out=0)	(Out=0)

■ ポイントは、状態変数を見つけること

■ 分岐条件の変数は状態変数か？

- 状態変数の定義
  - 状態変数の取りうる値は、有限個である
  - 状態変数は、内部で更新される



© Japan Embedded Systems Technology Association

リバースモデリングの作業手順



条件			処理
無条件実行			T=in1,S=in2
T==1			S++
	S<X1	flg==S1	(Out=S)
		flg==S2	(Out=0),flg=S3
		flg==S3	flg＝S1
	else	flg==S1	(Out=X2+X1)
		flg==S2	(Out=X1),flg=S2
		flg==S3	(Out=X1),flg=S2
T>4	flg==S1		(Out=S)
	flg==S2		/
	flg==S3		/
else			(Out=0)

flgはS1,S2,S3の有限値かつflgは内部で更新されている



flgは、状態変数だ

elseを有限値に置換

© Japan Embedded Systems Technology Association

リバースモデリングの作業手順



状態遷移表の作成

条件			flg		
			S1	S2	S3
無条件実行			T=in1,S=in2	T=in1,S=in2	T=in1,S=in2
T==1			S++	S++	S++
	S<X1		(Out=S)	/	/
			/	(Out=0),flg=S3	/
			/	/	flg＝S1
	else		(Out=X2+X1)	/	/
			/	(Out=X1),flg=S2	/
			/	/	(Out=X1),flg=S2
T>4			(Out=S)	/	/
			/	/	/
			/	/	/
else			(Out=0)	(Out=0)	(Out=0)

flgを
状態変数
として、状態に
移行する

© Japan Embedded Systems Technology Association

リバースモデリングの作業手順

条件		flg		
		S1	S2	S3
無条件実行		T=in1,S=in2	T=in1,S=in2	T=in1,S=in2
T==1		S++	S++	S++
	S<X1	(Out=S)	(Out=0),flg=S3	flg＝S1
	else	(Out=X2+X1)	(Out=X1),flg=S2	(Out=X1),flg=S2
T>4		(Out=S)	/	/
else		(Out=0)	(Out=0)	(Out=0)

うん、
状態遷移表に
なった



これで、やっとレビューができる！！



一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association
© Japan Embedded Systems Technology Association

ツール化の方針→できることを自動化しよう！



■ 自動化できそうなことを検討

- ソースコードから条件処理表は自動で抽出できそう
- 状態変数の選択は，ユーザ任せにしたい
  - どの変数を選択すると，意味のある状態遷移表になるのか？
  - 状態変数を選ぶこともコストが大きい
- 状態変数候補を自動的に求め，ユーザが状態変数を選択する補助を行えるようにする
  - 状態変数の条件を満たしている変数を見つけてくる
  - ソースコードでどのように使用されているかを見たい

© Japan Embedded Systems Technology Association 2016

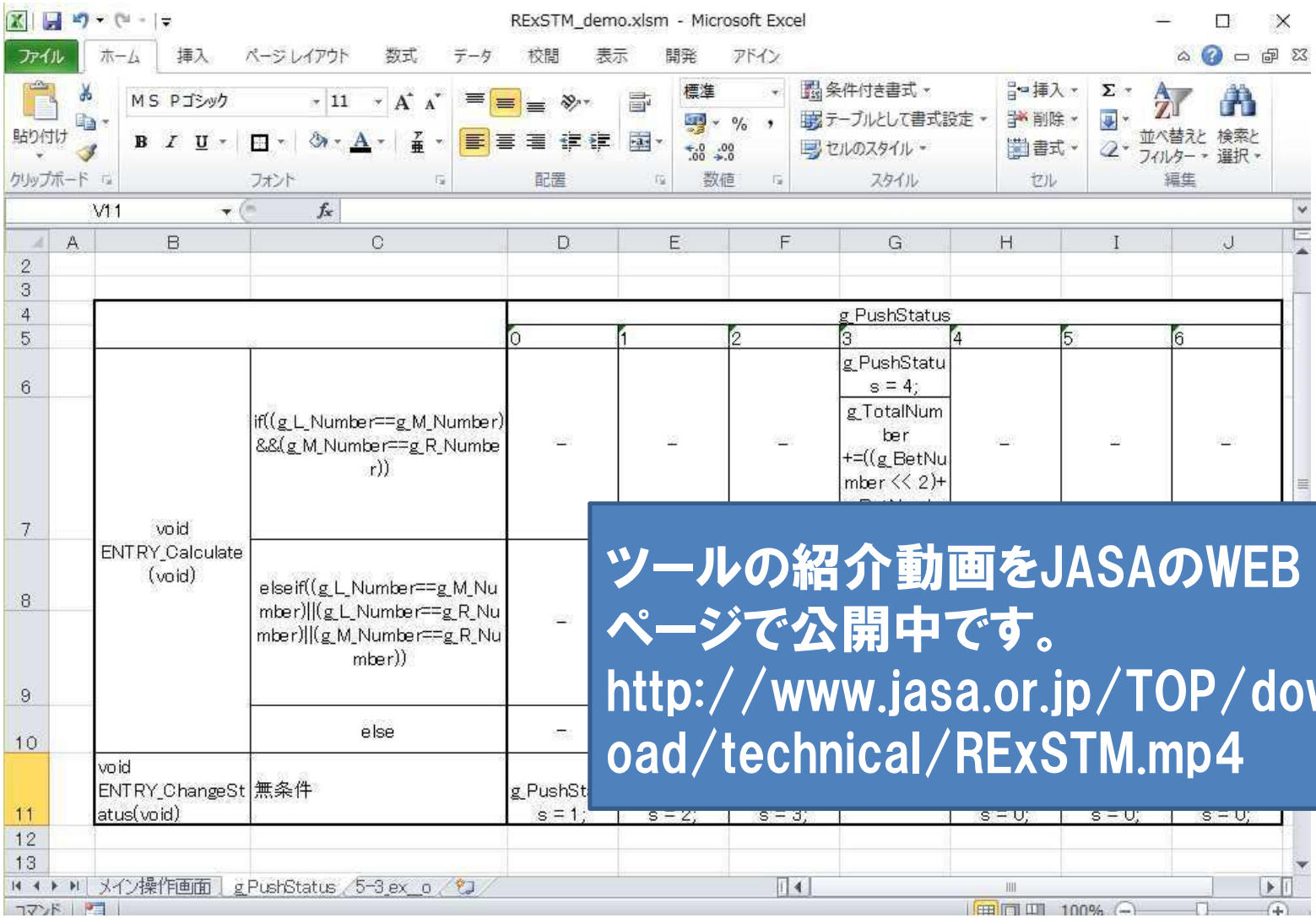
RExSTM for C　条件処理表・状態遷移表作成ツール



ソースコードの整形・解析を行い、状態遷移表生成のための情報整理を実施

© Japan Embedded Systems Technology Association

RExSTM for C　状態遷移表の出力



ツールの紹介動画をJASAのWEBページで公開中です。
http://www.jasa.or.jp/TOP/download/technical/RExSTM.mp4

© Japan Embedded Systems Technology Association

実験：状態遷移表のリバース 手動 VS 自動

名古屋大学の学生4名によるソースコードから状態遷移表を手動VS支援ツールで作成する実験を実施した。

実験対象ソースコード：自動車の動作の一部を表すプログラムLOC110
(プラットフォームや開発環境向けに使用することが可能なコード)
電子レンジの動作の一部を表すプログラムLOC190
(RTOS(TOPPERS/ASP)のアプリケーションを意識したコード)

手動で行う場合

被験者	修正回数	作成時間
学生1	1回	91分
学生2	2回	108分
学生3	4回	85分
学生4	4回	122分

実験：状態遷移表のリバース 手動 VS 自動

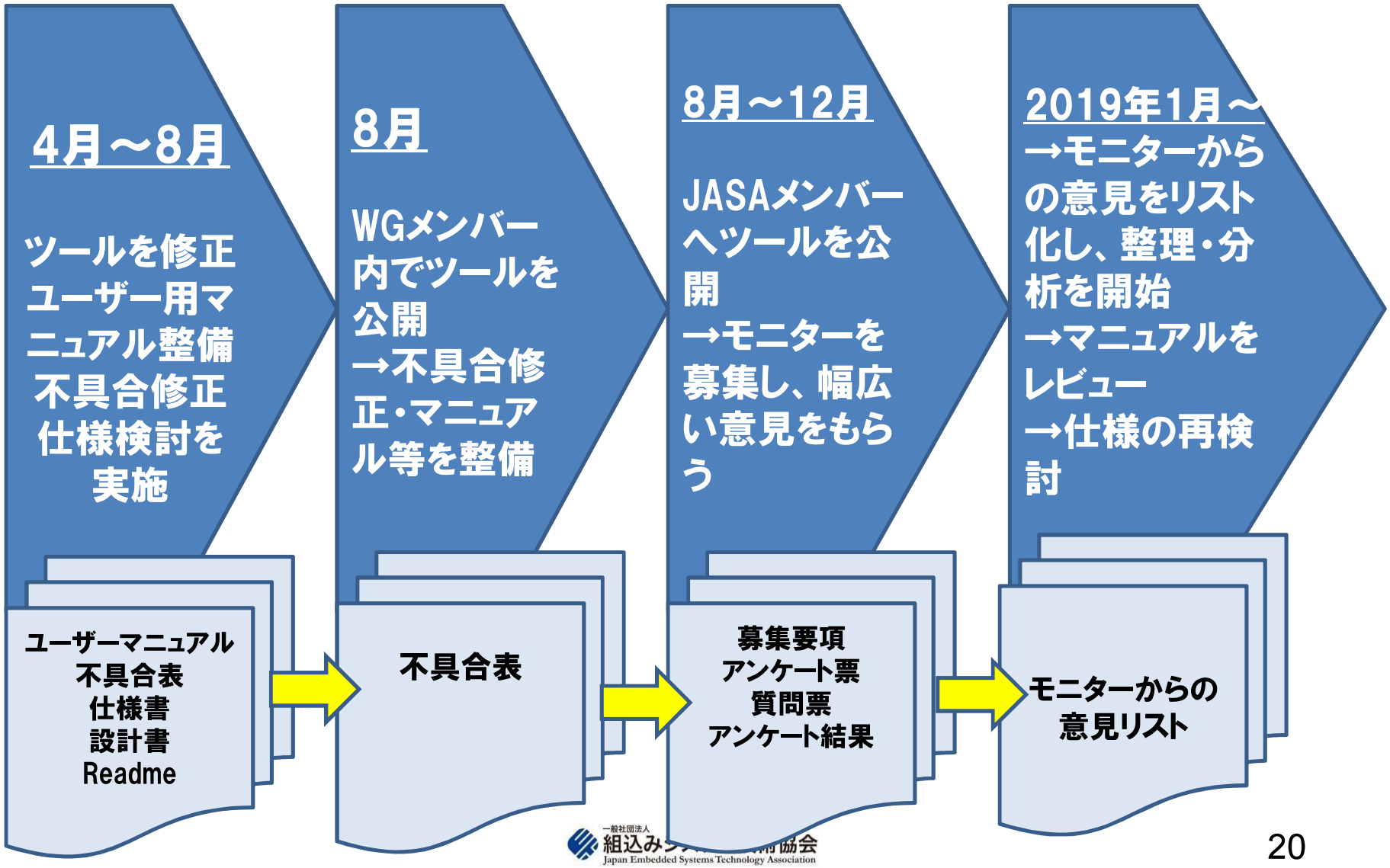


支援ツールを利用する場合

被験者	修正回数	作成時間
学生1	0回	19分
学生2	0回	12分
学生3	0回	10分
学生4	0回	20分

- 手動の場合は、間違いや見落とし等も発生しやすく修正回数が発生し、ツールを利用する場合と比べて作成時間も5倍から6倍程度かかる
- ソースコードと状態遷移表の対応がとれていない
- 関数が不足している
- 表現するべきではない箇所についても状態遷移表を作成している

2018年度の活動の流れ



© Japan Embedded Systems Technology Association

モニター募集要項(～2018年12月まで)

1. 状態遷移表自動生成・作成ツールのモニター募集について

技術本部状態遷移設計研究WGでは、2014年度よりレガシーコードの蘇生術と題して既存のCソースコードから状態遷移表を自動で生成・作成するためのツールを研究してまいりました。

このたび、ようやくツール化を実現・公開するという運びとなり、会員の皆様に公開させていただき、当ツールの利用具合に関するコメント等を頂戴できればと考えております。

なお、当ツールは最終的にはオープンソース化ができればと思っております。多くの会員の試用をお待ちしております。

1. ツール名称　RE x STM(レックス　エスティーエム) for C
2. 試用期間　お申込み後、2018年12月末まで
3. 試用申込み方法

① JASA会員専用サイトにサインインの上、「【会員限定】優待情報」⇒「【2018年度限定】「RE x STM forC」モニター申込フォーム」を選択すると、申込フォームにつながります。

② 申込みフォーム送信後、1週間以内にDL用のサイトアドレス・パスワードを送付します。

③ ダウンロード後、手順書にしたがい、ツールをインストールして下さい。

4. 試用アンケート

DL用のサイトアドレス・パスワードを発行後、30日を目安に、当WGよりヒアリング又はアンケートのご連絡をさせて頂きますので、ご協力の程、よろしくお願い致します。

© Japan Embedded Systems Technology Association

ツールモニターさんへサポートを実施

- ツールを使用していただいているモニターさんから質問をいただき、そのサポートを実施した。
- 質問は、使い方や、制限事項、アドバイスや、コメントなど幅広いご意見をいただくことができた。

問合せ票

日時：2018年XX月XX日（X ）
担当者：XXX　XX
連絡先アドレス：XXX@XX.com
PC環境：WindowsXXX、XXbit、OfficeXXX,
ExcelXXX、GCCバージョンXXX
仕様ツール：RExSTM for Cver.XXX
問合せ内容：
XXXX…

問合せは、ツールを使用していただいた企業の開発環境や使用している他のツールなどによって様々
当初は、個別で対応をしていたが、問合せ票を作成し内容を整理しやすいように変更
また、仕様については、WG内で議論を重ねることができた

© Japan Embedded Systems Technology Association

モニターアンケート一部抜粋

ツールを利用いただいた方について教えてください。

1.業界

(1)製造業(2)情報・通信系(3)車載系(4)家電系(5)その他

2.職種

(1)生産部門(2)生産技術・管理(3)設計・開発(4)調査・企画(5)営業(6)保守・安全(7)教育訓練(8)経営・管理(9)その他

3.就業年数

(1)3年未満(2)3年以上5年未満(3)5年以上10年未満(4)10年以上15年未満(5)15年以上

4.役職

(1)事務系一般社員・職員(2)技術系一般社員・職員(3)係長・主任クラス(4)課長クラス(5)部長クラス(6)経営者・役員クラス

(7)その他

5.従業員数

(1)1~29人(2)30人~99人(3)100人~299人(4)300人~499人(5)500人以上

ツールを知ったきっかけと、モニターを希望された目的を教えてください。

1.知ったきっかけ

(1)会社の指示(2)ホームページ(3)ちらし(4)情報誌・冊子・論文誌(5)他の機関からの紹介(6)知人の紹介(7)展示会

(8)その他

2.目的

(1)生産・開発現場における課題を解決するため(2)市場調査のため(3)新しい知識・及び技術ツールなどを習得するため

(4)同僚・部下へ伝達・指導するため(5)その他

© Japan Embedded Systems Technology Association

モニターアンケート結果一部抜粋

番号	フィードバック	
1	「tsvファイルが存在しません」というエラーが発生し、原因を調査するのが大変だったので、試用を諦めました。	エラー問題
2	今回、使用したソース群は、インクルードファイル、コンパイルスイッチなどもたくさん使用しています。 フォルダ構成も3階層ぐらいあります。	ネスト問題
3	また、ユーザーマニュアルの5．2で制限されている要件も満たされていないものファイルもたくさんあります。エラーを取り除くヒントがあれば、もう少し、問題を除去できたかもしれません。	制限問題
4	作成された一部の表はエビデンスや情報共有に用い、メンバー間での認識合わせに用いることができた。	
5	しかし、ネストの深さ、状態変数候補となる条件などの制限事項が想像以上に厳しかった。	ネスト問題
・・・	・・・他たくさんのご意見をツールのモニター企業からいただいた	・・・

© Japan Embedded Systems Technology Association

モニターアンケート結果

■ 相対パス・ネスト・Switch文・if文対応問題

- ツールの仕様を検討する段階で議論されていた問題「相対パス」・「ネスト問題」・「Switch文とif文対応問題」・「演算子の問題」等がやはりネックになっていることが明らかになった。また、今回は予算と時間の都合で上記を含めた制限事項を他にも設けていたが、その部分に対しても問題であることがわかった。

■ サンプルコードがもっとほしい

- ツールのモニター中に問合せがあり、サンプルコードを対応表として利用してもらうことも実施したが、やはり対応には限界があった。

■ 多様性を持たせてほしい

- ソースコードによっては、状態遷移表にすることができるが、多様性を持たせ、幅広い企業に利用してもらうためにはツールの改修が必要であることがわかった。また、改修するべき点も今回のモニターを実施することで明確になった。

使える部分を広げたい

もうちょっと改修されたら使いたい

モニターアンケート結果よりツール改修の必要性を感じた。

© Japan Embedded Systems Technology Association

オープンソース化についての議論

■ 定義

- オープンソースの定義は、ソフトウェアのソースコードへのアクセスが開かれている（ソースコードが公開されている）ことをうたうのではなく、オープンソースライセンスとして
  アの配布条
  ソースコー
  を利用する

■ レベル

1. 現状あるもの
2. 現状あるもの
3. 再利用性を重
4. GitHubなどを

■ 手順

1. 公開するべき範
   - 誰がハッピーに
   - 環境は？Win10
2. ライセンスの確認
   - Python・GCCなど
3. 公開ルール作成(2次的著作物・再配布についてなど)
4. パターン表作成(C→STM)

■ 勉強会

- JASAのOSS活用WGから竹岡さんにお話しを伺ったらどうか？

- **オープンソース化に向けて、定義やレベル、公開手順や対象者（誰がハッピーになるため）等を議論**
- **必要に応じて、仕様の見直しや不具合修正を実施**
- **ツール関連だけでなく、ツール周辺のドキュメントを整備**

	できる ソースコー
	アを変更、 すではいけ
	場合、それが 号であった場合、 ために、配布を行った者 はソフトウェアの名前やバージョン番号を変更したり、変更部分をパッチ*として公開することを要求できる
個人やグループに対する差別の禁止	特定の個人や団体に対して利用を制限することはできない。たとえば「共産主義国家では利用できない」といった制限を加えることはできない
利用する分野（fields of endeavor）に対する差別の禁止	分野によって利用を制限することはできない。たとえば商用利用や軍事利用を制限する条項は加えられない
ライセンスの分配（distribution）	再配布にあたって、何らかの追加ライセンスに同意することを求めてはならない。たとえば機密保持契約への同意を、配布を受ける条件としてはいけない
特定製品でのみ有効なライセンスの禁止	「ある特定の製品と組み合わせて使う場合のみ利用可能」といった制限を課してはならない
ほかのソフトウェアを制限するライセンスの禁止	頒布にあたって、そのソフトウェアと一緒に配布されるほかのソフトウェアに制限を課してはならない。たとえば同じ媒体に含まれるソフトウェアがすべてオープンソースでなければならないといった制限はできない
ライセンスは技術中立的でなければならない	特定の技術に強く依存するものであってはいけない。たとえば、GUIを備えていなければ利用できないケースや、OSに依存するような項目は加えられない

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

開発者用ドキュメント(一部抜粋)



最終更新日:2019年2月6日 **RExSTMビルド用ドキュメント**

■　　**はじめに**

このドキュメントはビルド用のものである．RExSTM¥programs内のファイルに対して
修正を行いRExSTMの動作
また，RExSTMの使用方法
る．
このドキュメントは事前にR
提としている．

内容は、検討段階なので、修正が必要だがオープンソース化したときに開発者が機能の追加などが行えるようにする

ビルド方法

■　　**目的**

RExSTM¥programs内のフ
めビルドを行う必要がある
ここではビルド方法につい

■　　**環境**

OS: windows10 64bit
Python: version3.4.0
においてビルドに成功することを確認している．**ビルド手順**

1.　**Pythonをインストールする**

・・・

© Japan Embedded Systems Technology Association

ソースコードのコメントを整備



```
#!/usr/bin/env python
# coding: Shift_JIS

# ##################################

# ファイル名: c_reform.py
# 機能: 1)解析対象のcファイルの中括弧を
#       2)解析対象のcファイル中のプロトタ
# 関数: c_reform
# 著者名: 山本椋太, 若林丈紘

# ##################################
```

ファイル名や機能の説明、関数名や著者名を記載する予定

© Japan Embedded Systems Technology Association

公開についての予定



■ 公開の日程
- ET-WEST2019を目標に公開予定

■ パブリックコメント
- モニター企業からいただいたコメントを整理・分析を実施
- リスト化をおこない優先順位を決定する

■ ツールの修正
- パブリックコメントの優先順位をベースにツールを改修する

■ オープンソース化
- 開発者用にマニュアルを整備
- ツールのソースコードを整理し、コメント等を追記
- ユーザマニュアル等をレビュー

© Japan Embedded Systems Technology Association

ご清聴ありがとうございました

© Japan Embedded Systems Technology Association



【ついに公開！レガシーコードをリサイクル】

2019/4/9 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

なぜオープンイノベーションは失敗したのか？ OpenELの失敗事例とその理由、そして解決策は？



2019年5月20日

OpenEL WG

中村憲一

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

目次

- JASA主導によるオープンイノベーション
- OpenELの開発
- 国際標準化の提案
- 失敗
- 解決策
- 活動の成果
- OpenEL WGのご紹介

© Japan Embedded Systems Technology Association

JASA主導によるオープンイノベーション

2000～2002年、Linux研究会

2002/10～、JASA Embedded Platform研究会

- 各種プラットフォームの調査研究
- オープンイノベーションについて研究

2011/5～、Platform研究会傘下にOpenEL-WGを設置し、ロボット向けのプラットフォームを調査

- アプリケーションレイヤーや、ミドルウェアレイヤーで動作するプラットフォームが多数
- ハードウェアに密接したレイヤーでのプラットフォームが必要との結論に

OpenELを提案

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

OpenEL(Open Embedded Library)とは？

■ ロボットや制御システムなどのソフトウェアの実装仕様を標準化する組込みシステム向けのオープンなプラットフォーム

OpenEL 3.1



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

国際標準化の提案



OpenEL



↓

HAL4RT



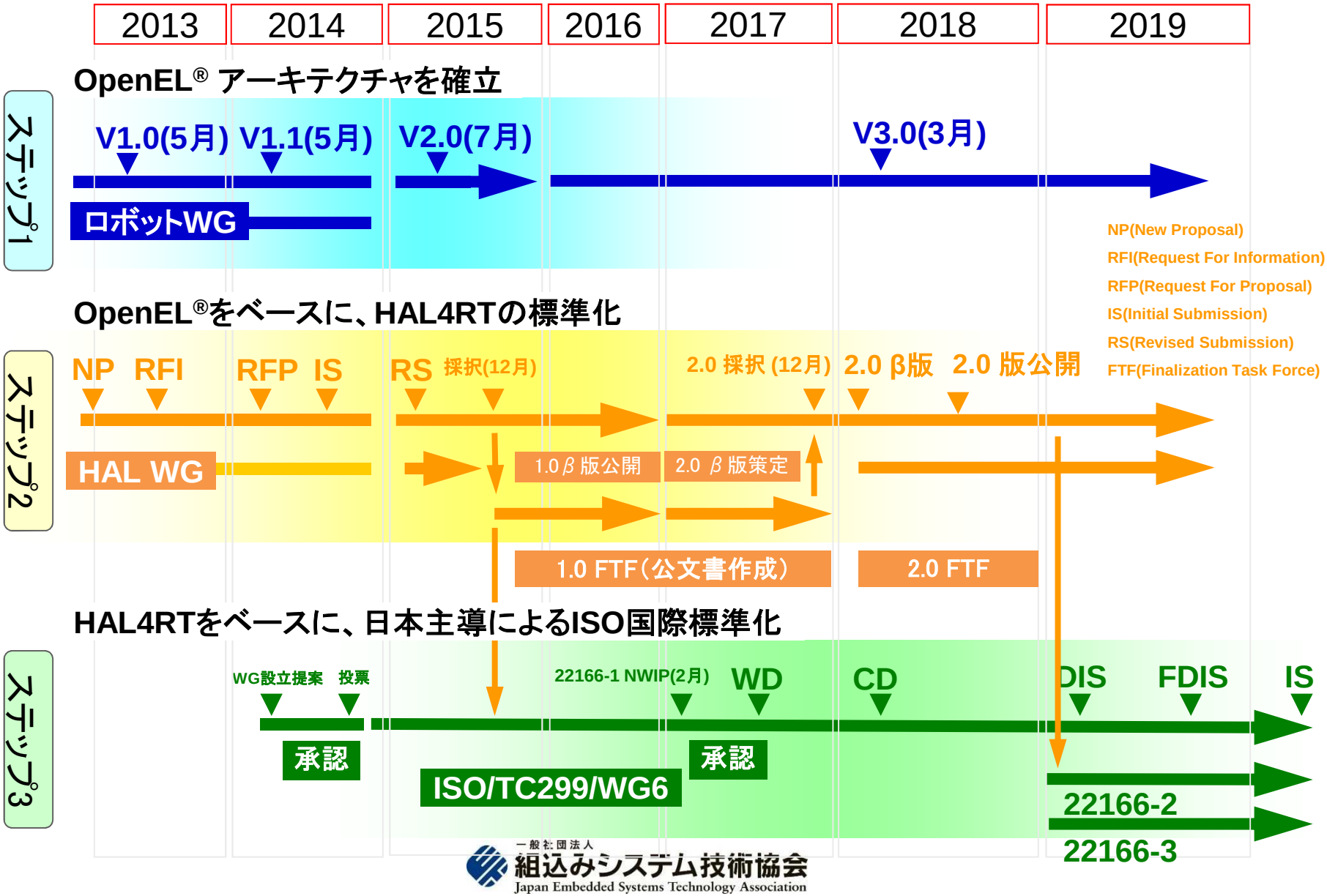
↓

ISO22166-1



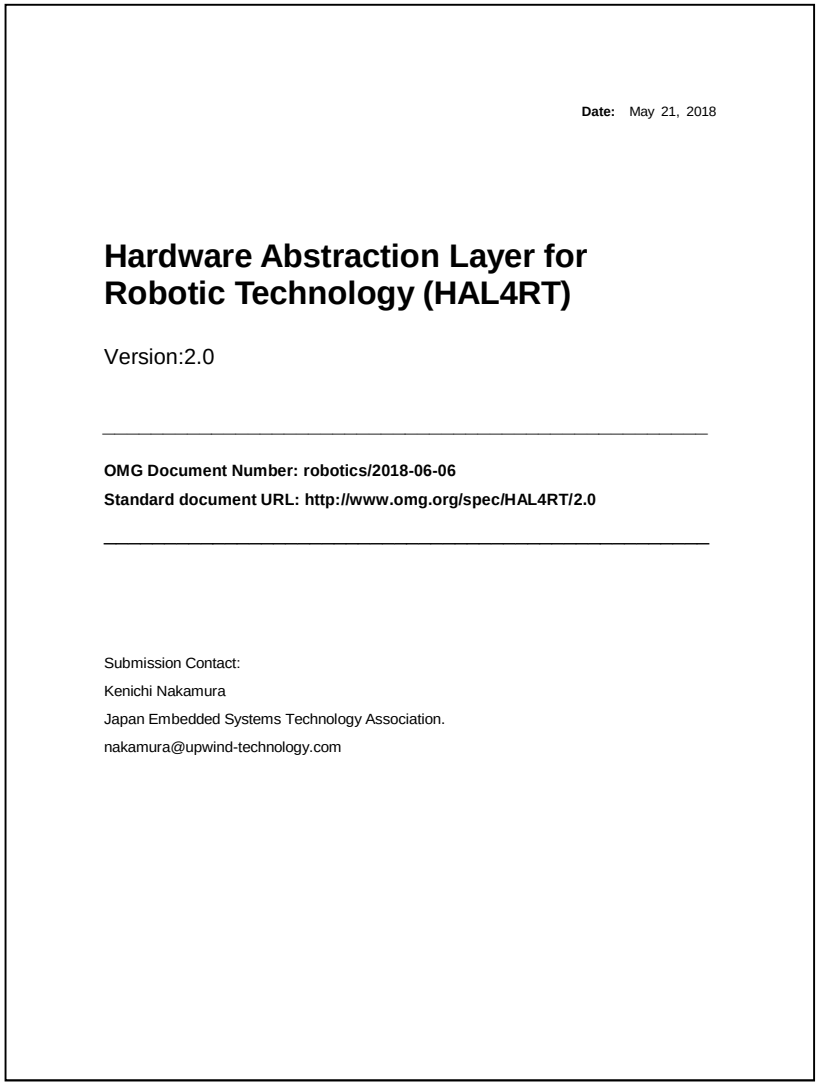
一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

国際標準化シナリオ(2017/11/1現在)



一般社団法人 組込みシステム技術協会
Japan Embedded Systems Technology Association

HAL4RT 2,0 RFC



Date: May 21, 2018

Hardware Abstraction Layer for Robotic Technology (HAL4RT)

Version:2.0

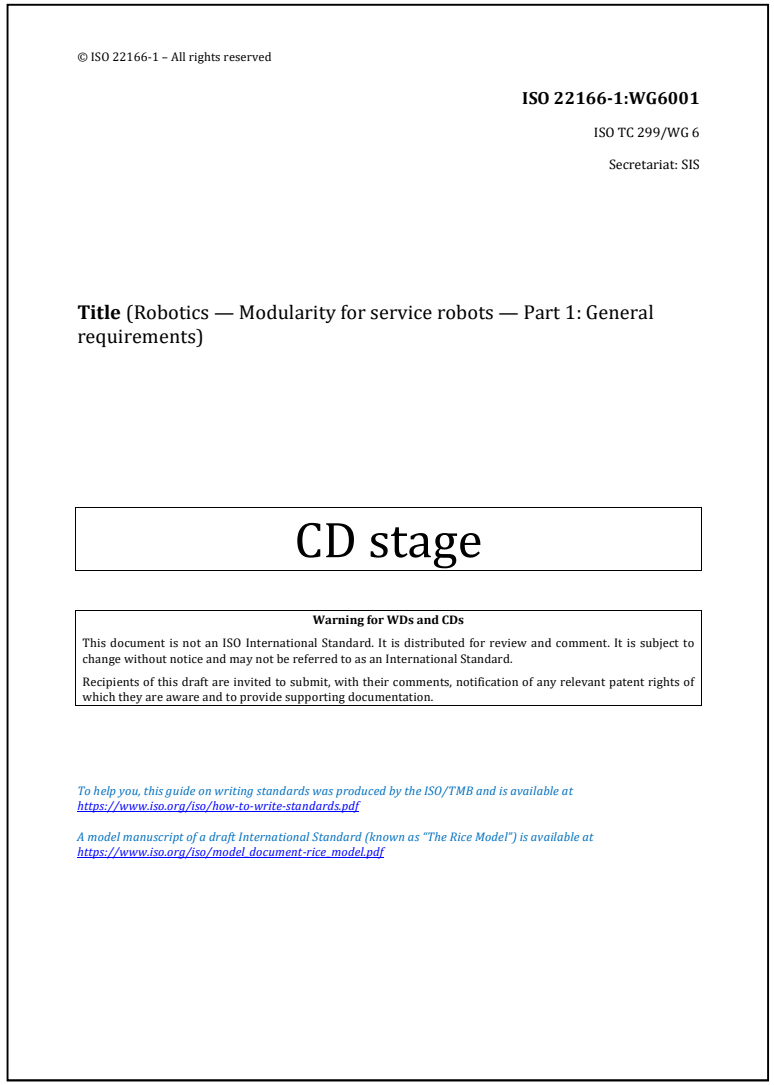
OMG Document Number: robotics/2018-06-06

Standard document URL: http://www.omg.org/spec/HAL4RT/2.0

Submission Contact:
Kenichi Nakamura
Japan Embedded Systems Technology Association.
nakamura@upwind-technology.com

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

ISO 22166-1 CD



© ISO 22166-1 – All rights reserved

ISO 22166-1:WG6001

ISO TC 299/WG 6

Secretariat: SIS

Title (Robotics — Modularity for service robots — Part 1: General requirements)

CD stage

Warning for WDs and CDs

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

To help you, this guide on writing standards was produced by the ISO/TMB and is available at https://www.iso.org/iso/how-to-write-standards.pdf

A model manuscript of a draft International Standard (known as "The Rice Model") is available at https://www.iso.org/iso/model_document-rice_model.pdf

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

ISO 22166-1 CD

7.3 Overview and requirements for software interfaces

7.3.1 General

Software module provides the software interface to access variables, invoke services, and process events. Hence a software module internally provide some functionalities such that remote variables are read and written via communication API or message format or shared variable, remote service is invoked and the results are returned, and the proper process is operating at occurrence of events, where remote variable and remote service is located in the other software modules. Also software modules shall access to HW devices via HAL and software device module and read the profile to initialize and operate properly them.

In addition message formats are provided for the control and maintenance of software module, whose examples are downloading and uploading of files such as software modules, profile, and application packages, and execution control of software module such as start, stop, suspend, resume, and so on.

7.3.2 Hardware abstraction layer

Hardware abstraction layer is a set of functions in software that link or emulate some platform-specific drivers, allowing software modules direct access to the hardware modules without modification of their codes. This standardized API increases the portability and reusability of these device drivers.

Hardware abstraction layer shall satisfy the following requirements:

a) Support hardware devices such as sensors, actuators, display device, power management, and controllers as platform independent models;

b) Use requirements and API defined in OMG Hardware Abstraction Layer for Robotic Technology (HAL4RT)(http://www.omg.org/spec/HAL4RT/).

7.3.3 API for exchange of information among software modules (aspects)

This API shall be used to exchange the information among software modules (or software aspects of modules), where information includes the values of variables, invocation of services (or functions), process of events, and files such as an executable code of software module, a profile, and a package. The variables, the services, and the events are defined in software modules (or software aspects of modules). Variables and parameters for service invocation shall be defined using data types in clause 7.2. The type of variables is classified into a periodic variable and an aperiodic variable. The type of services is classified into a blocking (or synchronous) service and a nonblocking (or asynchronous) service.

The protocols among two or more software modules (or software aspects of modules) is not specified because lots of international communication protocols exist. Note that remote access to the remote hosts is performed using the message format in clause 7.5 and is provided by the middleware. The middleware can also support exchange of information among software modules in the local host.

This API for exchange of information among software module (and software aspects of modules)shall support the following:

a) Read variables;

b) Write Variables;

c) Invoke Services;

d) Enrolment and Process of event;

e) Quality of Services is required for items a)-d). (e.g. safety-related values, Real-time characteristics, Security)

© ISO 22166-1 – All rights reserved

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

失敗



- 2018年5月のISO京都会議にて、OMG HAL4RTへの参照が含まれるISO/CD22166-1の改善のための議論が行われたが、HALに対して産総研から意義が出たために、HALの記述が削除されることになった。
- 2018年6月のOMG技術会議にて、HAL4RT 2.0 RFC(Request For Comment)文書案についてOMG Robotics-DTF全体会議でレビューおよび投票を行った結果、賛成5（JASA、富士通、MITRE、PTC、SPARX）、反対4（JARA、産総研、電通大、芝浦工業大学）で可決された。その後、アーキテクチャ委員会でレビュー及び投票を行ったが、産総研から意義が出たために、投票は否決された。

OMG HAL4RT 2.0 RFCへの反対意見



- ロボット用では、RTミドルウェア(OMG RTC)、ROSがすでにある
- 既存の標準との違いが不明確である
- 議論が足りない
- ロボットに特化した仕様ではなく、ロボット用(for Robotic Technology)とは言えない
  - 仕様名称にRTが含まれなければ、何ら問題ない
  - HAL4ET(Embedded Technology)?
- 実績が少ない
  - 商用製品は、アップウィンドテクノロジー社の二足歩行プラットフォームロボットUTRX-17-1のみ

解決策



仕様の範囲の明確化



推進体制の強化



豊富な実績

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

活動の成果



OpenEL 3.1仕様書を作成した。



OMG HAL4RT 2.0 RFC文書を作成した。



OMG HAL4RT 2.0 RFCに準拠したOpenEL 3.1を公開した。(https://github.com/openel/)



OMGのWGやタスクフォースの共同議長や議長を経験することができた。世界各国・各社の出席者と議論することができた。



ISOの日本のエキスパートとして、ISOに規格案を提出し、世界各国のエキスパートと議論することができた。



これらの結果、JASAとして国際標準化活動を推進するノウハウ、スキル、ナレッジ、経験を得ることができた。



経済産業省をはじめ各団体に対するJASAのプレゼンスが向上したと考える。

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

OPENEL WGのご紹介

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

OpenEL WGの活動概要



■ OpenEL WGでは、OpenELの仕様の強化、OpenELの国内外における普及、OpenELの国際標準化の可能性の調査を行います。

1. OpenELの仕様の強化
   - OpenEL バージョン3.2または4.0仕様書の発行（成果物：文書）
   - OpenEL バージョン3.2または4.0の実装例のGitHubでの公開（成果物：ソフトウェア）
   - OpenEL バージョン3.2または4.0のデモンストレーション（成果物：システム）
2. OpenELの国内外における普及
   - OpenELの推進のための強固な体制を構築します。
   - OpenELに関する情報をホームページなどで広く公開します。
3. OpenELの国際標準への提案
   - OpenELを国際標準とするための提案先や提案方法の調査を行います。

OpenEL WGの活動予定

WG(OpenELのステアリング・コミッティ)開催

・OpenELの仕様策定、普及・啓発およびその他の活動計画の立案、意思決定を行います。また、国際標準化の可能性についても検討します。WG傘下に、アクチュエーターSWG、センサーSWGを設置します。

アクチュエーターに関する技術調査(SWG会議)

・OpenELのアクチュエーターに関する仕様策定、アクチュエーターに関する技術の調査、講師を招いての勉強会、OpenELのアクチュエーターに関する仕様書執筆、実装などの具体的な作業を行います。

センサーに関する技術調査(SWG会議)

・OpenELのセンサーに関する仕様策定、センサーに関する技術の調査、講師を招いての勉強会、OpenELのセンサーに関する仕様書執筆、実装などの具体的な作業を行います。

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

OpenEL WGの開催日時



- 毎月1回（第4火曜日15:00～18:00、JASA会議室）
- 4/23（火）、5/28（火）、6/25（火）、
  7/23（火）、8/27（火）、9/24（火）、
  10/29（火）、11/26（火）、12/24（火）、
  2020/1/21（火）、2020/2/25（火）、2020/3/24（火）

OpenEL WGからのPR

- いくら世界に勝る技術力があってもルール作りで負けてはビジネスになりません。また、ルールは政府に作ってもらうものではありません。
- そこで、本WGでは、自分たちの手でOpenELというルールを作り、オープンイノベーションを起こします。
- 我々と一緒にゲームチェンジャーとなって、世界市場を狙いましょう。
- メーカーやデバイスベンダー、OpenELを顧客に提案したい方々だけでなく、OpenELを勉強したい方々やOpenELをすぐに使ってみたい方々の参加も歓迎いたします。

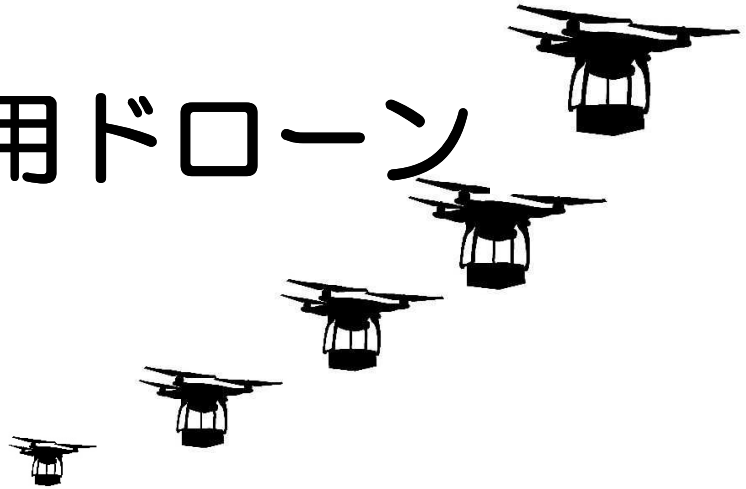
ご静聴ありがとうございました。

なぜオープンイノベーションは失敗したのか？
OpenELの失敗事例とその理由、そして解決策は？
2019/5/20 発行

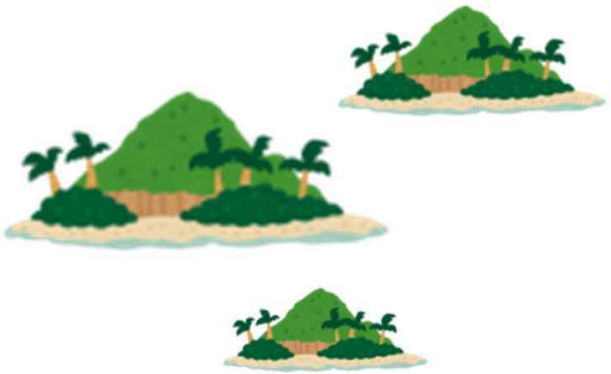
発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

「移動するIoT」産業用ドローンの機能安全を考える



～ 固定翼ドローン研究に向けて ～



2018年5月20日

(一社)組込みシステム技術協会
技術本部／IoT技術高度化委員会／ドローンWG
(株)金沢エンジニアリングシステムズ
小林 康博

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

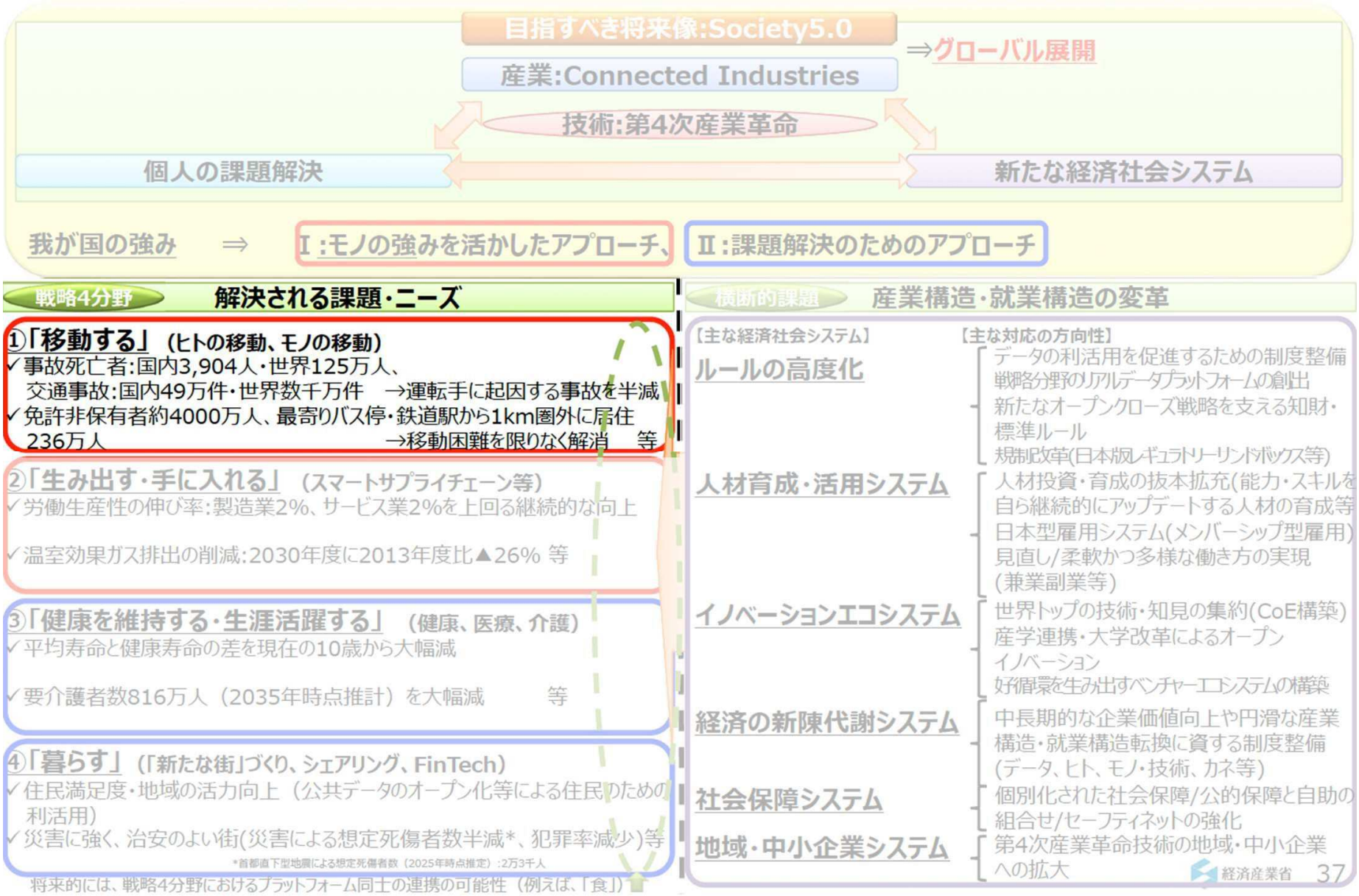
© Japan Embedded Systems Technology Association 2017

(一社)組込みシステム技術協会
技術本部 IoT技術高度化委員会
ドローンWG

１．活動概要

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

我が国の基本的戦略の全体像



© Japan Embedded Systems Technology Association 2017

日本の立ち位置 「移動する」

現状

※SAEレベル4 高度運転自動化：システムが全ての運転タスクを実施（領域限定的）で、予備対応時において利用者が応答することは期待されない

- 国内外の事業者は、**2020年頃のレベル4*自動走行車両の実現を目指し、ソフト系とハード系の事業者が強みを補完するために提携する動きが日常化**（日産とDeNA、FordとUber等）。
- とりわけ競争の鍵となる認知技術を中心とした自動走行技術開発については**サプライヤが競争をリード**（Bosch、Continental、Intel（mobileye）等）。更に、情報処理を支える**AI半導体・システムも、共通の情報処理基盤**として今後重要。
- 勝ちの絵姿が不透明な中、海外各国は、自動走行車両のいち早い実用化を目指し、**国内ルールの検討や、公道実験が可能な環境を整備し、自国内における自動走行技術開発の加速と移動関連データ取得等を奨励。日本政府も対応を加速。**
- ドローンについては、本体やコントローラー開発では海外企業（DJI、Intel等）が先行する中、ドローン単体の衝突回避等の技術開発に加えて、**複数ドローンを制御する運航管理システムの開発や、物流事業者やEコマース事業者などによる、事業性のあるサービス開発が重要に。**

日本の立ち位置・アプローチ

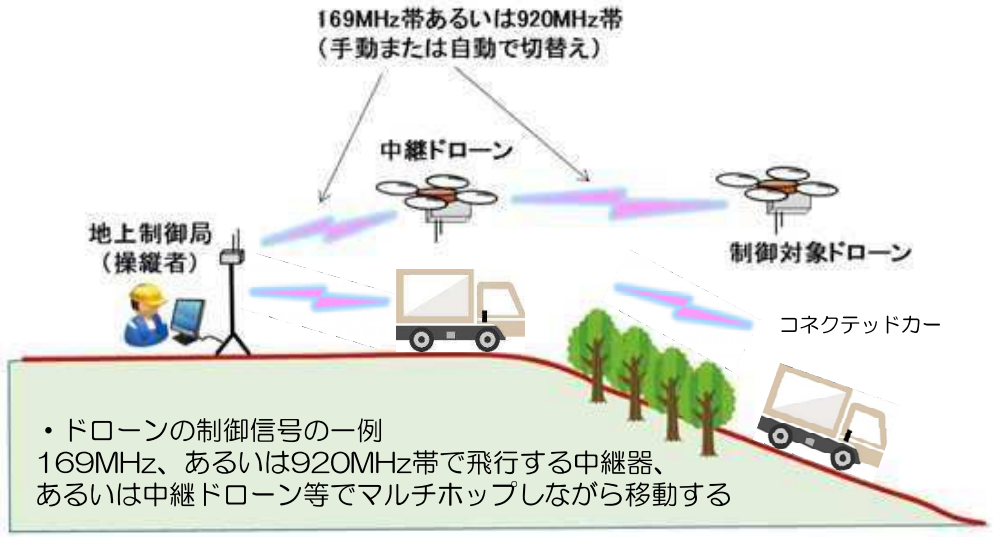
- 日系自動車OEMの世界シェア3割から生み出される運転制御・カメラ情報等の多様なリアルデータ、自動走行に必要な認知・判断・操作技術に係る日系サプライヤのグローバルな存在感など、**課題解決への貢献に必要なリソースを日本は有しており**、ひいては経済成長につながるポテンシャルを有している。ドローンについては、防災・物流など明確な出口分野が国内に存在。
- 日本がいち早く課題解決に貢献していくため、取得可能なリアルデータを元に、**（ⅰ）自律的なエッジ**（自動走行車・隊列走行車・ドローン等）**を実現**し、エッジを最適運用し、いち早く社会実装して移動サービスモデルを確立するための**（ⅱ）基盤となるシステム・インフラ**を整備し、ダイナミックマップ等の移動に係る「リアルデータのプラットフォーム」を創出するとともに、併せて**（ⅲ）ビジネス環境・ルール**の障害を取り除くことが肝要。

© Japan Embedded Systems Technology Association 2017

「移動するIoT」とは何か？

JASAドローンWGの取り組み

経済産業省「新産業構想ビジョン」
『移動する』（ヒトの移動・モノの移動）に応える
ドローンをテーマに活動している



マルチホップ中継制御による通信経路 出典：情報通信研究機構プレスリリース

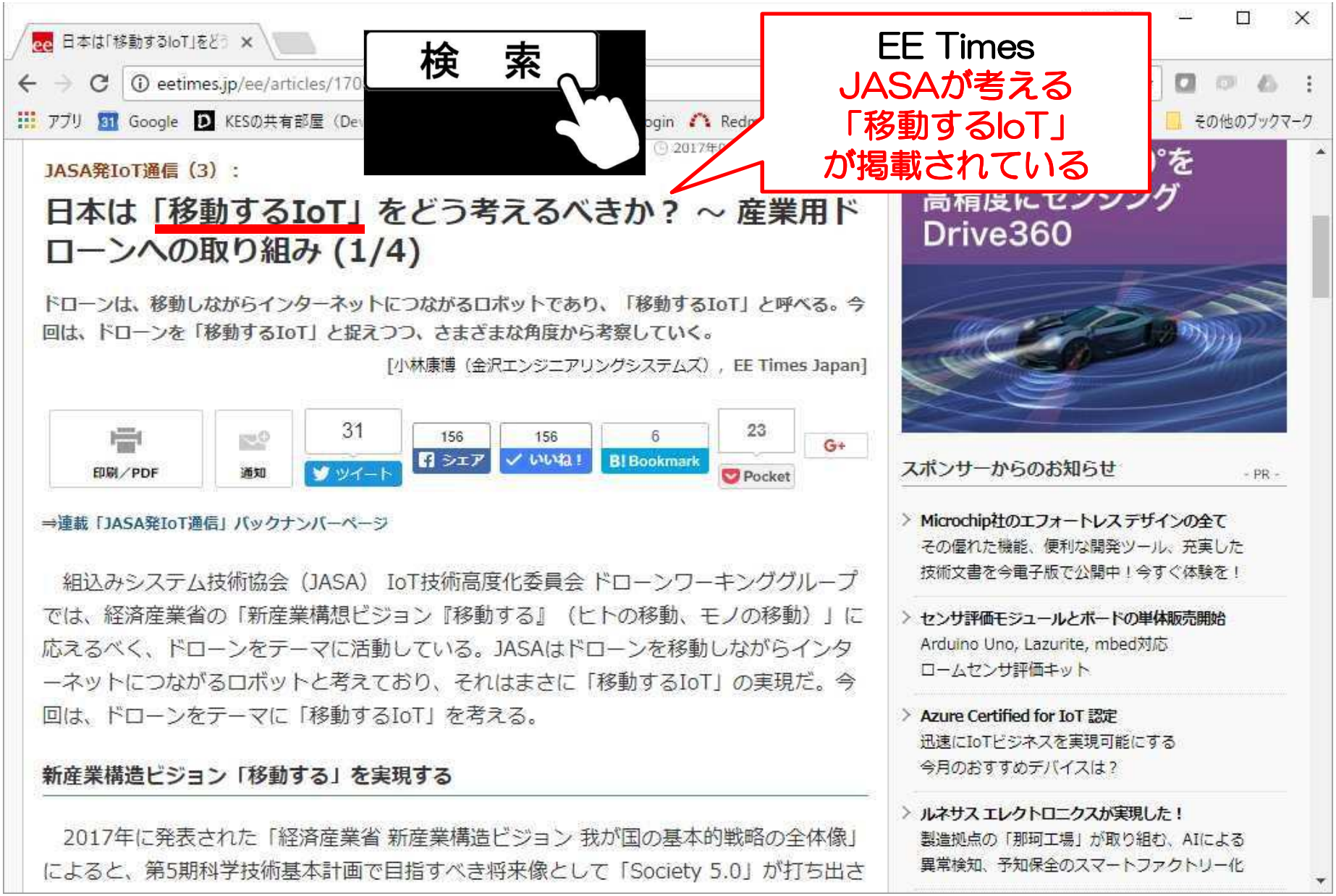
ドローンは移動しながら
インターネットに繋がるロボット

「移動するIoT」



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017


検　索
EE Times
JASAが考える
「移動するIoT」
が掲載されている
JASA発IoT通信（3）：
日本は「移動するIoT」をどう考えるべきか？ ～ 産業用ドローンへの取り組み（1/4）
ドローンは、移動しながらインターネットにつながるロボットであり、「移動するIoT」と呼べる。今回は、ドローンを「移動するIoT」と捉えつつ、さまざまな角度から考察していく。
[小林康博（金沢エンジニアリングシステムズ），EE Times Japan]
印刷／PDF
通知
31
ツイート
156
シェア
156
いいね！
6
B! Bookmark
23
Pocket
⇒連載「JASA発IoT通信」バックナンバーページ
組込みシステム技術協会（JASA） IoT技術高度化委員会 ドローンワーキンググループでは、経済産業省の「新産業構想ビジョン『移動する』（ヒトの移動、モノの移動）」に応えるべく、ドローンをテーマに活動している。JASAはドローンを移動しながらインターネットにつながるロボットと考えており、それはまさに「移動するIoT」の実現だ。今回は、ドローンをテーマに「移動するIoT」を考える。
新産業構造ビジョン「移動する」を実現する
2017年に発表された「経済産業省 新産業構造ビジョン 我が国の基本的戦略の全体像」によると、第5期科学技術基本計画で目指すべき将来像として「Society 5.0」が打ち出さ
Drive360
スポンサーからのお知らせ
- PR -
Microchip社のエフォートレス デザインの全て
その優れた機能、便利な開発ツール、充実した技術文書を今電子版で公開中！今すぐ体験を！
センサ評価モジュールとボードの単体販売開始
Arduino Uno, Lazurite, mbed対応
ルームセンサ評価キット
Azure Certified for IoT 認定
迅速にIoTビジネスを実現可能にする
今月のおすすめデバイスは？
ルネサス エレクトロニクスが実現した！
製造拠点の「那珂工場」が取り組む、AIによる異常検知、予知保全のスマートファクトリー化

新産業構造ビジョン「移動する」を実現する

「解決される課題・ニーズ」の筆頭項目 である「移動する」に応える。

1．産業用ドローン市場の立ち上げ

2．ドローンを題材にJASAの強みを活かした高信頼性、プラットフォームの設計

3．プラットフォームを<u>オープンソース</u>として提供・普及・拡大

JASA ドローンWG：http://www.jasa.or.jp/TOP/download/technical/droneWG.pdf

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

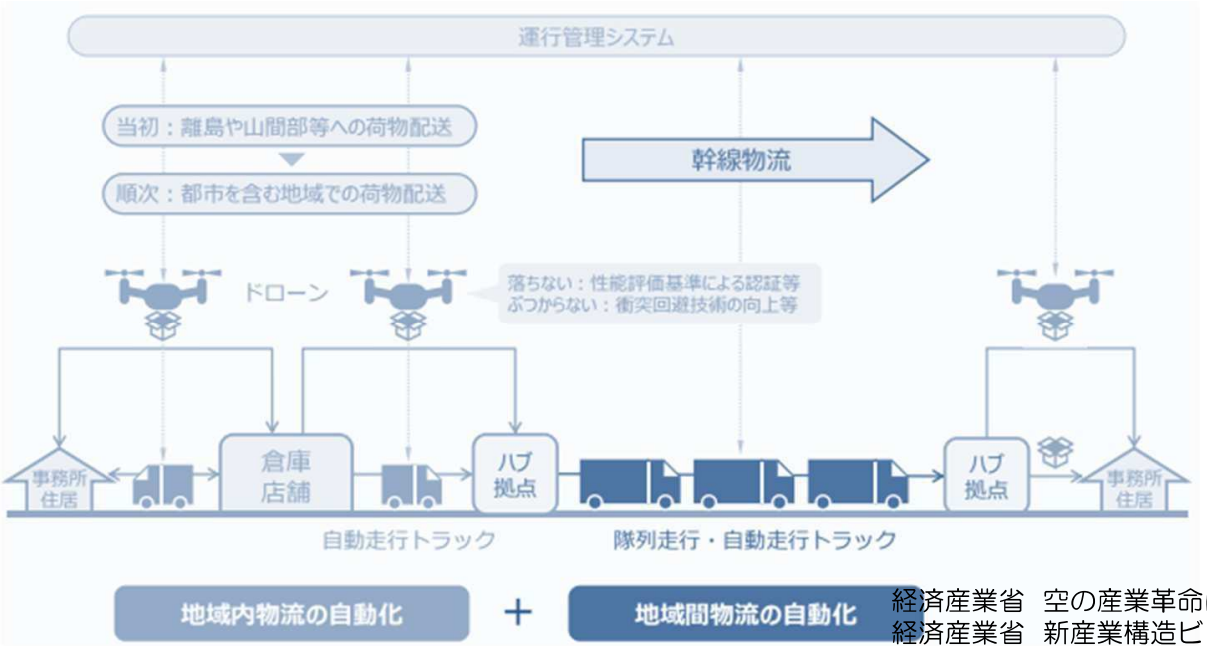
© Japan Embedded Systems Technology Association 2017

2. 離島に向けた取り組み

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

空の産業革命に向けたロードマップ

2017年　レベル１　目視内での操縦飛行
　　レベル２　目視内飛行（操縦なし）
2018年　レベル３　無人地帯での目視外飛行
2019年　レベル４　有人地帯での目視外飛行
　　：
　　：
　　：
2030年　レベルX　陸～空にわたる多様な輸送手段をシームレスに…



経済産業省　空の産業革命に向けたロードマップ（２０１７年5月１９日）
経済産業省　新産業構造ビジョン（２０１７年5月３０日）

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

空の産業革命に向けたロードマップ　小型無人機の安全な利活用のための技術開発と環境整備

平成29年5月19日
小型無人機に係る環境整備に向けた官民協議会



※ 各項目の詳細やその他の事項については補足資料（別紙）に記載する。[1]

© Japan Embedded Systems Technology Association 2017

長崎県 小値賀町（案）





メンバーの出身地で実証実験が可能、かつ、絶妙な大きさの島を発見！

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

ドローン が活躍する小値賀町（最終イメージ）

ドローンの町「小値賀町」構想

適度に離島間が離れており、実証実験・運用開始に理想的な環境と考える

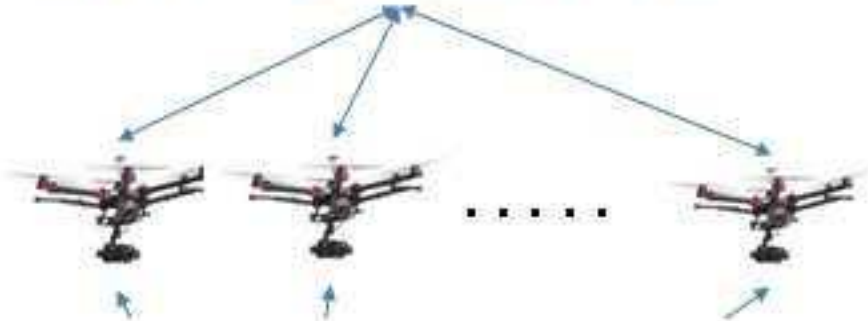


© Japan Embedded Systems Technology Association 2017

JASA Open Drone Platform

各社
運航管理サブシステム
を設計しサービスを提供

低軌道衛星通信（2020年実現）



・有事の際のフライトプラン変更
・GNSS含む各種センサーより状況を常時把握
・機体単体の故障診断



各地の離島にGWを配置



JASA Drone GateWay
通信距離50km実現
（Open Source）

各種ネットワークを使用し
オペレータへ

- **異常通知**
- **故障予測**
- **管制情報**
- **管制指示**

無人機航空管制（UTM）



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association



MCPC
モバイルコンピューティング推進コンソーシアム
Mobile Computing Promotion Consortium

３．利活用のユースケース

組み込みエンジニア
機能安全の観点から要件定義する

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

ドローンのシステム構造

産業用ドローンのサービス階層	関連団体・企業・規格
サービス層 	各サービスを提供する企業
UTM層 （無人航空機 管制システム） 	JUTM JAPAN UTM Consortium
SFP層 ストレージ層 	クラウド提供企業
Gateway層 オープンソース 	ARIB Association of Radio Industries and Businesses IoT Acceleration Lab MCPC モバイルコンピューティング推進コンソーシアム Mobile Computing Promotion Consortium IPA Drone Community JAPAN Association DCAJA OpenEL
フライトコントローラ層 オープンソース 国際標準 	一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association 組込システム開発企業

© Japan Embedded Systems Technology Association 2017

各省庁の目標達成へむけた利用シーンとは何か？

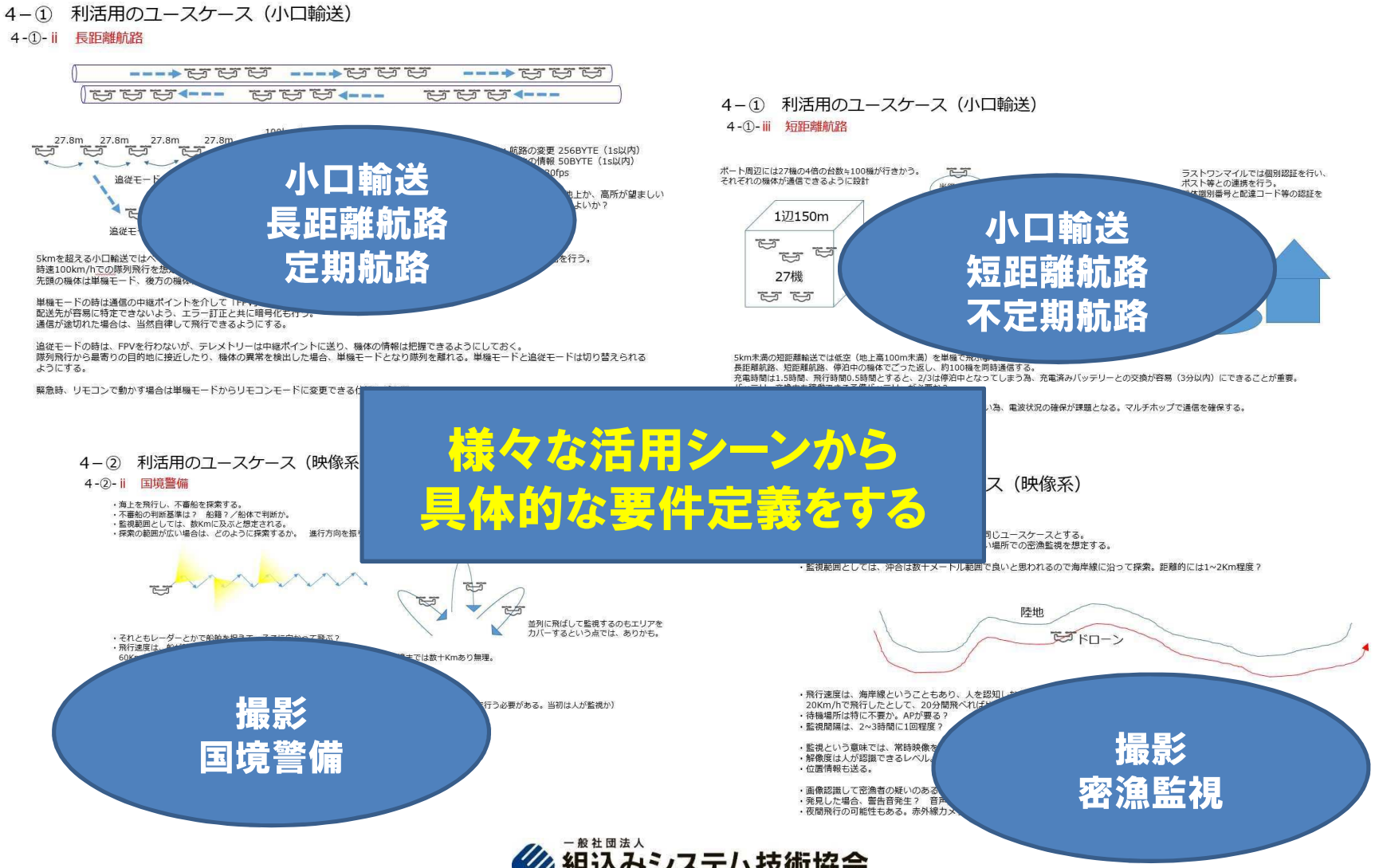
対象省庁の目標達成へむけた利用シーン（代表例１～２ケ）

		ロボット活用シーン　（機器＋ＤＡＴＡ）
総務省	日本再興戦略	健康長寿、産業力、 地域振興、国際競争力 安心できるﾛﾎﾞｯﾄ（電波） 政策と活用環境整備
経産省	新産業構造 ビジョン	移動革新 ｽﾏｰﾄｻﾌﾟﾗｲﾁｪｰﾝ等 価値創造　生産性革 快適、健康、介護 新たな産業 新たな街づくり、シェアリング、 フィンテック
農林水産省	ｽﾏｰﾄ農業政策等 ｽﾏｰﾄ1次産業	「農業 かつ低廉な農業資材の供給」と 「農産物流通・加工の合理化」（農業競争力強化支援法） 1次産業のＩＣＴ化（六次産業化・地産地消法） 海外出荷地方創生（ディスカバー農山漁村の宝）
国交省	ｉ ｺﾝｽﾄﾗｸｼｮﾝ	ＩＣＴ土木・防災
内閣府	未来投資戦略 2017 Society 5.0	移動革命、地方好循環、 健康革命、、世界に先駆けたｽﾏｰﾄｻﾌﾟﾗｲﾁｪｰﾝ・ｲﾝﾌﾗの生産性と都市の競争力の向上



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

利活用のユースケースの具現化



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

機体・無線等の要件（スペック）定義へ

No			1		2			
項目 ／ ユースケース			小口輸送		空撮			
			長距離航路	短距離航路	国境警備	密漁監視（海岸線沿い）	密漁監視（定点）	国有地・国立公園の調査
機体の性能	飛行速度		100km/h	30km/h	60km/h	20km/h	5km/h	20km/h
	飛行距離（片道）（速度と時間から算出とも限らない）					3km	2Km	←
	稼働距離					往復6Km（直線1-2Km）	↑	往復6Km
	高度					15-30m	←	←
	航続時間（移動込み）					←	←	←
	ペイロード				ーカー）	2kg（カメラ）	←	←
	頻度		フル稼働	←	3回／日	24回／日	48回／日	1回／月
	位置取得		GPS	←	←			
	制御（プロポ等）	用途	緊急時の操作	←	手動撮影			
		転送レート	100kbps？	←	←			

機体
飛行時間
ペイロード

要件定義から
産業用ドローンの要求仕様へ

No			1					
項目 ／ ユースケース			小口輸送					
				距離航路	国境警備	密漁監視（海岸	密漁監視	国有地・
無線通信	テレメトリ（+センシングデータ				←			
		品質			←			
	画像転送（低品質）FPV	用途	飛行の確認（安心）	←	人による大まかな判断			
		解像度	フルHD	←	←			
		フレームレート	15fps	←	←			
		品質要求（接続性）	低	←	高			
		用途	—	—	不審船検出	人		

通信速度
通信距離
チャンネル数

MCPCと無線についての
検証を行った

一般社団法人 組込みシステム技術協会
Japan Embedded Systems Technology Association

マルチユースの実現

①マルチユース（複数のことの同時実現） その1

マルチユース要素	今回の実証実験	今後の予定
1. 通信での情報提供	Wi-Fiでのアプリ通信性試験	用途ごとのアプリ通信（静止、すれ違い通信、リレー他）

検証シーン ドローンポートや特定通信エリアの上空飛行においてのアプリの通信をする。ドローンポートでは、ソフトの入れ変え、上空では計測情報を受信と今後の作業指示。



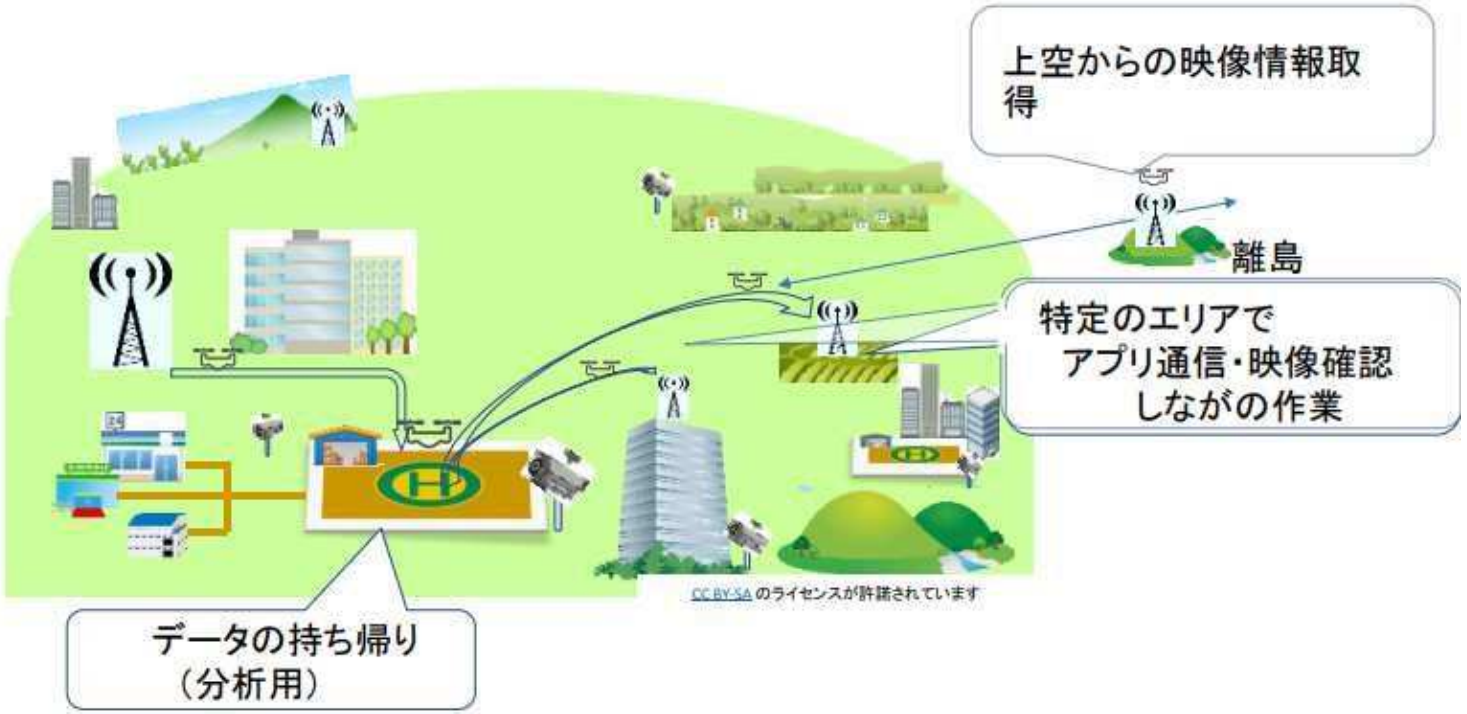
一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

マルチユースの実現

①マルチユース（複数のことの同時実現） その2

マルチユース要素	今回の実証実験	今後の予定
2. 映像	空撮	農林漁業毎や防災での映像活用 可視光を超えた映像活用

検証シーン　空撮でのDATA活用



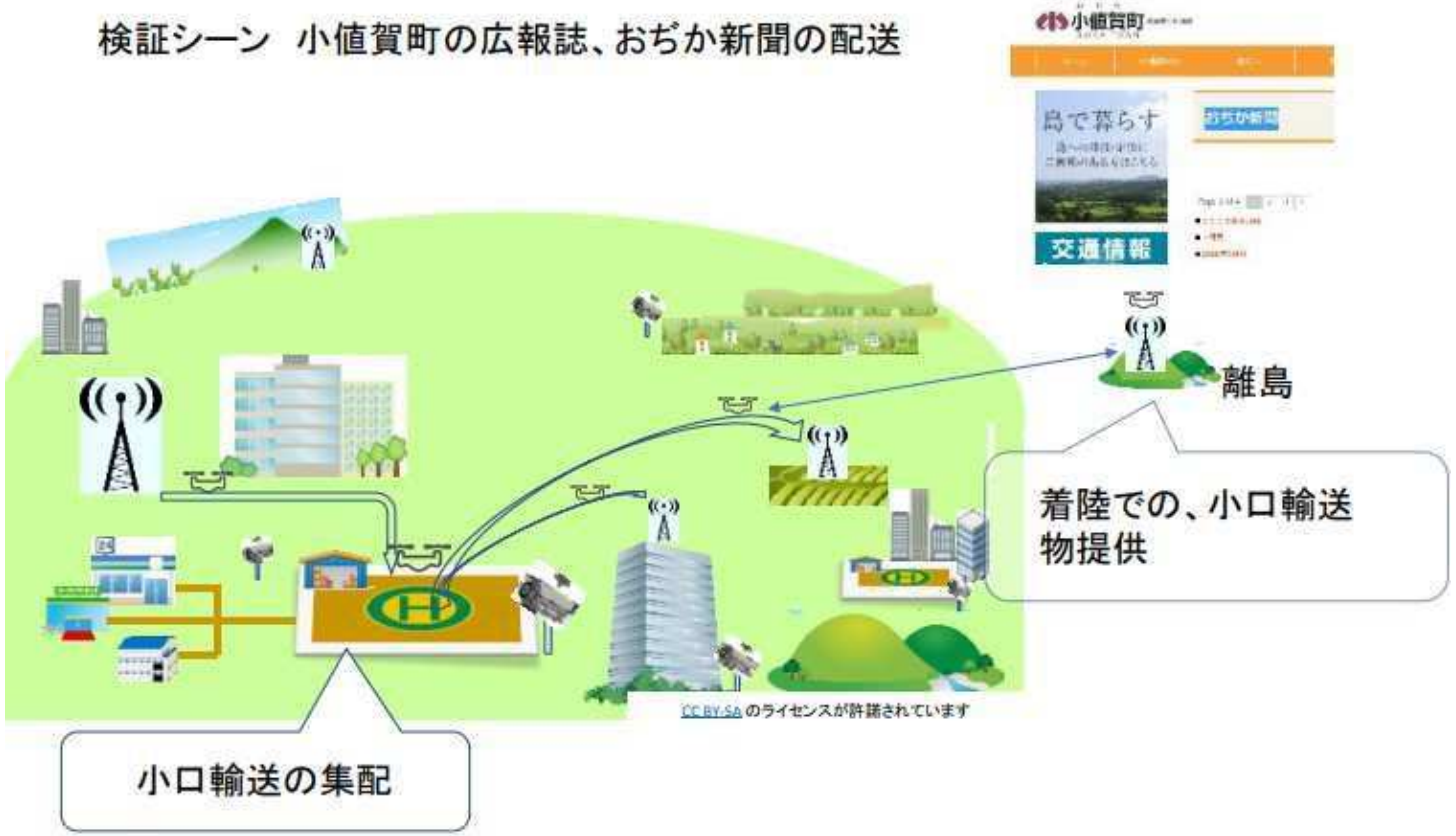
一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

マルチユースの実現

①マルチユース（複数のことの同時実現） その3

マルチユース要素	今回の実証実験	今後の予定
3. 小口輸送	町役場の地元 広報紙搬送	複数のフライト手段や他の自動 運転との連動

検証シーン　小値賀町の広報誌、おぢか新聞の配送



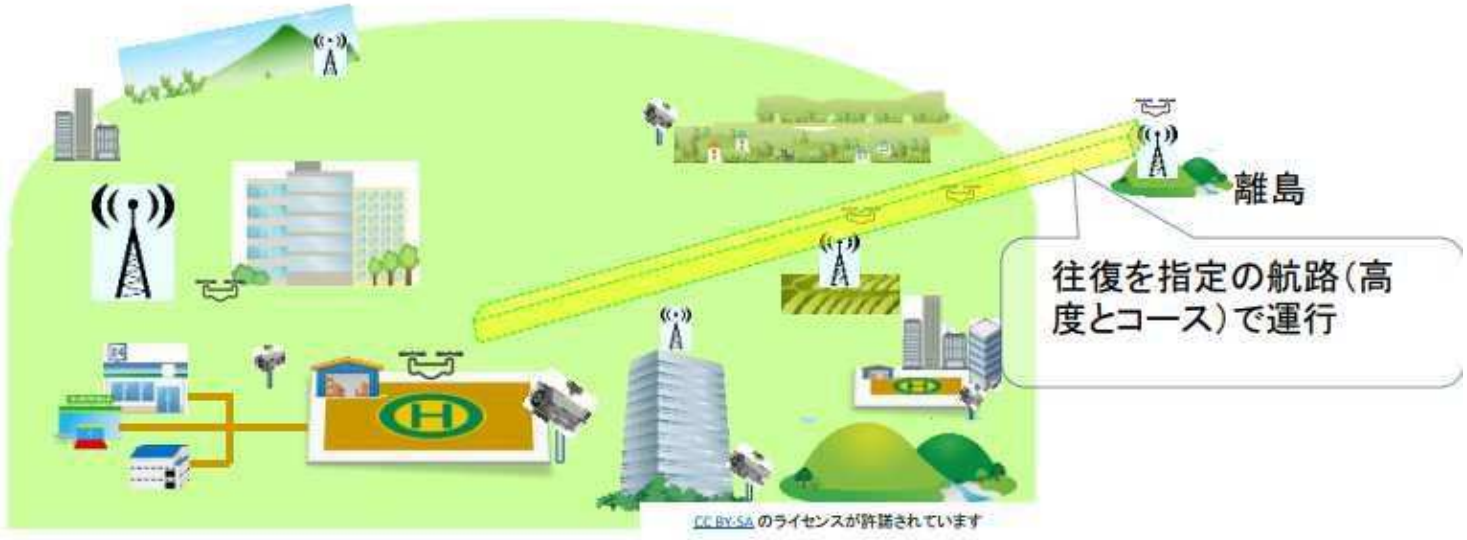
一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

マルチユースの実現

①マルチユース（複数のことの同時実現） その4

マルチユース要素	今回の実証実験	今後の予定
4. 複数機での航路活用	往復フライト試験	往復航路、登坂車線的復路活用 オープンソースでの協調フライト

検証シーン　飛行効率を考慮すると、航路を道路のように使い
集約的航空をするのが、有効
（上り、下り、登坂道路などの組み合わせへの発展へむけて
指定のコースを行き、戻ってくる



CC BY-SA のライセンスが許諾されています

長崎県 小値賀町での確認開始

（動画）



一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

MCPC
モバイルコンピューティング推進コンソーシアム
Mobile Computing Promotion Consortium



YRP
Yokosuka Research Park

(株)横須賀テレコムリサーチパーク

4．YRPでの無線の実証実験

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

LPWAテストベット＋横須賀ドローンフィールド

市街地エリア
・横須賀市役所
・久里浜駅、YRP野比駅、衣笠周辺
起伏帯エリア
・YRP
・横須賀市ドローンフィールド





一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

信頼できる飛行用通信　を　まず　検証。
　LPWA＋Wi-Fi通信で、ドローンを飛ばせる通信を　検証する

あわせて、はがき＆夕刊、写真　を　出来る範囲で　アプリとして実現

（　離島の小口輸送・タイムラグなく新聞が届く、漁場写真が確認できる。
　　これを、もう少し　島民が　ピンとくる　言葉に　直して　実検証に臨む）



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

白山ラジコンクラブでのLPWA試験



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

トリリオンノード

横須賀ドローンフィールドでのLPWA試験



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

MCPC レポート
ドローンのマルチユースについて

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

5．機能安全の研究から固定翼ドローンへ…

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

タッグを組む

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association



金沢工業大学

研究成果をインプット

- ドローンの要件定義
- 制度設計
- ニーズ
- 機体制作の基本伝授
- 機能安全を伝授

スピード感を活かした研究

- 適応制御とAI
- オソースで社会貢献
- 日本では真新しい固定翼ドローン

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

ドローンのレシピ



オープンソースを活用し、
ドローン業界のクリエイターの
裾野を広げる活動をすることで、
開発エンジニアを育てる



一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association



金沢工業大学
OSSコンソーシアム



一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

ドローンのレシピ

DCoJa
Yatagarasu Ver2を使った
自作ドローンのレシピ

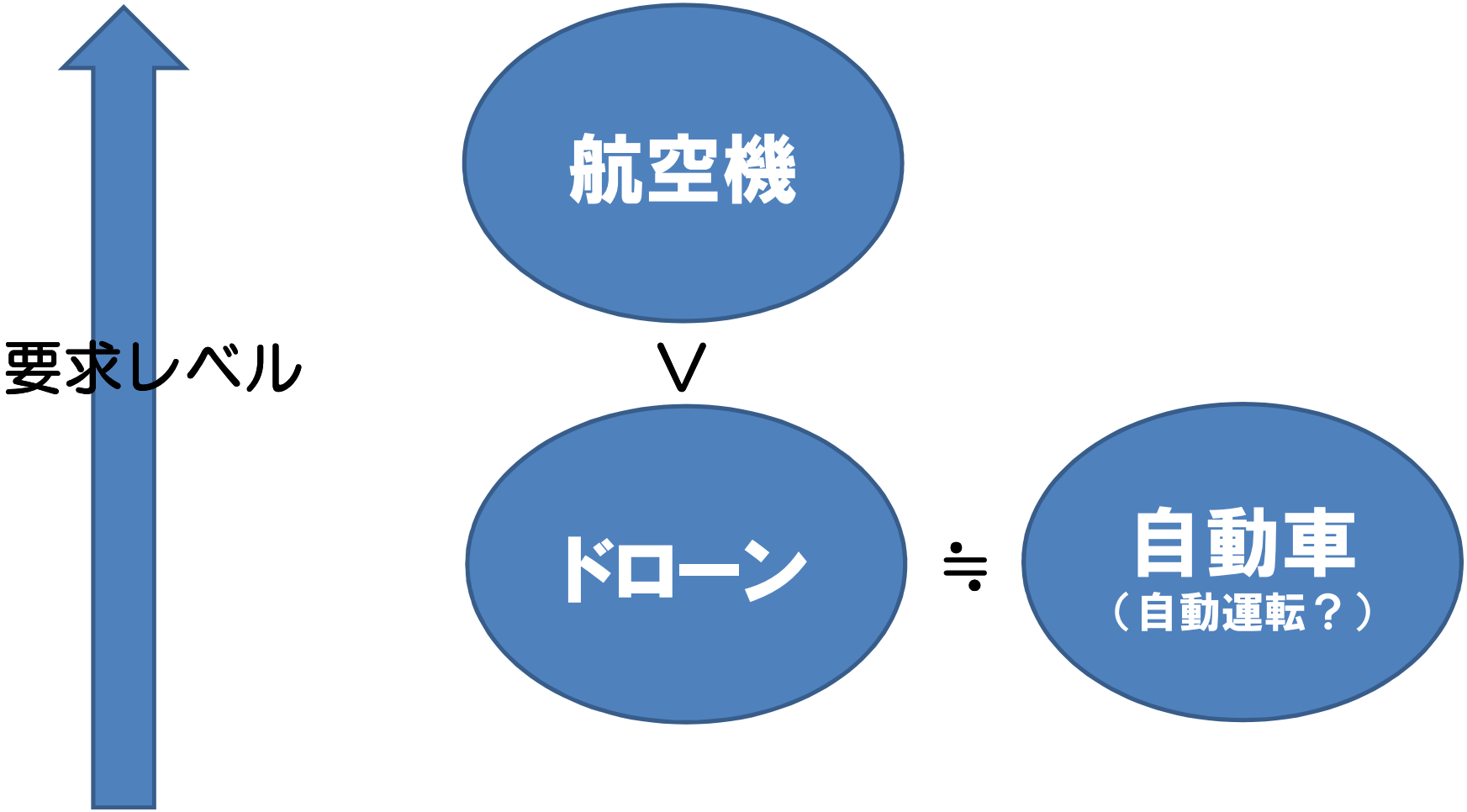
一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

金沢工業大学での自作

自作ドローン作成動画

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

ドローンの機能安全



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

ISO26262の安全度水準ASILのレベル分け

ASIL-AからASIL-Dの４段階が定められ、その割り当ては下記の３項目による。
障害によって被る危害の大きさの尺度（S）発生頻度（E)、障害発生時の制御難易度（C)

危害度	制御難易度 / 発生頻度	簡単 (C1)	通常 (C2)	不可能 (C3)
軽傷 (S1)	稀(E1)	—	—	—
	低(E2)	—	—	—
	中(E3)	—	—	A
	高(E4)	—	A	B
重傷 (S2)	稀(E1)	—	—	—
	低(E2)	—	—	A
	中(E3)	—	A	B
	高(E4)	A	B	C
致命傷 (S3)	稀(E1)	—	—	A
	低(E2)	—	A	B
	中(E3)	A	B	C
	高(E4)	B	C	D

▶ Source: https://www.ipa.go.jp/files/000004583.pdf 、表3.6

© 計画工学研究所

自動運転でASIL定義の再考必要か？

レベル 3,4 の自動運転では、「ドライバーが取り得る危険事象回避の可能性」が概して期待できないため、より高い ASIL の検討の可能性がある。

		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

レベル	名称	定義概要	安全運転に係る監視、対応主体
自動運転システムが（作動時は）全ての運転タスクを実行			
3	条件付運転自動化	システムが全ての動的運転タスクを限定領域'において実行 作動継続が困難な場合は、システムの介入要求等に適切に応答	システム（作動継続が困難な場合は運転者）
4	高度運転自動化	システムが全ての動的運転タスク及び作動継続が困難な場合への応答を限定領域において実行	システム
5	完全運転自動化	システムが全ての動的運転タスク及び作動継続が困難な場合への応答を無制限に（すなわち、限定領域内ではない）実行	システム

個々の自動運転車が有する性能及び使用の態様に応じた運行設計領域（ODD）を定め、走行環境や運用方法を制限し、自動運転システ ムが引き起こす人身事故であって、合理的に予見される防止可能な事故が生じな いことを確保する必要がある。

▶ Source: https://www.ipa.go.jp/files/000055801.pdf、http://www.mlit.go.jp/common/001253665.pdf

© 計画工学研究所

Drone Safety Integrity Level はどう定義するか？

制御難易度(C) → Fail-SafeからFail-Operationまでの難易度（F）で置き換えては？
障害によって被る危害の大きさの尺度（S）発生頻度（E)、Fail-Opeの要求or難易度（F)

		Fail-Safe 簡易 (F1)	Fail-Safe 車載並 (F2)	Fail-Ope 必須 (F3)
危害度	制御難易度 / 発生頻度	簡単 (C1)	通常 (C2)	不可能 (C3)
軽傷 (S1)	稀(E1)	—	—	—
	低(E2)	—	—	—
	中(E3)	—	—	A
	高(E4)	—	A	B

ASIL-A → D-SIL-1
ASIL-B → D-SIL-2
ASIL-C → D-SIL-3
ASIL-D → D-SIL-4

この定義は仮置き

© 計画工学研究所

Drone Safety Integrity Level (検討のたたき台)

ユースケース (大分類)	ユースケース(中分類)	障害物の可能性	想定される空域	提案する D-SIL	特記事項
小口輸送	長距離航路	低い	山間部・海・平地	D-SIL-2	自由落下防止は必要か
	短距離航路	高い	山間部・海・平地	D-SIL-2	自由落下防止は必要か
空撮	国境警備	低い	海	D-SIL-1	
	密漁監視（海岸線沿い）	低い	海	D-SIL-2	自由落下防止は必要か
	密漁監視（定点）	低い	海	D-SIL-1	
	国有地・国立公園の調査	高い	森林	D-SIL-3	発火防止（森林火災）
データ取集	見守り	高い	平地	D-SIL-4	
	インフラ監視・データ収集	高い	平地	D-SIL-4	防爆対策必要だろう
点検	定期航路	高い	山間部・海・平地	D-SIL-4	
	不定期航路（緊急事態）	高い	山間部・海・平地	D-SIL-3	

▶ Source: ドローン要件定義_V0.1(JASA Drone WG)、2018/11版

© 計画工学研究所

産業用ドローン立ち上げに向けて

１．組込みシステムを理解したエンジニアの意見

現在も日本の産業を下支えし、日本のストロングポイントである安心・安全を考えることが重要

２．オープンソースで産業を立ち上げる

産業用ドローンのプラットフォームをオープンソースで早急に立ち上げたい。
その為に仲間が必要。OSSワーキングと連携したい。

３．機能安全・セキュリティ要件

安全性向上委員会、情報セキュリティWGと連携し、セキュリティ要件定義の検討を始めた。
金沢工業大学と連携し、適応制御と機能安全の共同研究を行う。
JEITAとの連携を模索し、業界としての機能安全を検討出来ればと思う。

© Japan Embedded Systems Technology Association 2017

【講演タイトル「移動するIoT」をドローンで実現する為に…
～ 利活用のユースケースを考えた無線の実証実験～

2018/7/6 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

© Japan Embedded Systems Technology Association 2017

スマートライフにおけるQoL向上とエモーションデータの活用検討

スマートハウスにおける
スマート睡眠デモ事例紹介

2019年5月20日

IoT技術高度化委員会/スマートライフWG

國井 雄介

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

アジェンダ

1. スマートライフWG紹介
   - スマートライフWGの活動目的と位置づけ
   - スマートライフWG活動推移
2. 2018年度活動紹介
   - エモーションキャッチセンサ調査
   - スマートライフ QoL向上に向けた検討
   - スマート睡眠サービス検討
3. 2018年度活動実績
   - 東大 COMMAハウス　スマート睡眠デモ
   - ET2018　エモーションセンサフュージョンシステムプロトタイプデモ
4. 今後の活動について

© Japan Embedded Systems Technology Association

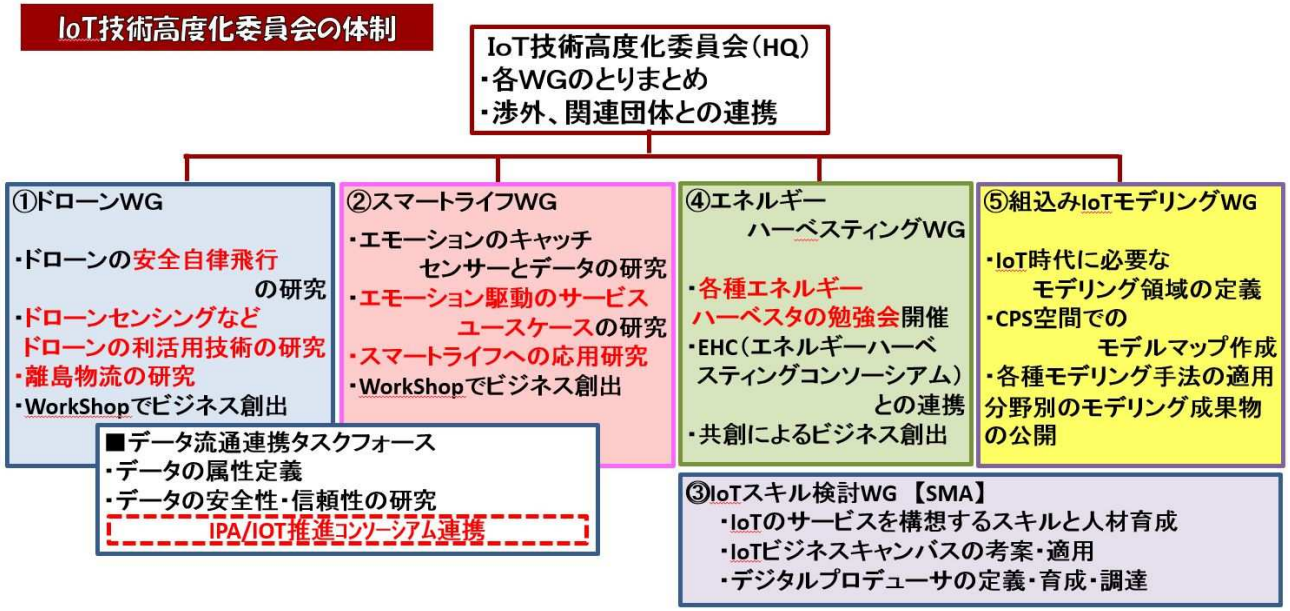
スマートライフWG紹介

© Japan Embedded Systems Technology Association 2017

スマートライフWG活動目的と位置づけ

活動目的

1. 生活における、さまざまなシーンの人の感情や状態に即したIoTサービスを実現するユースケースの研究
2. エモーションキャッチセンサとデータの研究
3. QoL向上のための生活用IoTの普及、課題解決を組込みの視点で検討



© Japan Embedded Systems Technology Association 2017

スマートライフWG活動推移

	2017 エモーションWG	2018 スマートライフWG
活動	エモーションキャッチセンサの調査 エモーションドリブンサービスの有用性の調査	エモーションキャッチセンサの調査 スマートライフ QoL向上に向けた検討 スマート睡眠サービス検討
成果	**ET2017** エモーションセンシングプロトタイプデモ展示 JASA IoT技術セミナー **EETimes Japan** 「感情とIoTを融合させた新しいサービスモデルの提案 」記事寄稿	**東大 生産技術研究所一般公開** 東大 COMMAハウス スマート睡眠デモ **ET-WEST2018** JASA IoT技術セミナー **ET2018** JASA IoT技術セミナー エモーションセンサフュージョンシステム プロトタイプデモ

本日は、2018年度の活動紹介

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

2018年度活動紹介

エモーションキャッチセンサ調査

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

エモーションキャッチセンサ種別

センサ種別	概要	特性
カメラセンサ（表情）	表情から、感情を識別	識別速度が速い。 画像での分類のため、誤認識がある。
バイタルセンサ（生体）	心拍、[illegible]情などから、	センシングデータの信頼度が高い。 人差があるため、チューニング で精度が上がる。 る。 判定にチューニングが必要となる。
マイクセンサ（音声）	音声抑揚から、感情を識別（音響解析型）	マイクを接続するだけで簡単に取得可能。 音声認識しにくい怒声なども検知することが可能。 音声（抑揚）からのみの場合、表面的。
	音声内容から感情識別（言語解析型）	テキストからの感情解析は精度が高い 複数の感情を表す言葉や、固有名詞を多く含む場合などは感情の判定が難しい。

エモーションセンサフュージョンシステムに利用

赤字：メリット
青字：デメリット

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

エモーションキャッチセンサ種別

センサ種別	概要	特性
脳波センサ（脳波）	脳波から、感情を識別	脳波から感情を識別するため、精度が高い。 脳波をセンシングするため頭部への接触が必要。 Rawデータから直接感情を識別することは困難。 個人差が大きい。
振動センサ（振動）		作するため、手軽にバイタルが 。 シングするため、ノイズに弱く、 リアルタイム性も低い。
サーモセンサ（体温）	温度（体温）から感情を識別	非接触でセンシングできるものある。 温度（体温）からのため、単体では識別が不可能。他のエモーションキャッチセンサと組み合わせる必要がある。

エモーションセンサフュージョンシステムに利用

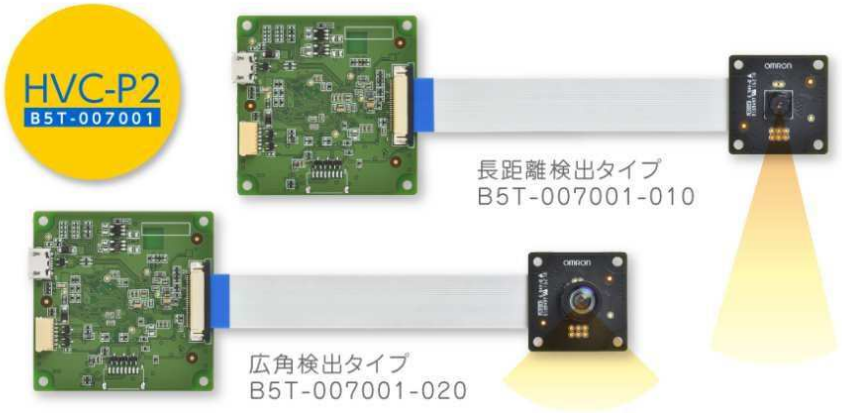
赤字：メリット
青字：デメリット

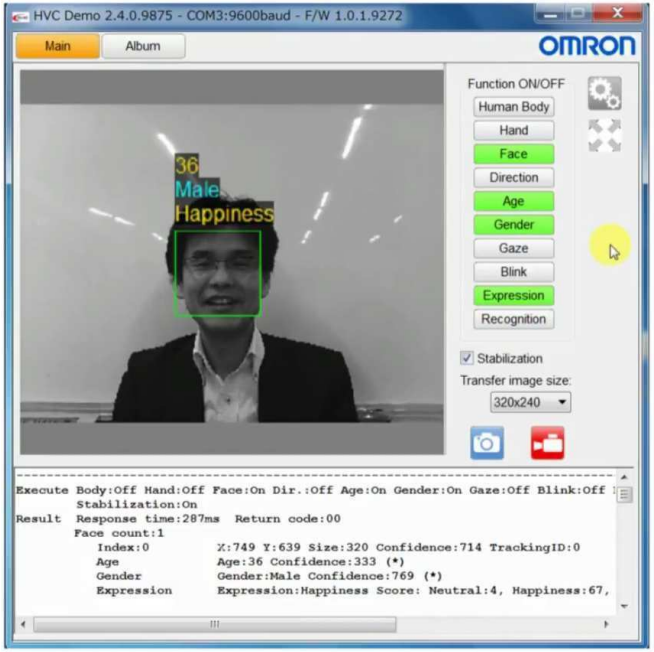
一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

カメラセンサ

- オムロン社製 HVC-P2
- 5表情（「真顔」「喜び」「驚き」「怒り」「悲しみ」）のスコア
- ポジティブ/ネガティブ度合い
- 年齢推定（年齢（0～75歳））
- 性別推定（男／女）



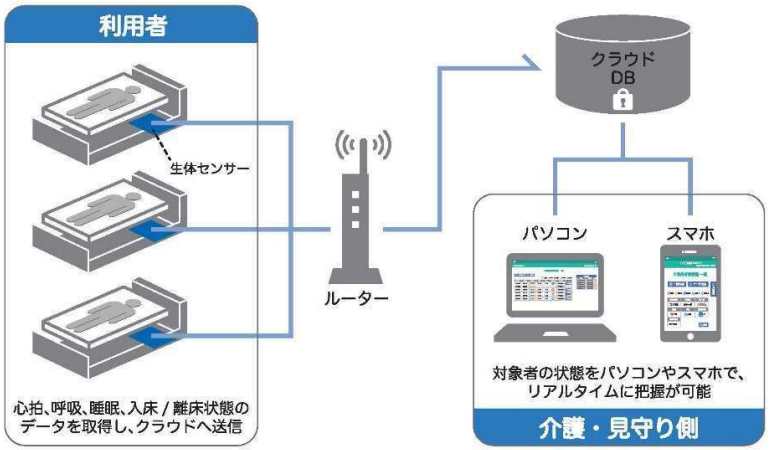


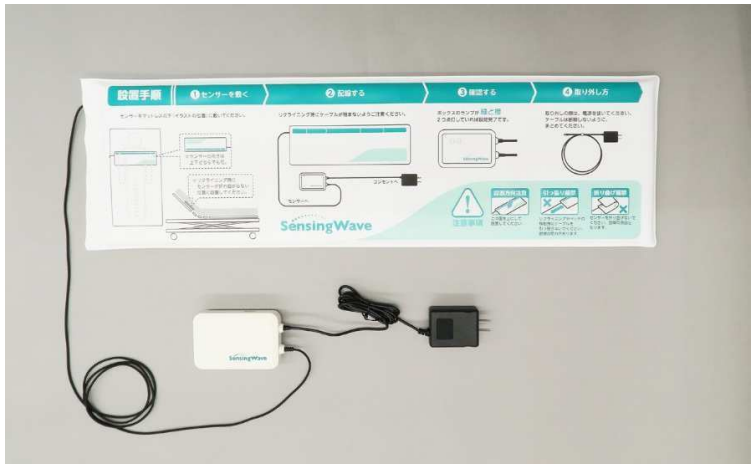
https://plus-sensing.omron.co.jp/product/hvc-p2.html

© Japan Embedded Systems Technology Association

バイタルセンサ

- 凸版社製非接触のバイタルセンサ
- 心拍や呼吸データを取得
- 覚醒、就寝、寝返りなど睡眠状態の取得が可能
- 生データとしてストレス値（リラックス度合い）も取得可能



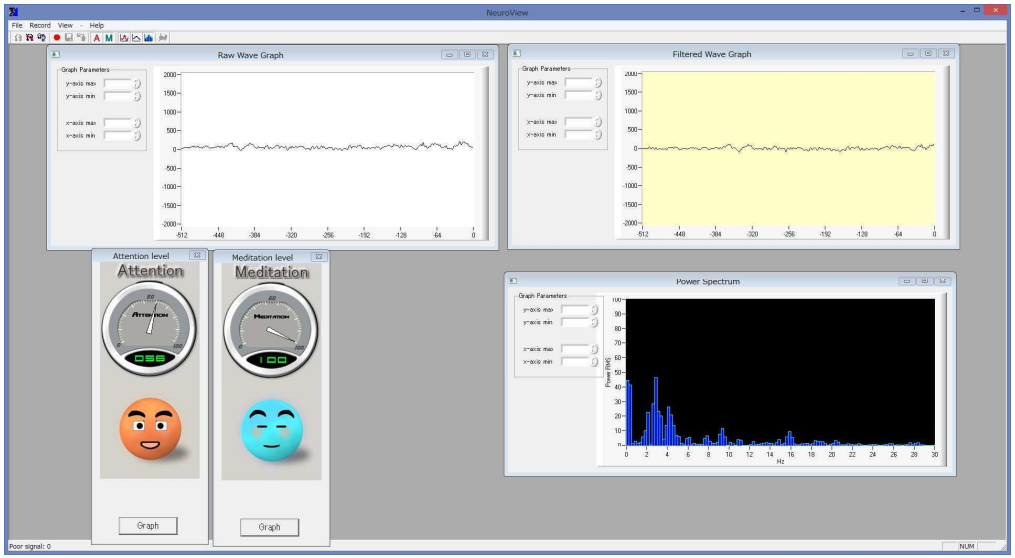


https://www.toppan.co.jp/news/2018/03/newsrelease180328.html

© Japan Embedded Systems Technology Association

脳波センサ

- NeuroSky社製 MindWave
- Rawデータ取得可能。
- 各種開発キットあり。
  （Android、iOS、Windows、非公式だけどPythonも）





http://www.neurosky.jp/products/

© Japan Embedded Systems Technology Association

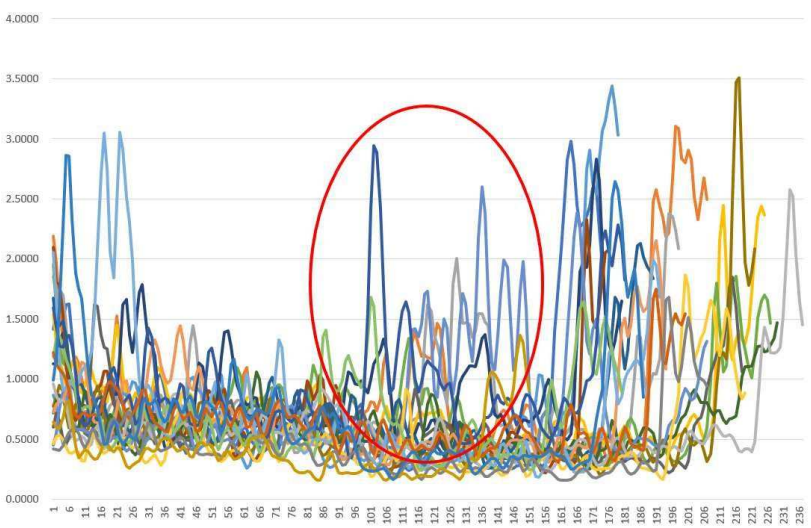
脳波センサ

- 脳全体に出現する「基礎律動（背景脳波）」を取得
- 耳とおでこの間の電位差をとって、基礎律動を測定
- delta, theta, lowalpha, highalpha, lowbeta, highbeta, lowgamma, midgammaの8種類を計測
- 独自アルゴリズムにより、ATTENTION（集中度）とMEDITATION（リラックス度）を算出

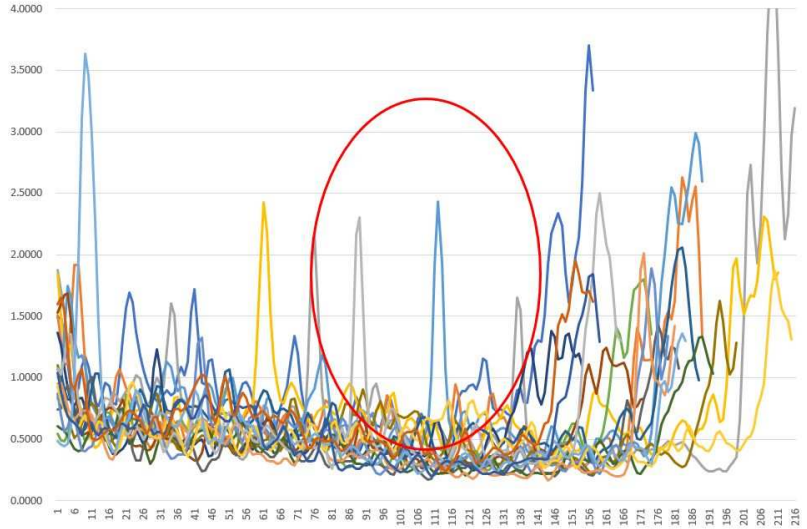
脳波のタイプ	周波数範囲	心理状態
デルタ	0.1Hz～3Hz	夢を見ない深い睡眠、ノンレム睡眠、 無意識
シータ	4Hz～7Hz	直感的、創造的、想起、空想、幻想、夢
アルファ	8Hz～12Hz	リラックス、ただし気だるくはない、平穏、意識的
低ベータ	12Hz～15Hz	リラックスしているが集中している、統合的
中ベータ	16Hz～20Hz	思考、自己および環境の認識
高ベータ	21Hz～30Hz	警戒、動揺

© Japan Embedded Systems Technology Association

脳波センサ



B波の変化（ババ）



B波の変化（ババ以外）

・脳波は、個人差が大きい
・有効な脳波を取得する為には慣れが必要
・Rawデータから、直接感情を読み取ることは難しい
・脳波は常に計測される為、時系列的な扱いが必要。

© Japan Embedded Systems Technology Association 2017

脳波センサ

脳計測機器		計測信号	対ノイズ	深部計測	画像
EEG	電位差	脳波 （神経群活動）	△	△	
NRIS	近赤外分法	血行動態変化	◎	×	
MRI	機能的磁気 共鳴画像	血行動態変化	×	◎	

脳波センサにも、様々なセンシングの種類があり、用途に応じて選択する必要がある。

© Japan Embedded Systems Technology Association 2017

2018年度活動紹介

スマートライフ QoL向上に向けた検討

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

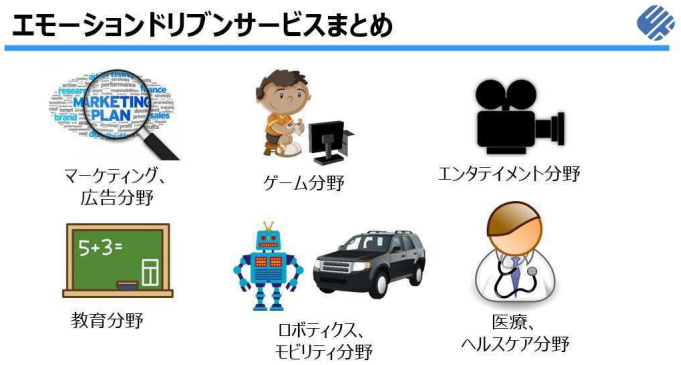
© Japan Embedded Systems Technology Association 2017

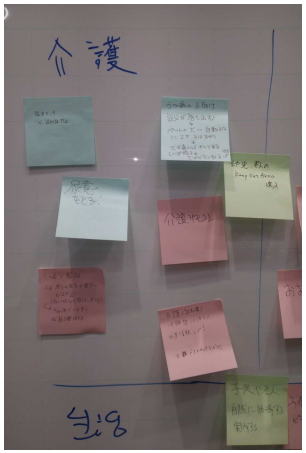
スマートライフ分野でのターゲット選定

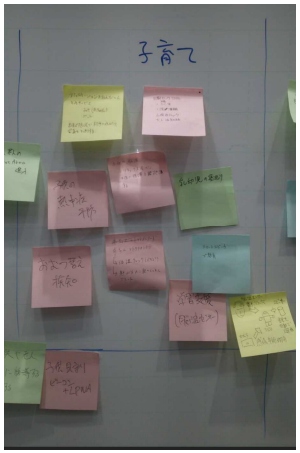
■スマートライフ分野でエモーションを活用したQoLを向上させるアイデアディスカッション

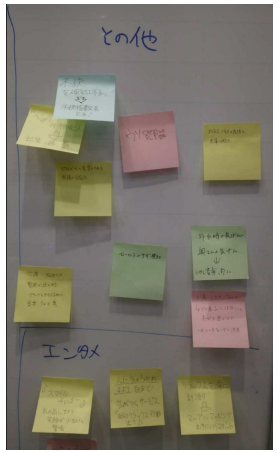
■生活
・雨を検知（ベランダにセンサー）
・寝苦しい夜のエアコン調節（温度＋リモコン）
・**睡眠の質をアップさせるためのサービス**
　脳波センサやシート型センサを使って
・子供や老人を自然に誘導する、案内する
■介護
・脳波センサで認知症防止
・うつ病の人向け
　気分が落ち込む→ペットの犬に自動的にエサ、またはおやつ
　→犬が喜んでやってくるしっぽ振る→エンドルフィン出る→治る
・介護リモコン
・尿意をとる
・幼児、老人のKeepOutArea侵入阻止
・うつ病×脳波
　→周りの空気で発言に敏感（ちょっとしたことで悲しむ、怒る）
　　→脳波でいち早く状態を伝える
・介護（認知症）
　→徘徊してしまう人の位置情報とかを取得。
　→ドアのロックをかける
■子育て
・コミュニケーション取れない人とのサービス
　- 子供（赤ちゃん）、ペット
　表情や脳波で相手に伝えたり、写真を撮ったりする
・自動ロックシステム
　- 柵、洗濯機
　- 子供はロック、大人はそのまま
・子供の熱中症予防
・子供の脳波
　- リラックス度から子供の機嫌を把握する
・乳幼児の寝返り
・おむつ替え検知
・赤ちゃん×トリリオンノード
　体温チェック、熱があれば親のスマホにアラート
・学習支援（脳波センサ）
・スマートスピーカで教育
・脳波センサ集中、リラックス＋カメラ（感情、年齢性別）
　RAPIRO＋スピーカ（音楽を自動で選曲）

などなどブレストでたくさんのアイデアが！









一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

睡眠サービスがなぜ重要か？



日本は世界有数の不眠大国



高血圧、糖尿病など生活習慣病だけでなく、
うつ病などのメンタル疾患を引き起こす原因



睡眠時間が短いなら
質の高い睡眠をとる必要性がある

皆さんは良く眠れていますか？
睡眠時間は足りてますか？

© Japan Embedded Systems Technology Association

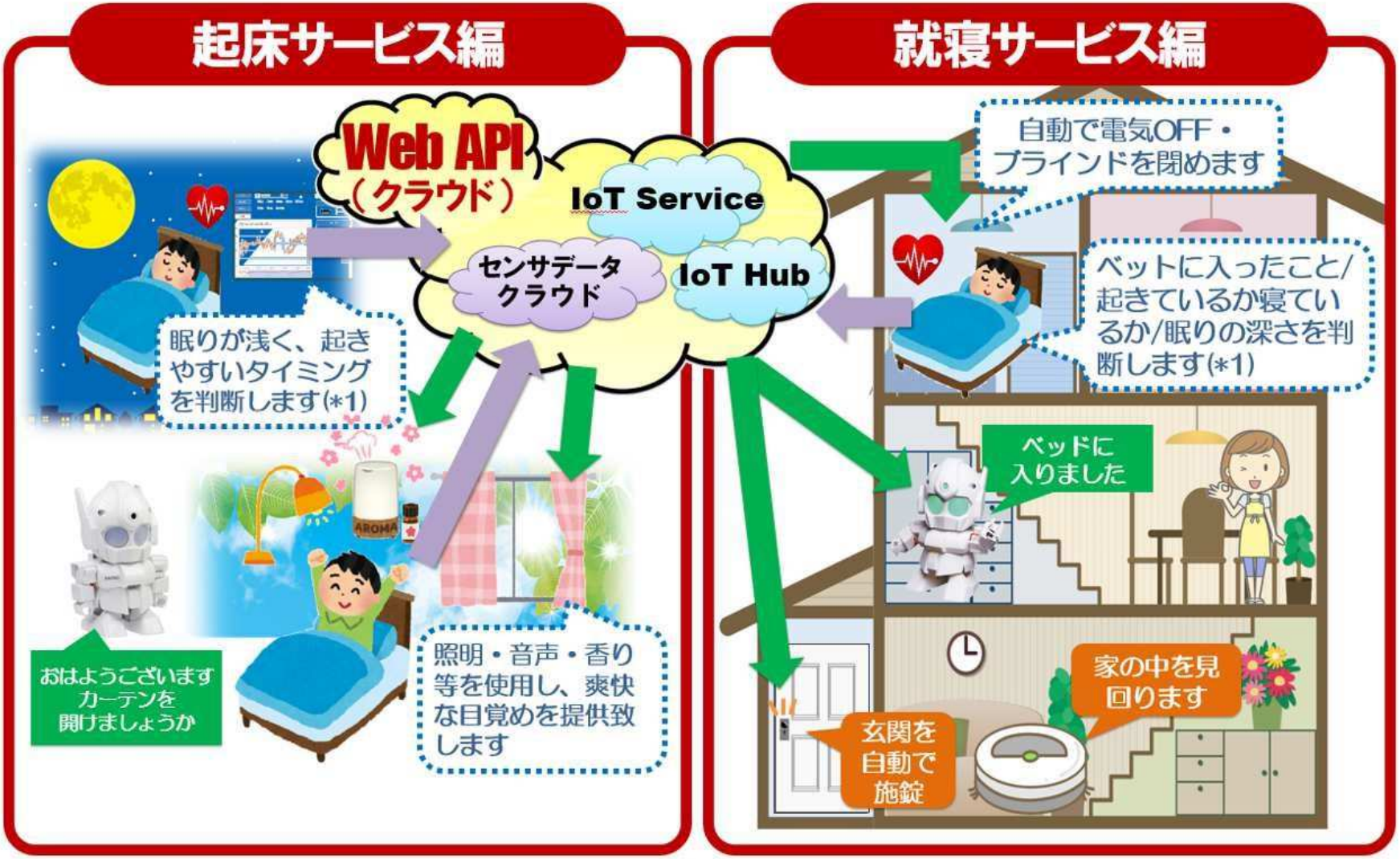
2018年度活動紹介

スマート睡眠サービス検討

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

睡眠サービス実現例



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

2018年度活動実績

東大 COMMAハウス　スマート睡眠デモ

© Japan Embedded Systems Technology Association 2017

東大 IoT特別研究会 （RC88）との連携

■ IoT特別研究会

- IoT社会の早期実現に向けた具体的方策を検討
- 「東大駒場リサーチキャンパス公開2018」（ 2018年6月8～9日）
- COMMAハウス（スマート実証ハウス）にて展示・発表

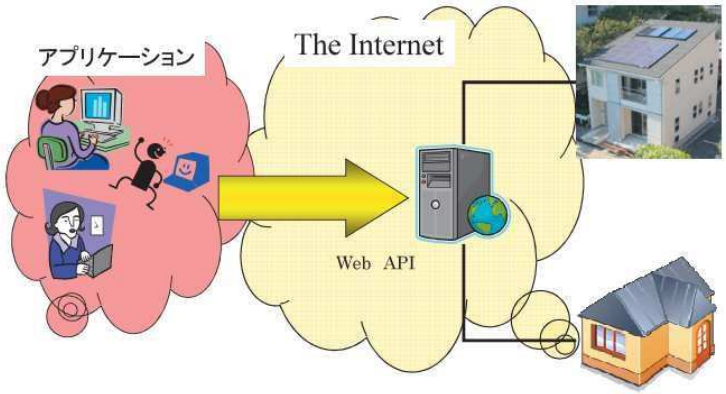


“COMMAハウス”

東京大学
生産技術研究所
スマート実証ハウス
（東京都目黒区駒場）



テストベッド構成図



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

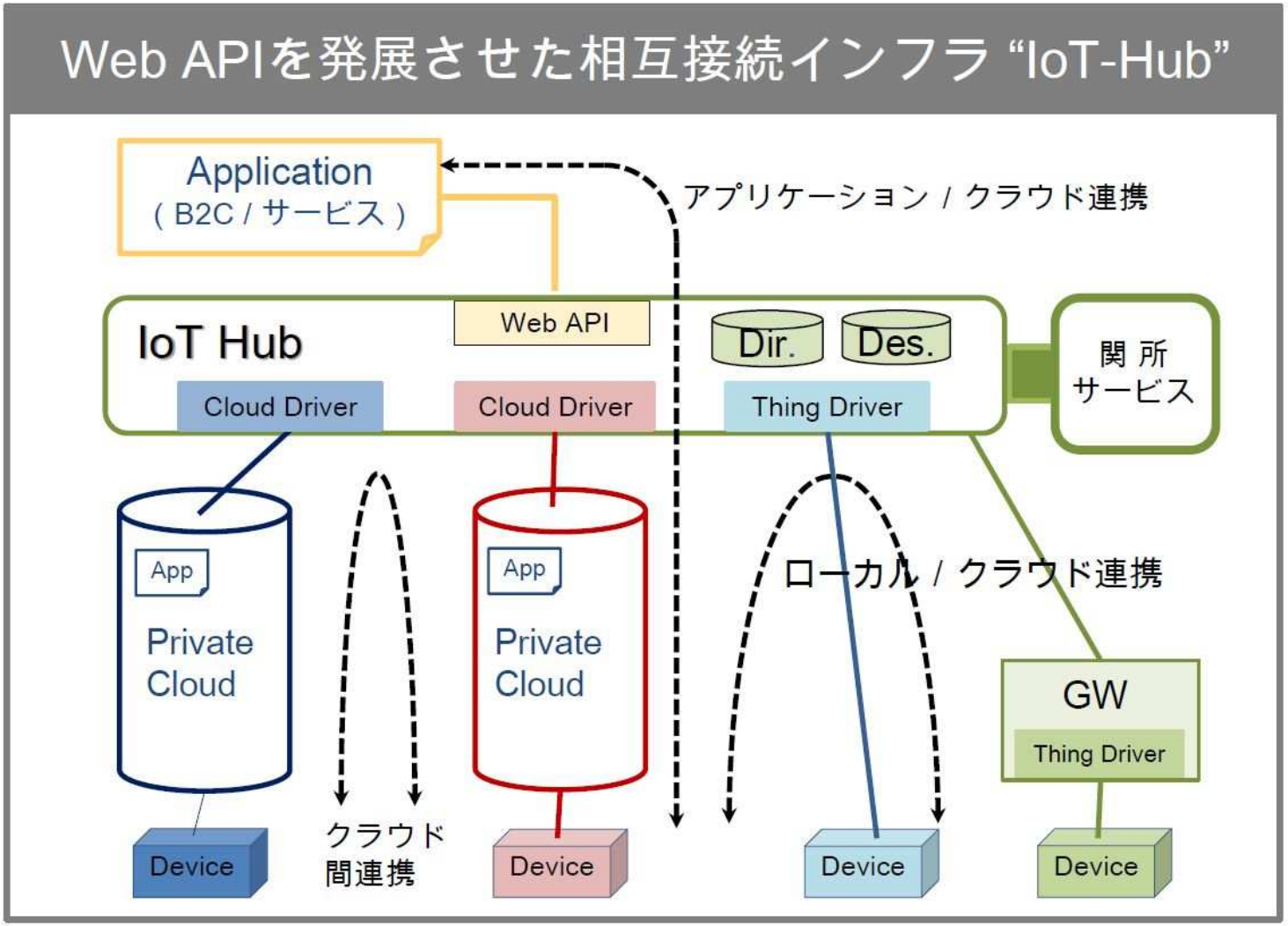
© Japan Embedded Systems Technology Association

東大COMMAハウスでのデモ動画



© Japan Embedded Systems Technology Association 2017

東大 IoT特別研究会 （RC88）との連携



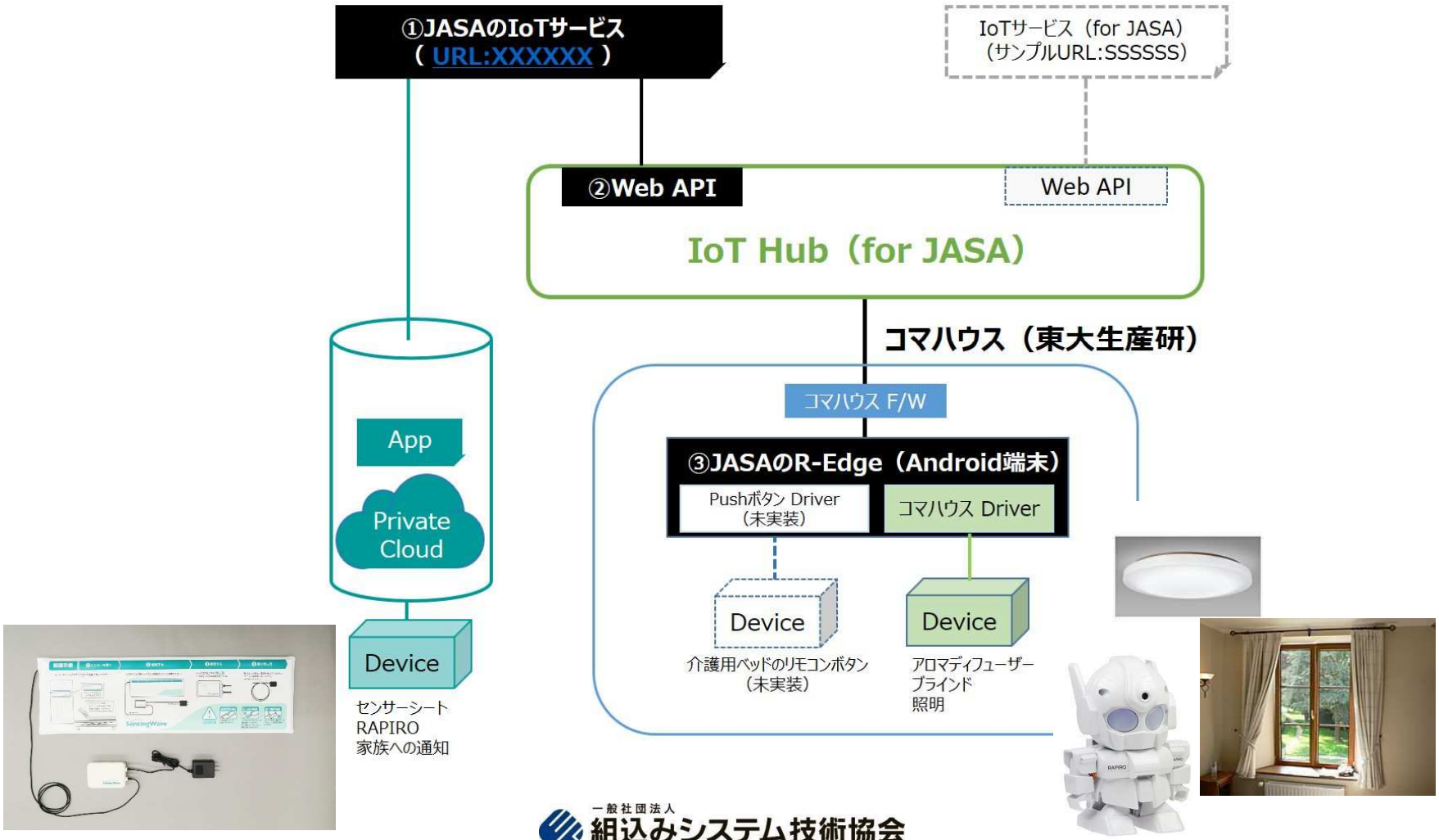
東京大学生産技術研究所提供

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

COMMAハウス展示デモシステム構成

IoT 相互接続構造図（相接WebAPI）



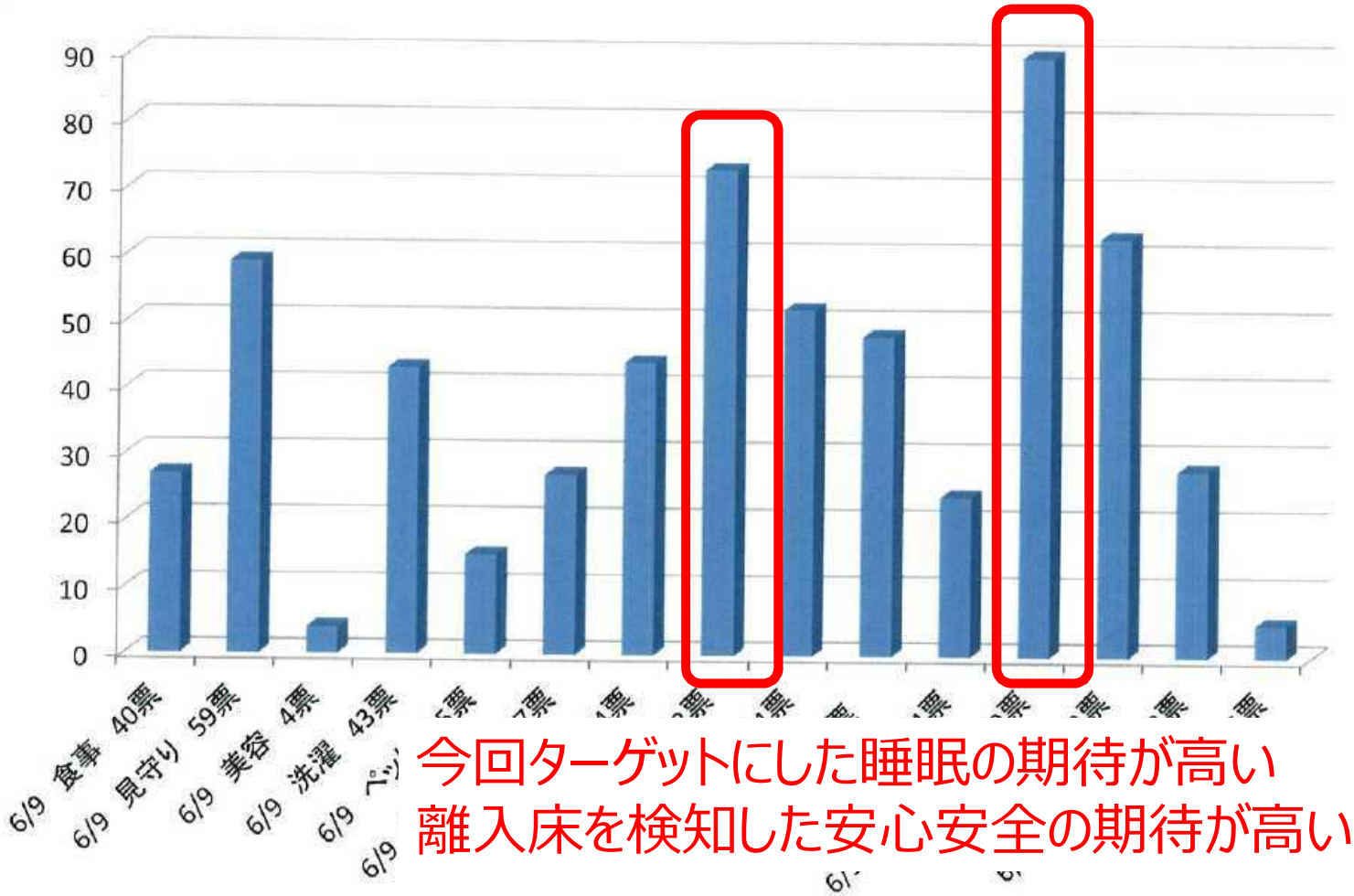
一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

24

© Japan Embedded Systems Technology Association

COMMAハウス展示アンケート結果

来場者アンケート（今後の生活シーンへの期待）



今回ターゲットにした睡眠の期待が高い
離入床を検知した安心安全の期待が高い

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

COMMAハウス展示デモアンケート結果

■ 個別アンケート回答

- 空間の明るさ、暗さをコントロールできるのが面白い。　（一般　30代男性）
- 睡眠サービスについては個人的にとっても欲しい。　（東大関係者　20代男性）
- 発達障害の子育て、夜叫症、不眠症、解離障害の子育てへの応用を期待。
  （一般　40代女性）
- 朝、気持ち良く起きられるような技術を使いたい。　（20歳未満男性）

■ 全般的な評価

- ロボットが優しい声で起こしてくれるのが良い。
- ロボットの目の表情が変わるのが、愛嬌があって良い。
- 遠隔地に住む家族の生活状況を見守れるのが便利で安心できる。
- 操作するのを忘れても安心。

© Japan Embedded Systems Technology Association

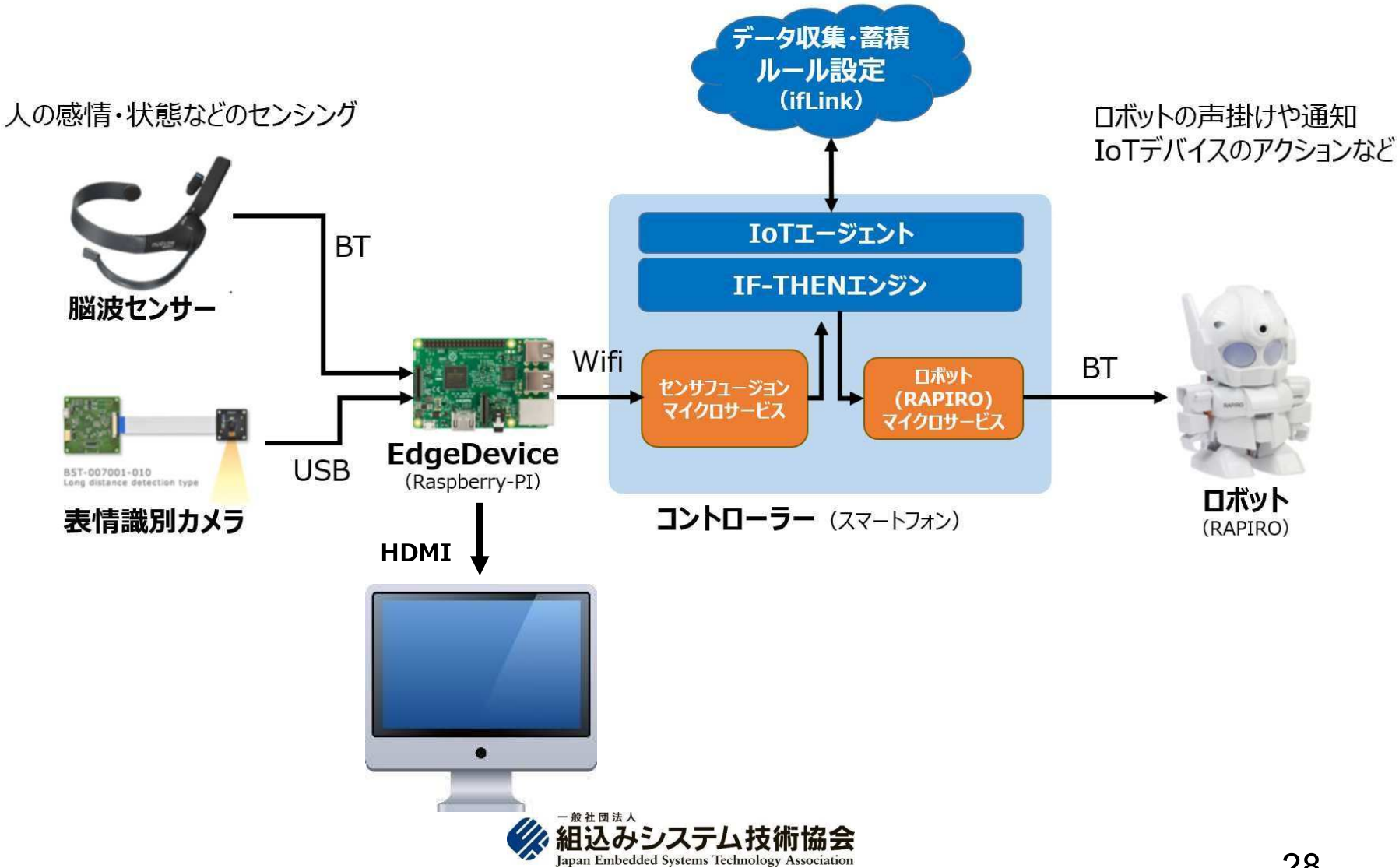
2018年度活動実績

ET2018　エモーションセンサフュージョンシステムプロトタイプ

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

エモーションセンサフュージョンシステムプロトタイプ構成



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

エモーションセンサフュージョンシステム利用例

■　受験勉強支援

・集中度を検知
・居眠り検知
・リラックス検知
・睡眠の質検知
・せき、くしゃみ検知



・脳波センサー内蔵
・カメラで表情を検知

・集中をサポート
・リラックスサポート
　⇒ヒーリング音楽を流す。
・風邪の早期検知
　⇒せきくしゃみを監視



・居眠りを警告！！
・睡眠の質をアップ



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

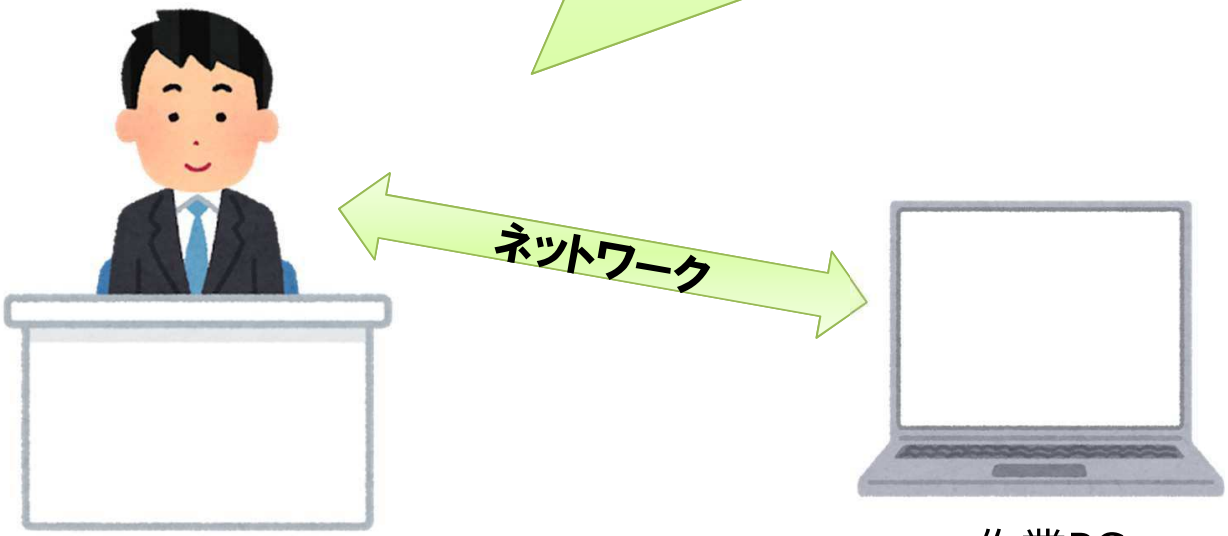
© Japan Embedded Systems Technology Association

エモーションセンサフュージョンシステム利用例

■　スマイルチェッカー

脳波とカメラで状態を検知
・表情検知（笑顔指数）
・ストレス指数（リラックス度）
・集中度

自身の気が付きにくい状態を
「ふわっと」通知。
・笑顔が少ない警告：● に光る。
・良く集中している：● に光る。
・イライラ警告：● に光る。



© Japan Embedded Systems Technology Association

ET2018デモ（like it checker）

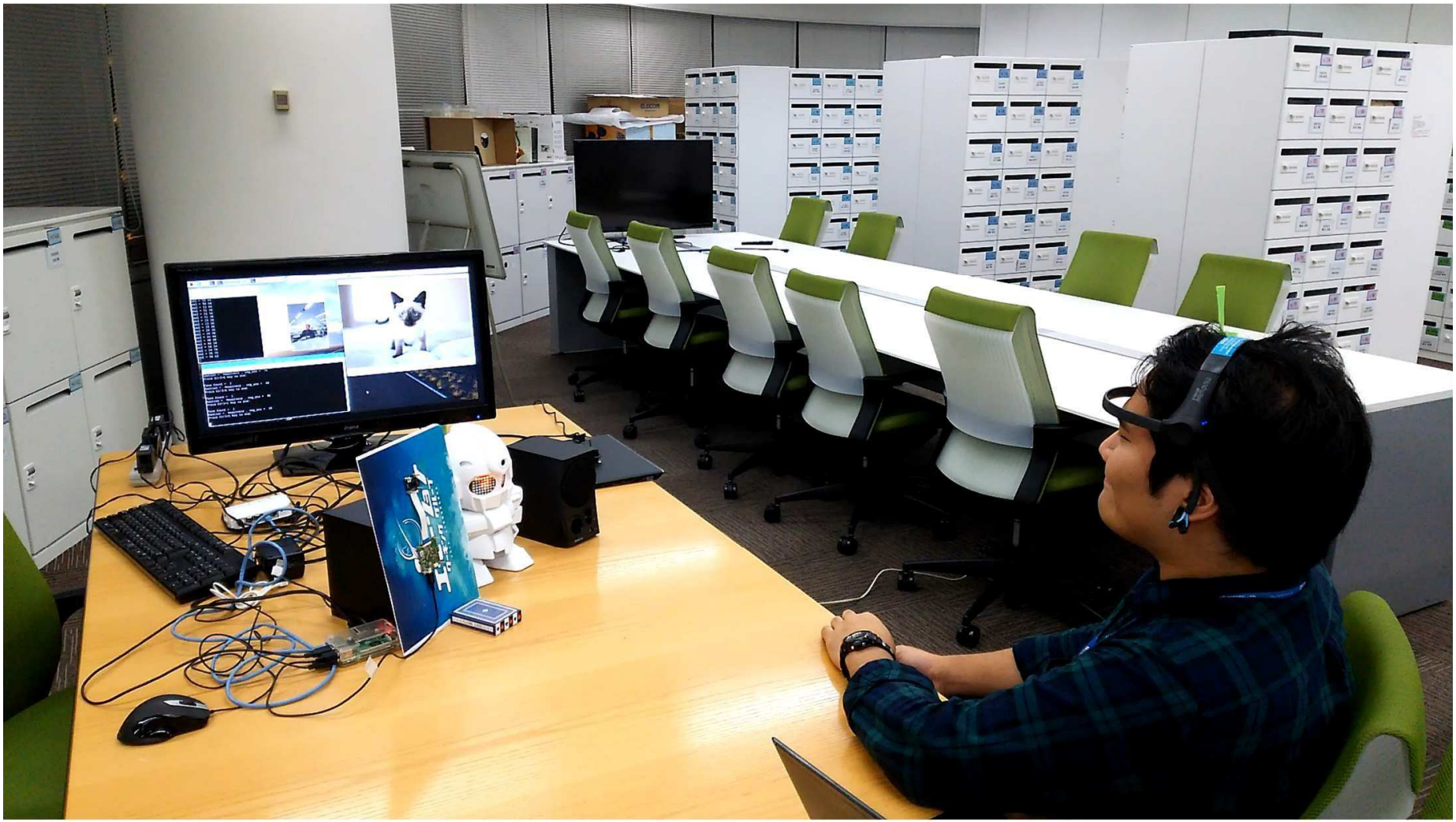
- 脳波と表情から、被験者がその画像が好きかどうかをチェックする
- 脳波は、リラックス度、表情はネガポジ値を利用し、判断
- 閾値の変更は、プログラムを変更せず、利用可能
- 判断結果は、RAPIROが音声と動作で通知



一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

エモーションセンサフュージョンシステムプロトタイプ構成



© Japan Embedded Systems Technology Association 2017

今後の活動について

© Japan Embedded Systems Technology Association 2017

2019年度スマートライフWG活動方針

1. スマートライフ分野における、QoL向上、課題解決のためのソリューションを組込みの視点で検討。

スマート睡眠をターゲット！

2. スマートライフに利用できるセンサとデータの研究

特にエモーションキャッチセンサ！！

3. 検討したソリューションについて、プロトタイプを作成し、サービスの有用性を検証する。

Built To Thinkで、よりよいソリューションへ！
イマジネーションを発展させる！！！

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2017

スマート睡眠評価システム



東大駒場リサーチキャンパス公開2019
COMMA ハウス
デモ展示予定
5月31日（金）、6月1日（土）
http://www.komaba-oh.jp/

© Japan Embedded Systems Technology Association

スマートライフWGからのメッセージ

JASA IoT技術高度化委員会
スマートライフワーキンググループでは、
一緒に活動を行うメンバーを募集しております。

お申し込みはJASA事務局まで。

© Japan Embedded Systems Technology Association 2017

【スマートライフにおけるエモーションデータの活用検討】

2019/05/20 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA）が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

© Japan Embedded Systems Technology Association

IoTの電力を担う
エネルギーハーベスティングの
可能性と課題

2019年5月20日

IoT技術高度化委員会

エネルギーハーベスティングWG

ユークエスト株式会社　冨岡　理

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

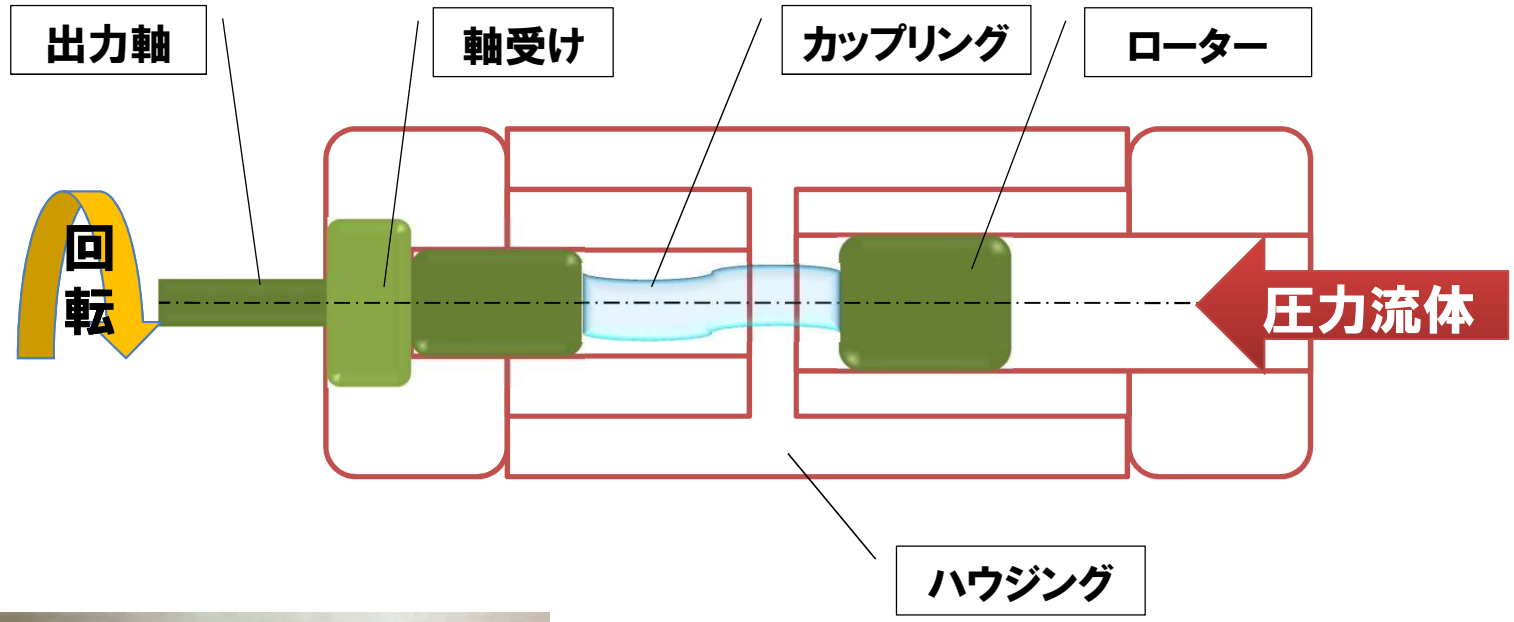
アジェンダ

- ゲストのお話を聞きました
- ETデモ（と出展社たち）
- エネハベ成功の条件
- WGのありかた

ゲストのお話

- 18年8月：中野製作所（水流発電）
- 18年9月：金沢大学（磁歪発電）
- 19年2月：東洋エレクトロニクス（振動発電）
- 19年4月：DSPC（エネハベ導入支援ツール）

中野製作所　水流発電





反発する磁石を近づけて流体で圧力をかけると磁石が回転する
原理が不明のため、最適なハウジングやクリアランスの設計が手探り（試行錯誤）

http://j-net21.smrj.go.jp/develop/energy/company/2013080801.html

東洋エレクトロニクス　振動発電

振動発電ビーコン

人の歩く振動で発電
無線通信モジュールを内蔵して電波を発信
日常生活防水対応



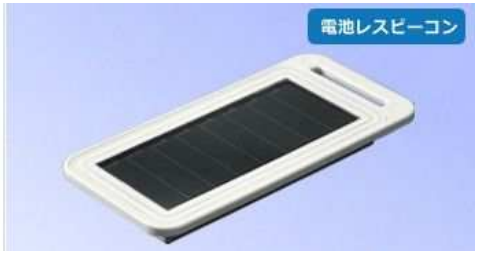


・人の歩行する振動で発電
　⇒約 5Hz の低い周波数に対応
・振動発電による電池レス
　⇒電池交換不要
　　メンテナンスフリー
　　発信間隔の考慮が不要
・人の動きを検出
　⇒行動の監視や分析に活用可能



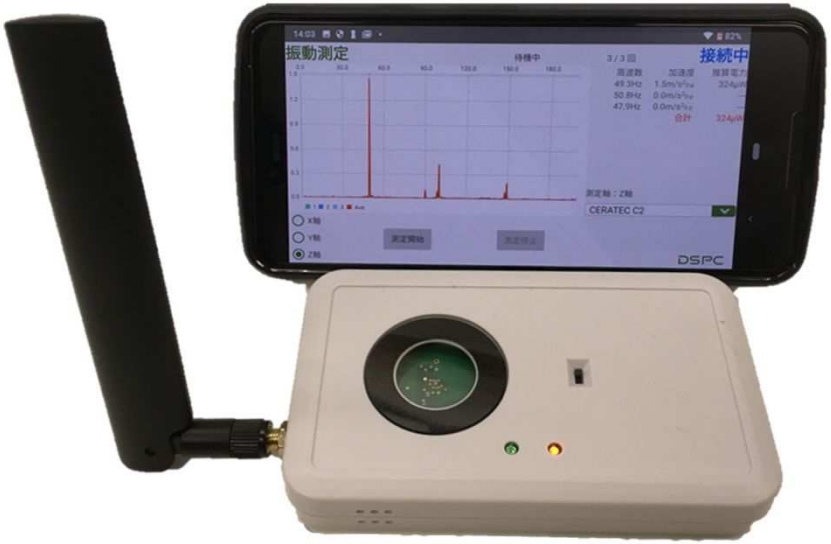
ソーラービーコン PB10-B

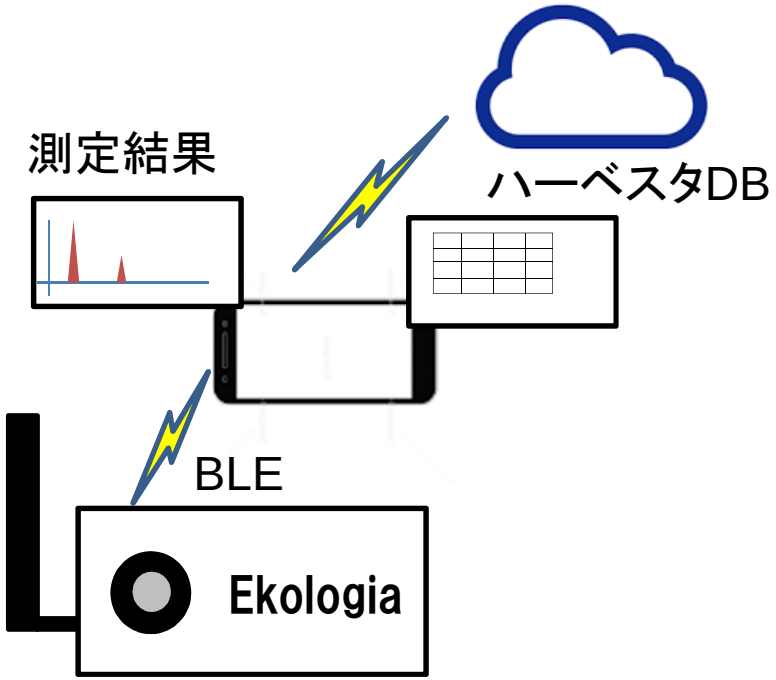
屋内灯で数秒毎に電波を発信
電池交換不要のメンテナンスフリー
iBeaconデータフォーマット対応
固定ビーコンにも動的ビーコンにも



一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

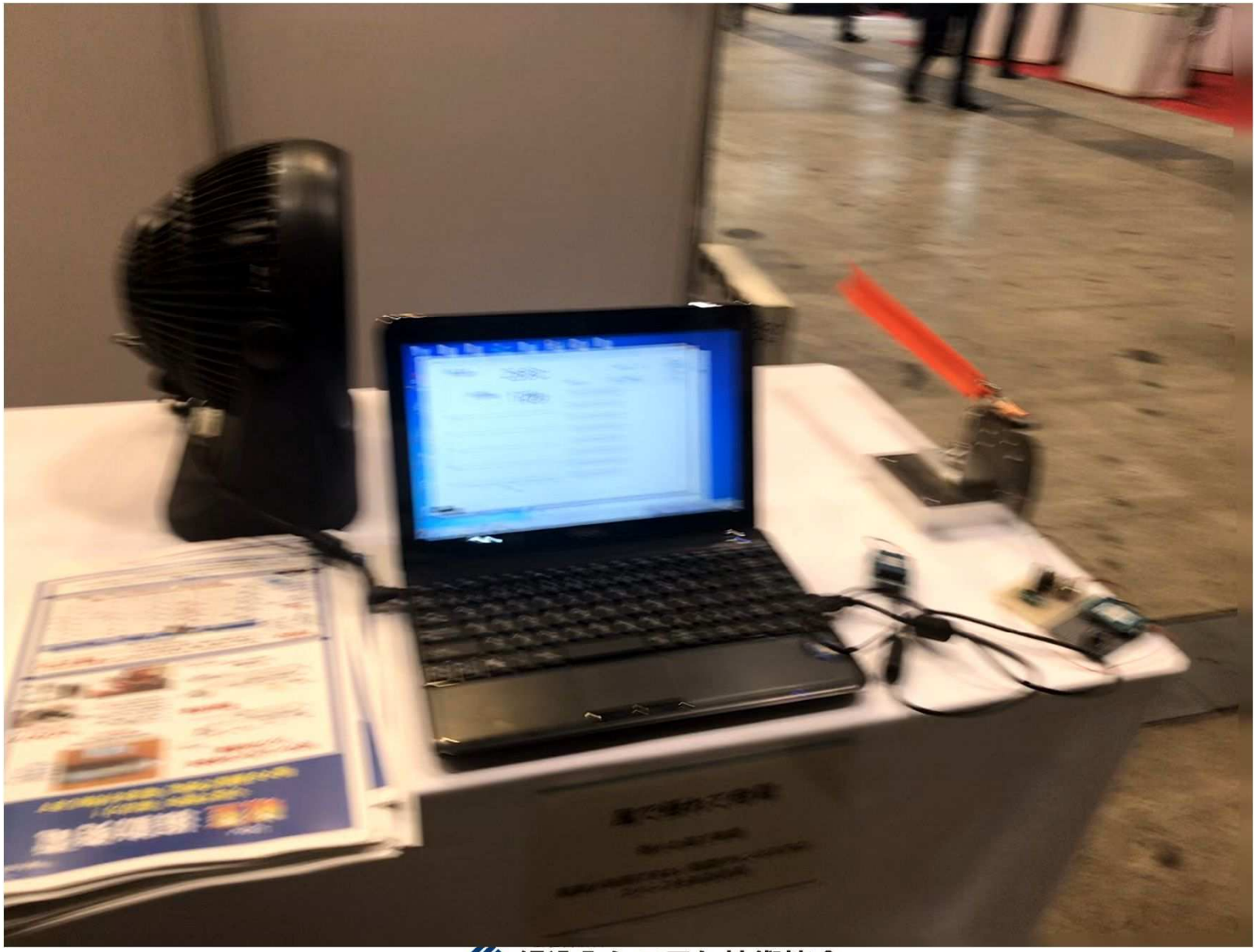
DSPC　エネハベ導入支援ツール（Ekologia　エコロジア）



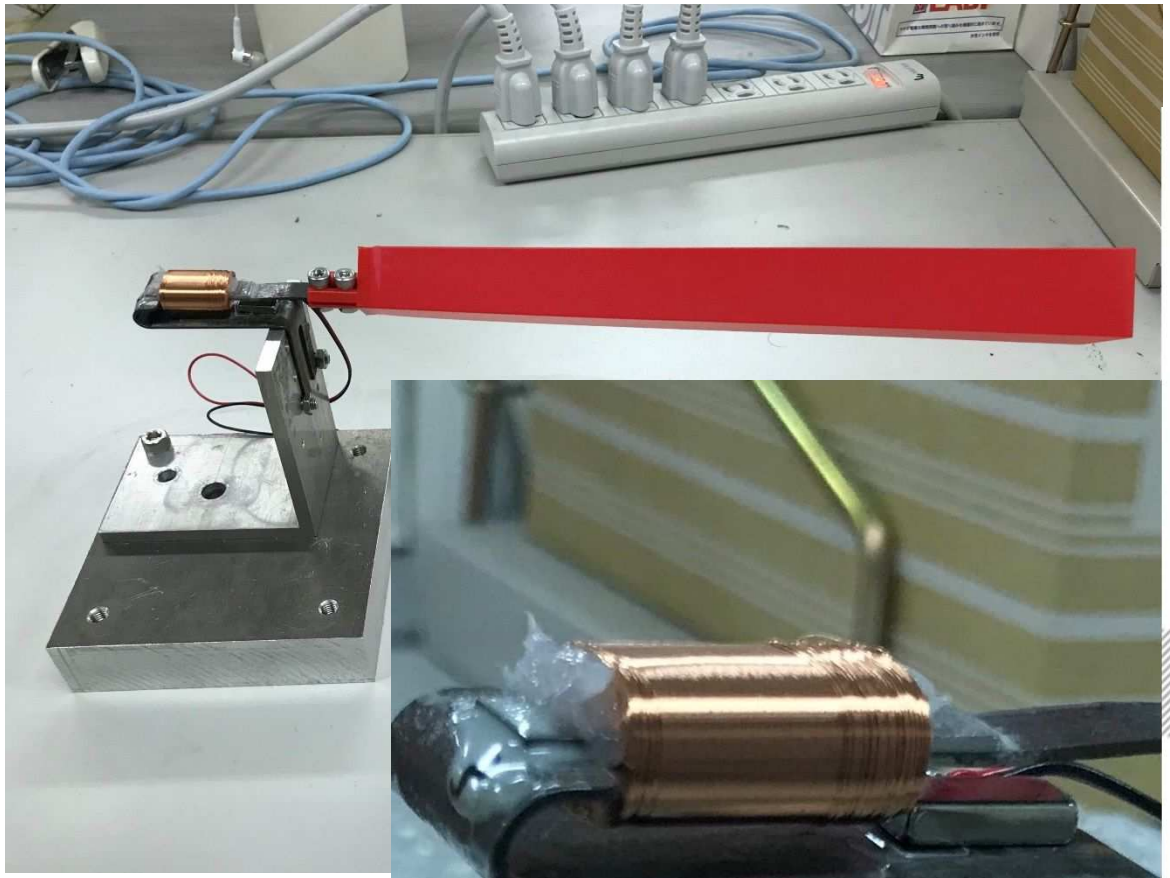


光・振動・温度差・電磁波の4つのエネルギー源を計測
各社ハーベスタ実力値のデータベースと照らし合わせ、最適なハーベスタをお勧め

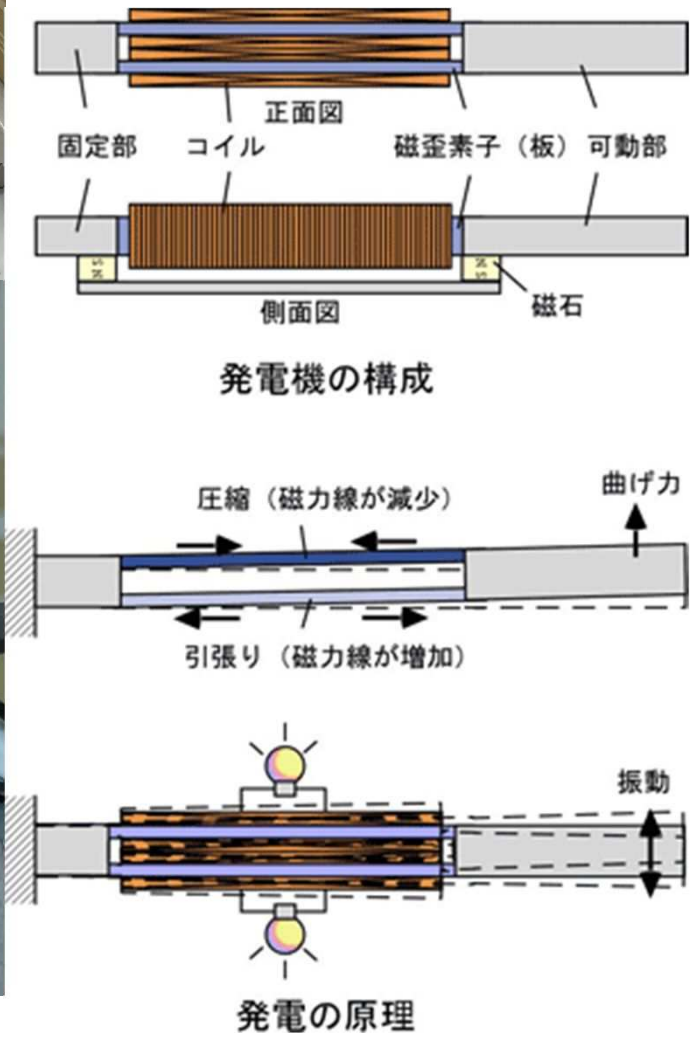
ETデモ（金沢大学　磁歪発電）



ETデモ（金沢大学　磁歪発電）



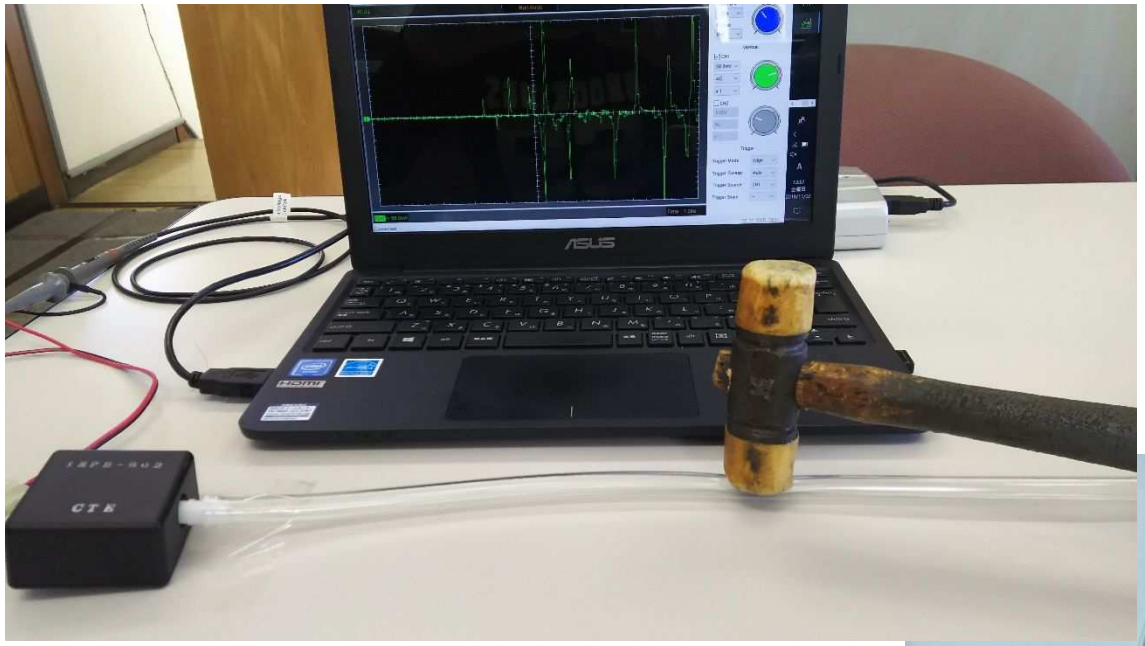


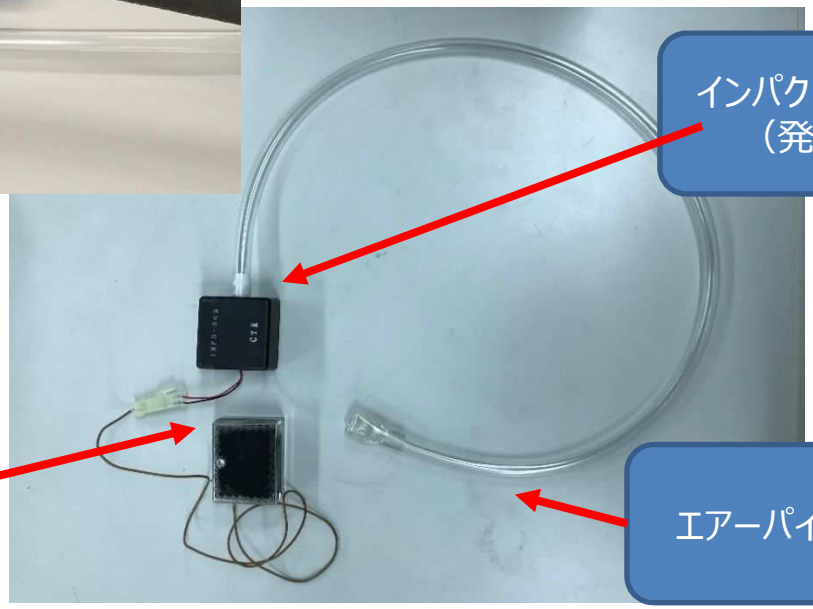


http://vibpower.w3.kanazawa-u.ac.jp/technical_information.html

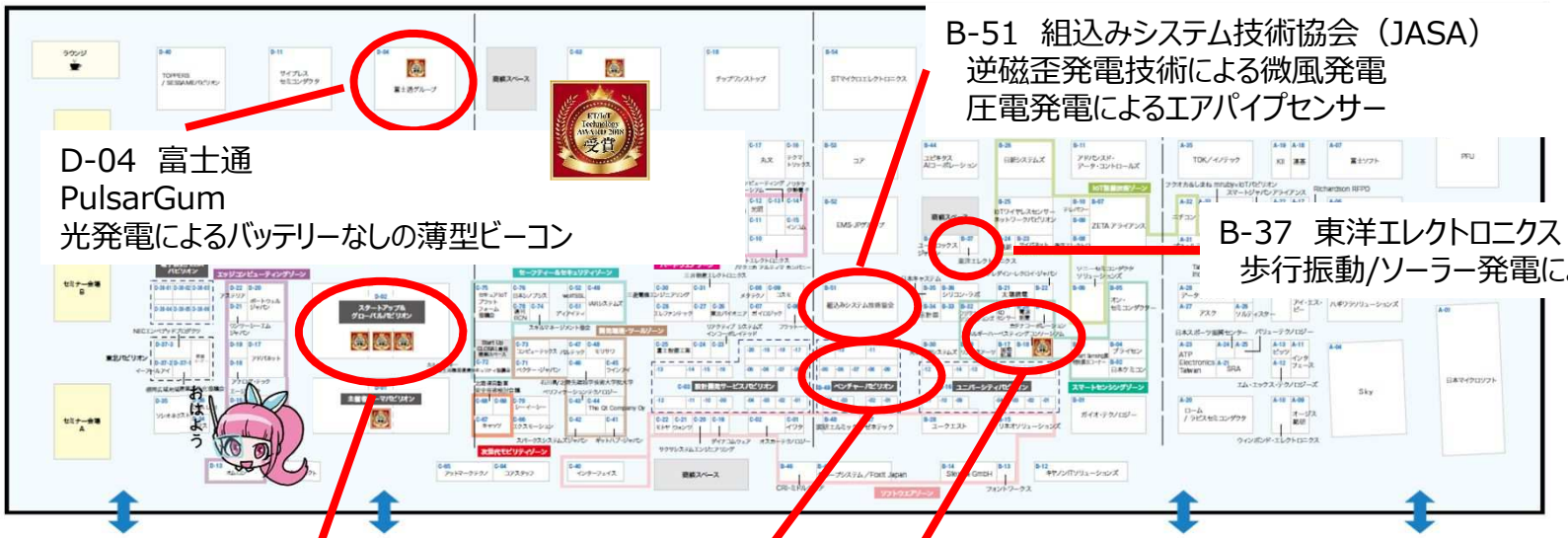
一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

ETデモ（セラテックエンジニアリング　圧電発電）





ET2018での展示（見落としあるかも）



D-04　富士通
PulsarGum
光発電によるバッテリーなしの薄型ビーコン

B-51　組込みシステム技術協会（JASA）
逆磁歪発電技術による微風発電
圧電発電によるエアパイプセンサー

B-37　東洋エレクトロニクス
歩行振動/ソーラー発電によるビーコン

D-02（グローバルパビリオン）Enerbee社（仏）
磁歪＋圧電による発電技術
Smart Vent

B-49⑥（ベンチャーパビリオン）モッタイナイ・エナジー
産総研発の熱電発電技術
鉄道模型を動かしている

B-18　エネルギーハーベスティングコンソーシアム
・アイシン高丘/アトムシステム
熱電発電モジュール

・アルプス電気
プッシュ/リリースの動作で発電する電磁誘導方式の小型発電器

・アルプス電気/NTTデータ経営研究所
通常無線の1/1000電力で高速セキュア通信

・宇宙航空研究開発機構（JAXA）
待機電力ゼロのワイヤレスセンサ（電波のエネルギーハーベスティング）

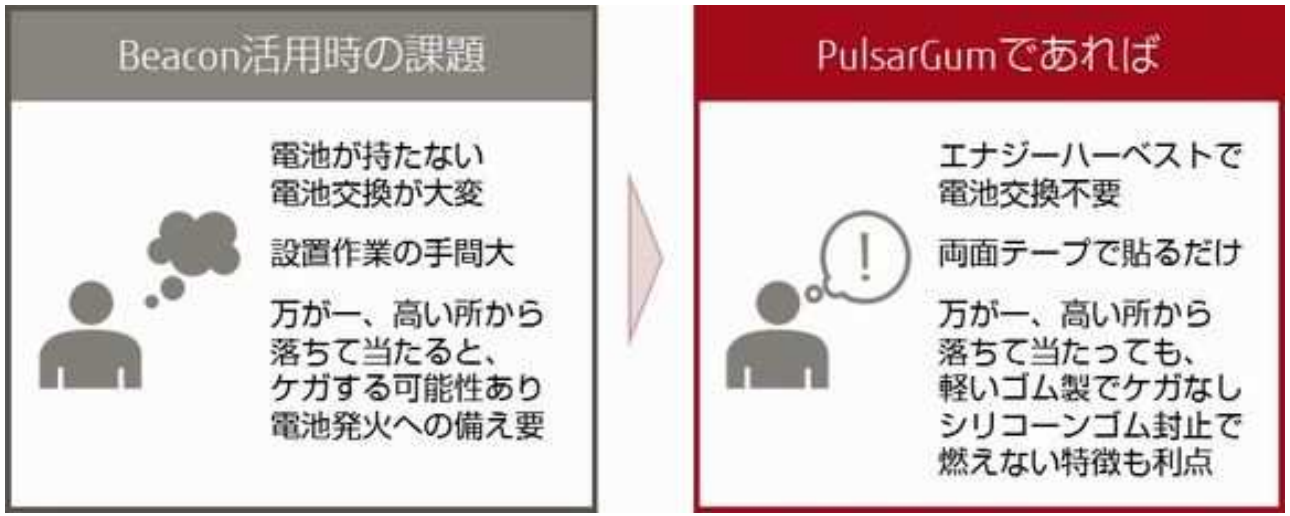
一般社団法人 組込みシステム技術協会
Japan Embedded Systems Technology Association

富士通　PulsarGum

バッテリーフリービーコン
太陽電池内蔵
300ルクスの照度で発電
シリコンゴムでコーティング







http://www.fujitsu.com/jp/solutions/industry/manufacturing/monozukuri-total-support/products/list-view/pulsargum/

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

モッタイナイエナジー





熱電変換ユニット

熱水と冷水を通すことで温度差発電

発電モジュールを直列接続

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

エネハベ成功の条件

- ハーベスタメーカーは短期間で消滅する例が多い
- WGの議論でもシーズから検討したりニーズから検討したりしても、ビジネスが成り立つかどうかは微妙なバランスの上に成り立っているように見える
- では、成功した人たちはなぜ成功したのだろう
  何に目を付けたのだろう

成功事例

■ Perpetuum(英)

https://perpetuum.com/



鉄道の台車に振動発電装置とセンサー、無線ユニットを取り付け、セルフパワーで状態を監視

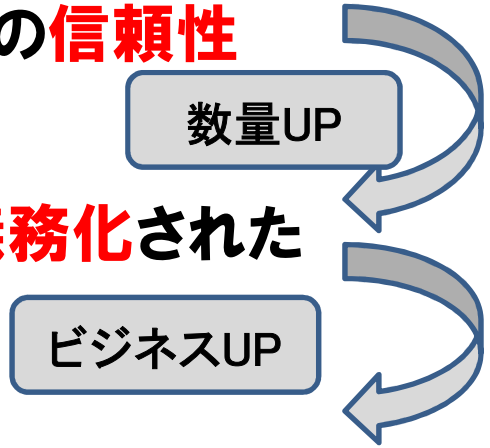


Saudi Railway Company

成功事例

■ Perpetuum(英)

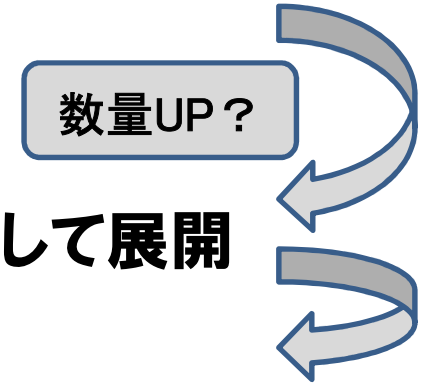
- デバイス開発
  - サウサンプトン大学でのMEMSでの発電デバイス研究がスタート
  - 顧客要求によって大型化/耐久性向上
    (当時小型化の要求はなかった)
  - GEやハネウェルなどにOEMするほどの**信頼性**
- ビジネス
  - 英国で列車の車軸モニタリングが**義務化**された
  - ベンチャー経験のある経営者
    デバイス売りから**サービス売り**へ



多くの場合

■ セラミック某社(米)

- デバイス
  - ― HEAD社のテニスラケットに採用
    「衝撃を吸収する」
  - ― スキーやスノボにもスタビライザーとして展開
  - ― ・・・・連絡が取れなくなった



スケールアップに失敗

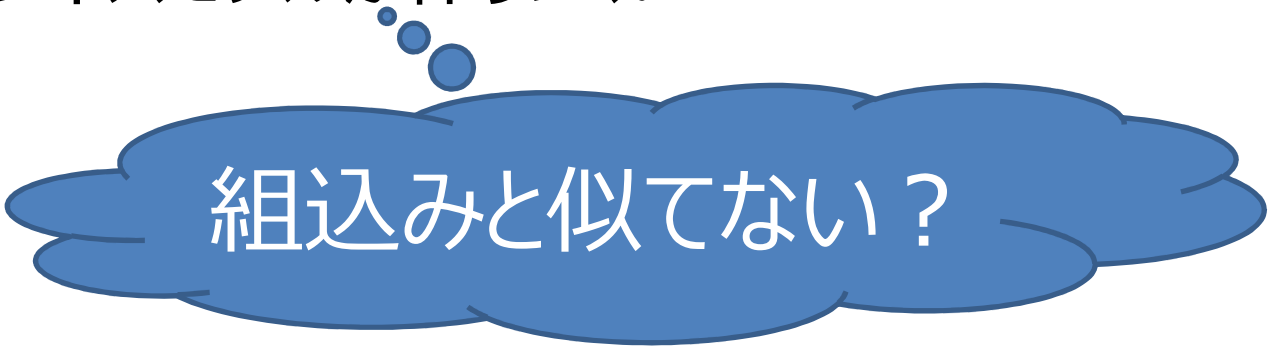
このような例は数多い。
大企業にとって十分なスケールアップができないという理由で断念するケースも同じ
「スケールアップ」のスケール感とは？
100億の売上が必要か1億の売上でもよいのか。

成功のポイント

- 義務化/法制化
  - 下水道の水位（日本）
  - シャッター
- 余技
  - ペルチェ素子メーカーの熱電発電事業
  - ステッピングモーターからの応用
- 電池交換が大変
  - 数年～数十年に一度しか発報しないだろう警報装置
  - 完全密閉（防水防塵筐体）
  - 遠い/不便
- 電源持たないモノのモノ探し
  - トレーラー台車、パレット、使用後の酸素ボンベ、高機能フィルムの使用後の芯（特許の塊）、建機（高所作業車、アタッチメント）、レンタサイクル
- そのほか、ハーベスタメーカーが見落としている市場があるのではないか
  - 数が出ない
  - ニーズ側が気付いていない
  - 対環境性（電池の廃棄問題）

JASAとエネルギーハーベスティング

- 発電量・発電タイミング・蓄電量・消費電力のバランスが難しい
- 数多くのニッチなユースケース
  - 用途やアプリケーションが多様なため大量生産でビジネスモデルが作りにくい



- ニーズを作り出す難しさ

WGのありかた

■ 協業領域と競争領域

- 協業領域をWG内で実施
  - 調査研究
  - 汎用的な実験
  - モノの購入/委託作業をWG費用、WGメンバーの人件費は手弁当
- 競争領域は自社に持ち帰り

WGメンバーの目的

- 自社の製品やソリューションをエネハベで付加価値つけたい/課題解決したい（①自社付加価値）
  - 自社のIoTスターターキット（工場向け）の電池レス化をする技術がないか。
  - インドで考えている物流トラッキングの仕事に活用できる技術を探す
  - 自社の製品や技術に活用できないか
- エネハベ技術でお客さんを捕まえたい（②EHビジネス）
  - IoT系の既存顧客に電源レスを提案
  - エネハベを使って〇〇をしたいという人に対してSIをする
- エネハベシステムを実現するために自社の技術やプラットフォームを使えないか（③ツールビジネス）

WGメンバーの目的

① 自社付加価値（自社の製品やソリューションをエネハベで付加価値つけたい/課題解決したい）
- 一覧表を作りたい
  - 入手性、価格、性能
- 実験をしたい

② EHビジネス（エネハベ技術でお客さんを捕まえたい）
- 一覧表を作りたい
- 客がイメージを広げられるデモを作る
- エネハベ相談所として機能する
  - 相談所があるよアピールをする

③ ツールビジネス（エネハベシステムを実現するためのツール）
- 測定する対象や方法を標準化する

やるべきこと

■ 一覧表作成
- 入手できるハーベスター一覧
- 実発電能力（実験した結果）
- 使う側の電力測定

■ 相談所の開設
- ET、ETWest、ET名古屋
- 九州ものづくりフェア
- EETimesコラム
- EHCが出展するところについでに出してもらう
- デモの作成

WGとしての成果物と企業での活用

■ 成果物

- 一覧表
- 相談所に集まったニーズの共有
- 測定仕様

■ 企業での活用

- 一覧表を用いて自社製品にEHを利用する
  （①自社付加価値）
- 相談が来た企業との個別商談
  （②EHビジネス）
- 測定ツールの開発
  （③ツールビジネス）

最後に（成果発表なのに・・・）

- 一覧表作成のための測定作業がなかなか進みません
  - 月に一度の活動では限界がある
  - どこまで本格的に測定するか？
  - 使う側にとって役に立つ一覧表はどんなものか？
- エネハベ難しい
  - あちらを立てればこちらが立たず
- 一緒になって悩んでくれる人募集

【IoTの電力を担うエネルギーハーベスティングの可能性と課題】

2019/5/20 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

ロボット、機械学習AI OSSの紹介と OSSの品質についてのアプローチ

2019年5月20日
OSS活用WG/技術本部本部長

竹岡尚三 (株)アックス

© Japan Embedded Systems Technology Association 2017

今、OSS
(オープンソース・ソフトウェア)
なのか?

© Japan Embedded Systems Technology Association 2017

自動運転もOSS!「Autoware」

·東京大学/名古屋大学 加藤真平先生 日本で最も進んだ自動運転 研究
·名古屋大学 開発 自動運転ソフトウェア「Autoware」サポート
·名古屋大学ら、開発済み自動運転システム一式をオープンソース化…加藤准教授「時間をジャンプ」
　　·Autowareは、オープンソース・ソフトウェアとして無償配布されている
http://response.jp/article/2015/08/26/258648.html





Autoware応用



ティアフォー「Milee（マイリー）」
https://clicccar.com/2017/12/11/538774/ より引用



ティアフォー無人 輸送車「Postee」
https://response.jp/article/2019/01/17/318151.html より引用



日本郵便／実証実験、自動運転でゆうパック

2018/3月

http://logistics.jp/media/2018/03/12/2361 より引用





(株)ティアフォー、ヤマハモーターパワープロダクツ（YMPC）、(株)マクニカが提供する「アカデミックパックPRO」
自動運転開発用のプラットフォーム
https://jidounten-lab.com/w_academic-pro-autonomous-car より引用

Autoware応用「ただの畳」

- 「ただの畳」
  - 悟空のきもち(ヘッドスパ)、関西電力、損保ジャパン日本興亜　3社共同プロジェクト
  - 時速1km/h でゆっくり走る
  - リゾート地で



https://www.asahi.com/articles/ASM1H46XHM1HPLFA001.html より

ロボット用ミドルウェア

RTミドルウェア



■産総研などが開発しているロボット用ミドルウェア
　・RTコンポーネントは、OMGにて、国際標準化
■「OpenRTM-aist」は、RTミドルウェアの産総研によ
　・ライセンスは、LGPLおよび産総研と個別に契約す
　　アルライセンス方式

画像引用元,http://www.openrtm.org/openrtm/ja/node/161#toc5
参考
http://www.openrtm.org/openrtm/ja/content/openrtm-aist-official-website

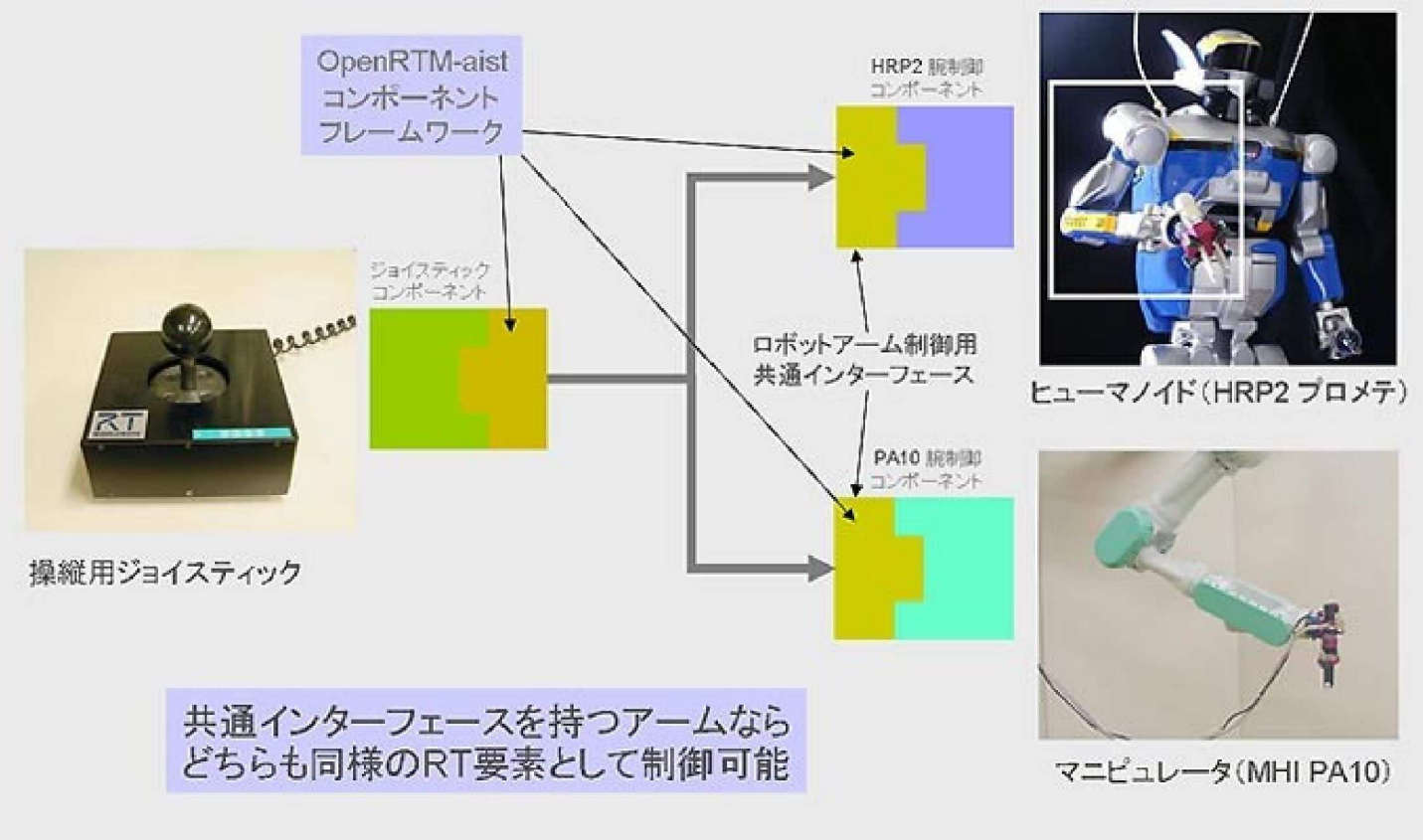
ROS

■データの流れに応じて、コンポーネントをつなぐ
■自動車の自動運転でも採用
■OpenCVも含まれている
■雑に言ってしまえば…
　• ロボットを作るためのソフトウェア部品の多くが
　　含まれている



画像の引用元
http://www.ros.org/news/resources/2010/poster2color_revis.jpg

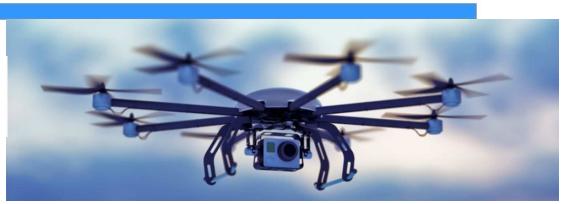
RTミドルウェア(ロボット用ミドルウェア)



引用元
http://www.aist.go.jp/aist_j/press_release/pr2005/pr20050224/pr20050224.html

Dronecode

https://www.dronecode.org/



■Linux Foundationの取りまとめで

■無人飛行体 Unmanned Aerial Vehicles (UAVs)を開発
オープンソース・ソフトウェア + オープンソース・ハードウェア

■オン・ボード(機体上)と、オフ・ボードのソフトウェアがある

- オンボード: 機体の制御、センサー制御
- オフボード: リモコン、データ収集/記録

■Dronecode は、いくつかのプロジェクトとリポジトリでできている。

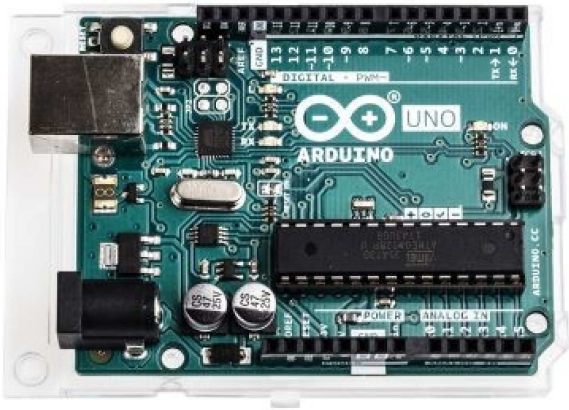
• ソースコードは、下記から

Ardupilot https://github.com/diydrones/ardupilot
PX4 https://github.com/PX4
Pixhawk https://github.com/Pixhawk
MAVLink https://github.com/mavlink
UAVCAN https://github.com/uavcan
ROS https://github.com/ros
Other repositories https://github.com/Dronecode

画像引用元
https://www.dronecode.org/sites/dronecode/files/styles/dronecode_header/public/front_page_slides/images/drone_video_slide.png?itok=7C7IFYIP

IoT,M2Mもオープンソース技術

- Arduino
  - オープンソース・ソフトウェア (OSS)
    ＋
  - オープンソース・ハードウェア (OSHW)
  - OSS
    - 開発環境 IDE
    - ライブラリ: 膨大な量のライブラリ
    - OSSで、世界中のみんなが開発
  - OSHW
    - 回路図
    - 基板レイアウト
  - お手軽試作にバッチリ
    - ハードウェアは、アマチュア品質だが…
- Raspberry Pi
  - Linux搭載、ハードウェア情報公開





OSS使用の要請



- サービス主体
  - 自動車でさえ、サービス主体になってきた
  - ソフトウェアがプラットフォームとして重要
  - 全体が1モジュール
    - Android,Linuxなど

- 原理試作,早い試作は、OSS使用
  - 動作が確認できたら、プロプライエタリソフトウェアに置き換えていく
  - 一般的になりつつある(か?)

- OSSは、どのあたりに使えるか?
  どの程度の品質ならば、どのあたりに使えるか?

© Japan Embedded Systems Technology Association 2019

OSSの本当の利点

■「無料だから嬉しい」とか言ってると、負ける

■特定企業のOSとは違い、ソースがあるので、
理解し、独自の改良が可能
■デファクト・スタンダードがOSSなら、
特定ベンダに囲い込まれない
■自分の都合でシステムをリリースできる
- 特定のソフトウェアのリリース(バージョンアップ)に引っ張られない
  - 独自に品質を上げたソフトウェアを、自由に維持できる

OSSに乗るしかない!

- 顧客が、OSS前提で仕様を決めている
- サービス主導の時代→OSSで素早くシステム構築
  - Windows+ブラウザで、できていることは、できて欲しい
- 高度なロボットを、すぐに作りたい
- 人工知能(AI)が入った機器をすぐに作りたい

- OSSを活用する世界をみんなで考えよう

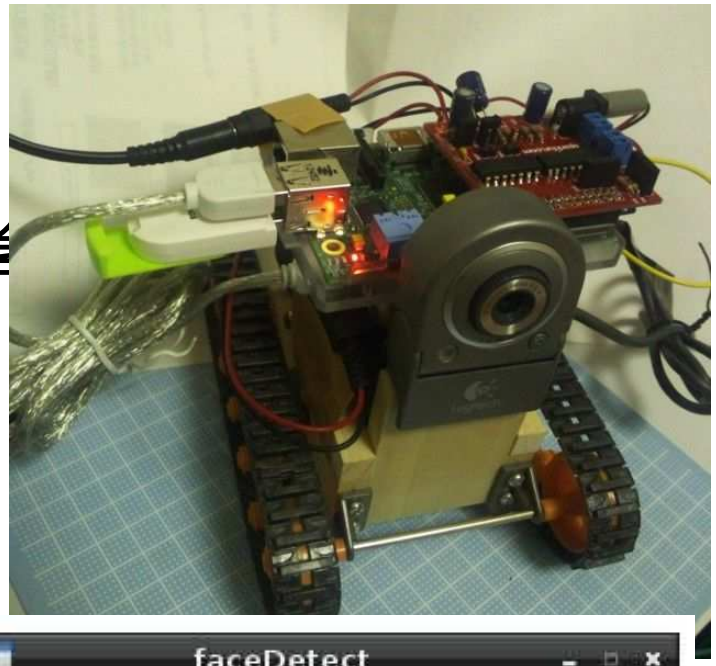
→ JASA　OSS活用WG

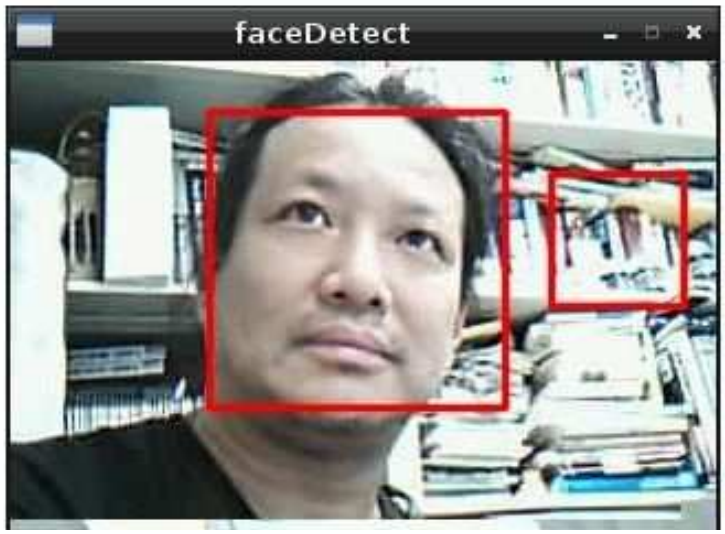
© Japan Embedded Systems Technology Association 2019

機械学習(AI)
OSS

© Japan Embedded Systems Technology Association 2016

OSS機械学習ロボット





■人間の顔を見つけたら、追尾する
- 顔の位置を判断して、右へ行くか、左へ行くかを決める
- 阪急電車も判別可能(機械学習によって)

■Linux+OpenCV+OpenEL

■OpenCVは
- 機械学習(SVMなど)による顔認識などを含

■総合的な画像処理ライブラリ

■DCモータの制御は**OpenEL**
- OpenELはJASAなどが推進している
  ロボット用の下位ハードウェアの
  抽象化層の規格

■ハードウェア
- RaspberryPi
  —ARM11@700MHz
  —512MBytesRAM
- USBカメラ
- DCモータ

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

組み込みで使える Deep Learning(深層学習) OSS



- 組み込みで使えるものを紹介
  - 独立して動作
    - クラウドとかサーバは不要
  - 割と新しいもの
  - でも、学習時には、GPUが必要かも…

- TensorFlow (Deep Learning OSS)
  - Googleが開発
  - Apache 2.0ライセンス
  - AndroidやiOSでも動作
  - Pythonから使いやすい
  - https://www.tensorflow.org/

•Chainer (Deep Learning OSS)
  - 日本の Preferred Infrastructure 社が開発
  - ライセンス
    - 基本的に、改変、再配布など可能。詳細は下記
    - https://github.com/pfnet/chainer/blob/master/LICENSE
  - Pythonから使いやすい
  - X86 CPU以外で動作している情報が無い…
  - http://chainer.org/

•Caffe (Deep Learning OSS)
  - UC BerkeleyのBerkeley Vision and Learning Center (BVLC)が開発
  - 速い
  - Pythonから使える
  - http://caffe.berkeleyvision.org/

Deep Learning の推論には、たいしてCPUパワーは要らない



■半導体 会社が提供しているDeep Learning

■STM32Cube.AI (X-Cube-AI)
- ST32で実行(推論)できる
- クラウドで AI の学習を行う
- 学習済みモデル,プログラムをダウンロード&ST32で実行

https://www.st.com/en/embedded-software/x-cube-ai.html?icmp=tt9145_gl_pron_dec2018

■推論のみの車載機にGPUは不要だろう

機械学習はDeep Learning だけじゃない!



■深層学習 (Deep Learning)
- 多層パーセプトロン(Perceptron)の強化形
  - パーセプトロンは、1950年代末期からある
  - バック・プロパゲーションを色んな層に
- 良い学習のためには、演算が多い
- 学習結果、判定結果は解析不能
  - 本質的に。

■サポート・ベクタ・マシン (Support Vector Machine)
- 試料をベクトル化して、比較
  - ベクトルの要素は、機械が決定する。が、人間がベクトルを見て、解析することは可能
- Deep Learningと比べて演算が少ない

■Google自動運転でDeep Learningが有名になるまでの10年間ほどは、SVMがとても多く使われていた

Support Vector Machine(SVM) OSS



- SVMは、計算が少ない
- ARMなどの組み込みCPUで十分に動作可能
- SVM-Light
  - Thorsten Joachims @Cornell Universityが開発
  - Linear kernelの学習時間は短い
  - 精度はいい
  - http://www.cs.cornell.edu/People/tj/svm_light/index.html
- SVM-perf
  - Thorsten Joachims @Cornell Universityが開発
  - 学習時間は短い
  - 精度はいまいち
  - http://www.cs.cornell.edu/People/tj/svm_light/svm_perf.html
- LIBSVM
  - 国立台湾大学のChih-Chung Chang と Chih-Jen Linが開発
  - 学習時間は長い
  - Linear kernelの精度は SVM-Light並
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- LIBLINEAR
  - 国立台湾大学 Machine Learning Group が開発
  - 学習時間がかなり短い
  - 精度はいい
  - http://www.csie.ntu.edu.tw/~cjlin/liblinear/
- CvSVM (OpenCV)
  - OpenCVに入っている
  - よく使われている
  - http://opencv.jp/opencv-2.1/cpp/support_vector_machines.html

OSS品質評価
の
アプローチ

© Japan Embedded Systems Technology Association 2016

OSSって面倒くさい？



- 品質問題
  - 適当に作られた無料のソフトウェアの品質は?
  - どこの誰だか、知らない人が作ったものでしょ…
  - OSSの品質が、とても良い、ということは、しばしばある
- 品質の押さえ方は、大事
- **OSSの品質評価の問題点**
  - 開発手法で押さえられない
    - (導入したい)OSSは開発が終了しているから
  - コードレビューにそぐわない
  - 巨大OSSだから、使用したい
    - 小さいソフトウェアなら、新たに書き下ろせる
  - 巨大OSSのコードをすべて精査するのか???
    - 通常のサイトには、その能力がない

© Japan Embedded Systems Technology Association 2019

OSSの品質評価の解決方法



- テストする
- 他者の商品ソフトウェアを購入した場合
  - 製作者である他者が、品質を保証
    - テスト結果で、品質を示すことも多い
      - 必ずしもそうではないが…

- OSSもテストして、品質を確認

© Japan Embedded Systems Technology Association 2019

OSSのテスト方法の難点



- OSSもテストして、品質を確認したい…
- OSSのテストの難点
  - 仕様がいまひとつ明確でない
  - どこを重点的にテストすればいいか判らない
    - コードをレビューしていないから、
      難しそう&重要そうな部分が判らない
      ↓
  - 境界条件テストが難しい
  - 限界値テストは、可能だろう
    - 仕様が不明確な場合もあるが

© Japan Embedded Systems Technology Association 2019

OSSのひとつのテスト方法



- 境界条件テストが難しい
- 限界値テストは、可能だろう
  - 仕様が不明確な場合もあるが

    ↓
- いろんな引数を、対象OSSに食わせる
- 異常な振る舞いをしたら、危険
  - 危険な値が、入力されないように、外で処置

    or
  - OSSを修正

© Japan Embedded Systems Technology Association 2019

OSSのひとつのテスト方法:Fuzzテスト



- いろんな引数を、対象OSSに食わせる
- Fuzzテスト
  - ファズ（英：fuzz）（予測不可能な入力データ）を与えることで意図的に例外を発生させ、その例外の挙動を確認するという方法を用いる。ファズテストと呼ばれることもある。
  - https://ja.wikipedia.org/wiki/%E3%83%95%E3%82%A1%E3%82%B8%E3%83%B3%E3%82%B0 より引用

© Japan Embedded Systems Technology Association 2019

Fuzzテストを試行



- OSS活用WGで、fuzzテストを試行する
- 対象OSS
  - OpenCV :機械学習,画像処理
    - ニーズ高い
  - OpenRTM-aist (候補)
    - 規模が大きそうなので、やや躊躇している
  - OpenEL (候補)
    - JASAが中心となって策定中の規格
    - OSS版ソフトウェアを対象に行いたい

© Japan Embedded Systems Technology Association 2019

Fuzzテスト試行により



- 品質評価の基準を探る
  - Fuzzテストの結果がどの程度だと、安心だと言えるのか

    or
  - Fuzzテストの結果からは、安心だと言えないのか
- テスト・コストを探る
  - どれぐらいテストをすると
    - 問題が発現するのか
    - 十分にテストした、と言えるのか/言えないのか?

© Japan Embedded Systems Technology Association 2019

Fuzzテスト試行コストの測り方



- テスト・コストを探る
  - 「異常がすぐわかった」を 1として、
  - 他の異常が読み取れるまでのコストを、比で表す。

- ※テスト実施したものが優秀で、すべて「すぐできた」という報告になってしまった…

© Japan Embedded Systems Technology Association 2019

2016年度OpenCVにFuzzテスト試行した



- 機械学習（認識）に関する関数をテスト

- `void cvReleaseImage( IplImage** image )`
- `void cvReleaseHaarClassifierCascade( CvHaarClassifierCascade** cascade )`
- `int cvNamedWindow( const char* name, int flags=CV_WINDOW_AUTOSIZE )`
- `CvCapture* cvCreateFileCapture( const char* filename )`
- `CvCapture* cvCreateCameraCapture( int index )`
- `int cvSetCaptureProperty( CvCapture* capture, int property_id, double value )`
- `bool CascadeClassifier::load(const string& filename)`
- `CvMemStorage* cvCreateMemStorage( int block_size=0 )`
- `IplImage* cvQueryFrame( CvCapture* capture )`
- `void detectMultiScale( const Mat& image, vector<Rect>& objects, double scaleFactor=1.1, int minNeighbors=3, int flags=0, Size minSize=Size())`
- `void cvRectangle( CvArr* img, CvPoint pt1, CvPoint pt2, CvScalar color, int thickness=1, int line_type=8, int shift=0 )`
- `void cvShowImage( const char* name, const CvArr* image )`
- `void cvReleaseMemStorage( CvMemStorage** storage )`
- `void cvReleaseCapture( CvCapture** capture )`
- `void cvDestroyWindow( const char* name )`

© Japan Embedded Systems Technology Association 2019

OpenCVにFuzzテスト試行した結果(2016)



大きな問題がありそうなもののみを、表に掲げる

関数	引数	値	結果（大きな問題がありそうなもののみ抜粋)
CvCapture* cvCreateFileCapture(const char* filename)	filename	NULL	Segmentation fault
IplImage* cvQueryFrame(CvCapture* capture)	capture	NULL	静止画使用時は黒い表示、カメラ使用時はNULLを返す
void detectMultiScale(const Mat& image, vector<Rect>& objects, double scaleFactor=1.1, int minNeighbors=3, int flags=0, Size minSize=Size())	scaleFactor	inf	呼び出してから戻ってこない
void cvShowImage(const char* name, const CvArr* image)	name	存在しないウィンドウ名	(Hankyu+Face Detect:12853): GLib-GObject-CRITICAL **: g_object_unref: assertion 'G_IS_OBJECT (object)' failedを表示し、新たなウィンドウ生成
	image	NULL	真っ黒の表示。(Hankyu+Face Detect:16287): GLib-GObject-CRITICAL **: g_object_unref: assertion 'G_IS_OBJECT (object)' failed表示。

© Japan Embedded Systems Technology Association 2019

OpenCVにFuzzテスト試行した(2016)



- ファースト・インプレッション
  - OpenCVって、なかなか品質が高い（主観的感想）
  - でも、バグを見つけたぜ！
    - 異常値の対応不備。通常使用では、問題にならない

© Japan Embedded Systems Technology Association 2019

2017年度OpenRTM-aistにFuzzテスト試行した



- OpenRTM-aist を対象にFuzzテストを試行した
  - OpenRTM-aist :RTMの産総研の実装,OSS
    -
    - 関数/対象　引数/設定項目(コンフィギュレーション・パラメータ)
    -

bindParameter関数　　param_name

RTC::RingBufferのコンストラクタ　　length(long int,バッファ長)
RTC:Properties::load　入力ストリームへ与える文字列（キー:値）
RTC::Manager::createComponentcomp_args(const char *)
RTC::Manager::init　　argc, argv
RTC::RTObject_impl::setInstanceName　instance_name(const char *)

rtc.conf中のネームサーバへの登録名

※もともと、RTMは、通信の枠組みなので、設定可能なパラメータが少ない

© Japan Embedded Systems Technology Association 2019

OpenRTM-aistにFuzzテスト試行した結果(2017)



問題がありそうなもののみを、表に掲げる

関数/対象	引数/設定項目	値	得られた結果 (怪しいもののみ)	備考
コンフィギュレーション・パラメータ(coil::Propertiesのコンストラクタ、char* []引数でキー・値の並びを渡す)	キー名	キー名:nullptr	Segmentation Fault	
bindParameter関数	param_name	nullptr	0	第3引数で指定したデフォルト値(10)とならない。
RTC::RingBufferのコンストラクタ	length(long int,バッファ長)	-1	Component create failed. 中止 (コアダンプ)	仕様では、2以下を指定するとバッファ長2で初期化される、となっている。
		0	length()が0	仕様では、2以下を指定するとバッファ長2で初期化される、となっている。
		1	length()が1	仕様では、2以下を指定するとバッファ長2で初期化される、となっている。
RTC:Properties::load	入力ストリームへ与える文字列(キー:値)	キー・値共にaを3173回繰り返し	値だけが923文字に切り詰められる。また、例外送出されることがあった。(17回中2度発生した)。	terminate called after throwing an instance of 'CORBA::BAD_INV_ORDER'
RTC::RTObject_impl::setInstanceName	instance_name(const char *)	nullptr	Segmentation fault	

© Japan Embedded Systems Technology Association 2019

OpenRTM-aistにFuzzテスト試行した（2017）



- ファースト・インプレッション
  - 品質が悪くない（主観的感想）
    - 問題が出たのも、初期化で使用するところ
  - でも、バグを見つけたぜ！
    - 異常値の対応不備。通常使用では、問題にならない

© Japan Embedded Systems Technology Association 2019

2018年度OpenEL OSS版にFuzzテスト試行した

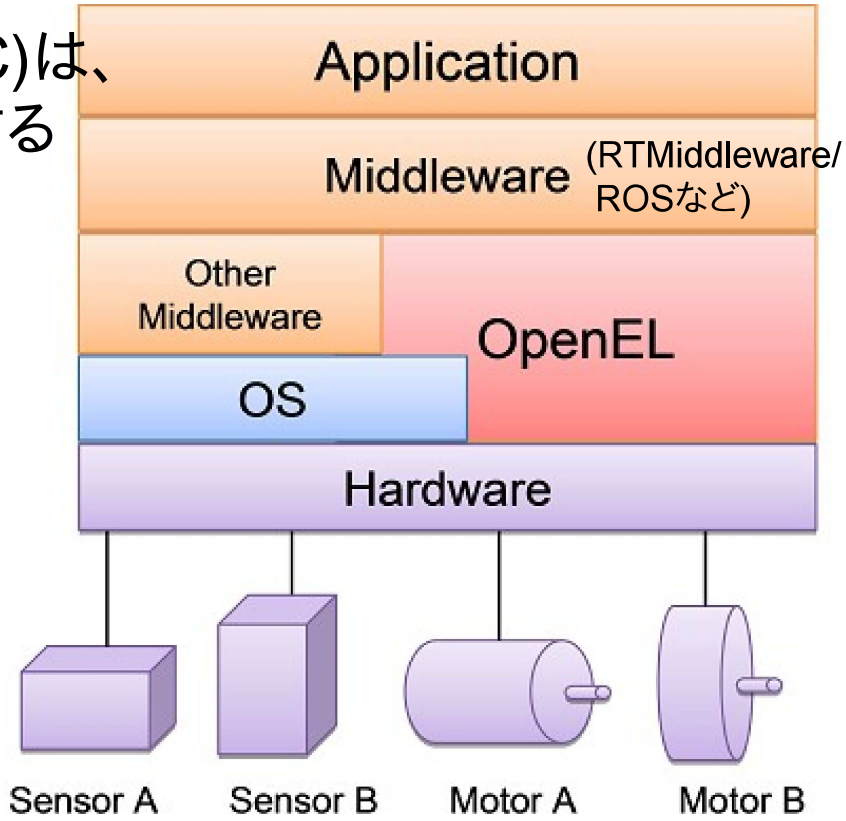


- OpenEL(OSS版)を対象にFuzzテストを試行した
  - OpenELのJASAによる仕様確認版
    - 関数/対象　引数/設定項目(コンフィギュレーション・パラメータ)
- **内部で、引数チェックを行わず、軽い**

© Japan Embedded Systems Technology Association 2017

OpenEL

■JASA,産総研などが推進しているフレームワーク
■国際規格にするべく、OMGに提案中
■RTミドルウェアなどのコンポーネントの可搬性を高める
■ハードウェアに近い層を、抽象化
■OpenEL準拠で書かれたソフトウェア(RTC)は、
■ハードウェア・ドライバに依存せずに動作する



参考
http://jasa.or.jp/openel/Main_Page/ja#OpenEL.E3.81.AE.E6.AD.B4.E5.8F.B2.E3.81.A8.E6.99.AE.E5.8F.8A.E3.83.BB.E5.95.93.E7.99.BA.E6.B4.BB.E5.8B.95

引用元 http://jasa.or.jp/openel/Main_Page/ja

OpenELにFuzzテスト試行した結果1/2(2018)



-

関数	引数	値	結果（怪しい結果は黄色で塗りつぶしてある）	備考
HALCOMPONENT_T* HalCreate(int32_t vendorID, int32_t productID, int32_t instanceID)	vendorID	0x00000001(未定義のベンダID)	NULLを返す	ソースコード内の関数仕様の返り値の記述に誤り有り
	productID	0x00000011(未定義のプロダクトID)	NULLを返す	
void HalDestroy(HALCOMPONENT_T* halComponent)	halComponent	NULL	Segmentation fault	
HALRETURNCODE_T HalInit(HALCOMPONENT_T *halComponent);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalReInit(HALCOMPONENT_T *halComponent);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalFinalize(HALCOMPONENT_T *halComponent);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalAddObserver(HALCOMPONENT_T *halComponent, HALOBSERVER_T *halObserver);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	halObserver	NULL	Segmentation fault	HAL_ERRORを返すべき？
	halObserver	イベント発生時のコールバック関数をNULLにする	Segmentation fault	何もしない方が良い？（仕様が無いためなんとも言えない）
HALRETURNCODE_T HalRemoveObserver(HALCOMPONENT_T *halComponent, HALOBSERVER_T *halObserver);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	halObserver	NULL	HAL_OKを返す	HAL_ERRORを返すべき？
HALRETURNCODE_T HalGetProperty(HALCOMPONENT_T *halComponent, HALPROPERTY_T *property);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	property	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalGetTime(HALCOMPONENT_T *halComponent, int32_t *timeValue);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	timeValue	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalEventTimerStartTimer(HALEVENTTIMER_T *eventTimer);	eventTimer	NULL	Segmentation fault	HAL_ERRORを返すべき？

© Japan Embedded Systems Technology Association 2017

OpenELにFuzzテスト試行した結果2/2(2018)



HALRETURNCODE_T HalEventTimerStartTimer(HALEVENTTIMER_T *eventTimer);	eventTimer	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalEventTimerStopTimer(HALEVENTTIMER_T *eventTimer);	eventTimer	NULL	関数呼出し後，帰ってこない	HAL_ERRORを返すべき？/プラットフォーム依存関数に飛ぶため，この挙動もプラットフォームに依存する
HALRETURNCODE_T HalEventTimerSetEventPeriod(HALEVENTTIMER_T *eventTimer, int32_t eventPeriod);	eventTimer	NULL	Segmentation fault	HAL_ERRORを返すべき？
	eventPeriod	-100	タイマイベントが発生しない	常にHAL_OKを返すような実装/実際の挙動はプラットフォーム依存/仕様がないためなんとも言えない
HALRETURNCODE_T HalEventTimerAddObserver(HALEVENTTIMER_T *eventTimer, HALTIMEROBSERVER_T *timerObserver);	eventTimer	NULL	Segmentation fault	HAL_ERRORを返すべき？
	timerObserver	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalEventTimerRemoveObserver(HALEVENTTIMER_T *eventTimer, HALTIMEROBSERVER_T *timerObserver);	eventTimer	NULL	Segmentation fault	HAL_ERRORを返すべき？
	timerObserver	NULL	HAL_OKを返す	HAL_ERRORを返すべき？
HALRETURNCODE_T HalActuatorSetValue(HALCOMPONENT_T *halComponent, int32_t request, HALFLOAT_T value);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	request/valueに関しては，挙動がドライバの実装に依存するので未実施			
HALRETURNCODE_T HalActuatorGetValue(HALCOMPONENT_T *halComponent, int32_t request, HALFLOAT_T *value);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	value	NULL	Segmentation fault	HAL_ERRORを返すべき？
HALRETURNCODE_T HalSensorGetValueList(HALCOMPONENT_T *halComponent,int32_t *size, HALFLOAT_T *valueList);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	size	NULL	Segmentation fault	HAL_ERRORを返すべき？
	valueList	NULL	Segmentation fault	HAL_ERRORを返すべき？
（※この関数の仕様について：呼び出し後にsizeに値の個数が，valueListに複数の値が代入されるが，呼び出し前は値の個数が不明なので配列valueListをどの程度の長さ確保すれば十分かわからない；ドライバの仕様に含めておくのかもしれない）				
HALRETURNCODE_T HalSensorGetTimedValueList(HALCOMPONENT_T *halComponent,int32_t *size, HALFLOAT_T *valueList, int32_t *timeValue);	halComponent	NULL	Segmentation fault	HAL_ERRORを返すべき？
	size	NULL	Segmentation fault	HAL_ERRORを返すべき？
	valueList	NULL	Segmentation fault	HAL_ERRORを返すべき？
	timeValue	NULL	Segmentation fault	HAL_ERRORを返すべき？

（※HalActuator*/HalSensor* 関数は与えられたコンポーネントがアクチュエータ/センサであるかのチェックは行っていない．）

© Japan Embedded Systems Technology Association 2017

OpenELにFuzzテスト試行した（2018）



- ファースト・インプレッション
  - 引数チェックをサボっている（^^；

- OpenELは、軽い方がいい
  - 呼び出し側が、正しい引数を与えていれば問題ない
  - 下の層（ハードウェア）の取りうる値が、OpenELではわからない
    - チェックできない

© Japan Embedded Systems Technology Association 2017

指標を考える:API関数あたりの行数



	エントリ関数の数	全行数(*.c , *.cpp)	関数あたり平均行数
OpenCV	623	111480	178.94
OpenRTM-aist	1644	85358	51.92
OpenEL	51	3467	67.98

- API関数あたりの行数
  - 行数が少なければ、手抜きの可能性ある(?)
- OpenELは、平均行数は結構多い
- 記述言語によるか…
  - オブジェクト指向言語で、共通部分が多ければ、行数は減る可能性あり
- OpenRTMは、API関数が多すぎ(?)

© Japan Embedded Systems Technology Association 2017

指標を考える



1. API関数あたりの行数
   - 行数が少なければ、手抜きの可能性ある(?)と考えたが…
     - 単純には、相関が無さそう

2. どれぐらいのバグ/問題なら、安心して使えるか?
   - 今回の3つは、どれもバグがあっても、使えている
   - 関数ごとに判断すればよい
     - fuzztestで問題が出たものを、個別に判断
     - ex. 初期化時の異常 入力は、ほとんど問題にならない

© Japan Embedded Systems Technology Association 2017

2018年度は最終年度



- 企業の商品もやってみたい、というアイデアがあった
- Fuzz testは、他のテストに比べると、ローコストに実施できる
- Fuzz testで問題のある関数を発見してから、判断すれば、コストパフォーマンス良い

© Japan Embedded Systems Technology Association 2019

「ロボット、機械学習AI OSSの紹介とOSSの品質についてのアプローチ」

2019/5/20 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町 6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL:http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。

一般社団法人 組込みシステム技術協会 Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association 2019

DEEP LEARNING講習で作成したデモの紹介

2019-05-20

株式会社Bee 中村仁昭

TL;DR

- Deep Learning講習の内容紹介
  - GPUが無料で使えるGoogle Colaboratoryで講習
  - フレームワークはTensorFlow + Keras
  - グループ間の情報共有にSlack
- 講習参加メンバー作成のDeep Learningデモ紹介
  1. サラリーマン川柳を作ってみよう!
  2. 人の表情から感情を識別する
  3. 文章感情分析から顔文字生成

DEEP LEARNING講習の内容紹介

環境

- 講習にはGoogle Colaboratoryを使用
  - Colaboratoryはクラウドで実行されるJupyterノートブック環境
    - 無料でGPUやTPUが使える
    - 連続では12時間しか使えないなど、いくつか制限もあり

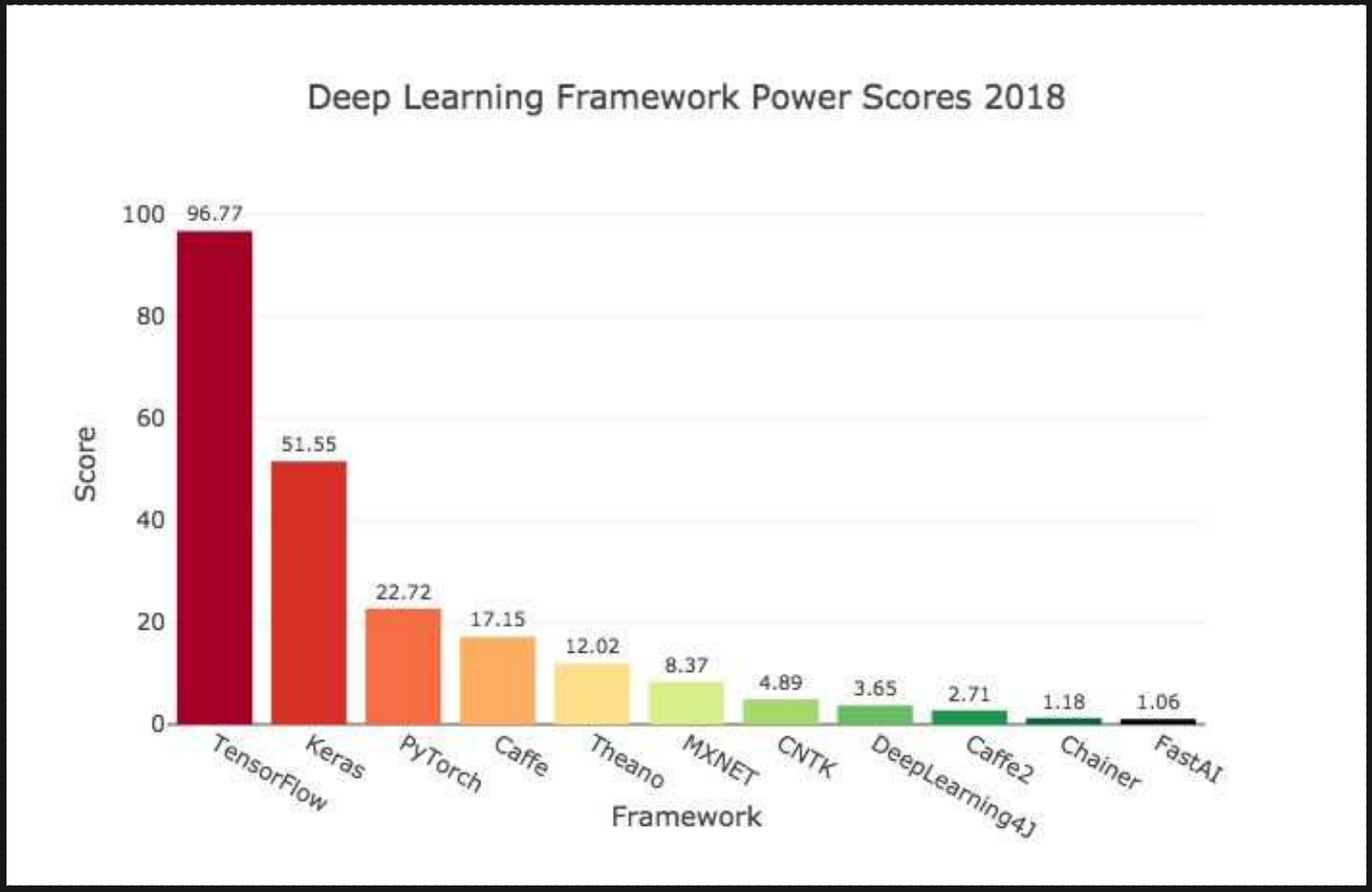


- ちなにみ初年度は…
  - ローカルPCにインストールしたPython + Jupyterノートブック環境で実施
  - Windows環境だとPythonインストールだけでももの凄い手間で不評…
  - GPUもないため動作が遅い…

フレームワーク

- フレームワークはTesorFlow + Keras
  - TensorFlow単体ではDefine-and-runなためネットワークの記述が大変
    - TensorFlow 2.0からはDefine-by-runなEager Executionがデフォルトになるが
  - フロントエンドにKerasを使うことによりDefine-and-runだがネットワークの記述が簡潔
  - シェア的にKerasかPyTorchの2択




Deep Learning Framework Power Scores 2018
Score
100
80
60
40
20
0
96.77
51.55
22.72
17.15
12.02
8.37
4.89
3.65
2.71
1.18
1.06
TensorFlow
Keras
PyTorch
Caffe
Theano
MXNET
CNTK
DeepLearning4J
Caffe2
Chainer
FastAI
Framework

情報共有にSLACK

- グループ間の情報共有と全体連絡用にSlackを使用
- 会社によってはSlackが使用禁止な場合もあり3グループ中1グループはメールで情報共有していた…
- Slackの活用は2019年度も継続して実施



セミナー内容

- 第1回 2018-08-08
  - Deep Learningとは？
    - ざっくりとした概要
  - Deep Learningの最近の動向
    - CVPR2018の動向
      - Computer Vision分野のトップ会議
    - ACL2018の採択論文を眺める
      - 自然言語処理のトップ会議
    - Kaggleの説明
      - データ分析コンペのサイト
    - 参考になる情報源や参考書の紹介
  - Python基礎
  - MNISTデモ

- 第2回 2018-09-12
  - Neural Networkの解説
  - Neural Networkの学習アルゴリズムの説明
  - 課題発表に向けたグループ分け

- 第3回 2018-10-17
  - ハイパーパラメータなど学習にあたってのテクニックの解説
  - CNNの解説
  - Keras MNISTの学習結果の可視化デモ
  - グループごとに課題の検討

- 第4回 2019-01-23
  - グループごとに決定した課題の内容と進捗発表
  - 課題を進めるにあたっての相談･検討

- 第5回 2019-03-06
  - グループごとに課題の成果デモを発表

講習参加メンバー作成のDEEP LEARNINGデモ紹介

サラリーマン川柳を作ってみよう！

- インターネットで公開されているサラリーマン川柳を収集
- 川柳をRNN/LSTMで学習
- 川柳の生成

文章感情分析から顔文字生成

- Twitterから絵文字つきツイートを収集
- 絵文字を教師データ(感情)にツイート文を1次元CNNで学習
- 感情と顔文字も学習させ、文→感情→顔文字の2段階で生成

人の表情から感情を識別する

- Google Imagesからツールで感情キーワード(anger, smileなど)で画像収集
- 収集画像からノイズを人力フィルタリング後CNNで学習
- Webカメラの入力から推論させて現在の感情を出力

IoT時代から見たハードウェア技術者のタスク

2019年 5月20日

ハードウェア委員会

碇山 真悟

一般社団法人
組込みシステム技術協会
Japan Embedded Systems Technology Association

© Japan Embedded Systems Technology Association

IoT時代から見たハードウェア技術者のタスク

目次

- はじめに
- ものづくりプロセスの例（IoT製品の場合）
- ものづくりプロセスの説明
- 具体的なタスクの内容（製品仕様）
  - 技術基準を確認する
  - 品質目標を決める
  - 取決めをする
  - その他
- 今後の活動予定
- 結びに

© Japan Embedded Systems Technology Association

■ はじめに

当委員会では、これまで時代と共に変わる組込みハードウェア技術者の役割を見てきた。

そして、「我々中小企業の多くが、それぞれの特定顧客への請負開発だけに業務が偏っている」ことに危機感を持った。

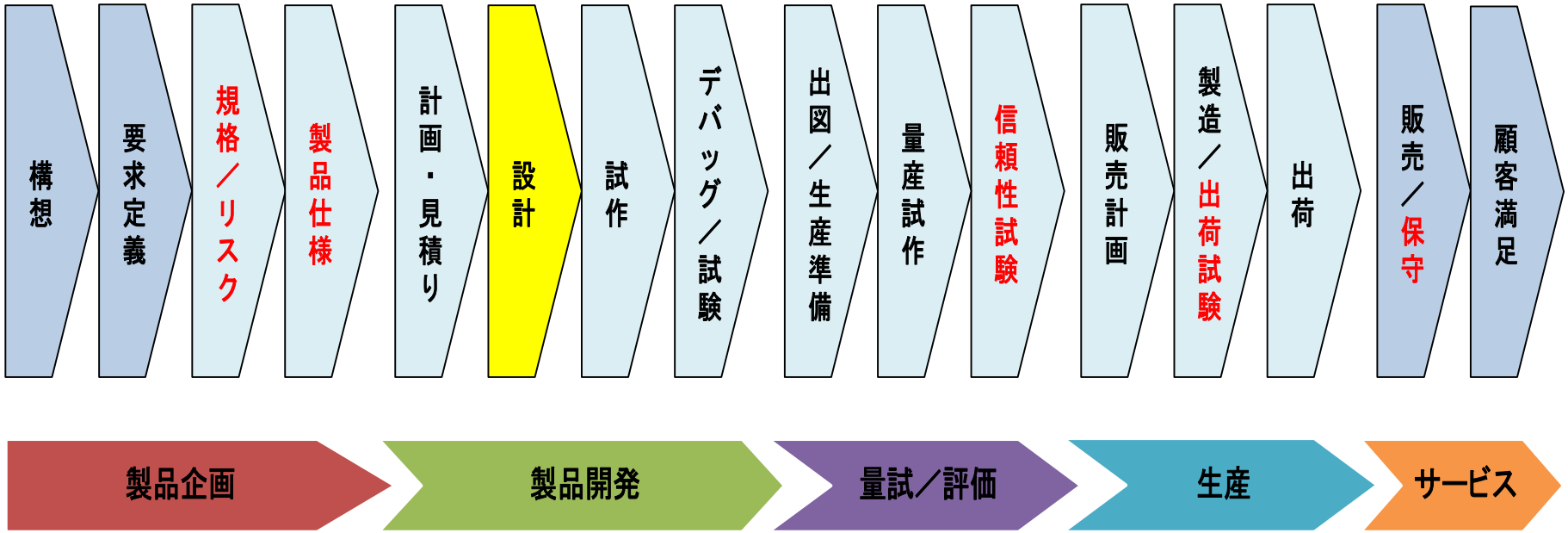
それで当委員会では、今後のハードウェア技術者はより上位（仕様・提案）へ業務拡張しなければならないと結論づけた。

⇒ スキルアップ＋ステージアップ

業務拡張することで実際にどのようなタスクが技術者に求められるか、IoT製品の開発を例に挙げて具体的に説明する。

© Japan Embedded Systems Technology Association

■ものづくりプロセスの例（IoT製品の場合）　ハードウェア委員会



© Japan Embedded Systems Technology Association

■ものづくりプロセスの説明

➢ ものづくりプロセスの5つの大きなステップ

1. 製品企画：①商品構想～②市場調査／顧客要求～③**法規・規格／リスク調査**～④**製品仕様（提案）**
2. 製品開発：⑤開発計画／見積り～⑥設計～⑦試作～⑧デバッグ／テスト
3. 量試／評価：⑨出図／生産準備～⑩量産試作～⑪**信頼性試験**
4. 生産：⑫販売計画～⑬製造／**出荷試験**～⑭出荷
5. サービス：⑮販売／**保守**～⑯顧客満足（妥当性の確認）

➢ 上位へ業務拡張して増える業務とは

- 現状は、製品開発ステップを請負契約で受注し、それ以外の製品企画、量試／評価、生産、サービスはお客様が実施する場合が多い。
- 上位へステージアップするとは、先ずは製品企画ステップの④製品仕様に携ること。そのためには赤で記した③法規・規制・リスク調査、及び量試／評価ステップの⑪信頼性試験が大きく影響してくる。
- 協力工場と共創して量試／評価及び生産ステップを受注することもできる。
- 業務が増えることは、その分責任範囲が増えるが付加価値も増える。

© Japan Embedded Systems Technology Association

■具体的なタスクの内容（製品仕様）

➢ 製品企画ステップ-④製品仕様（提案）のタスク

最初に染み出しを目指す業務で且つ究極でもある。主なタスクは以下の4つだが、先ずは技術的な内容をお客様に提案することから始める。

1. お客様の曖昧な要求を製品仕様に落とす。　＞本丸
○ 2. 技術基準を確認する。（法規＝法律、政令、省令、その他条例）
○ 3. 品質目標を決める。（お客様と交渉、リスク管理）
○ 4. 取決めをする。（責任範囲の明確化、リスク管理）
5. その他

© Japan Embedded Systems Technology Association

■具体的なタスクの内容

2.　技術基準を確認する。

電気用品が原因の危険・障害の発生防止や、公共性に影響があるものには適正な運営を目的とした法規が定められており、また、それに対応する技術基準（規格）が存在する。IoT製品が関係する主な法規は以下の通りだが、輸出する場合には相手国の法規を遵守しなければならない。

- 電気用品安全法　⇒経済産業省HP、経済産業局、JET、TIRI
  IoT製品でPSEの対象になることは少ないが、リチウムイオン蓄電池を使用した電池ユニットを本体と分離して構成した場合、モバイルバッテリーとみなされてPSEの対象になる。（平成30年2月1日付け通達改正）

- 電気通信事業法　⇒総務省HP
  MVNO（仮想移動体通信事業者）のSIMを使用したサービス、LPWAサービスを商売として始める場合に届け出る。

- 電波法　⇒総務省HP、総合通信局、登録証明機関
  基本的に技適マークの部品だけで構成すれば不要だが、もし新規でアンテナを製作する場合には装置として技適認証を受ける必要がある。

© Japan Embedded Systems Technology Association

■具体的なタスクの内容

3.　品質目標を決める。 ※市場投入後に問題になることが多い

法規以外の性能に関する品質項目を定義して、それぞれ目標を決める。
また、顧客要求が無い項目については、自社で設定してお客さまの合意を得る。リスクはなるべく品質目標を達成することで回避できるようにする。

- 安全性
- 信頼性

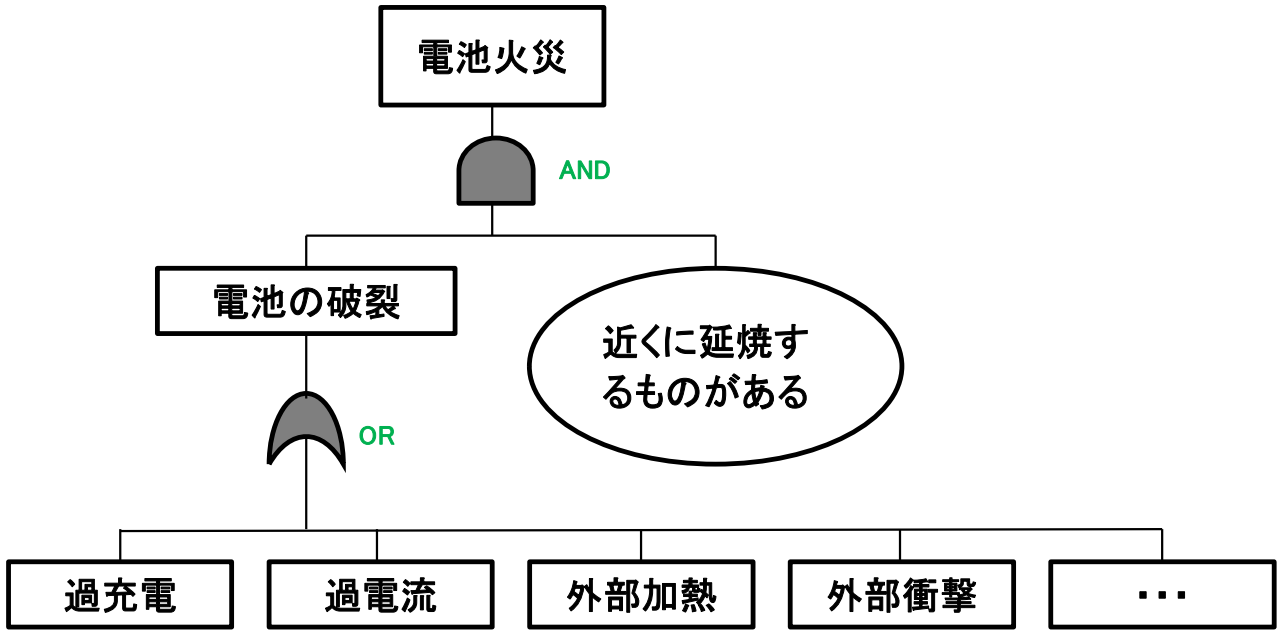
© Japan Embedded Systems Technology Association

■具体的なタスクの内容

3. 品質目標を決める。

- 安全性

人・環境・使われ方をイメージして、製品に起きてはならないこと（発火、感電、怪我など）が潜んでいないか可能性を挙げ、それぞれに対してその発生経路をFTAで論理的に明確にする。 ⇒方針を決め設計に盛込む



（FTA図の参考例）

© Japan Embedded Systems Technology Association

■具体的なタスクの内容

3.　品質目標を決める。

- 信頼性

□耐用年数、製品寿命

□振動、落下　　：疲労の蓄積や経年劣化の目標値は分からない　⇒経験を資産

□環境下動作

a. 温度動作

b. 温度サイクル

c. 防水　　：防水規格（IP表示）

d. 防塵

e. 耐候　　：自然環境に起因する劣化への耐性　⇒変色、変形、ひび

f. 塩水噴霧　　：腐食（さび）

□静電気放電(ESD)

□EMC評価

© Japan Embedded Systems Technology Association

■具体的なタスクの内容

■防水規格(IP表示)への過信

IP□□ →防水等級(8段階)
↓
防塵等級(6段階)

IPX6ケースが野外使用で浸水した例

①ケースの継ぎ目のガスケットの部分に雨水が溜まる。
②ケース内外の温度差等により水分がケース内部に少しずつ吸収され、水蒸気として内部にたまる。
③ケース内の水蒸気が結露し、ケース内面等に付着する。
④数か月後、ケース内の基板が浸水による不具合を起こした。

IEC規格／JIS規格	種類	保護の程度
IPX8	水中形	継続的に水没しても内部に浸水しない
IPX7	防浸形	一時的(30分)に一定水深(1m)の条件に水没しても内部に浸水しない
IPX6	耐水形	あらゆる方向からの強い噴流水によって有害な影響がない
IPX5	防噴流形	あらゆる方向からの噴流水によって有害な影響がない
IPX4	防沫形	あらゆる方向からの飛沫によって有害な影響がない
IPX3	防雨形	垂直から左右60度の範囲で落ちてくる水滴によって有害な影響がない
IPX2	防滴II形	垂直から左右15度の範囲で落ちてくる水滴によって有害な影響がない
IPX1	防滴I形	垂直から落ちてくる水滴によって有害な影響がない
IPX0	無保護	特に保護されていない

© Japan Embedded Systems Technology Association

■具体的なタスクの内容

4. 取決めをする。

万全を期して開発しても、IoT製品は市場に投入した後で思わぬ不具合を起こすことがある。特に品質問題については、いくら評価試験に合格したとしても不具合が起こってしまえばお客様からクレームが来る。

お客様と争いたくはないが、だからこそリスクになりそうな事項は明確にして取決めを行い、内容を記録する。

- 納入仕様（機能、性能）を作成する
- 保守体制を決める

  保守契約、保証書（期間、開始日、責任の範囲、免責事項）、取扱い説明書

- 保険を検討する
  - PL保険（製造物責任）
  - その他損害保険（カスタマイズ）

    a)リコール、b)歩留り、c)その他の賠償請求

© Japan Embedded Systems Technology Association

■具体的なタスクの内容

5. その他

IoT製品の製品開発～量試／評価～生産ステップが成功するか否かは、この製品仕様の「出来」に大きく左右される。

その他の注意点を以下に示す。

- 業界団体から情報を得る
- レビュアーに有識者の参加は必須
- ノウハウの蓄積と活用（失敗事例）

今回はハードウェア技術者が関連する部分を中心に説明したが、IoT製品の製品仕様を決める（提案する）ためには、ソフトウェア開発はもちろん、お客様、サービス内容、製造、保守などの観点も必要になる。

これらについては共創をキーワードに考えており、引き続きまとめて報告する予定。

© Japan Embedded Systems Technology Association

■今後の活動予定

- 2019年度ハードウェア委員会の活動予定

1. 設計・開発フェーズだけを請負うのではなく、製品あるいはサービス全体に業務範囲を広げるスキルを検討する。
   ⇒ 体験設計のワークショップを通して、お客様の曖昧な要求から本質的要求を抽出できるようになる。

2. IoTのハードウェアには独特の技術要素がある。例えばセンサー、無線通信、電池駆動・省エネ動作、耐候性等は、個々を見ると特に目新しくはないが、IoT端末には必ず備えていなければならない技術である。そのような「ある程度標準的な技術」でありながら中小にとっては共通の技術課題となっている項目を取り上げ、開発の参考になる情報にして発信する。
   ⇒ 2つ程度のテーマを選択し、チームに分かれてクローズされたネットで情報交換する。適宜まとめて「技術者が知っておくべき○○○」的な資料にまとめる。

© Japan Embedded Systems Technology Association

■ 結びに（再掲）

近頃ハードウェアへの意識低下を感じる場面もあるが、IoTには必ずハードウェアが存在する。

だがそこに自社の役割があるかが重要だ。

開発に於ける技術力の向上はもちろん大切だが、上位の領域（例えば企画や提案など）にハードウェア技術者が関与して提供するサービスの質を向上させることが重要と考える。そしてその領域で活躍できる技術者の育成を望む。

当委員会の活動が、組込みハードウェア技術者の一助となれば幸いに思う。

© Japan Embedded Systems Technology Association

【IoT時代から見たハードウェア技術者のタスク】

2019/5/16 発行

発行者　一般社団法人 組込みシステム技術協会
東京都中央区日本橋大伝馬町6-7
TEL: 03(5643)0211 FAX: 03(5643)0212
URL: http://www.jasa.or.jp/

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。