JEITA ワークショップ

IoTをソフトウェアエンジニアリングする: 品質の作り込みと評価

2017年11月2日

芝浦工業大学 情報工学科
中島 毅
E-mail: tsnaka @ shibaura-it.ac.jp

本日の話題

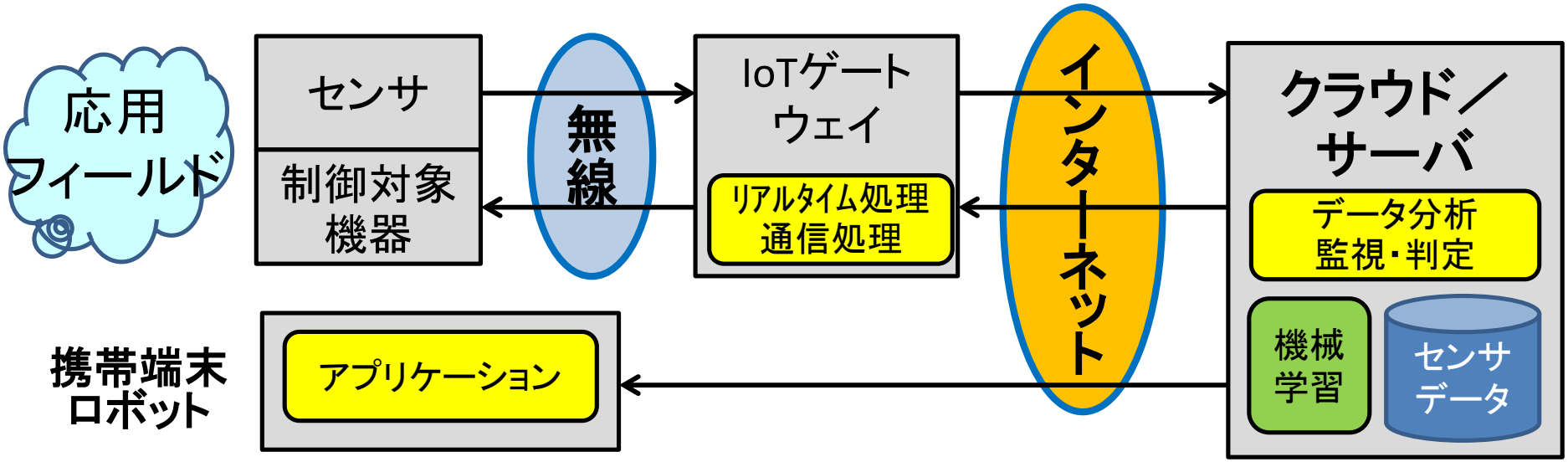
- Internet of Things (IoT) とその特徴
- 具体例でIoTの課題を考える
- IoT と Software Engineering
- 将来動向

IoT とは

- IoT:　Internet of Things
- M2M：Machine to Machine
  - 多数のセンサや機器をインターネットに接続し、人手を介さずに、様々なサービスを提供するシステム
  - センサ技術、無線通信／インターネット技術、情報処理技術の融合であり、横断的なシステム技術を要する

情報収集→通信→情報処理→デバイス制御

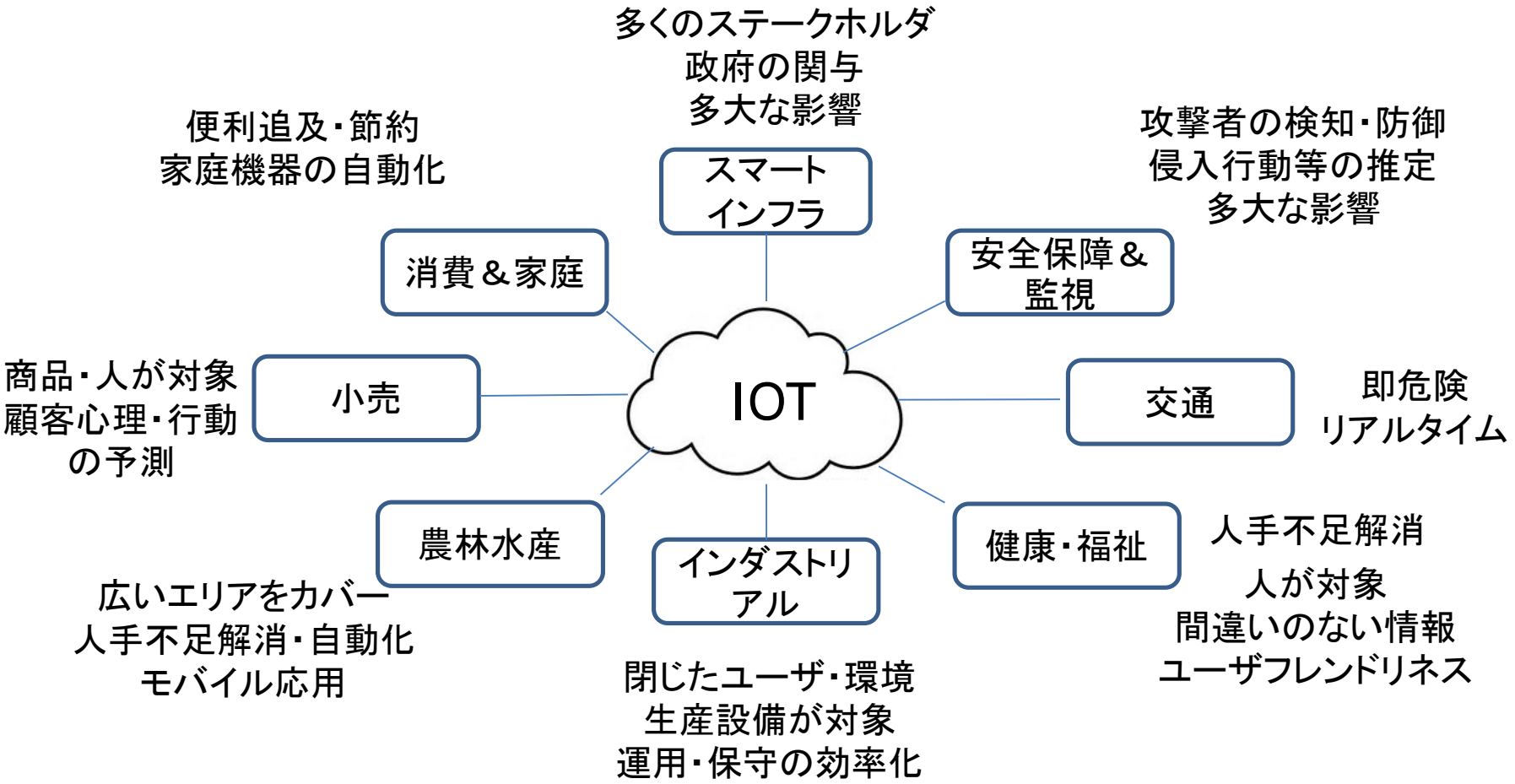
あらゆる「もの」がデータになる



IoTの特徴（1）

Razzaque, Mohammad Abdur, et al. "Middleware for internet of things: a survey." *IEEE Internet of Things Journal* 3.1 (2016): 70-95.

■ 多様な応用： 異なるステークホルダ，異なる対象・環境，異なる価値・リスク



IoTの特徴(2)

■ 多様なデバイス・リソース制約

RFID センサ デバイス

センサ ネットワーク

XBee
BLE

ローエンド コンピューティング デバイス

Ardiuno

ミドルエンド コンピューティング デバイス

Raspberry Pi

クラウド／サーバ

SCADA
フロントエンド

Edge rich

センサ＋
アクチュエータ

ドローン

ロボット

スマートフォン

強い　リソース制約（メモリ，処理能力，通信容量）　弱い

IoTの特徴(3)

■ 超大規模・動的ネットワーク

– 多数の「もの」がつながり(2020年 260億個), やり取り

⇒多数のイベント

–「もの」が移動・無線通信を行う

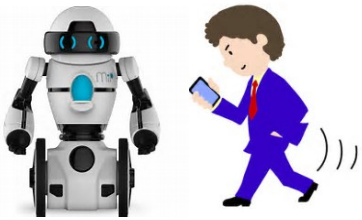
⇒ 動的なネットワークへの参加／離脱

⇒ 断続的インタラクション

動的にグループを作る

参加

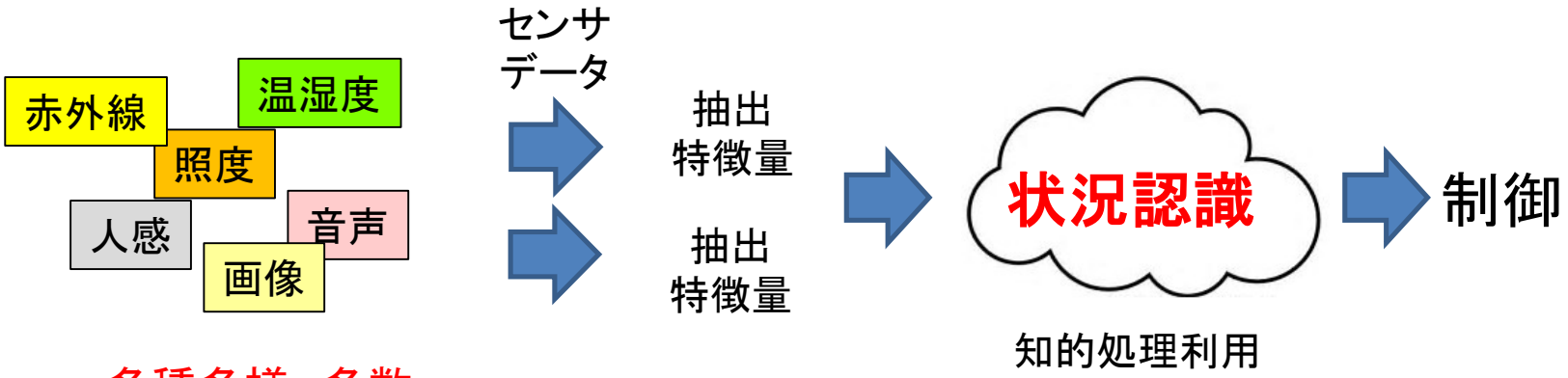
離脱



IoTの特徴(4)

■ 状況認識依存

– データは解析され解釈されて，意味をもつ



多種多様，多数，
異なる配置位置のセンサ群

– 位置情報が重要(絶対位置，相対位置)：

- 対象のいる場所(＋場所にひもづいた情報)
- 対象とセンサの位置(認識の正確さ)

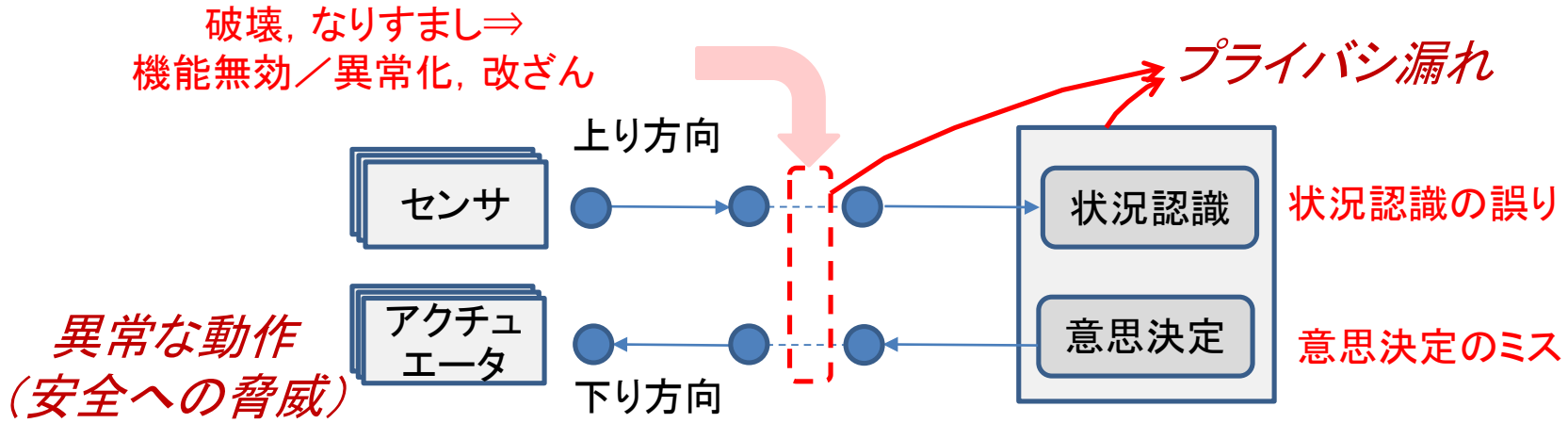
IoTの特徴(5)

■ 本質的な脆弱性と被害・影響の大きさ

– 識別／認証が極めて重要：多数のもの，動的な参加／離脱

– ロー／ミドルエンドのデバイスは脆弱

• 攻撃界面 の増加 ＋ 物理エリアによる遮断が困難



– 社会インフラの場合：　被害が甚大

– 監視対象が「人」の場合：　プライバシ漏れのリスク

本日の話題

- Internet of Things (IoT) とその特徴
- 具体例でIoTの課題を考える
- IoT と Software Engineering
- 将来動向

具体例でIoTの課題を考える

① 測定 と 知りたいことのギャップ
② 機械学習： 使いどころと，学習手段
③ クラウドコンピューティング VS
エッジコンピューティング

課題① 測定 と 知りたいことのギャップ

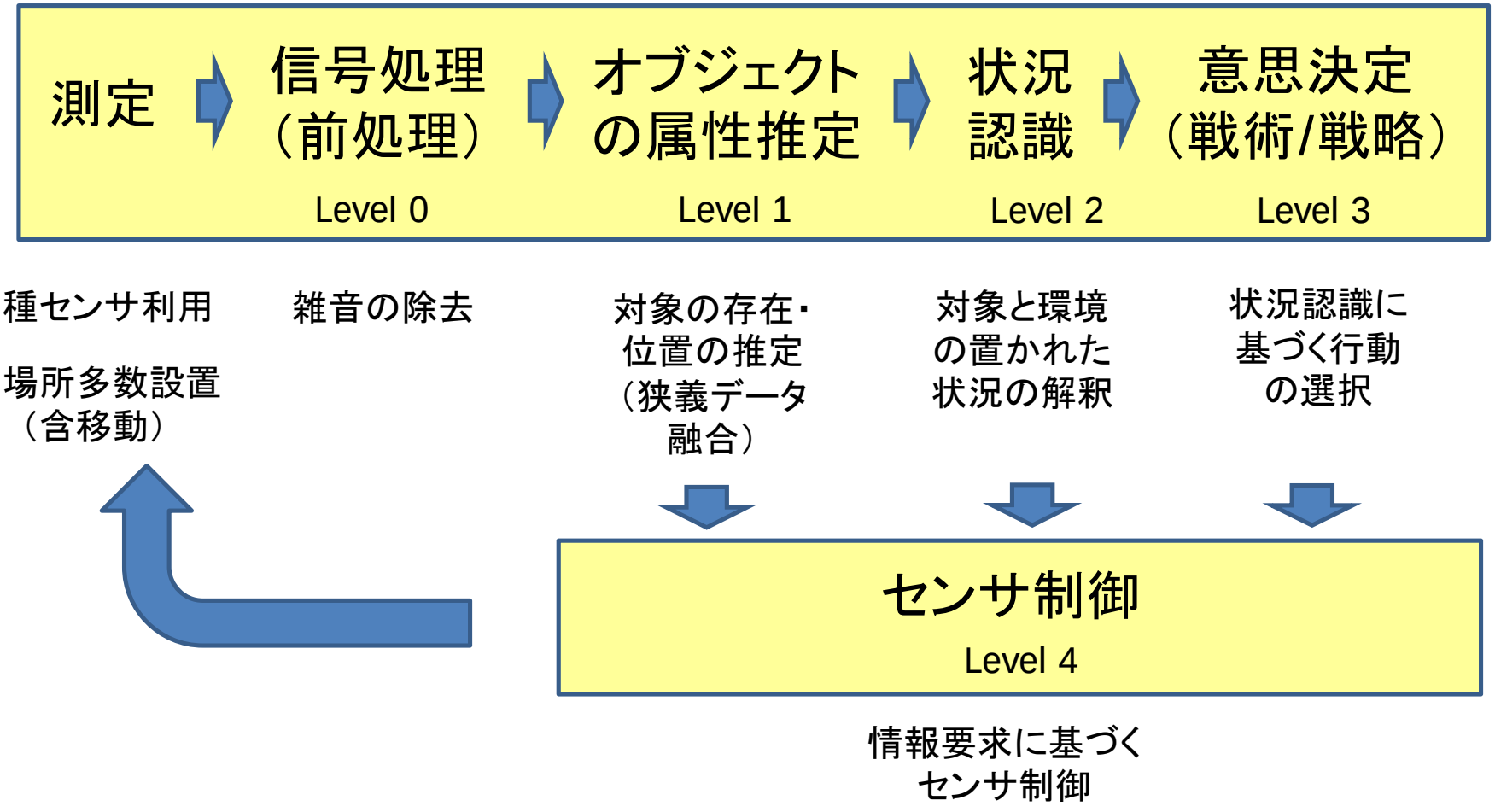
知りたいこと(状況認識): 対象の存在や状況

センサによる測定値: 物理的現象の観測

- **単機能センサ**: 観測状況により必ずしも正確にならない
  - 直接計測: 温度、湿度、照度、加速度、・・・
    - センサの計測可能範囲、計測精度
    - センサの設置場所: 設置の正しさ、外乱
    - 対象との関係: 距離、角度・・・
  - 間接計測: 人感、歩数、距離、・・・
    - 人感センサの場合: 赤外線／超音波／可視光、指向性
- **音声**: 環境音の識別は未開拓
- **画像**: 物体識別、状況認識は個別に(機械学習の領域)
  - 観測環境(照度、反射)
  - 観測機器の性能(解像度、処理性能)

データ融合： センサのことを整理して考える

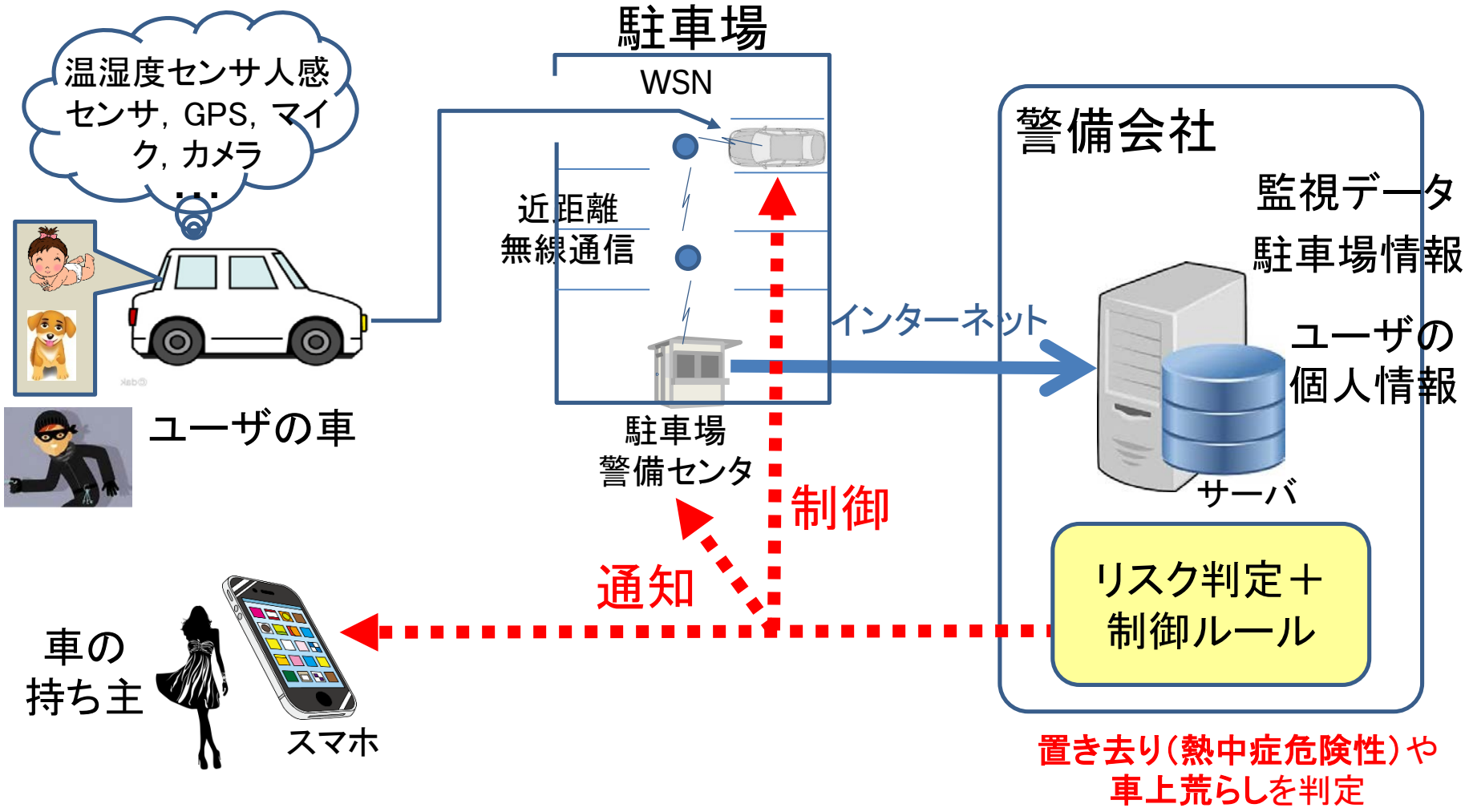
データ融合のモデル



Castanedo, Federico. "A review of data fusion techniques." *The Scientific World Journal* 2013 (2013).

状況認識問題：車内見守りシステム（15～）

車内にセンサを設置し，駐車場における 置き去り子供・ペットの熱中症、車の盗難、車上荒らし等を検知し通知するシステム

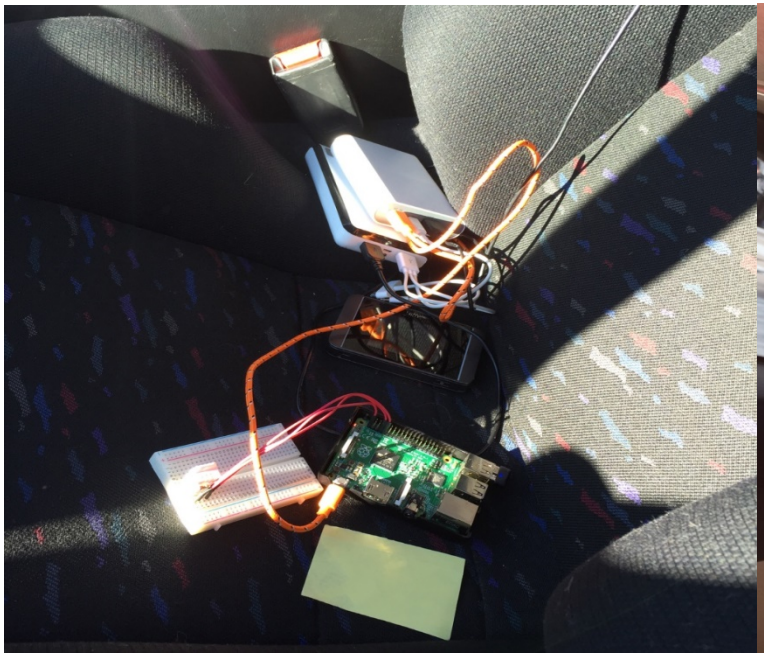


例: 温湿度センサ設置位置に関する妥当性実験

実験目的: 天井にセンサを設置することの妥当性評価
⇒ 天井の温湿度の測定値が，座席とどう違うか

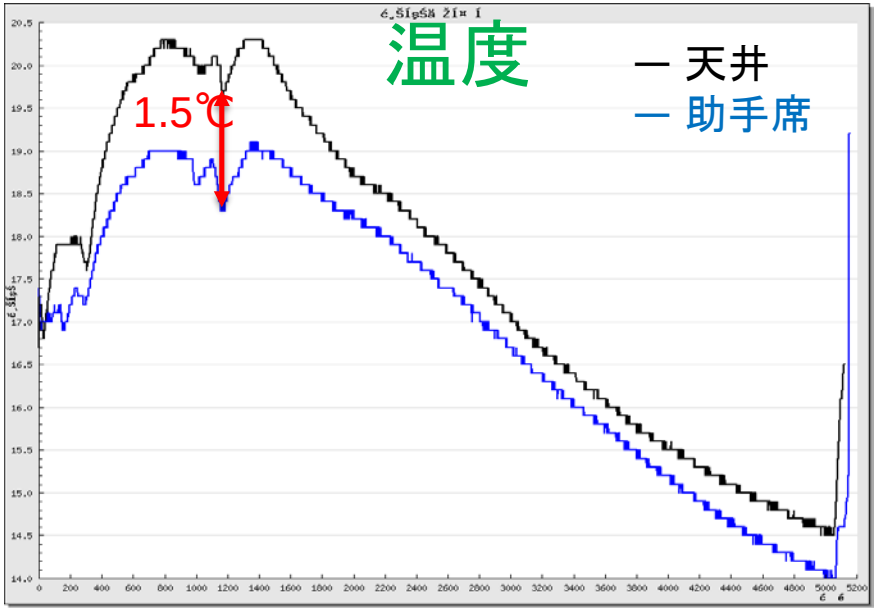
■実験

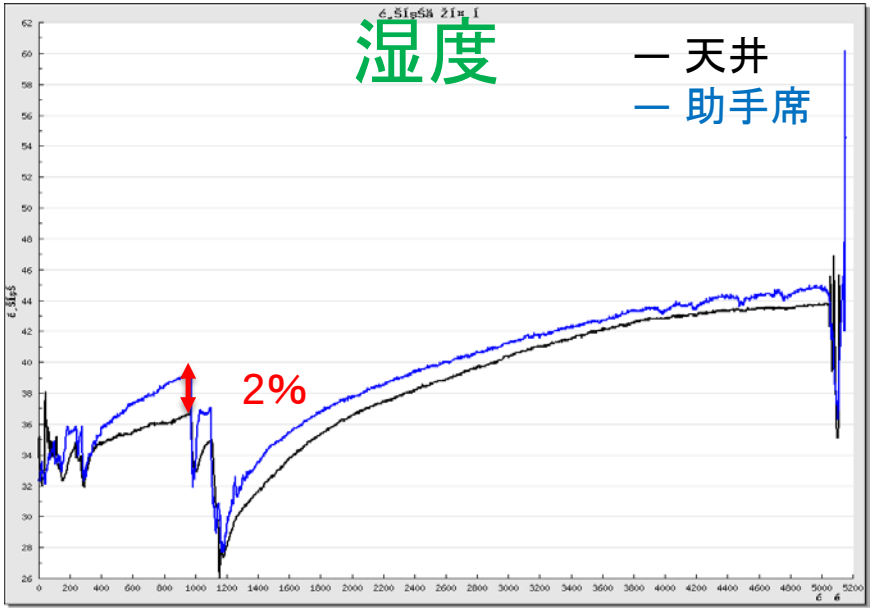
- 実験内容: 天井と座席との温湿度の差異を評価
- 実験方法: 実車で測定、天候の異なる二日間



実験1の結果（1日目）

天候：くもり（14：00～17：00）

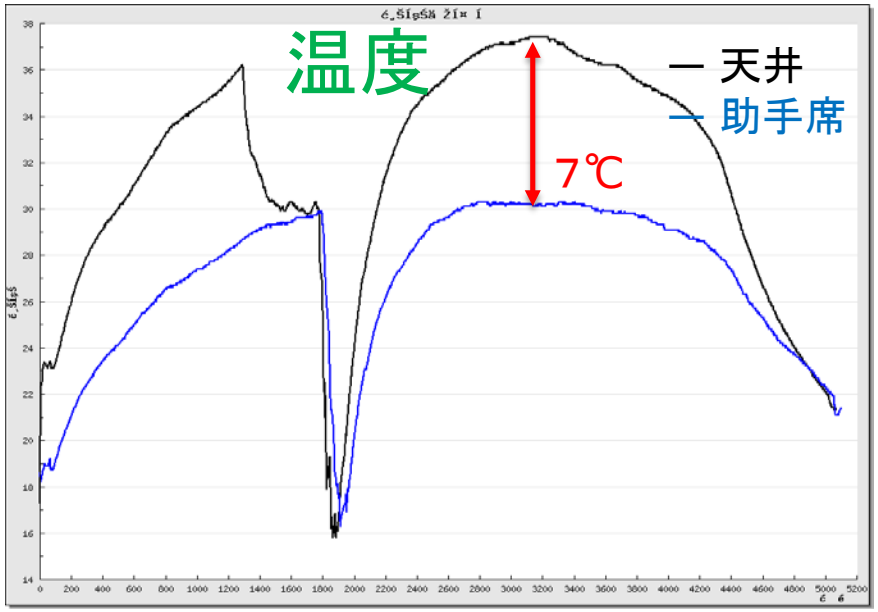


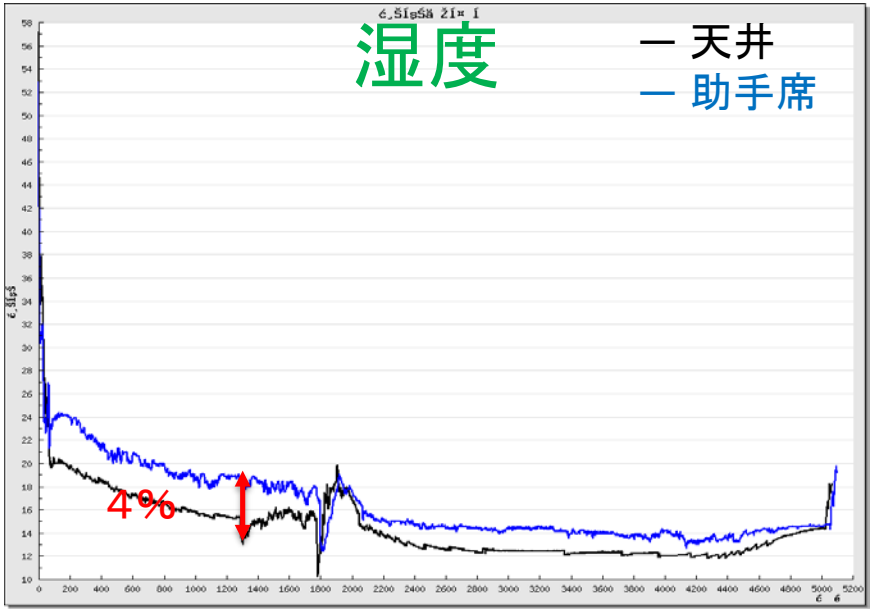


- 温度、湿度ともに追随
- 天井のほうが温度が高い（最大誤差1.5℃）
- 助手席のほうが湿度が高い（最大誤差2%）

実験1の結果（2日目）

天候：晴れ（14：00～17：00）





- 温度、湿度ともに追随
- 天井のほうが温度が高い（最大誤差7℃）
- 助手席のほうが湿度が高い（最大誤差4%）

データ融合の例： 状況認識

知りたい状況

①後部座席に子供が置き去りになっている

②前方座席に猫が置き去りになっている

③ドアが開いて車上荒らしが車内に侵入し、
物を盗んだ後にドアを閉める

- 画像と音を融合して判定する
  - 判定精度の向上：画像からの判定結果 OR 音による判定結果
  - 一連動作を見る：画像からの判定結果 AND 音による判定結果

状況認識： どのデータを組み合せるか

表： 検知したい事柄と必要なセンサ

検知したい事柄		重要性（〇は重要 △は付加的）	利用するデータ（〇主、△従）			
			人感センサ	温湿度センサ	音	画像
熱中症	①対象の存在	〇	〇		△	△
	②対象が人か動物か	〇			△	〇
	③温湿度	〇		〇		
	④対象の動作	△				〇
車上荒らし	⑤対象の存在	〇	〇		△	△
	⑥対象の動作	△				〇
	⑦ガラスが割れた音	〇			〇	
	⑧ドアが開いた音	〇			〇	

シナリオ①、②

シナリオ③

状況認識：認識率評価

総合判定Zは次式によって導出する

$$Z = \begin{cases} (1-(1-x)(1-y)) & (\text{ORの場合}) \\ x * y & (\text{ANDの場合}) \end{cases}$$

x: 音声における検知率(0～1)
y: 画像における検知率(0～1)

評価結果

シナリオ	画像部門	音声部門	総合判定Z
①	100%	61%	61%
②	87.50%	44%	93%
③	100%	98.70%	99%

- シナリオ①はカメラとマイクだけでは不十分
- シナリオ②、③はよい精度で把握できている

② 機械学習： 使いどころと，学習手段

使いどころ

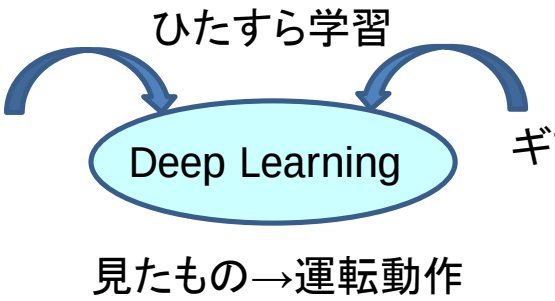
状況認識

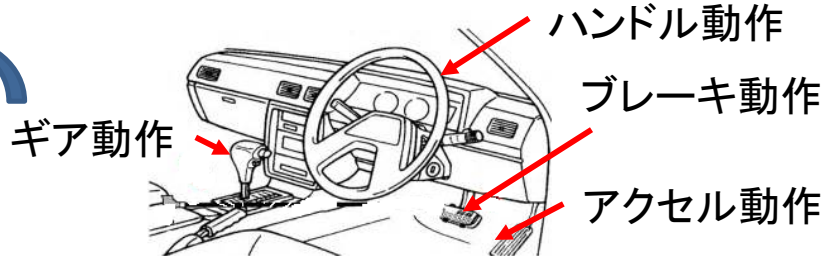
AND/OR

意思決定

入力：大量の走行時画像

出力：入力時の運転者の動作



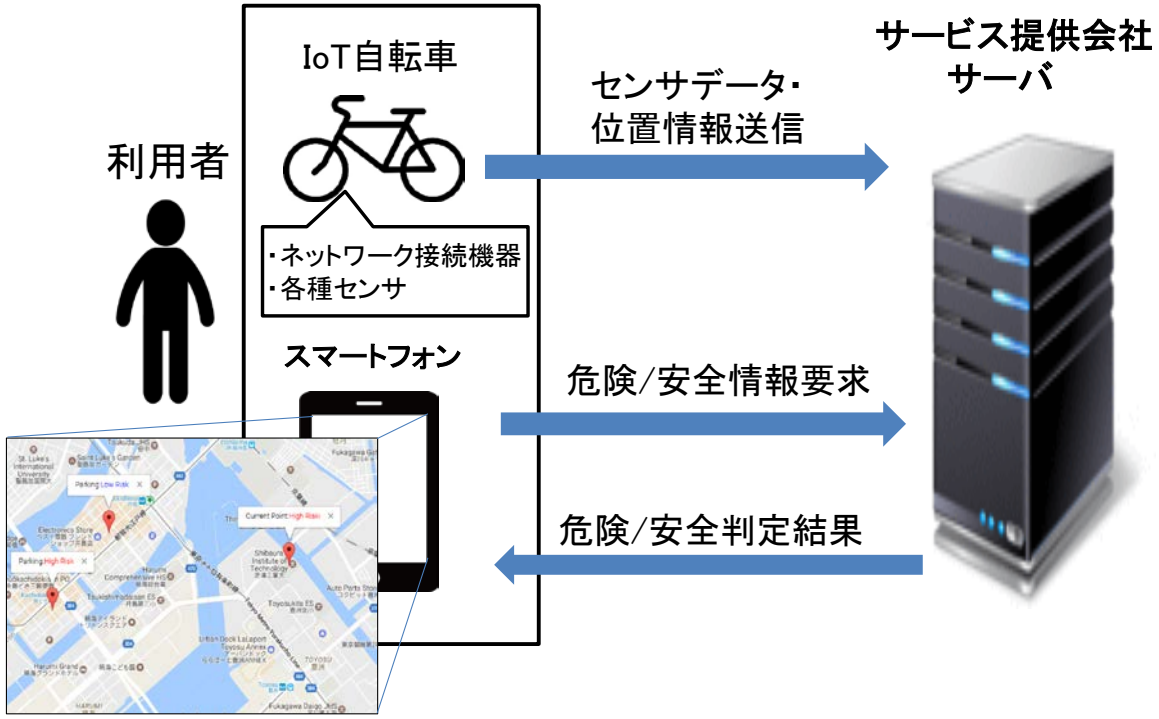


機械学習のエンジニアリング・V&V

- Python: ベクトル・行列演算に優れた言語
  → ほとんどのすべての機械学習がライブラリ化
  - TensorFlow(ディープラーニング)
  - scikit-learn(SVM, Baysian Network, その他多数)
- Engineering:
  - データ設計が命, プログラミング的な要素はほとんどない.
    - 出力データに対して「何が善悪か」を判断できる
    - 入力データを，多数観測／収集できる
    - 良い入力データを設定できる（出力との因果関係）
- Verification & Validation
  - 検証ではなく妥当性確認（ひたすら現実への適合性をみる）

AI応用：駐輪危険情報提供システム（16～）

自転車盗難防止・対策

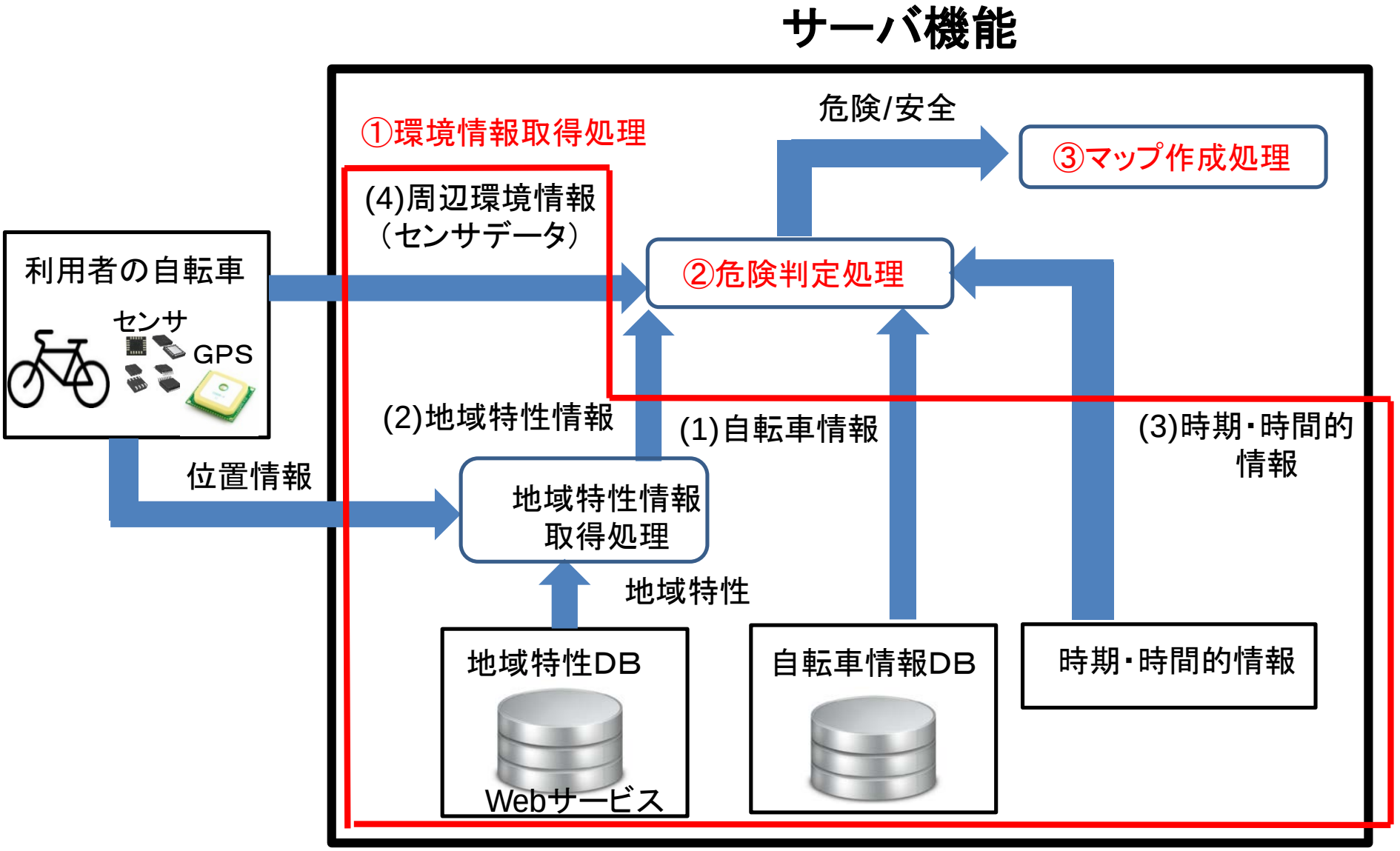


- センサデータの取集
- 危険／安全判定：機械学習
  サポートベクタマシン
  ベージアンネットワーク
- Webプログラミング
- 危険度の量化 と 根拠提示

IEEE ICIOT 2017 @Hawaii

国際学会にて発表
（4年生成果）

サーバ上の処理


サーバ機能
①環境情報取得処理
危険/安全
③マップ作成処理
(4)周辺環境情報
（センサデータ）
利用者の自転車
センサ
GPS
②危険判定処理
(2)地域特性情報
(1)自転車情報
(3)時期・時間的
情報
位置情報
地域特性情報
取得処理
地域特性
地域特性DB
Webサービス
自転車情報DB
時期・時間的情報

環境情報：　取得情報と取得方法

環境情報	取得情報	取得方法
(1)自転車情報	色・値段 ・車種・鍵の個数	利用者が予め登録
	駐輪時間	利用者が毎回入力
(2)地域特性情報	犯罪件数・人口密度・地域別平均年収	位置情報とシステムに登録された情報を照合する
(3)時期時間的情報	時間・月・曜日・平日／休日	ユーザリクエスト時に入手
(4)周辺環境情報	人通り量	人感センサを利用：一分間あたりの反応秒数
	天気	GPSを利用：位置情報と天気情報の照合
	照度	照度センサを利用：センサデータの値
	温湿度	温湿度センサ：センサデータの値

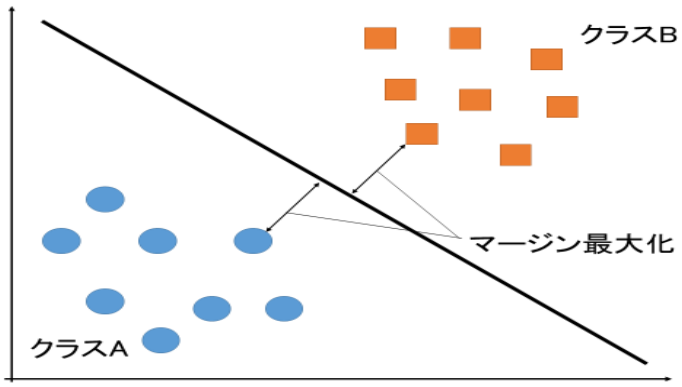
危険判定処理

機械学習手法

SVM（Support Vector Machine）を利用

SVMとは

識別対象を2クラスに分類する機械学習手法
マージン最大化を利用



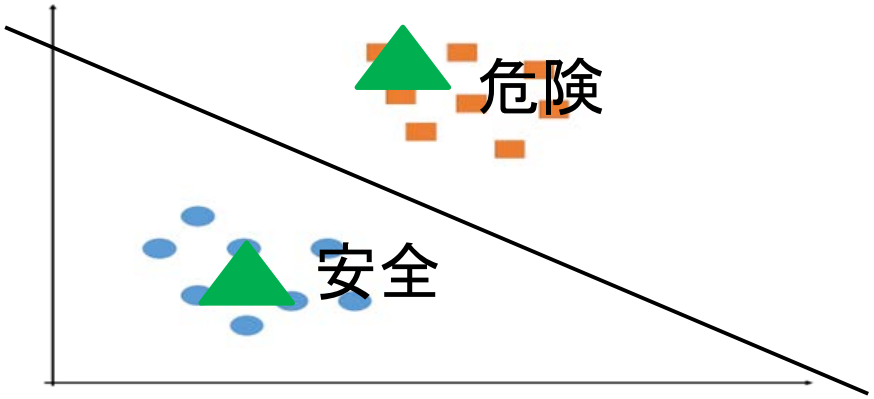
危険判定処理にどうSVMを用いるか

学習データ

盗難発生環境データ

盗難未発生環境データ

盗難発生/未発生の境界を作成



判定対象
データ

判定データがどちらに分類されるかで危険/安全を判定

適応上の課題：学習データをどのように作成するか

盗難発生環境データ → Web公開データあり，しかし周辺環境情報が欠落

盗難被害データ（Web公開情報）	＋	位置情報→統計データ	＋	時間・状況→現地調査
位置(0)＋自転車 (1)＋時間(3)＋その他状況		地域(2)		周辺情報(4)

盗難未発生環境データ → 公開情報等が存在しない

- 盗難被害未発生駐輪場の中から無作為抽出
- 時間と自転車(乱数で決定)を設定，現地調査

駐輪場データ（Web公開情報）	＋	位置情報→統計データ	＋	無作為抽出	時間・状況→現地調査
位置(0)＋その他状況		地域(2)		時間＋自転車(1)	周辺情報(4)

有効性評価：

調査データ計37件：

8割（盗難19件、非盗難11件）を学習データ

2割（盗難4件、非盗難3件）を評価用データ

		真の値	
		盗難	非盗難
判定結果	盗難	4	2
	非盗難	0	1

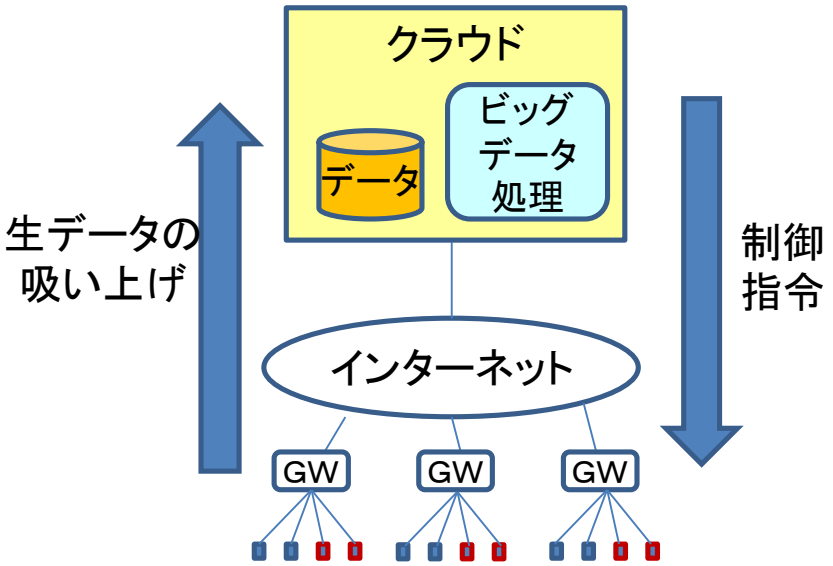
正解率は約71．4%、非盗難の誤判定は0%

本システムにおいてSVMは有効であると評価

課題③：クラウドコンピューティング（集中） VS エッジコンピューティング（分散）

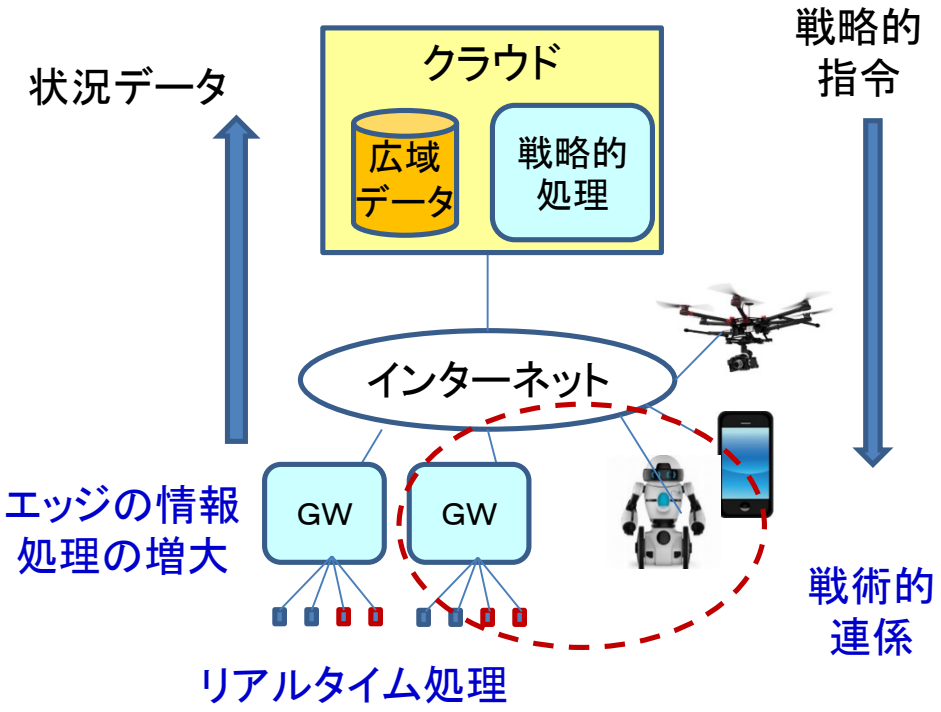
設計上の集中／分散のバランスは常に課題

クラウドコンピューティング



- 処理の簡素化
- 保守の容易化
- 知識の集約

エッジコンピューティング



- 認知→制御の即時性
- 通信負荷の低減
- 知識のローカライズ

授業の教材から：ドローンを使った放牧の自動化

放牧地にカメラを有する複数のドローンを展開し、家畜の食事、牧場までの追い込みを行うシステム

- 広域の画像撮影⇒群れの位置把握、猛獣の認識
- 必要に応じ連携して群れの追跡・追い込み，猛獣を追い払い
- 監視者は、自宅で情報を把握．追い込み・追跡開始など指令

処理方式？

トレードオフ事項は？

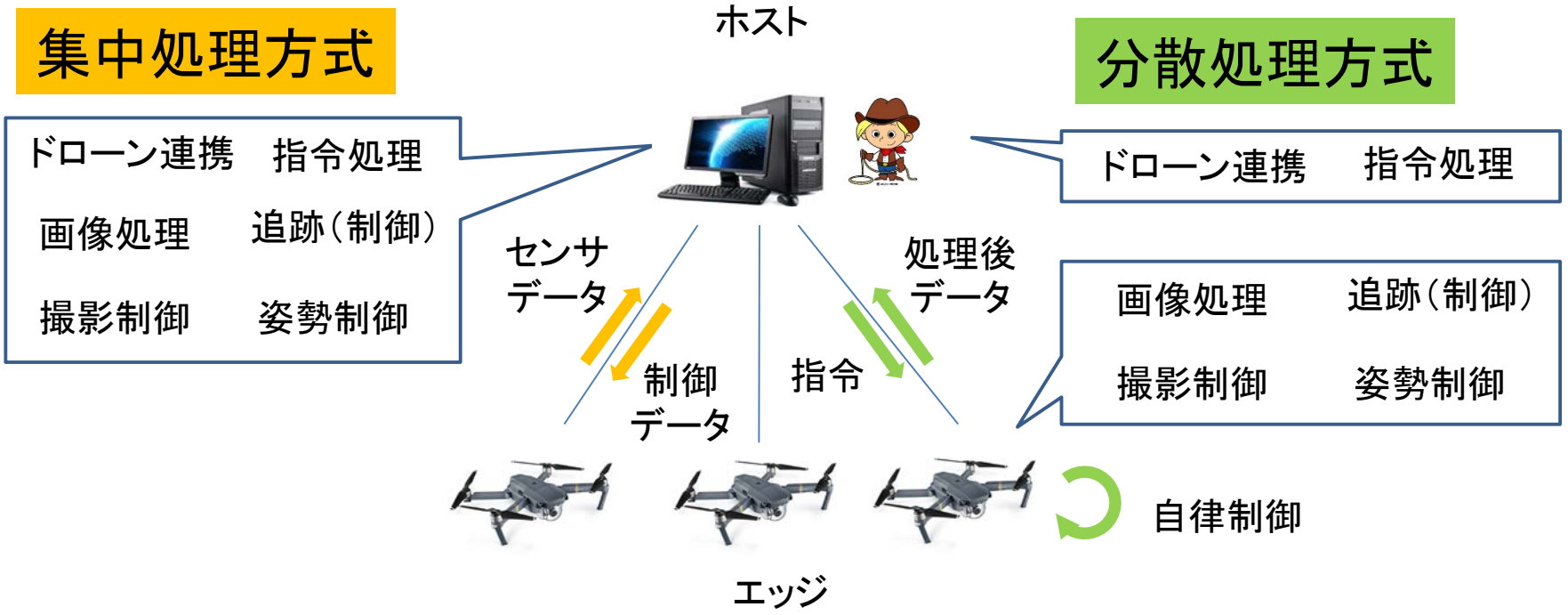
- リアルタイム性、
- 耐故障性
- 負荷（通信、処理）
- 変更性、試験性

やるべき処理 と 処理方式

やるべき処理

- 姿勢制御，撮影制御，画像処理，追跡（制御）
- ドローン連携，指令処理

処理方式



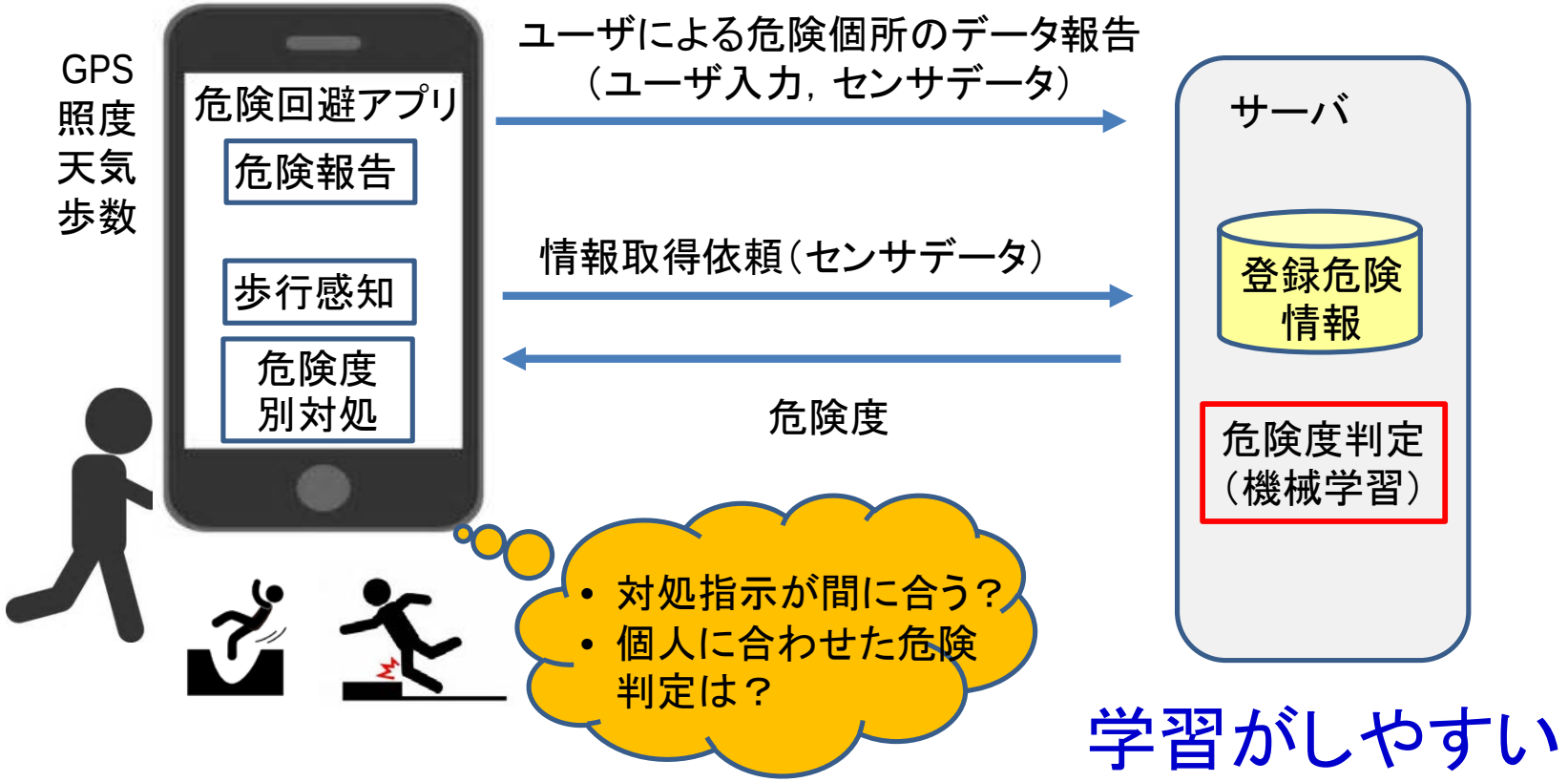
トレードオフ： 集中（クラウド）VS 分散（エッジ）

設計（方式）にはトレードオフがある

トレードオフ事項		集中処理方式		分散処理方式	
ノード処理負荷	ホスト	大（ほぼすべての処理）		小（エッジ間調整，データ統合等）	
	エッジ	小（センサデータ処理，HW制御）		比較的大（リアルタイム性高い処理）	
通信負荷		高い		低い（エッジで処理後のデータ）	
データの一貫性		データはホスト上で一元管理されるため、一貫性を維持可能		データの入力源、所在が分散するため、一貫性を保つしくみが必要	
システム標準化		○	標準化容易	×	エッジ処理は標準化が困難
効率性	リアルタイム処理	×	通信時間により困難	○	エッジ処理で高速処理可能
保守性	運用管理	○	容易。HWもSWも集中管理	×	HWやSWの管理が困難
	テスト容易性	○	容易。ホストのテストに集約	×	種々のパターンがあり困難
	エッジ個々のニーズ対応	×	対応不可	○	エッジへの機能付加・性能改善で対応可能
	拡張性	○	ホストで一元的に対応可	×	エッジ設置・試験の負荷大
信頼性		×	耐故障性小（障害時停止）	○	耐故障性大（障害時でも、大半の処理はエッジで対応可）
セキュリティ		○	ホストの対策に集約	×	各エッジで対策が必要

エッジ：歩きスマホ危険回避システム（16～）

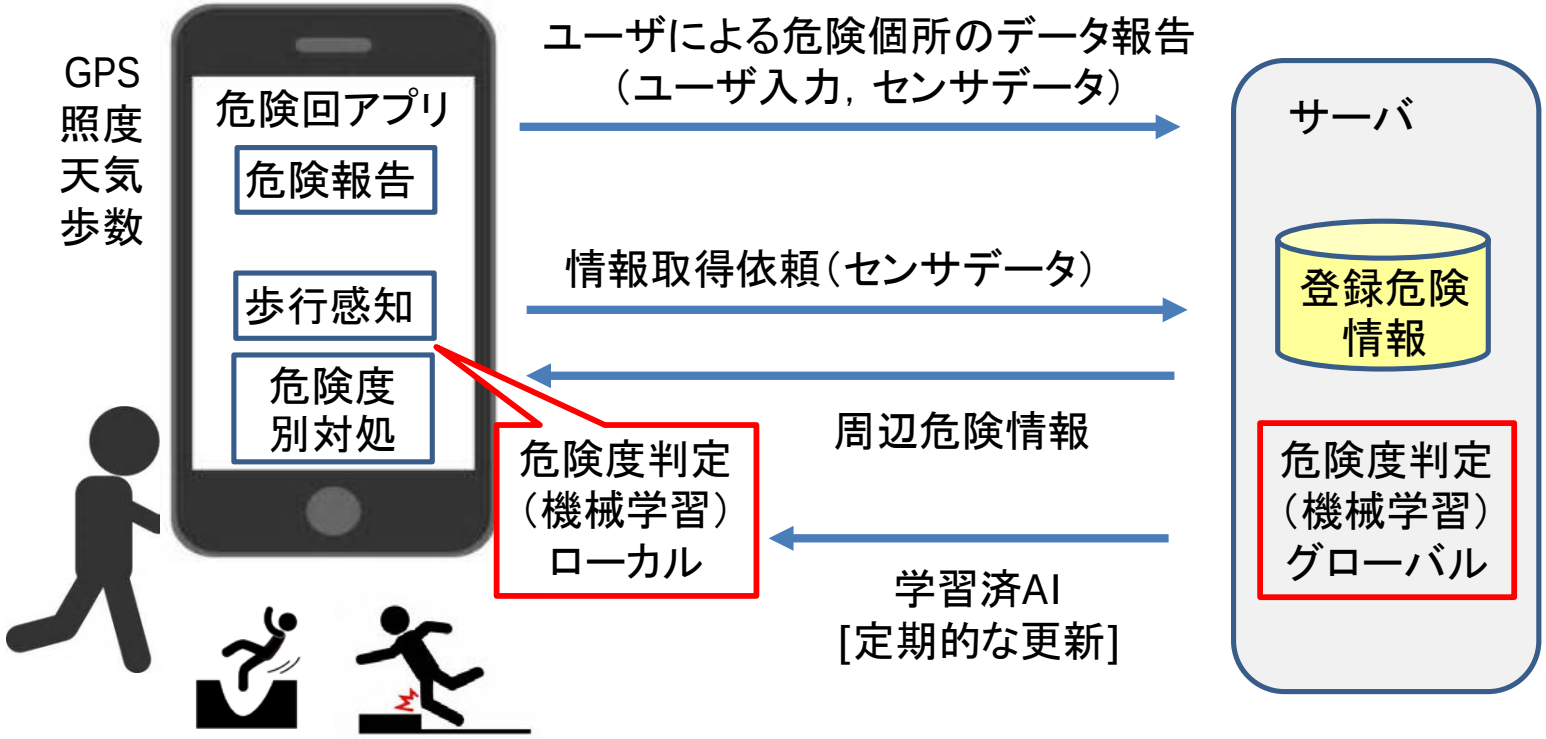
歩きスマホの危険を，危険に応じて通知/遮断



学習がしやすい

エッジ：歩きスマホ危険回避システム（16～）

歩きスマホの危険を，危険に応じて通知/遮断



- 対処指示の即時性の確保
- 個人に合わせた危険判定

AIの別の課題

- 学習の分担
- 学習済データの配布・統合

本日の話題

- Internet of Things (IoT) とその特徴
- 具体例でIoTの課題を考える
- IoT と Software Engineering
- 将来動向

話題： IoTソフトウェア工学における問題

- 要求定義
  - 運用下での動的な要求獲得
  - 品質要求の干渉・バランスの達成
- 設計と実装
  - 標準化未成熟→ノウハウの塊

IoTはENIAC時代を彷彿させる

- 保守
  - 多種多様なノード，それらからなるシステム
  - 構成管理：動的な構成変化，DevOps設定での保守
- 品質要求の実現
  - セキュリティ，信頼性，使用性，性能効率性

Spinellis, Diomidis: Software-Engineering the Internet of Things, IEEE Software, Vol. 34 No. 1 (2017)

品質要求の定義と実現

- SQuaREシリーズとIoT
- 品質要求の実現

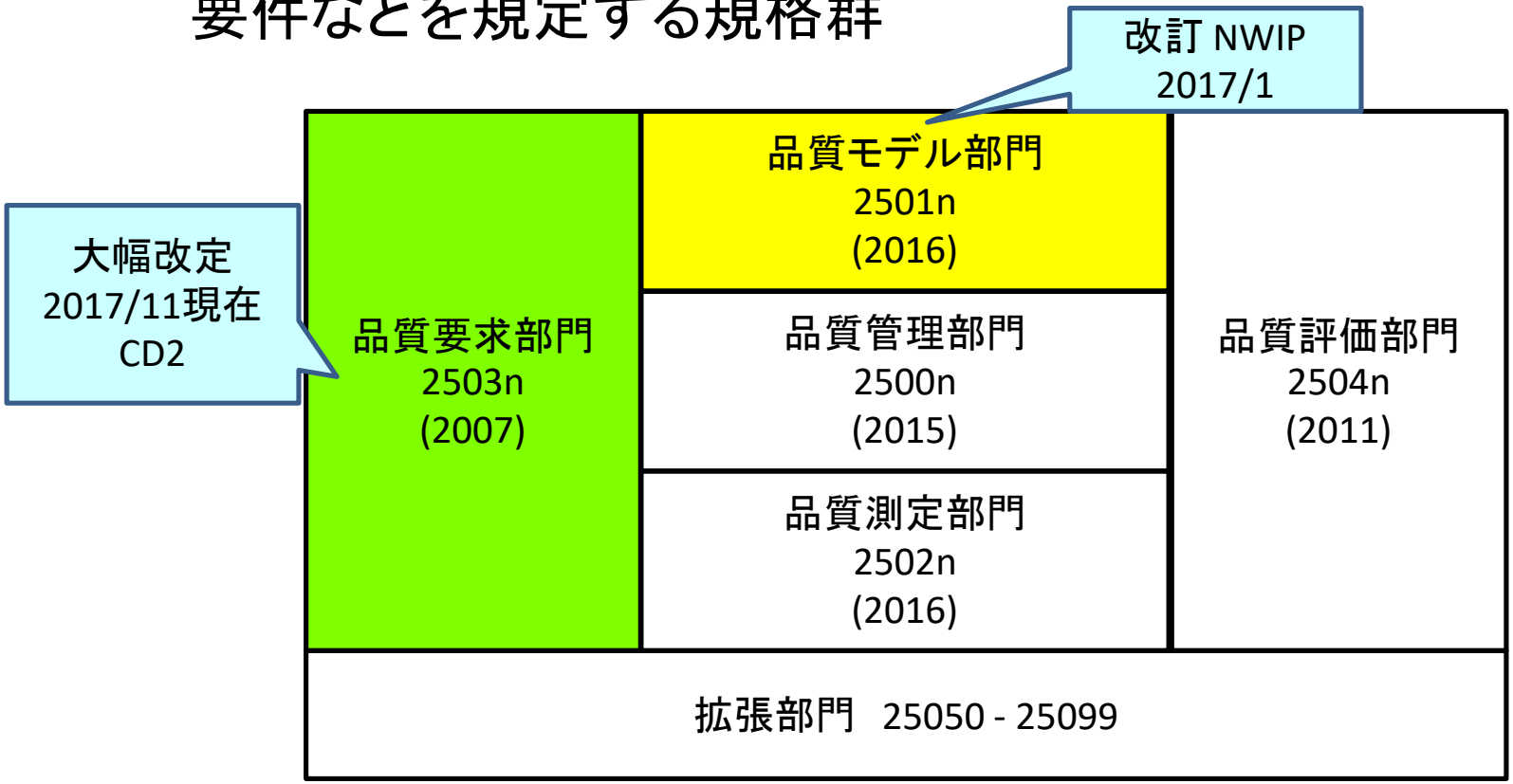
システム＆ソフトウェア品質に関する国際標準

SECBOOK「つながる世界のソフトウェア品質ガイド」

JIS閲覧　http://kikakurui.com/

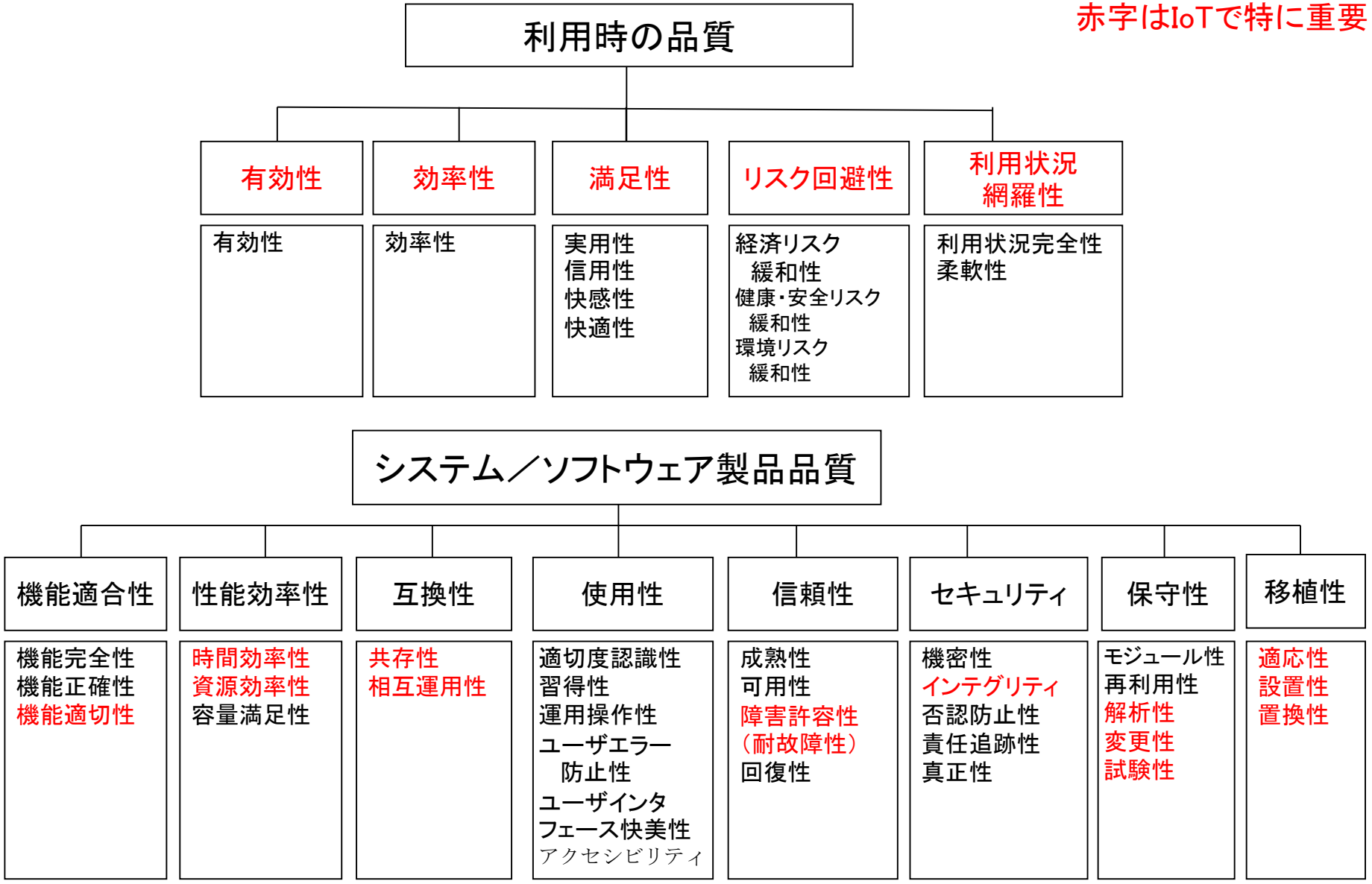
SQuaRE（ISO/IEC 25000 ）シリーズ

システム＆ソフトウェアの品質要求定義に関して、
基本的な概念、具体的な品質の定義や測定方法、作業要件などを規定する規格群

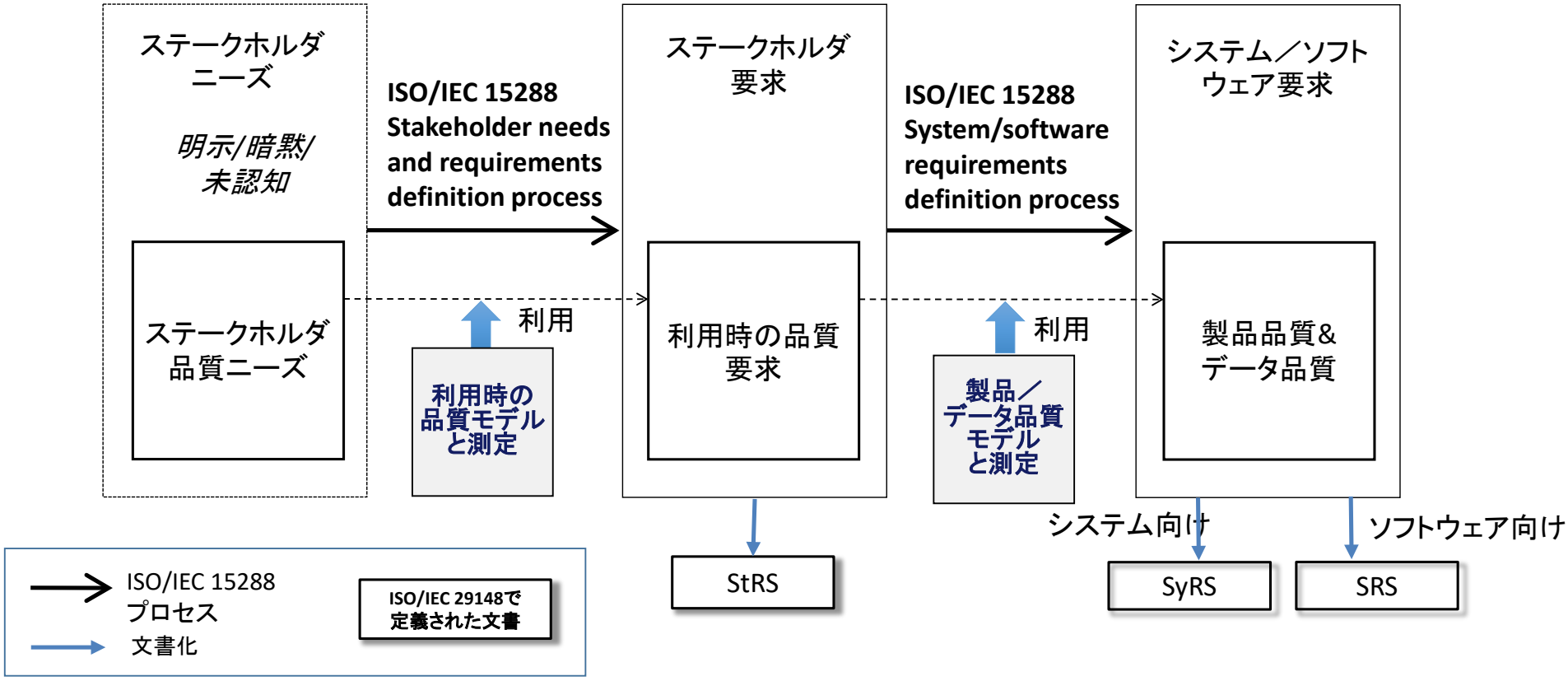


品質モデル： (ISO/IEC 25010:2011)

赤字はIoTで特に重要



ISO/IEC 25030-Rのスコープ



ステークホルダからの品質要求の獲得

- 品質要求の多様なソースに注意する
- 最終的には要求の利用者がまとめる
- ユーザ以外に影響をうける人に注意する

品質要求		ステークホルダ					
		ユーザ			その他のステークホルダ		
		一次ユーザ	二次ユーザ	間接ユーザ	開発者	調達者／独立評価者	影響を受ける人
利用時の品質要求		S	S	S	U	U	R
製品品質要求	ふるまい的	S	S		S, U	U	
	内部的				S, U	U	
データ品質要求		S	S	S	S, U	U	

S: 要求のソース, U: 要求の利用者, R: システムにより(悪)影響を受ける人

ステークホルダ とビジネスモデル

ステークホルダ	役割	システム利用目的 （期待する効果）	義務→責任	経費
独居老人	対象 （契約者）	①自分の異常を判断し治療や対処をしてもらう ②自分の体や行動をデータにより客観的に知る	・ウェアブルセンサをつける ・家の中にセンサを置く ・システムの助言に従う →義務違反による問題は自己責任	・月一定額を管理会社に支払う ・スマホの通信料
管理会社	システム運用者	①契約者からサービスフィー（×人数）を得る	・端末＋センサの購入・設置・保守・除去 ・判定ロジックの保守 ・システムの保守 ・判定ロジックの保守 →情報の正確性，適時性に関する責任 ・質問への対応	・システム保守費 ・運用人件費 ・端末＋センサの購入，設置，除去費用
家族	情報受益者 （代理契約者）	①対象の異常を知る ②対象の体や行動をデータにより客観的に知る	・異常通知の受信 ・分析レポートを受け取る	（対象に対する対価，義務，責任を代理する場合がある）
看護担当者	情報受益者 （オプション）	①対象の異常を知る ②システムによる対象の危険状態の一次判断をもらう	・異常通知の受信・対処 →通知見落しは責任問題 ・治療方針の立案 ・処方	・情報をモニタ，異常対処するのためのリソース
政府	規制者		・法律・条令の遵守確認 →利用者の安全性確保	

ステークホルダからの要求の獲得例：インターネットショップ

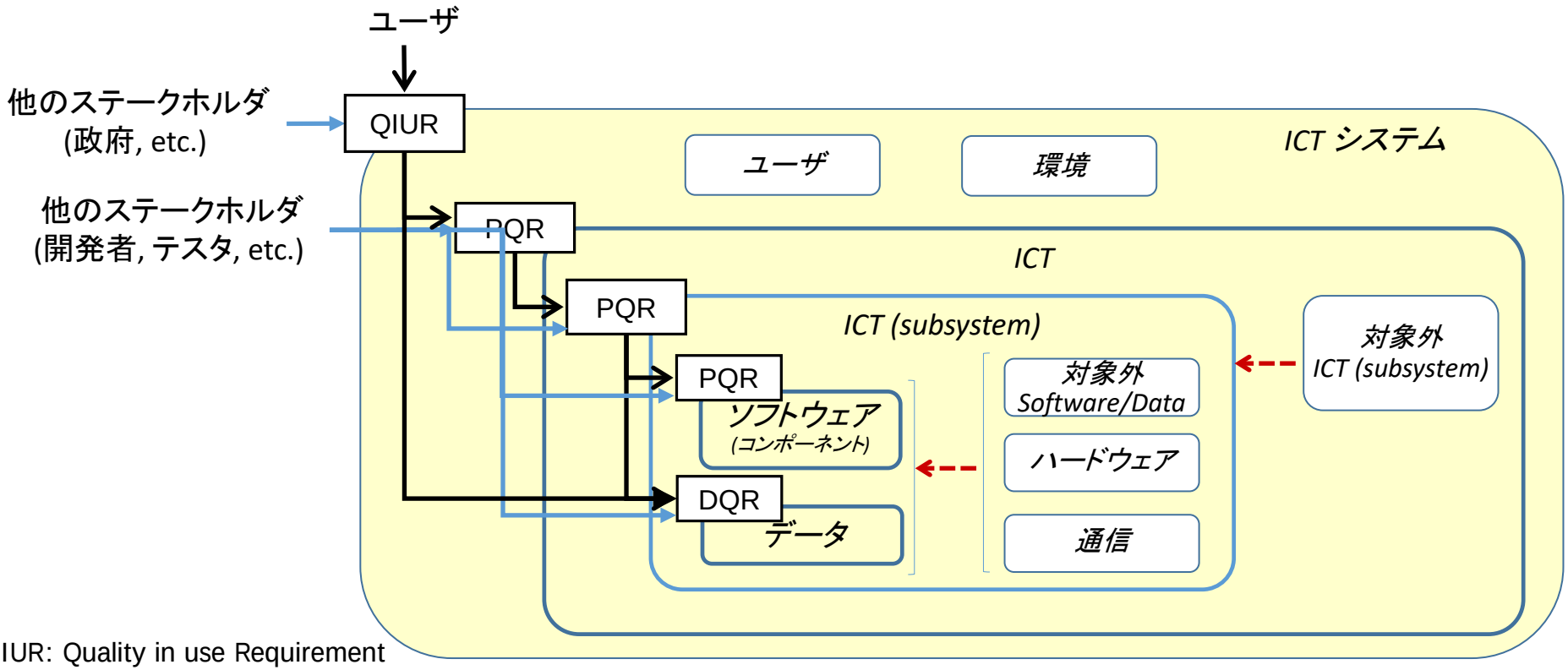
ステークホルダ / 対象		ユーザ				
		1次ユーザ		2次ユーザ		間接ユーザ
		インターネット買い物客	*ショップ内オペレータ*	*買い物ヘルパー*	*売り場部門の管理者*	*ショップのオーナ*
ICTシステム（人とシステム）			When he answers to questions from end users; **Effectiveness, Efficiency** When he changes/delete user's order; **Freedom of risk**	When he answers to questions from shoppers; **Effectiveness, Efficiency** When he changes/delete user's order; **Freedom of risk**	On the business process of his department; **Effectiveness** **Efficiency**	For all assets, scenes and threats; **Freedom of risk**
ICT (システム)	**ふるまい的**	When he searches and buys items using a browser; **Usability**	When he uses XX terminal; **Usability**			

ステークホルダ / 対象		他のステークホルダ			
		製品設計者	*テスタ*	*製造関係者*	*配送担当者*
ICTシステム（人とシステム）				On timeliness and accuracy of goods orders to them; **Effectiveness** **Efficiency**	On timeliness and accuracy of goods orders to them; **Effectiveness** **Efficiency**
ICT（システム）	**ふるまい的**		When preparing test environment; **Analyzability** **Testability**		
	内部的	To meet the business requirements on the product lifecycle; **Reusability** **Modularity**			

ISO/IEC 25030R-CD2

品質要求の展開

- 利用時品質は，製品品質・データ品質へ展開する.
- 製品品質は，サブシステム，ソフトウェア，コンポーネントへと展開する.

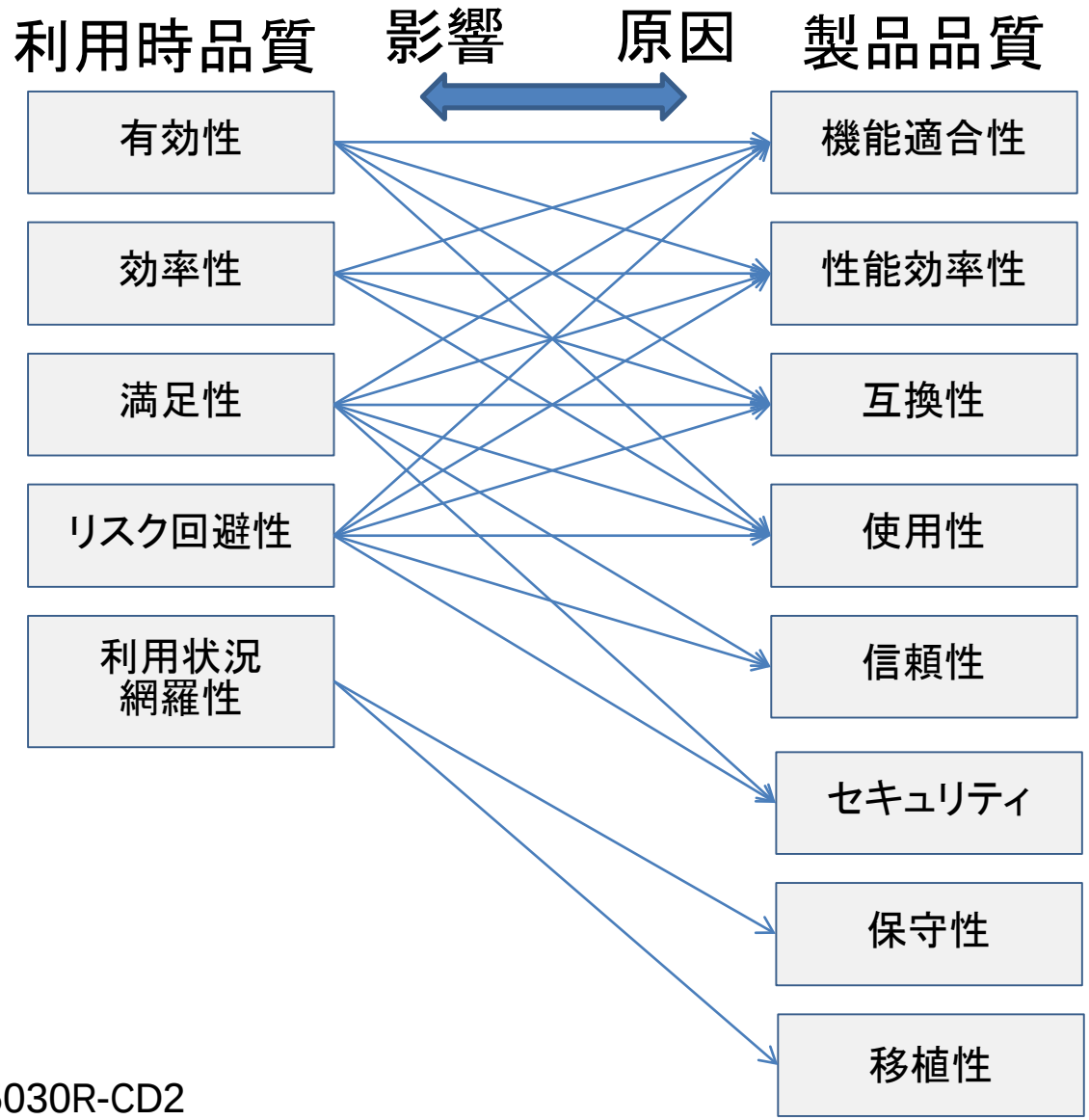


QIUR: Quality in use Requirement
PQR: Product Quality Requirement
DQR: Data Quality Requirement

導出される
二次的入力として要求を与える(例： ガイドライン)
制約として要求を与える (ICT 要求)

ISO/IEC 25030R-CD2

利用時品質と製品品質



ISO/IEC 25030R-CD2

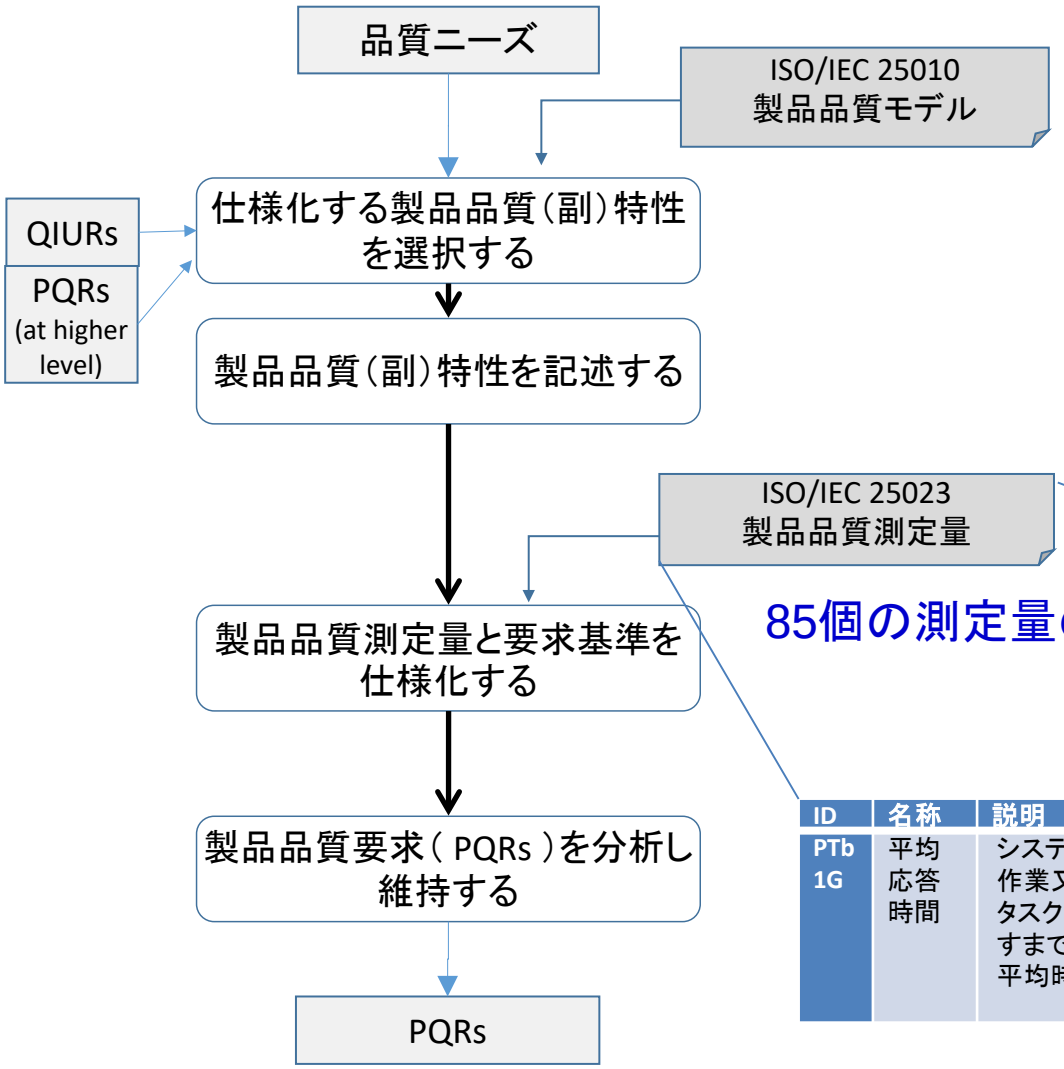
品質要求間の相互影響

影響先 / 影響元	機能適合性	信頼性	性能効率性	使用性	セキュリティ	互換性	保守性	移植性
機能適合性		−	−	−	−	−	−	−
信頼性		+		+	+		+	
性能効率性		−	−	+/−				
使用性			−	+	−	−	+/−	
セキュリティ	−	−	−	−	+	−	−	−
互換性	+		−		−	+	+/−	+
保守性	+	+/−	−		−	+	+	+
移植性			−			+	+	+

+ : 正の影響-: 負の影響 (干渉の可能性あり)

ISO/IEC 25030R-CD2

品質要求の記述の流れ



品質要求:

- 対象エンティティ
- 選択品質特性
- 記述
- 品質測定量
- 目標値と許容範囲

85個の測定量の定義

ID	名称	説明	測定関数
PTb 1G	平均応答時間	システムが利用者の作業又はシステムのタスクへの応答を返すまでにどのくらいの平均時間を要するか。	X = ∑(Ai)/ n i=1 ～n Ai = i番目の測定時において，システムが特定の利用者の作業又はシステムのタスクに応答を返すのに要する時間 n = 測定された応答の回数

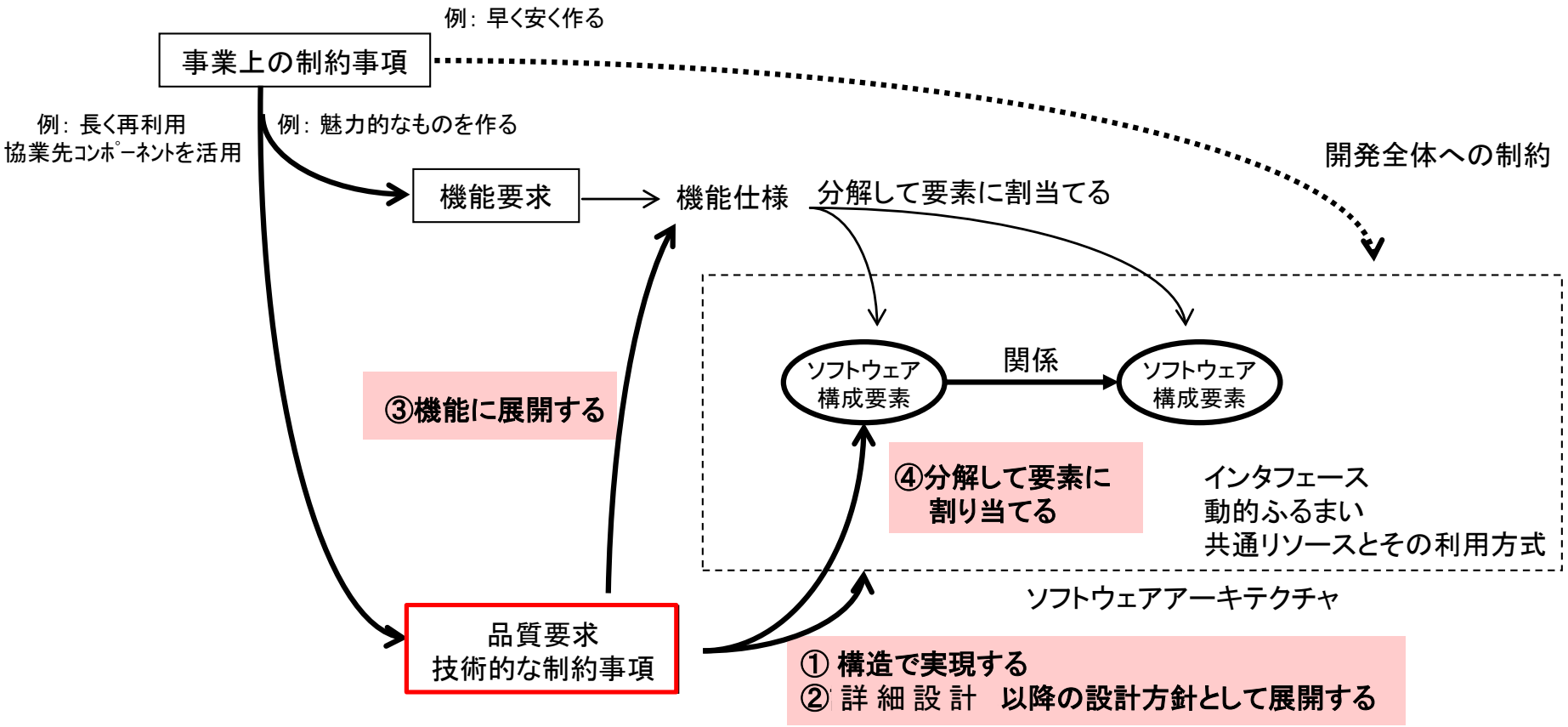
品質要求の定義と実現

- SQuaREシリーズとIoT
- 品質要求の実現

アーキテクチャを決定する要求

- 事業上の制約事項（製品戦略、ビジネスドライバ）
  - 製品の売りとすることは何か
  - 製品展開はどうするのか（シリーズ開発戦略）
- 機能要求（主要なユースケース）
- 技術的制約事項（テクノロジの利用　例：FW，購入品）
- 品質要求

品質要求の4つの展開



品質要求は，ソフトウェアアーキテクチャを決める主役

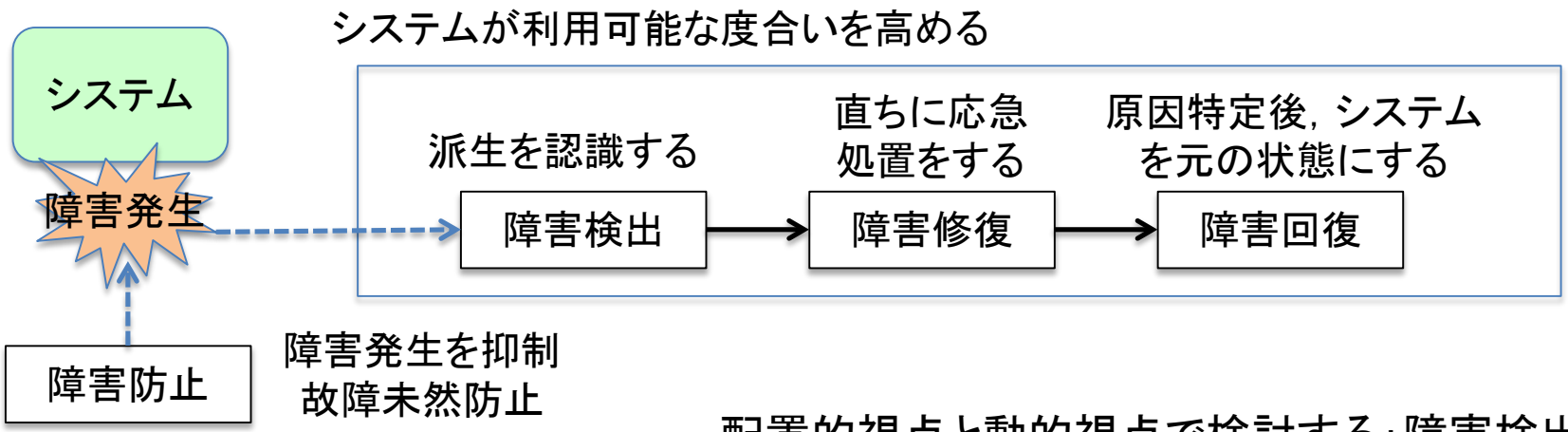
品質要求実現のための手順

1. 要求品質の識別と優先順位の認識
2. アーキテクチャの構築・評価（繰り返し）
   a. 実現戦略を立案：アーキテクチャ実現戦略
      Len Bass、Rick Kazman: 実践ソフトウェアアーキテクチャ,日刊工業新聞社(2005)
   b. 実現戦略を実装するアーキテクチャを選択/作成
   c. 評価する品質特性に合った「視点」での評価・バランス

IoTシステムでも使える！

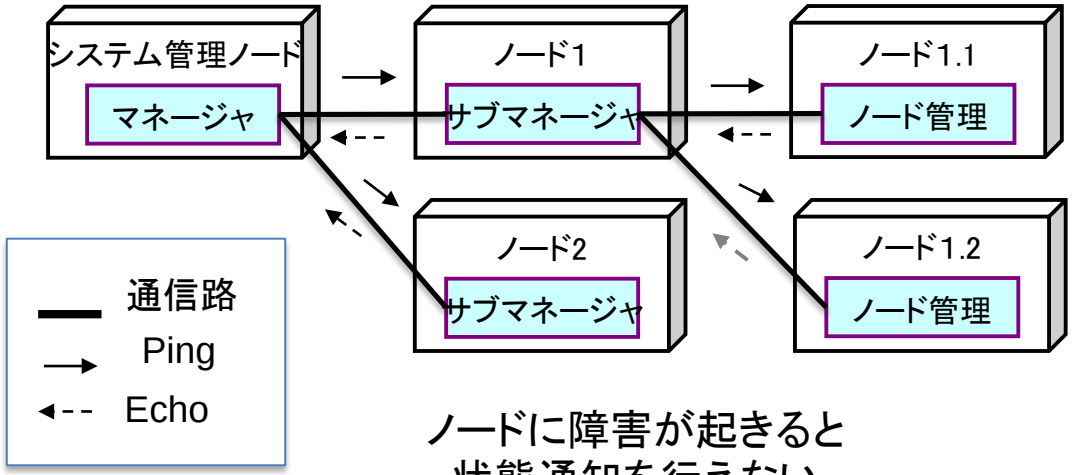
アーキテクチャの実現戦略：可用性

可用性：使用することを要求されたとき，運用操作可能およびアクセス可能な度合い



戦略	実現方式
障害検出	Ping&Echo
	ハートビート
	例外
障害修復	投票（Voting）
	アクティブ冗長性(Hot restart)
	パッシブ冗長性(Warm restart)
	予備(Spare)
障害回復	シャドウオペレーション
	状態同期化
	チェックポイント/ロールバック
障害防止	サービスからの除去
	トランザクション
	プロセス監視

配置的視点と動的視点で検討する：障害検出



ノードに障害が起きると
状態通知を行えない

アーキテクチャ実現戦略：修正性

修正性：欠陥の取り込み又は既存の製品品質を低下させずに，有効的にかつ効率的に修正することができる度合い

設計時
変更の影響を最小にする
→ 変更箇所の局所化

修正・変更時
他機能への波及を防ぐ
→ 連鎖反応の防止

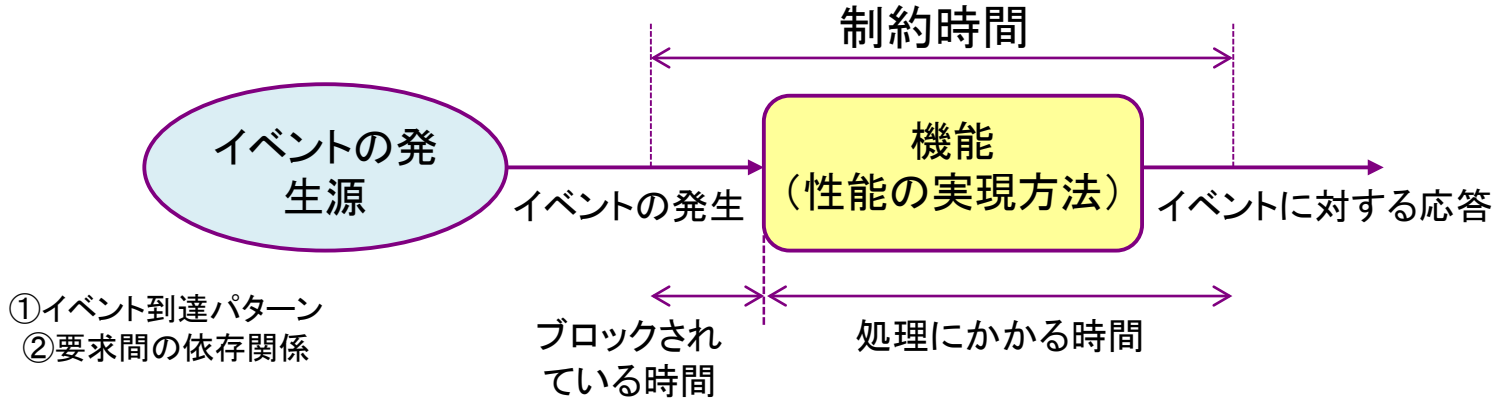
実行時
振る舞いの変更を容易にする
→ バインディング時期の遅延

戦略	実現方式	クラス図	コンポーネント図	パッケージ図
変更箇所の局所化	凝集度の向上	○	○	
	起こり得る変更の推測	○		
	モジュールの汎用化			○
	選択肢の制限	○		
	共通サービスの抽出			○
連鎖反応の防止	情報隠蔽	○	○	
	既存インタフェースの維持	○	○	
	通信経路の制限	○	○	
	仲介の利用		○	
バインディング時期の遅延	実行時のバインディング	○		
	構成ファイル			○
	ポリモフィズム	○		
	コンポーネントの交換		○	
	定義プロトコルを遵守		○	

静的視点で検討する

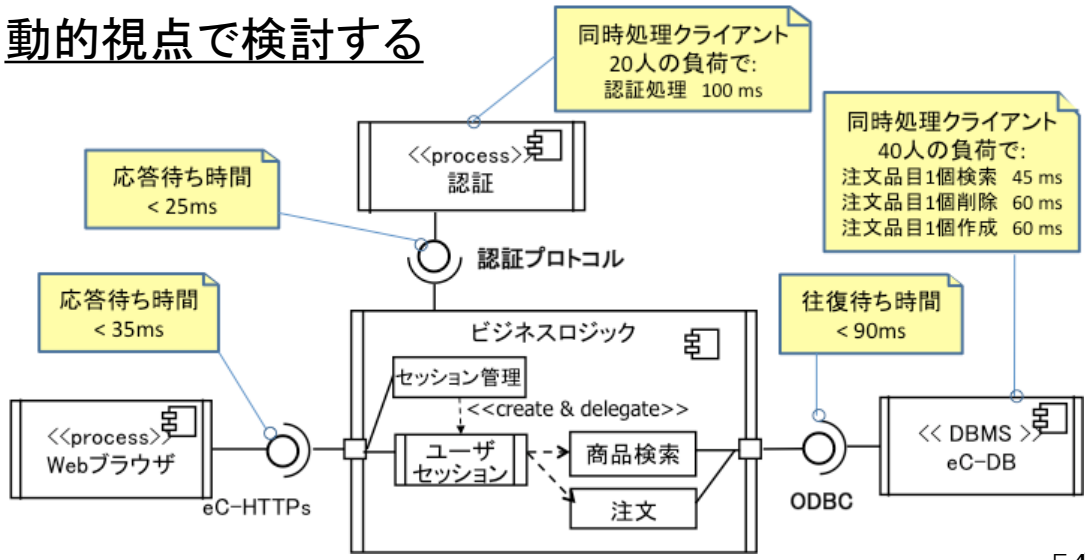
アーキテクチャ実現戦略：時間効率性

時間効率性：システムの機能を実行するとき、システムの応答時間及び処理時間、並びにスループット速度が要求事項を満足する度合



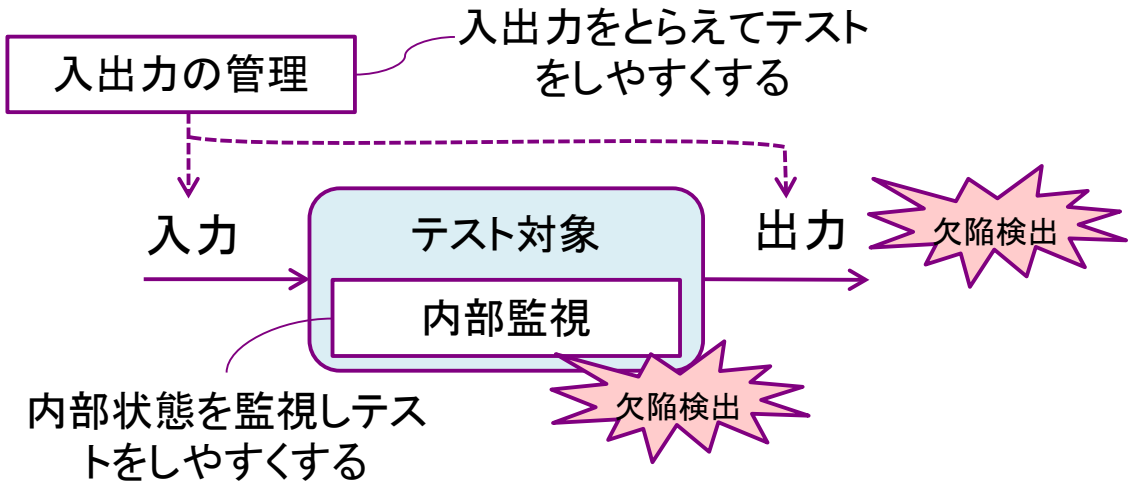
戦略	実現方式
リソース消費の低減	演算処理効率の向上
	無駄なリソース消費の削減
	イベント発生率の管理
	イベントのサンプリング頻度を抑制
	リソース使用の抑制
リソース管理	並列性の導入
	複数の複製を確保
	使用可能資源の増強
リソースの調停	スケジューリング方針

動的視点で検討する



アーキテクチャ実現戦略：試験性

試験性：ソフトウェアの追加や変更を実施した際に、コストを抑えてテストできるようにすること



戦略	実現方式
入出力の管理	記録と再生
	実装とインタフェースの分離
	特殊なアクセス経路/インタフェース
内部監視	組み込まれた監視

動的視点で検討する

コンポーネント図：インタフェース
パッケージ図：　　影響範囲

品質特性の検討に必要な視点と図法

- 品質特性を検討する上で，重要な視点と図法は異なる.
  → システムにより重要な品質特性が異なる
    → 方式設計で求められる記述が異なる

視点	図法	品質特性						
		セキュリティ	時間効率性	可用性	適応性	試験性	解析性	再利用性
コンテクスト	コンテクスト図	△			△	△	△	
機能	クラス図/データフロー図	△	△		○	△		○
配置的	配置図	○	○	○			○	
静的	レイヤ・パッケージ図	△			○	○	△	○
	コンポーネント図	△	△		○	○	○	○
動的	コンポーネント図（プロセス図）		○	△	△	○	○	

本日の話題

- Internet of Things (IoT) とその特徴
- 具体例でIoTの課題を考える
- IoT と Software Engineering
- 将来動向

将来動向

- ミドルウェア・フレームワーク・標準化
- 大規模なIoTものづくりに必要な道具立て
- 異分野IoT人材の育成

ミドルウェア・フレームワーク・標準化

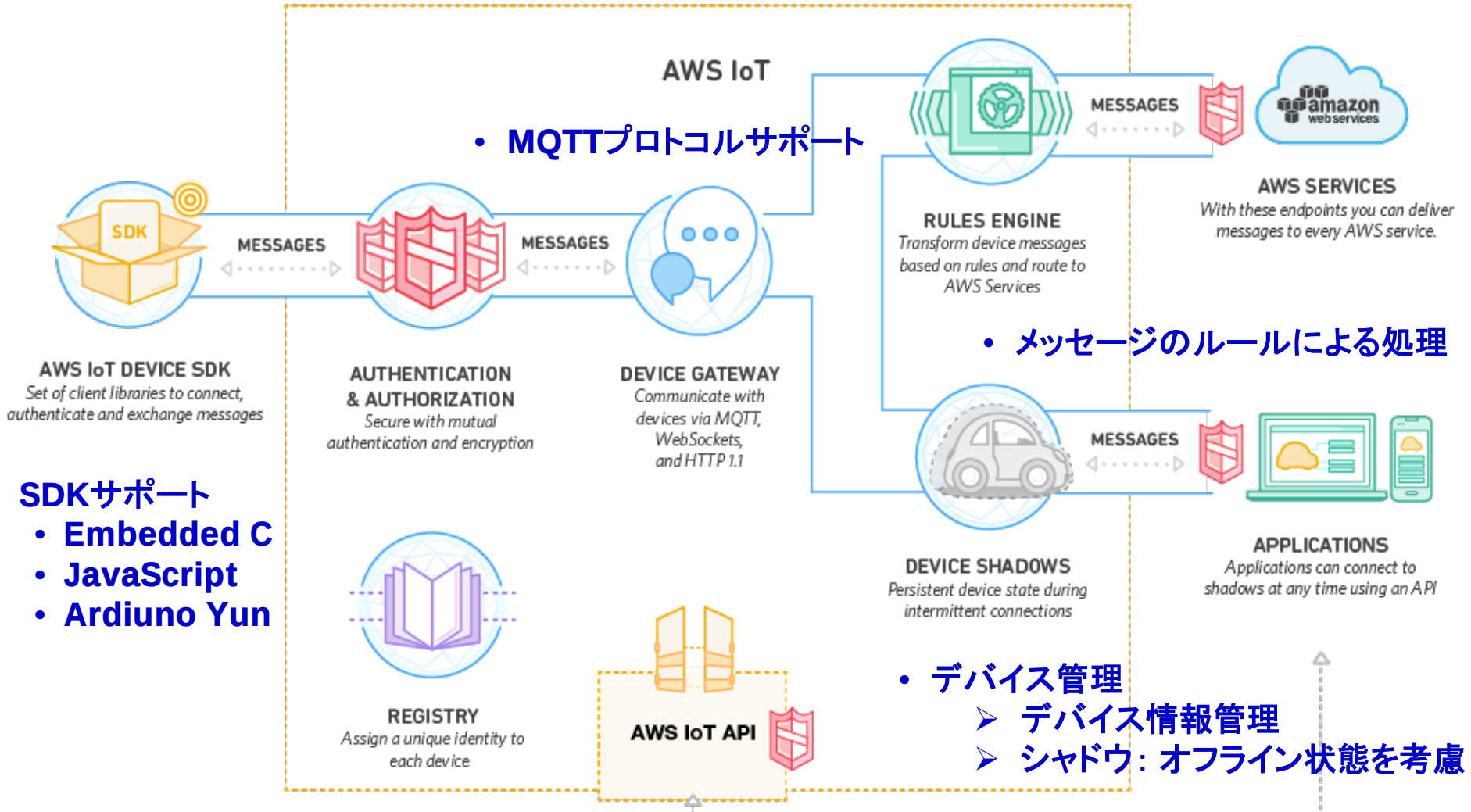
- クラウドサイドのフレームワーク
- センササイドのモジュール化・標準化

⇒

- IoTインフラ

AWS IoT：クラウドサイドのミドルウェア

クラウド(AWS)における簡単で安全なクラウドへのデバイス接続

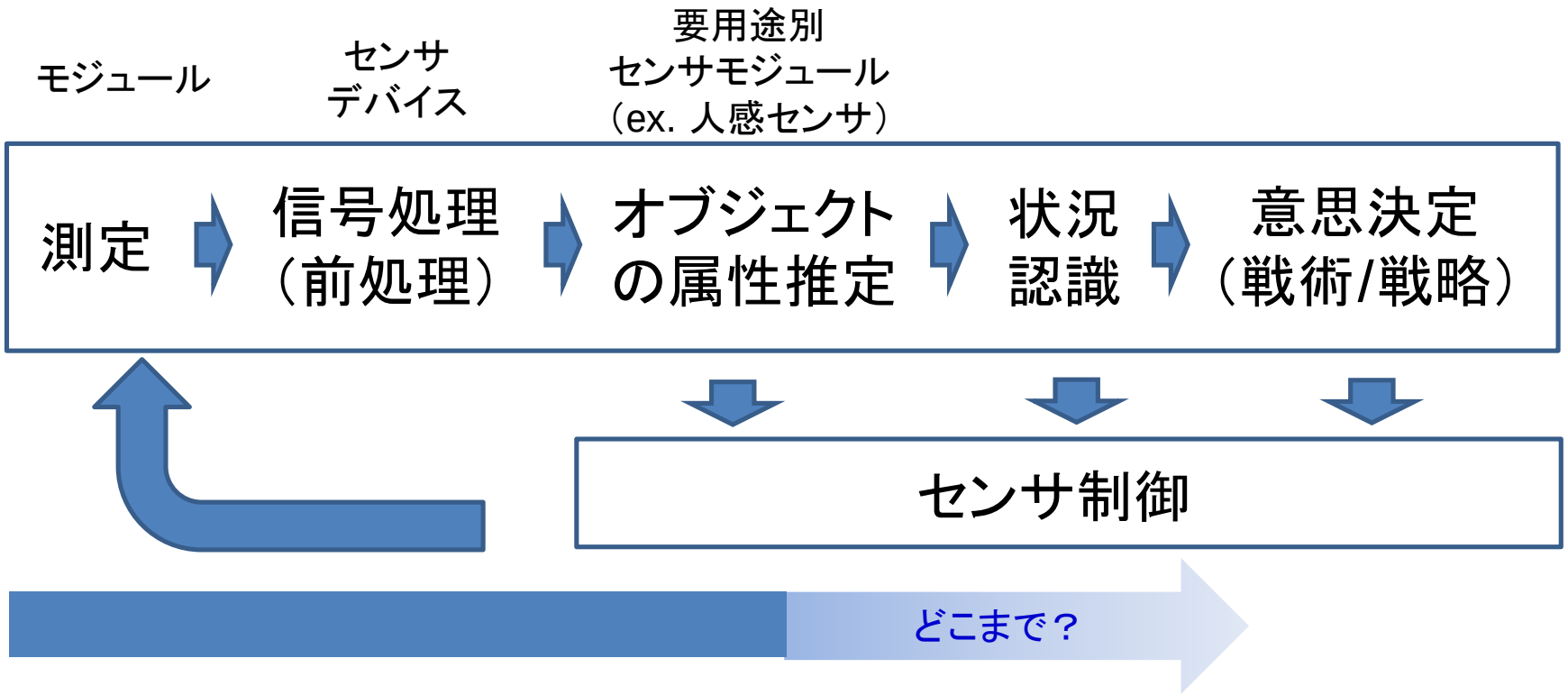


https://aws.amazon.com/jp/iot-platform/how-it-works/#gateway

センササイドからのモジュール化・標準化

- 標準化
- センスパイヤ：次世代センサ協議会
  - センサノードを賢く（エッジコンピューティング）

データ融合モデルを意識するとよい



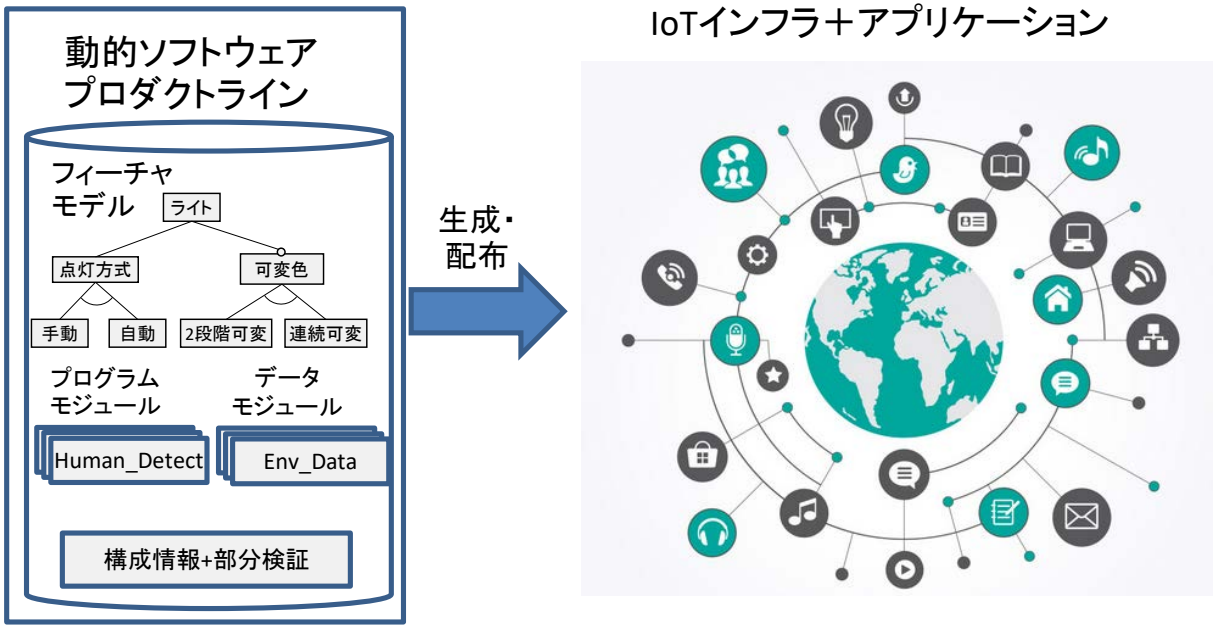
IoTインフラ に求められること

- Everything as a service（XaaS)モデルへ
  - 多数のものがつながる ⇒ サービスの集合として成長
  - RESTモデル： 4つの原則
    - やり取りされる情報は完結して解釈可能
    - 情報を操作する命令を予め定義・共有
    - 情報は汎用的な構文で一意に識別
    - 情報の内部に別の情報へのリンクを付帯可能
- Sustainability
  - 耐故障性
    - 状況認識の正しさの担保
    - 運用の継続性
  - スケーラビリティ（適応性）
  - セキュリティ
  - 保守： システム管理

大規模なIoTものづくりに必要な道具立て

- 組立・検証の技術：PF，通信，MW，コンポーネント
  - 構成管理
  - 組合せに関する知識 （Webより良質な情報を抽出）
  - 部分検証の技術（ネットワークの診断に似た）

⇒ 動的プロダクトライン

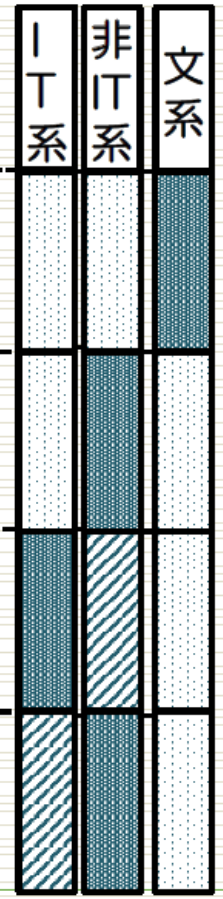


異分野IoT人材の育成

- IoTはアイデア勝負： 現場にこそ答えがある（Field-based Approach)
- 異分野のIoT人材が必要，でも構築には広範囲な知識を必要とする.
- 人材分野ごとに狙いとすること，必要な知識・技術が異なる

M2M/IoTシステム構築技術

技術区分	説明
M2M/IoT 応用技術１	文系のM2M/IoTアプリケーション仕様　社会・教育・経営など
M2M/IoT 応用技術2	理工系のM2M/IoTアプリケーション仕様　電気・機械・農業・制御
M2M/IoT システム 技術	クラウド／サーバ、ゲートウェイ、マイコン、OS、DB、ネットワーク、プログラミング
M2M/IoT デバイス 技術	センサー、アクチュエータ、制御・組合せ技術



構築内容例

・**IT系**：プログラミングを中心に構築し、通信ﾌﾟﾛﾄｺﾙ、ﾌｨｰﾄﾞﾊﾞｯｸ制御を実現する内容

・**非IT系**：温湿度，ｶﾗｰｾﾝｻｰとﾌﾙｶﾗｰ**LED**を**IoT**デバイスとして監視・制御する内容

・**文系**：温度センサー値のグラフ化と**LED**ランプ点灯によるクラウド含めて**IoT**を体験する内容

I/F　機能　構造

ご清聴ありがとうございました