ディペンダビリティ工学講座

土屋研究室

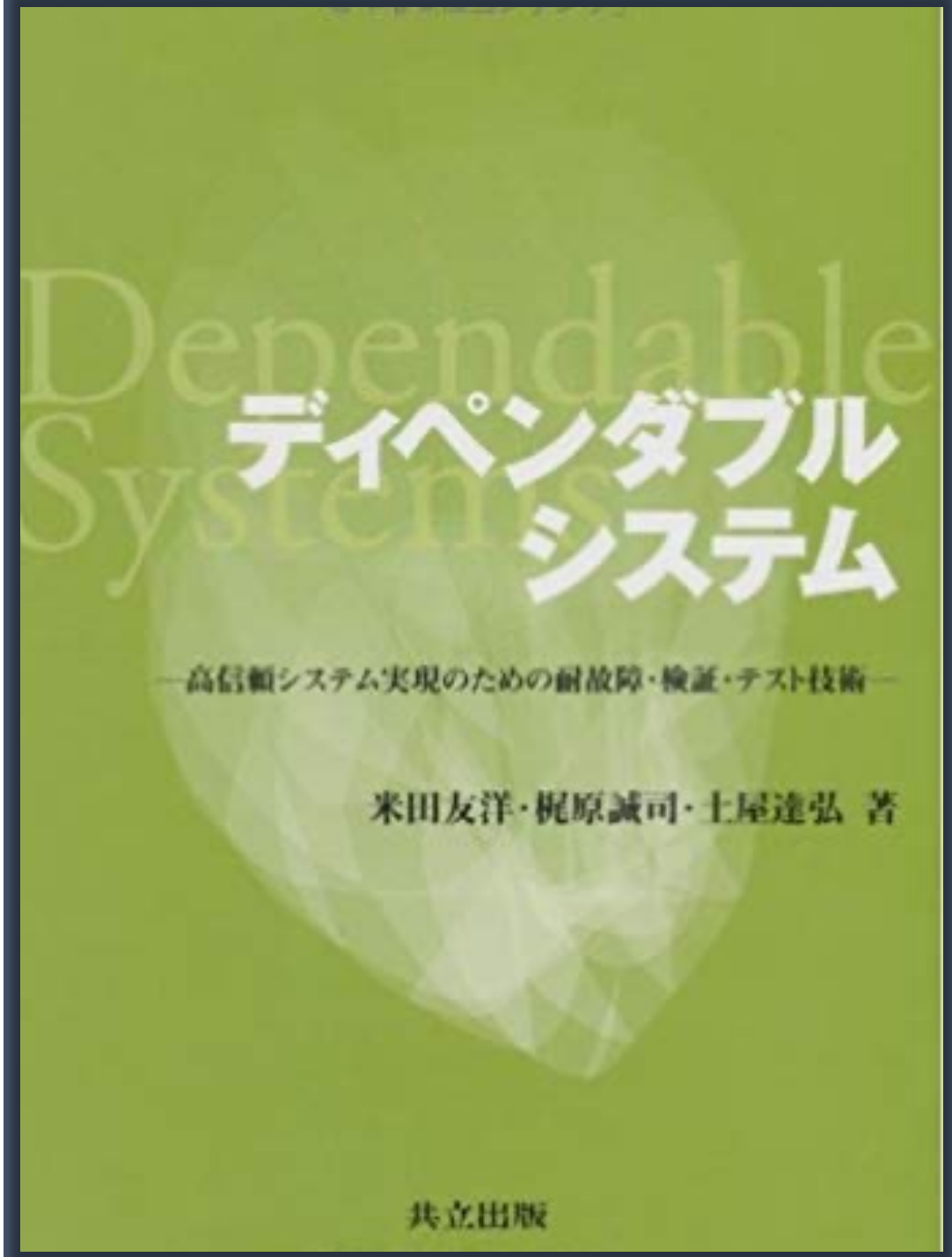
D3: 2名
D1: 1名
M2: 3名
M1: 5名

留学生5名
(中国4名，ギニア1名)

小島 英春

中川 博之

土屋 達弘



ディペンダビリティ工学講座

土屋研究室

ディペンダブル
＝ depend+able

信頼　できる

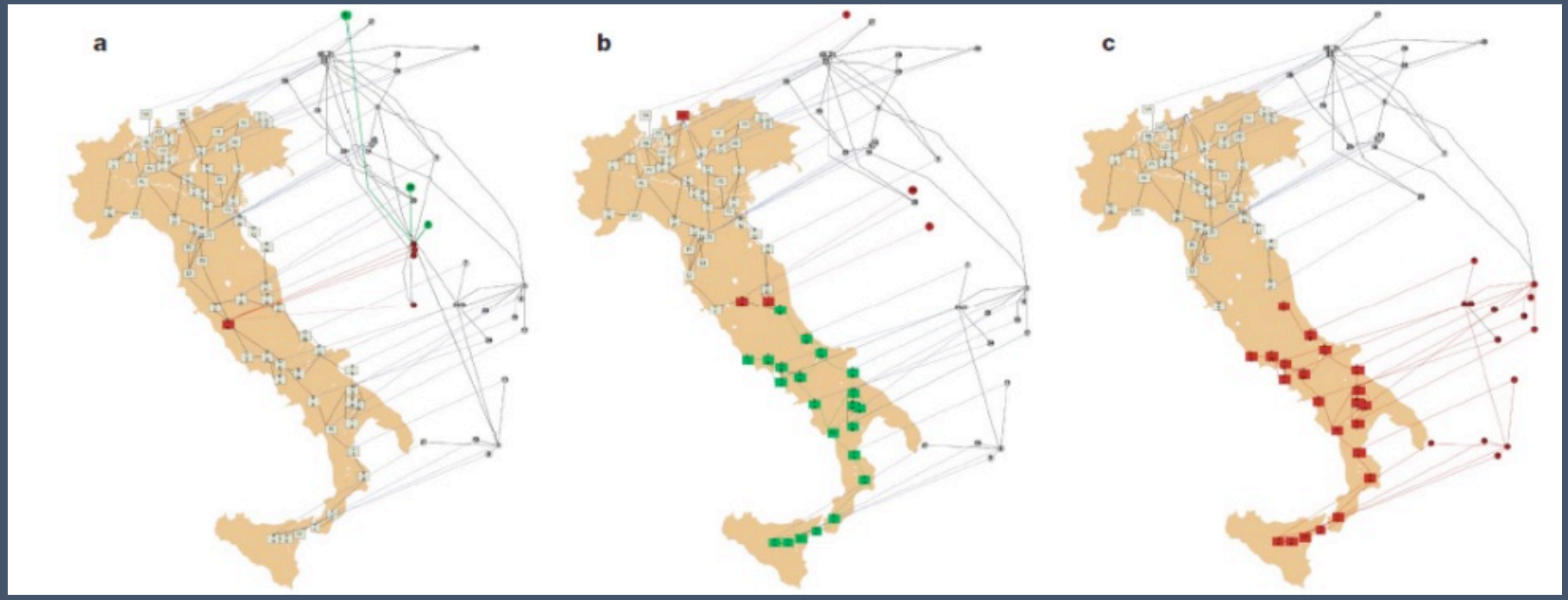
ディペンダビリティ工学講座
土屋研究室

OS	Browser	Protocol	Result	Result
Home	Chrome	IPv4	Pass	Pass
Home	Edge	IPv6	Pass	Fail
Pro	Edge	IPv4	Pass	Pass
Pro	Chrome	IPv6	Pass	Pass
Ent	Chrome	IPv6	Fail	Pass
Ent	Edge	IPv4	Pass	Pass
Ent	Chrome	IPv4	Fail	Pass
Home	Edge	IPv4	Pass	Pass
Pro	Edge	IPv6	Pass	Fail

組み合わせテスト (Combinatorial testing)

ツール CIT-BACH：「 CIT-BACH 」で検索

ディペンダビリティ工学講座
土屋研究室

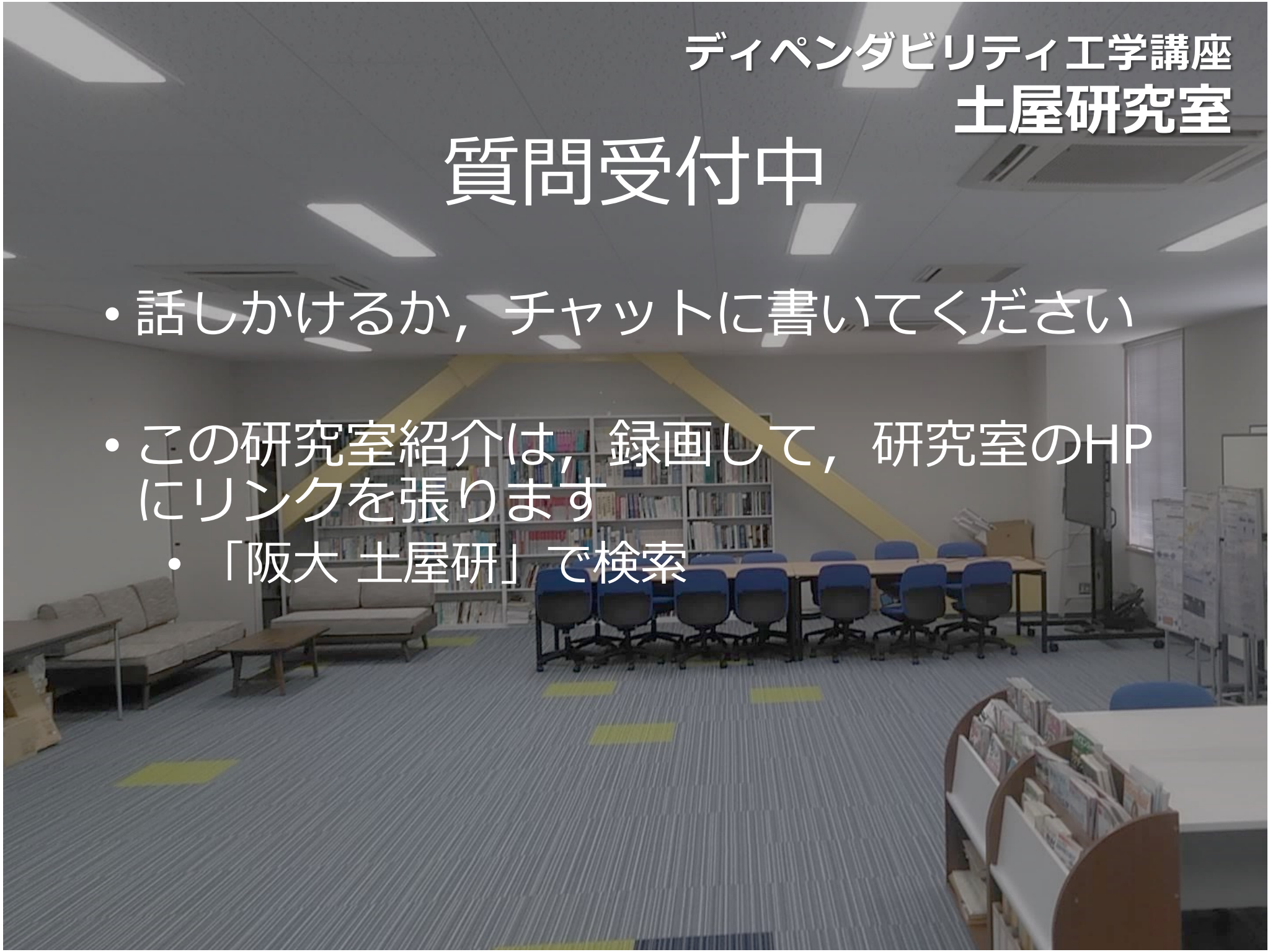


Nature, Vol.464, pp.1025-1028, 2010

複雑ネットワークの脆弱性解析

最後に

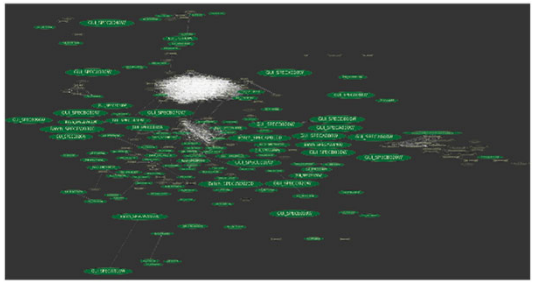
- 大学院までを見据えて研究に取り込んでほしい
- 修士を修了するまでに，1度は国際会議で論文発表してほしい
- 束縛が少ない分，自発的に研究に取り組んでほしい

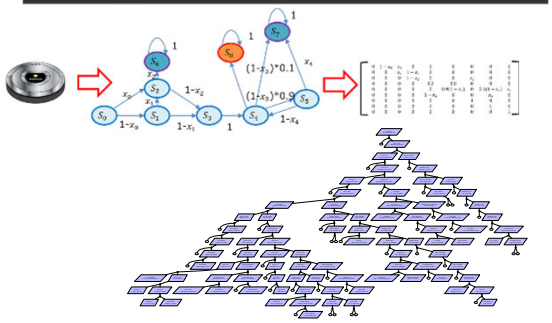
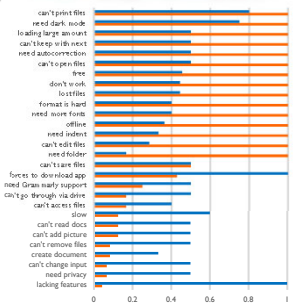

ディペンダビリティ工学講座
土屋研究室
質問受付中
• 話しかけるか，チャットに書いてください
• この研究室紹介は，録画して，研究室のHPにリンクを張ります
• 「阪大 土屋研」で検索

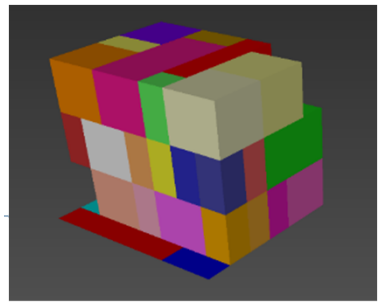
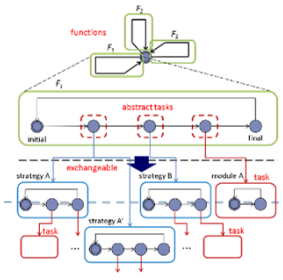
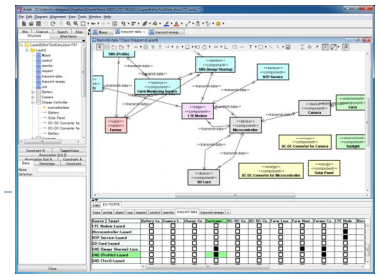
自己紹介： 中川博之

- ディペンダビリティ工学講座 (土屋研) 准教授
- 担当講義：計算論A, 情報論理学, 情報科学序説

- 研究内容：
  - 自己適応システム
  - 要求工学
  - 知的ソフトウェア
  - ソフトウェア開発の自動化
  - マルチエージェントシステム



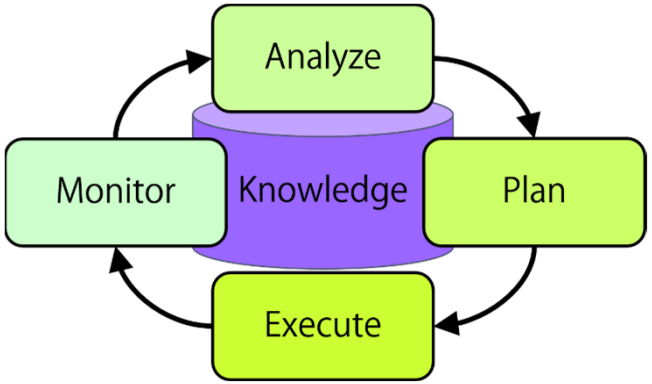
- I

研究内容紹介：自己適応システム

- 自己適応システム (self-adaptive systems):
  - 自発的に振舞いを替えることで
    環境に柔軟に適応できるシステム

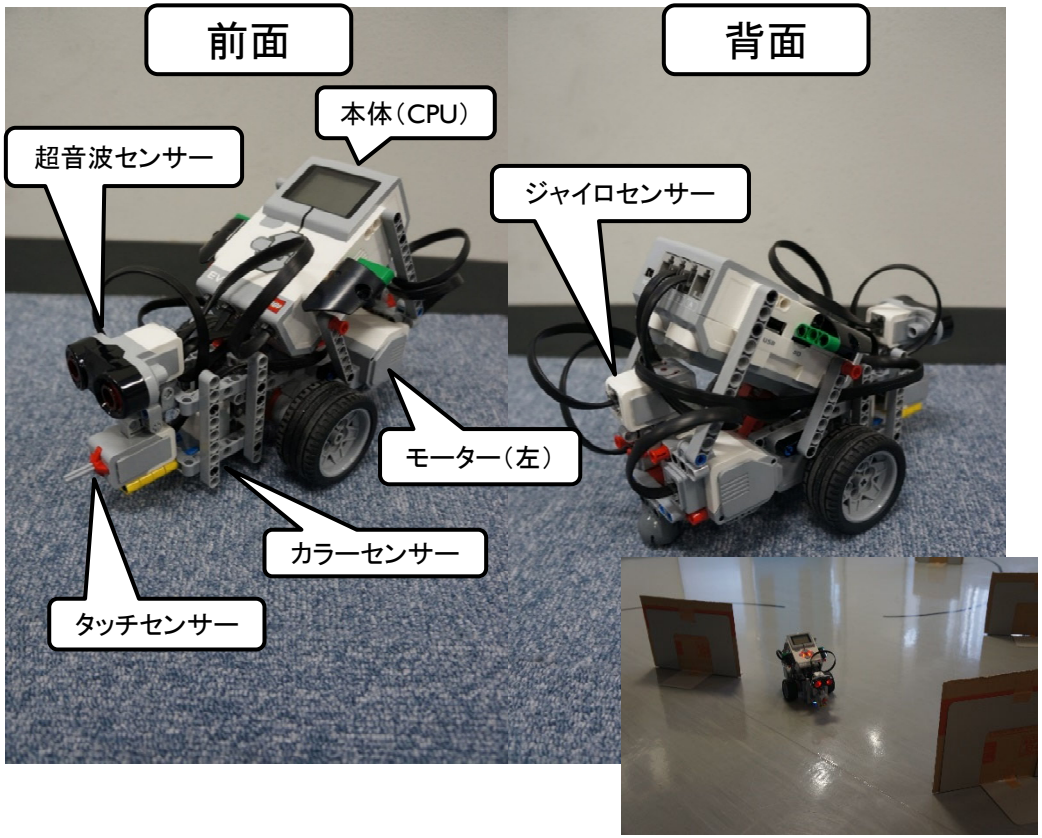
- 以下の能力が必要
  - 環境を観測する能力 (Monitor)
  - 現在の状況を分析する能力 (Analyze)
  - 望ましい振る舞いを決定する能力 (Plan)
  - 振舞いを切り替える能力 (Execute)

MAPE loop

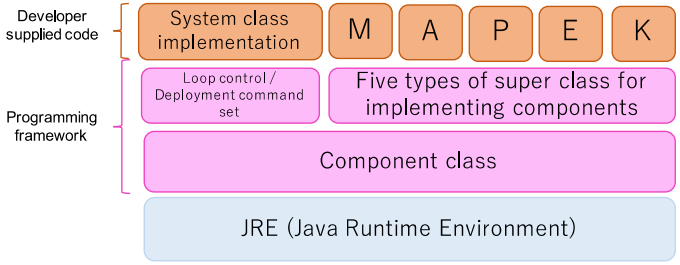


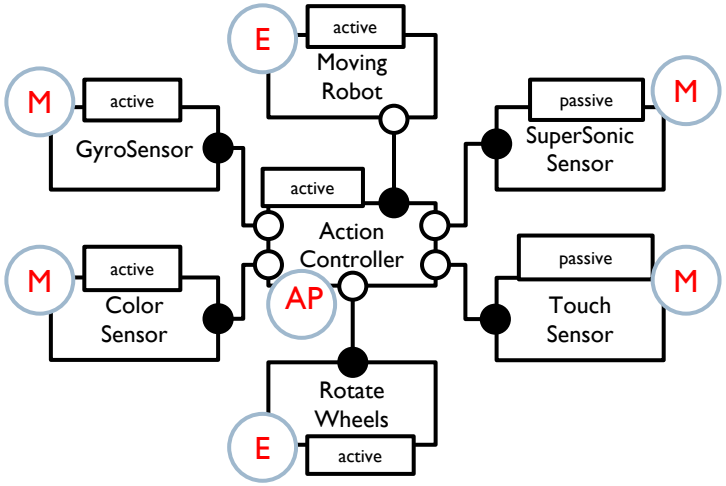
自己適応システムフレームワーク

- 実世界の環境変化に適応するロボット
  - プログラミングフレームワークを実装
  - LEGO Mindstorms上に実装




```
public class MonitorEvent extends Monitor{

  SerialCommunication serial = null;
  KnowledgeState knowledge = null;

  public MonitorEvent(SystemEventConverter se, String name) {
    super(se, name);
    this.serial = new SerialCommunication();  Set a serial port
    this.knowledge = (KnowledgeState) super.knowledge;
  }                                           Set a Knowledge component
  @Override
  public Object getEvent() {

    while(true) {

      // check whether button events happened
      int button_event = -1;
      if((button_event = this.serial.getButtonEvent()) != -1){
        switch(button_event) {
          case CLEAN: return "Clean";
          case SPOT: return "Spot";
          default: break;
        }
      }

      // check whether internal events happened
      if(knowledge.getEvent()!=null) {
        return this.knowledge.getEvent();
      }
    }                                         Monitor new events
  }
  [...]
}
```

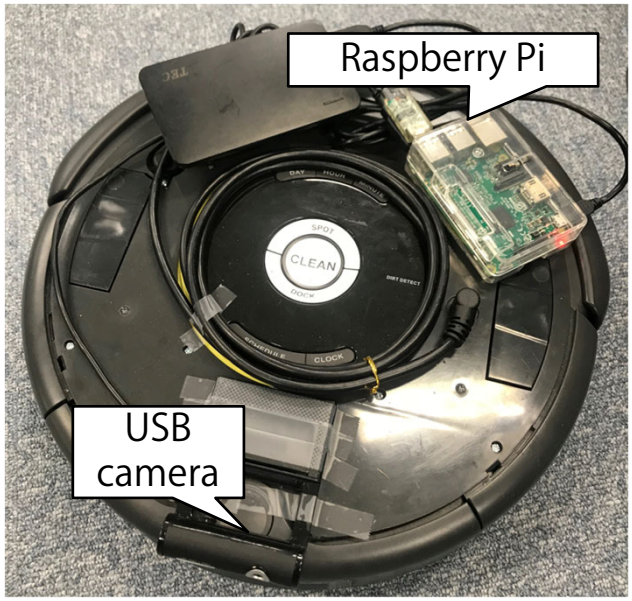


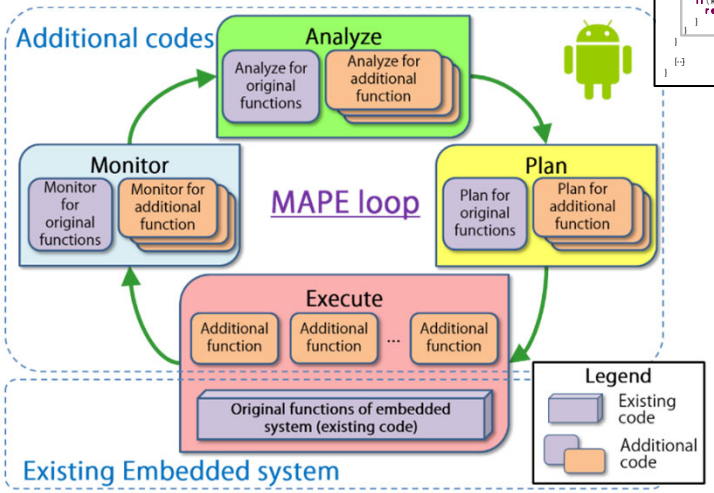


3 [Tsuda15] H. Tsuda, H. Nakagawa, T. Tsuchiya, “Towards Self-Adaptation on Real-World Hardware: a Preliminary Lightweight Programming Framework”, in SASO 2015, IEEE, 2015.

MAPEループの応用

- MAPEループを利用した組み込みシステムの機能拡張
  - 例題: 清掃ロボットの機能拡張
  - MAPEループの外付けによる機能の追加, 変更

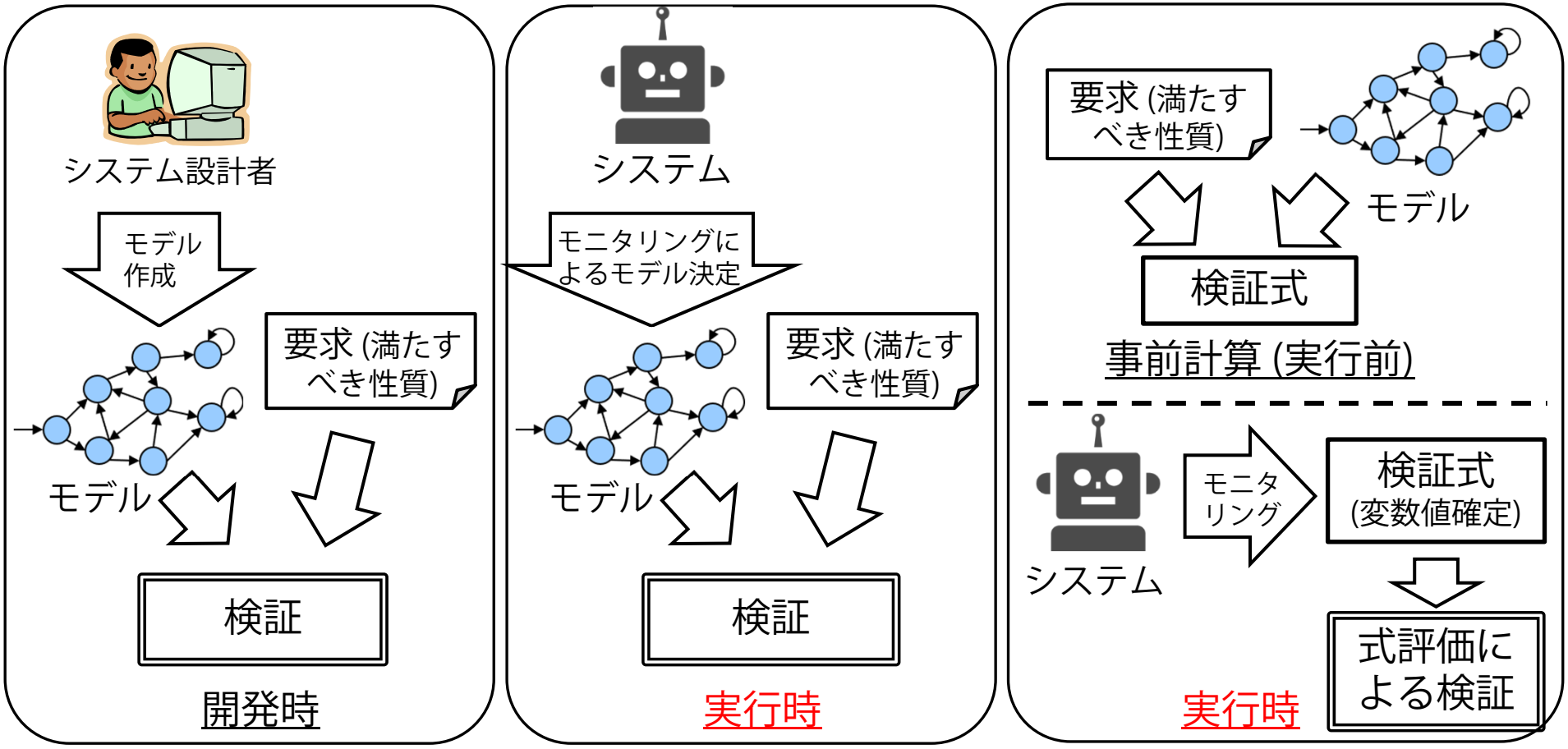





```
public class MonitorEvent extends Monitor{

  SerialCommunication serial = null;
  KnowledgeState knowledge = null;

  public MonitorEvent(SystemEventConverter se, String name) {
    super(se, name);
    this.serial = new SerialCommunication(); Set a serial port
    this.knowledge = (KnowledgeState) super.knowledge;
  }
                    Set a Knowledge component
  @Override
  public Object getEvent() {

    while(true) {

      // check whether button events happened
      int button_event = -1;
      if((button_event = this.serial.getButtonEvent()) != -1){
        switch(button_event) {
          case CLEAN: return "Clean";
          case SPOT: return "Spot";
          default: break;
        }
      }

      // check whether internal events happened
      if(knowledge.getEvent()!=null) {
        return this.knowledge.getEvent();
      }
    }
  }                 Monitor new events
  [-]
}
```

4 [Tsuchida 18] S. Tsuchida, H. Nakagawa, E. Tramontana, A. Fornaia, T. Tsuchiya, “A Framework for Updating Functionalities Based on the MAPE Loop Mechanism”, in COMPSAC 2018, IEEE, 2018.

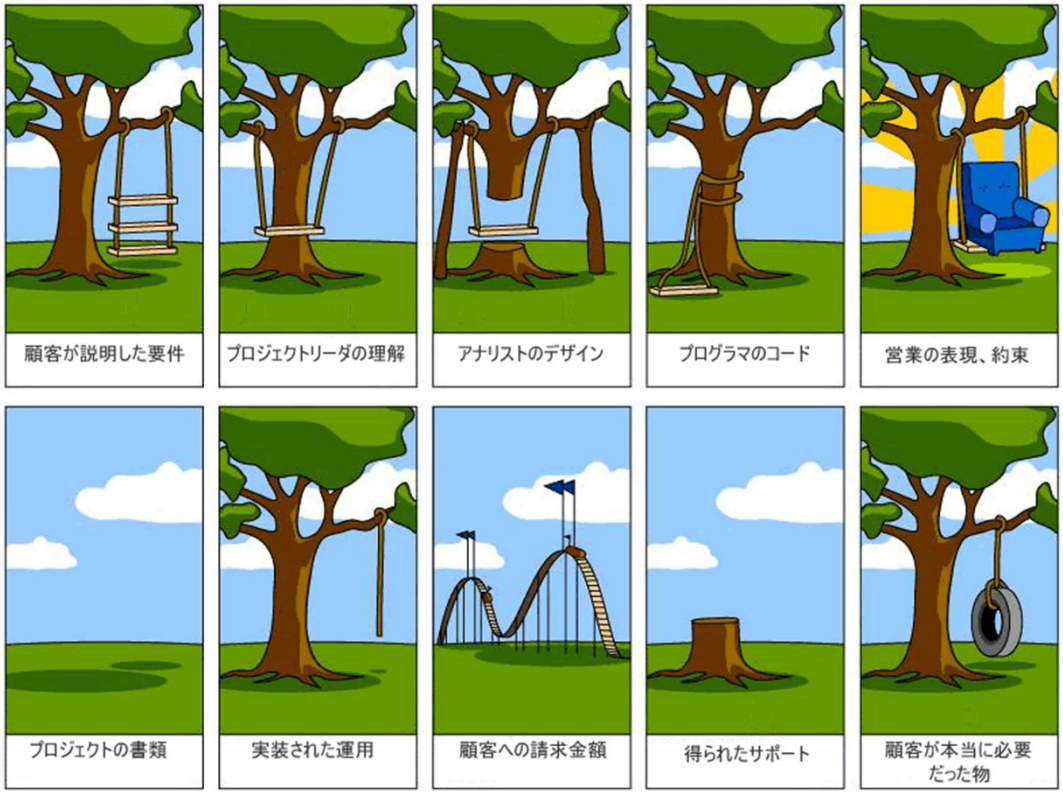
検証技術



自己適応システムの研究 ≒ ソフトウェア構築技術の自動化

研究内容紹介：要求工学

- 要求工学の重要性: StandishのCHAOSレポート
  - プロジェクト失敗の原因：
    - 2位「不完全な要求と仕様」，3位「要求と仕様の変更」



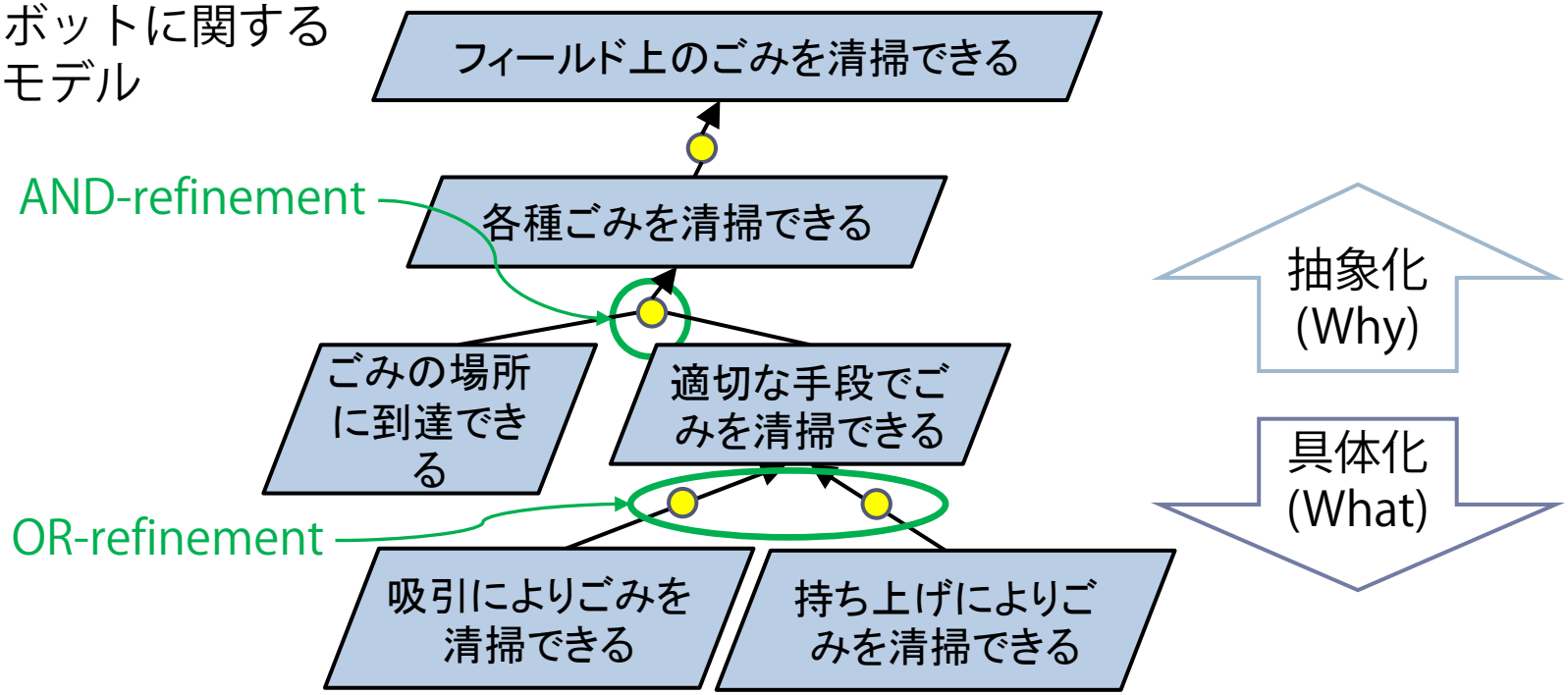
@ITより引用
原著：University of London Computer Center Newsletter, No.53, March 1973

[Standish] The Standish Group “The Standish Group Report - CHAOS”, 1995.

ゴールモデル

- ソフトウェアが実現すべき要求 (ゴール)を構造化し，目標から手段へと階層的に展開したモデル
  - 分析漏れや要求間の矛盾発見に効果的

例: 清掃ロボットに関するゴールモデル



ゴールモデル構築支援ツール

実装ツール

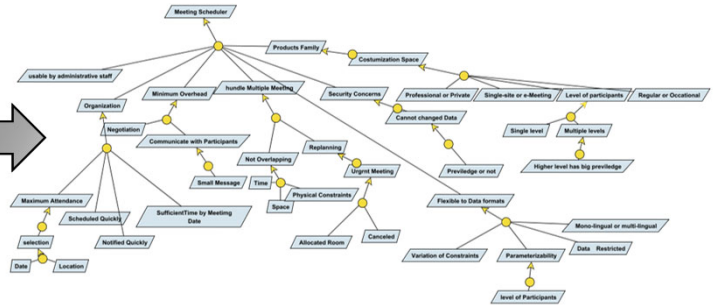
The ATM will communicate with the bank's computer over an appropriate communication link.
…

要求記述(入力)

1. 要求記述を文分割

2. 質問への回答に従いゴールを配置

3. ゴールモデルの画像を生成



ゴールモデル (出力)

回答

質問

要求分析者

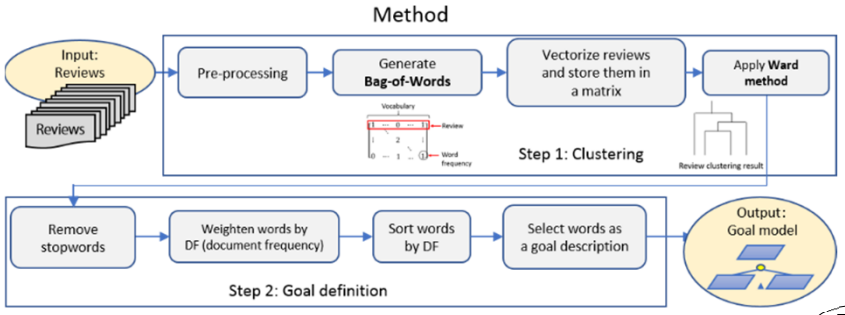
```
Q. What is the goal description of following sentence?

   If the sentence does not include requirements, press Enter key to skip this question.

   Requirement:
          The ATM will service one customer at a time.

   Description: service one customer
Q. Is the following goal: "service one customer"  the subgoal of the goal: "ATM System"?
   Input y/n.
   Description: service one customer

Selection: y
"service one customer" is subgoal of "ATM System"
The goal "service one customer" has brother "communication with bank"

Q. To satisfy the goal "service one customer", which goals should be satisfied?
     Which is apporopriate?
```

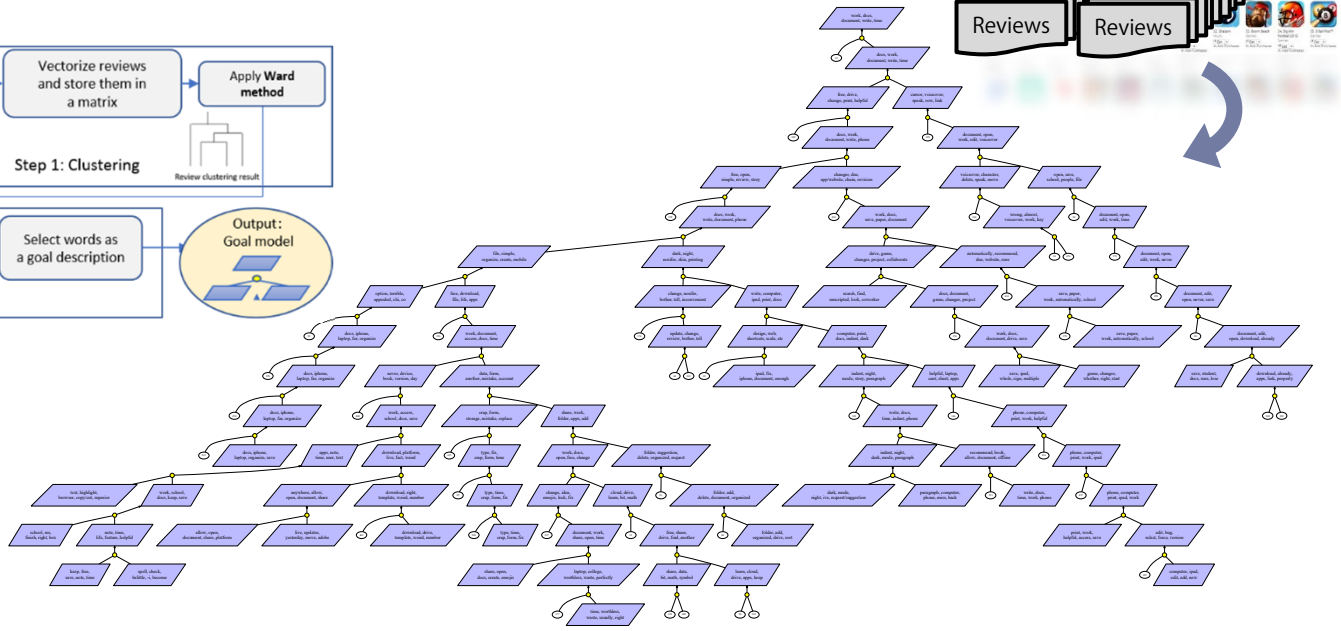
要求分析者はゴールモデルを意識せずに質問に答えるだけ！

H. Nakagawa, H. Shimada, T. Tsuchiya, “Interactive Goal Model Construction Based on a Flow of Questions”, IEICE Transactions on Information and Systems, Vol.E103-D, No.06, pp. 1309-1318, Jun. 2020 (2020. 6).

レビューからのゴールモデルの自動生成

- 膨大なアプリケーションレビューから要求を抽出
  - 類似レビューを分類し，ゴールモデルを構築
    - クラスタリングによりゴールを発見し構造化
    - 自然言語処理技術, Deep Learningを用いて
      ゴールラベリング

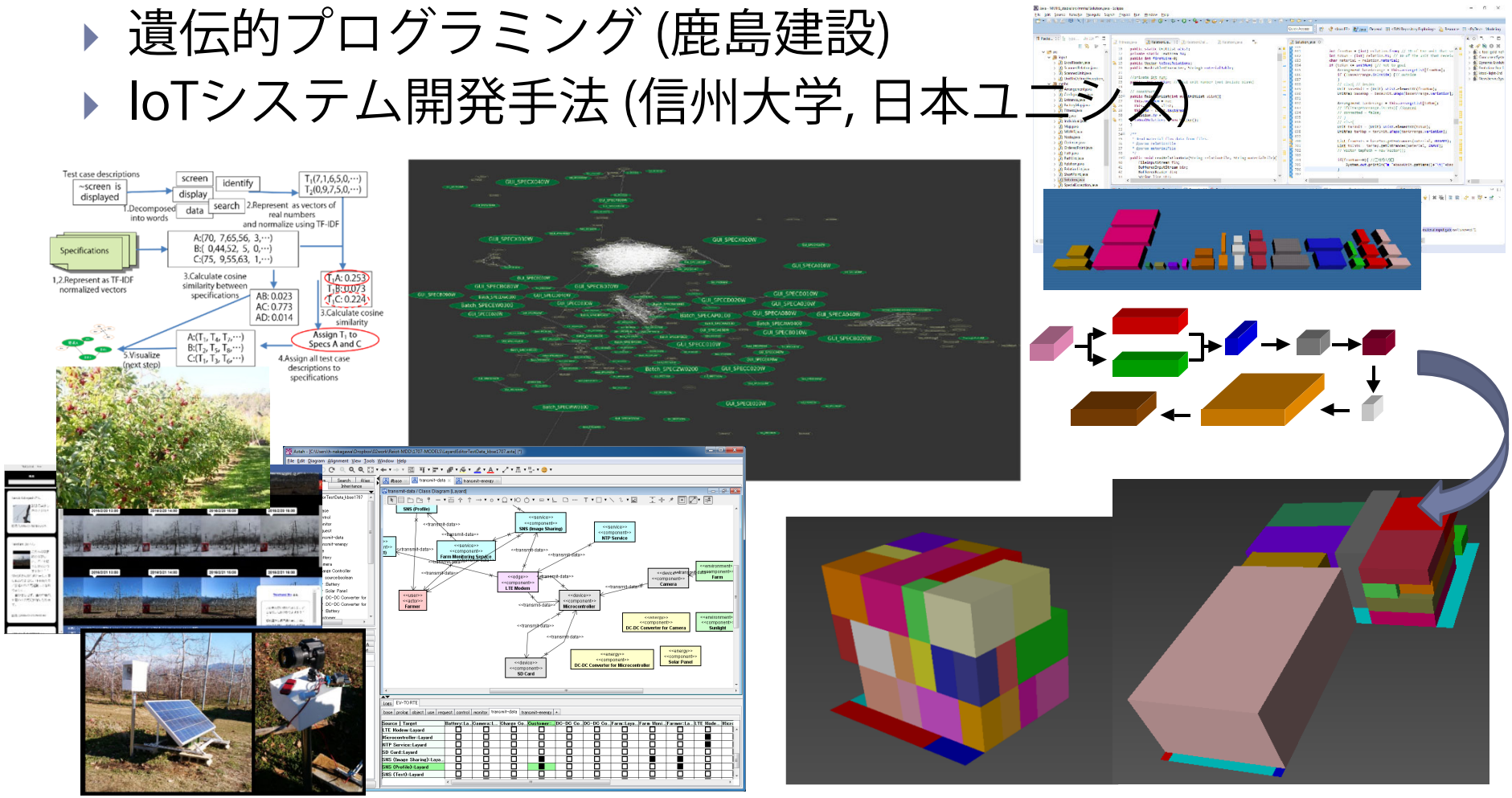




9 [Shimada19] H. Shimada, H. Nakagawa, T. Tsuchiya, “Goal Model Construction Based on User Review Classification”, in REFSQ 2019.
[Ren20] S. Ren, H. Nakagawa, T. Tsuchiya, “An Automated Goal Labeling Method Based on User Reviews”, in SEKE 2020.

研究内容紹介：その他 (共同研究など)

- 可視化ツール (SUNTORY関連会社)
- 遺伝的プログラミング (鹿島建設)
- IoTシステム開発手法 (信州大学, 日本ユニシス)

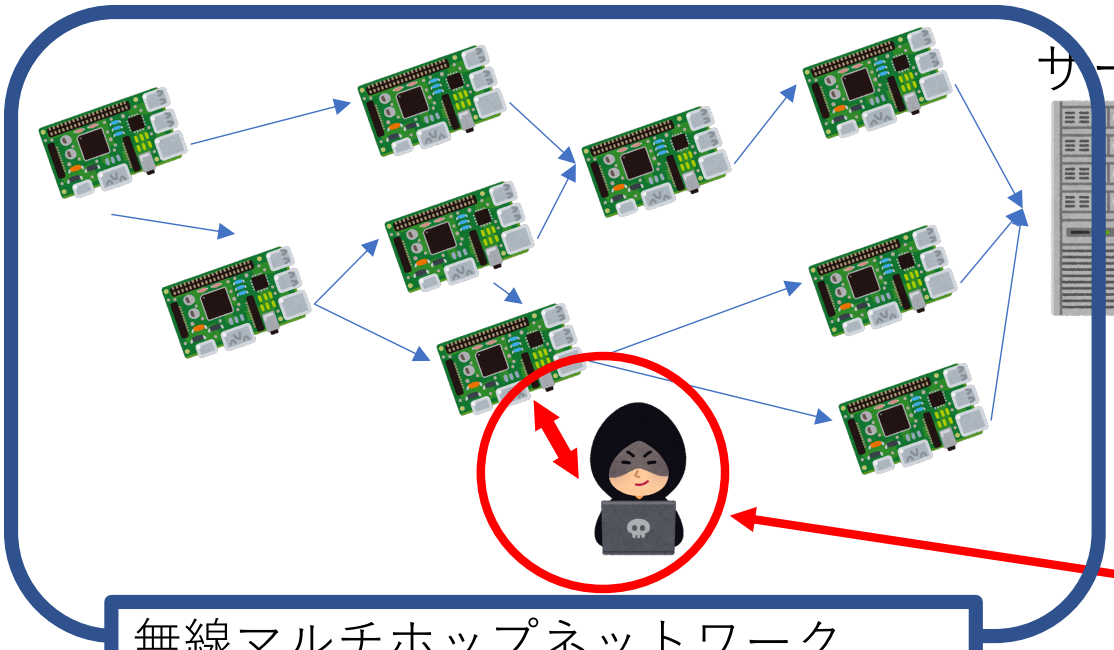


10 ソフトウェア開発, プログラミング, アルゴリズム設計に興味のある方を募集します！

研究テーマ

- 経路保証プロトコルの実装と評価
- 経路保証プロトコルの安全性検証
- Hyperledger Fabricのテストベッド構築
- アクセス制限をしたスマートコントラクト

経路保証プロトコルの実装



・役割
端末とサーバ間に
経路を作成する
・特徴
経路の正しさを保証する

攻撃者が嘘の経路情報を
ネットワークに送信したとしても
騙されずに正しく経路を構築する

無線マルチホップネットワーク
直接通信ができな場合は
バケツリレー方式で宛先まで届ける

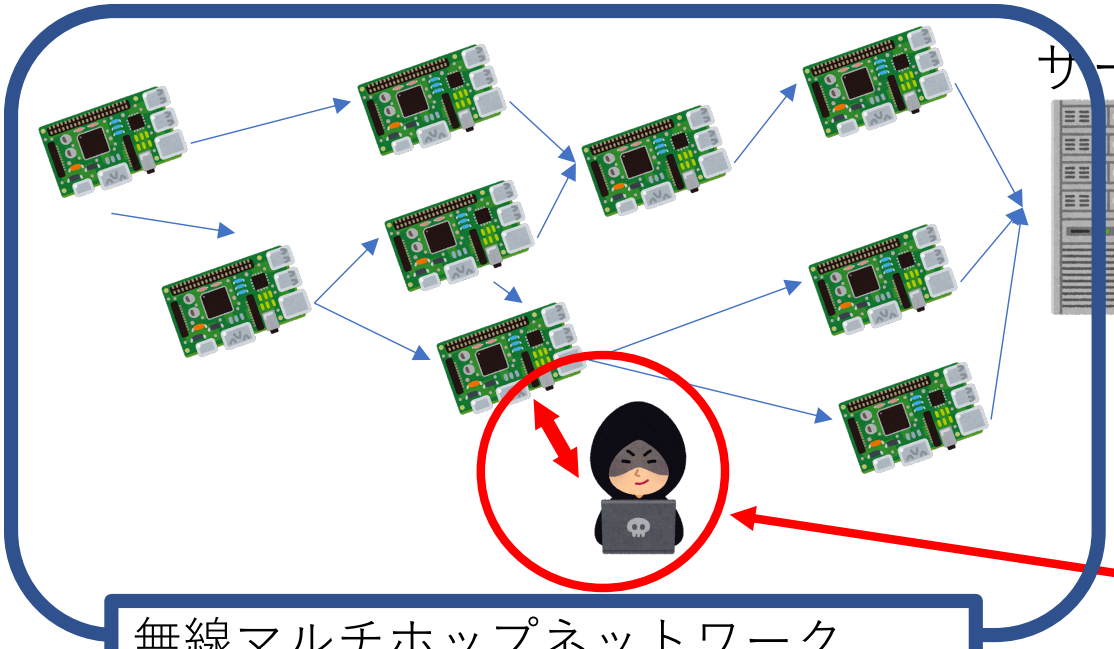
テーマ：経路保証プロトコルの実装と評価
複数のプロトコルを実装しそれらを評価する

実装言語：JAVAもしくはC++

内容

- 利用するツール Mininet-wifi
  - 無線環境をエミューレートするツール
  - Ubuntu上で動作させるのが便利
- 実装言語
  - Java または C++
- 対象プロトコル
  - ISDSR(済), SRDP(済), RSAbase(済),
  - Aran(済？), IDSAODV, Ariadne,　他
- 可能なら実機でも試したい

経路保証プロトコルの検証



本当に経路情報を
保証することができる？

攻撃者が嘘の経路情報を
ネットワークに送信したとしても
騙されずに正しく経路を構築する

無線マルチホップネットワーク
直接通信ができな場合は
バケツリレー方式で宛先まで届ける

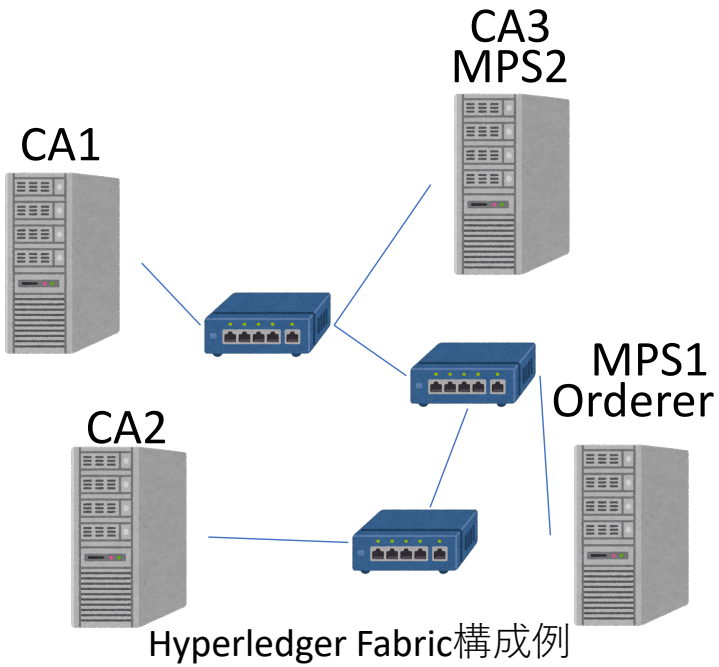
テーマ：経路保証プロトコルの安全性検証
複数のプロトコルを対象にモデルを作成して検証

内容

- 利用するツール はProverif
  - セキュアプロトコルの検証に利用される
  - WPAやTLSの安全性の検証に利用された例もある
  - 検証対象が無線マルチホップ通信プロトコルであるため工夫が必要

- 対象プロトコル
  - ISDSR, SRDP, RSAbase,
  - Aran, IDSAODV, Ariadne,　他

Hyperledger Fabricのテストベッド構築



Hyperledger Fabric構成例

Hyperledger Fabric：
Orderer：チェーンコードの実行順序などを管理する
MPS：チェーンコードの実行可能なユーザの登録などを管理
CA：ブロックの生成，チェーンへの追加，チェーンコードの実行について合意を取る

ネットワーク上にこれらの役割を持つノードが存在してスマートコントラクトを利用するシステムが構成される

問題点：
この様なシステムが正しく動作するかをテストするために，
実環境を用意することは非常に手間がかかる

テーマ：Hyperledger Fabricを対象にしたテストベッドの構築
エミュレータ上で様々なトポロジを試すことができる様にすることで
実際の環境を用意することなく様々なテスト環境を用意することができる

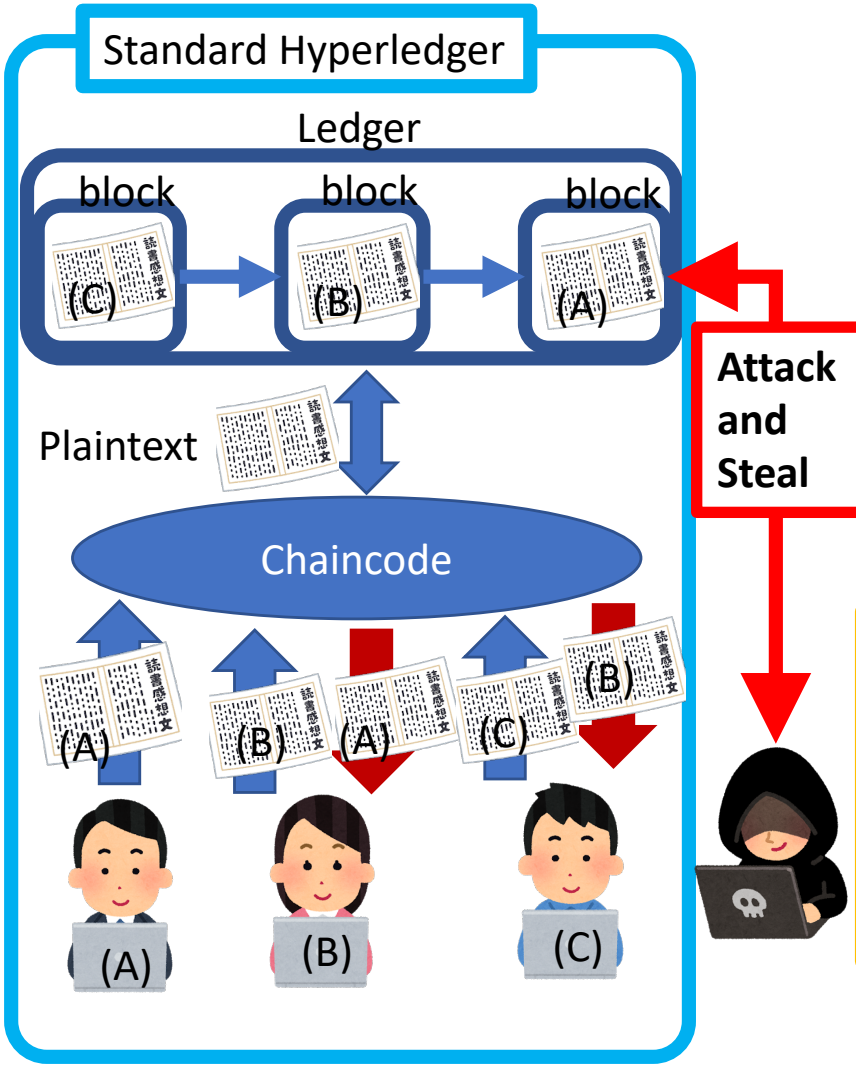
エミュレータとしてContainernetを利用する，仕様言語はPython

内容

- 利用するツール Containernet
  - Dockerコンテナを利用することができる
  - ネットワークエミュレータ
  - Ubuntu上で動作させるのが便利

- 複数のコンテナが端末で動作するようにする
- 設定ファイルでコンテナをネットワーク上に配置することができるようにする

アクセス制限をした
スマートコントラクトの実装



Hyperledger Fabric
Ledgerと呼ばれるデータベースをChaincodeと呼ばれるスマートコントラクトで操作する

問題点：
- LedgerとChaincode間のデータは平文
- Ledgerに保存されるデータは平文
- 攻撃者によってデータが摂取されると内容が漏洩する

テーマ：アクセス制限をしたスマートコントラクトの実装と評価
- やり取りするデータを暗号化
- Ledgerに保存するデータを暗号化

実装言語：JavascriptもしくはGo

内容

- HyperledgerFabricのサンプルChaincodeを改良
- データベースとの通信データを暗号化
- 実装言語
  - Java, Javascript, Goのどれか
  - 暗号部分はC++(既存のライブラリを利用する)
  - 暗号部分を利用できるようにする必要がある