シミュレータと実機を併用した開発実習やってみた

LED-Camp実行委員会

祐源英俊

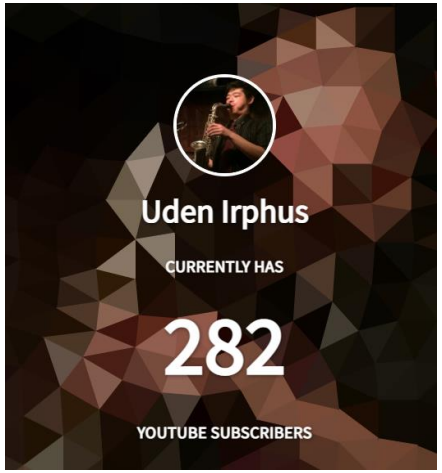
発表者自己紹介

祐源英俊（ゆうげんひでとし）

- 2018年 LED-Camp6参加
- 2018年より実行委員（4年目）
- 教材開発グループに所属
- オムロン株式会社 勤務(社会人2年目)
- ジャズの演奏が趣味

@Ke_N_551

@ken551



Uptown Jazz Underground

この発表では

組込み初学者向け開発実習 LED-Campでの教材について喋ります

実施報告とともに、ハイブリッド（実機&シミュレータ）形式の教材が

- どうやって実現されたか？
- 実際に使ってみてどうだったか？

などを説明します。

加えて、開発型の研修・教育活動やその教材について、
議論・意見交換をしましょう！

発表目次

1. LED-Campってどんなイベント？
2. 実機TankとシミュレータTank
3. “なぜ”併用したの？
4. “どうやって”併用したの？
5. 併用して”どうだった”の？
6. 終わりに

発表目次

1. **LED-Campってどんなイベント？**
2. 実機TankとシミュレータTank
3. “なぜ”併用したの？
4. “どうやって”併用したの？
5. 併用して”どうだった”の？
6. 終わりに

そもそも，LEDCampってなんなん？

- 組込み初学者（学生，社会人1,2年目など）
  を対象とする合宿形式の研修
- アジャイル，スクラム，MDDなど開発手法の講
- 競技会に向けたソフトウェアをチームで開発する

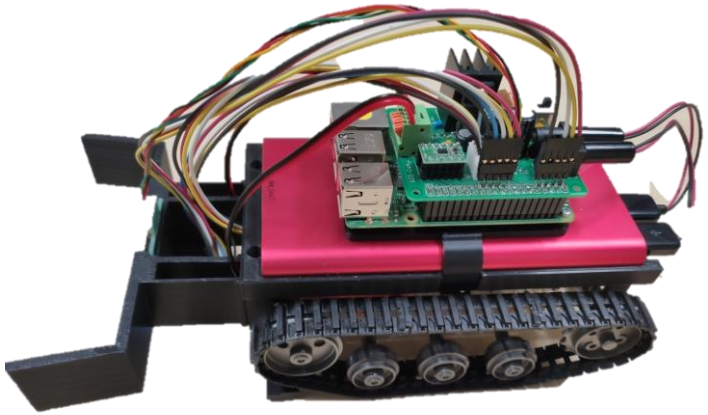
→チーム開発のコツ・難しさなどを経験的に学べ

発表目次

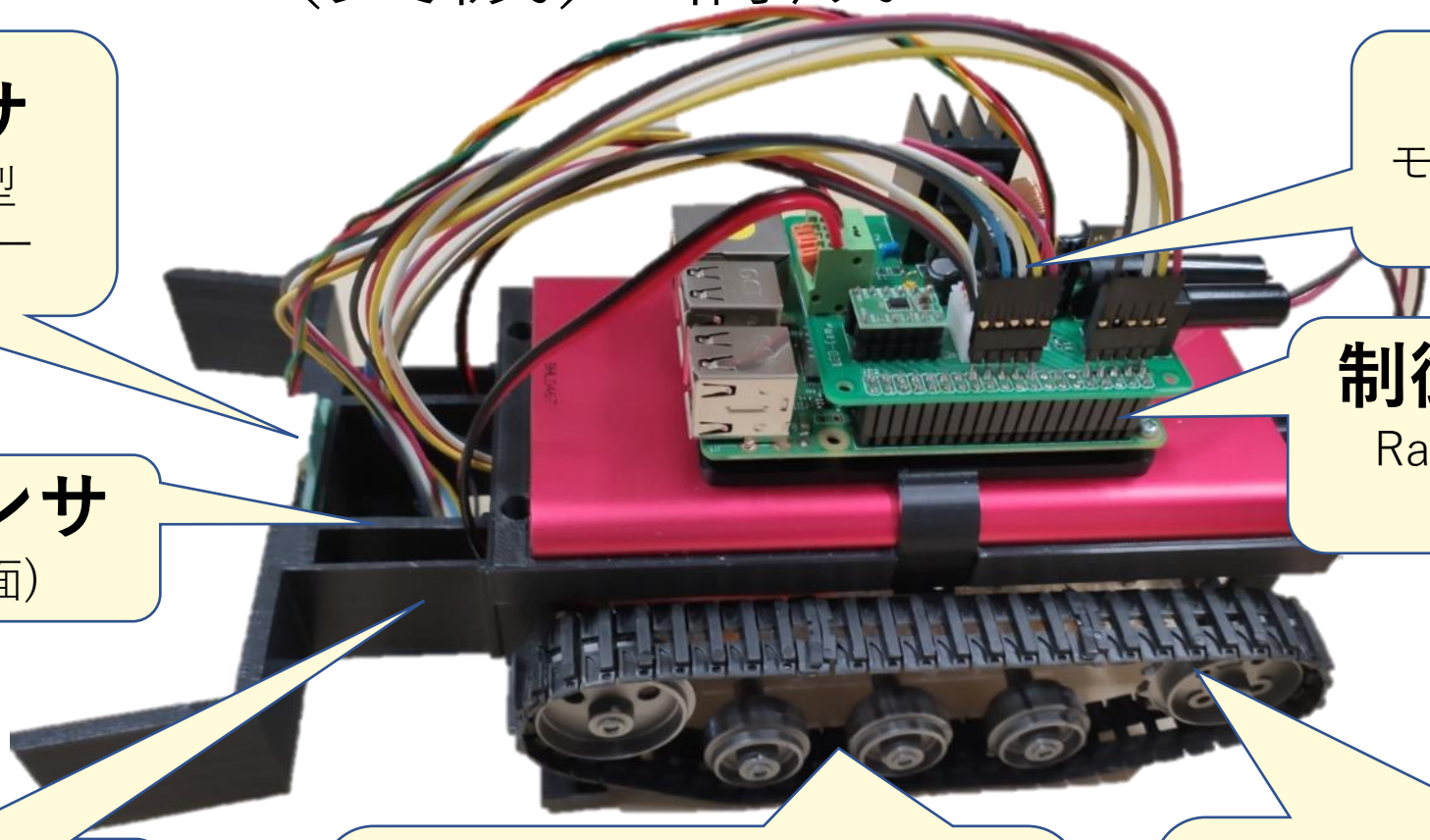
1. LED-Campってどんなイベント？
2. **実機TankとシミュレータTank**
3. “なぜ”併用したの？
4. “どうやって”併用したの？
5. 併用して”どうだった”の？
6. 終わりに

オンサイト形式のLED-Camp

- 下呂温泉（山形屋）での**三泊四日の合宿**
- 講義，開発実習，発表会など密なスケジュール
- ナイトセッションを始めとする交流の機会も
  （今年はコロナ対策で控えめ…）
- Camp7（2019年）では
  LED-Tank改（右写真）を教材に使用



LED-Tank（実機）構成



測距センサ
VL53L0X ToF型
レーザーセンサー

内製拡張基板
モータドライバ DRV8835
各センサ接続

制御コンピュータ
Raspberry Pi Model 3 B
Raspbian OS

カラーセンサ
BH1745(底面)

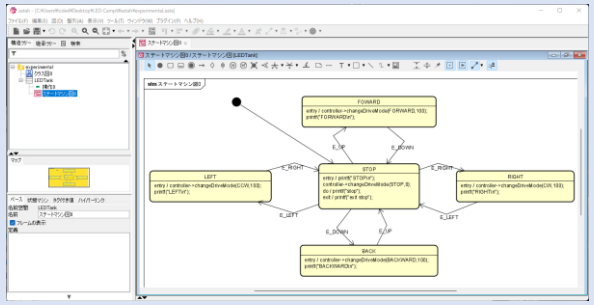
ラインセンサ
内製フォトセンサ基板

内製キャタピラ台車
タミヤのキャタピラキット
＋内製（3Dプリンタ出力）部品
＋モバイルバッテリー

内製エンコーダ基板
DCモータ
＋ロータリーエンコーダー
(LBT-131フォトインタラプタ使用)

実機を使用する開発環境

参加者PC



UML図（astah*上）

コード生成

C++

Tank制御コード
（実機向け）

Tank上RasPi

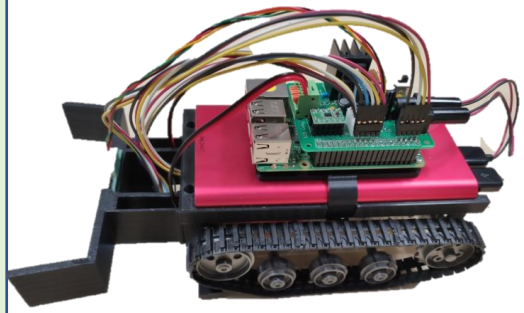
C++

Tankドライバ

コンパイル・リンク

a.out

実行ファイル



実行！

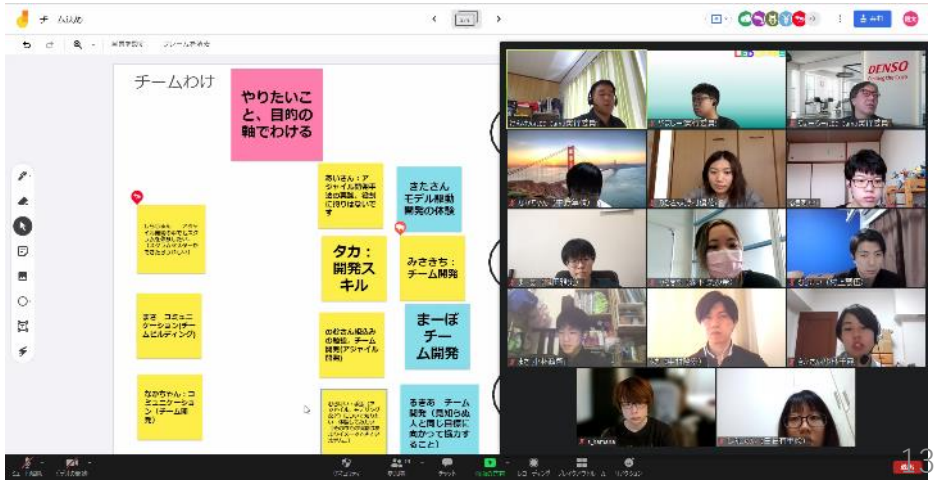
実機での競技会（Camp7より）

実機、持ってきてます！

とくとご覧あれ

オンライン形式のLED-Camp

- 昨年のLED-Camp9は、初の**完全オンライン開催**
- これまでの内容に加え，**オンラインコミュニケーション**および**オンラインチーム開発**の経験を目的に
- ロボットシミュレータWebotsを用いたバーチャル競技会を実施



LED-Tank（シミュレータ）構成

測距センサ
DistanceSensor
(Laser)

カラーセンサ
Camera

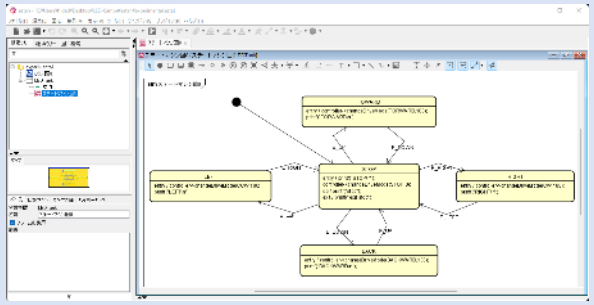
ラインセンサ
DistanceSensor(IR)

内製3Dモデル
Fusion360で作成
3Dプリント用データを一部使用

モータ回転角センサ
Motor
+PositionSensor

実機を使用する開発環境

参加者PC



UML図（astah*上）

コード生成

C++

Tank制御コード
（シミュレータ向け）

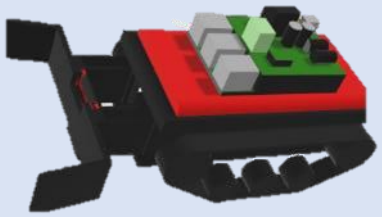
C++

Tankドライバ
（シミュレータ向け）

コンパイル・リンク

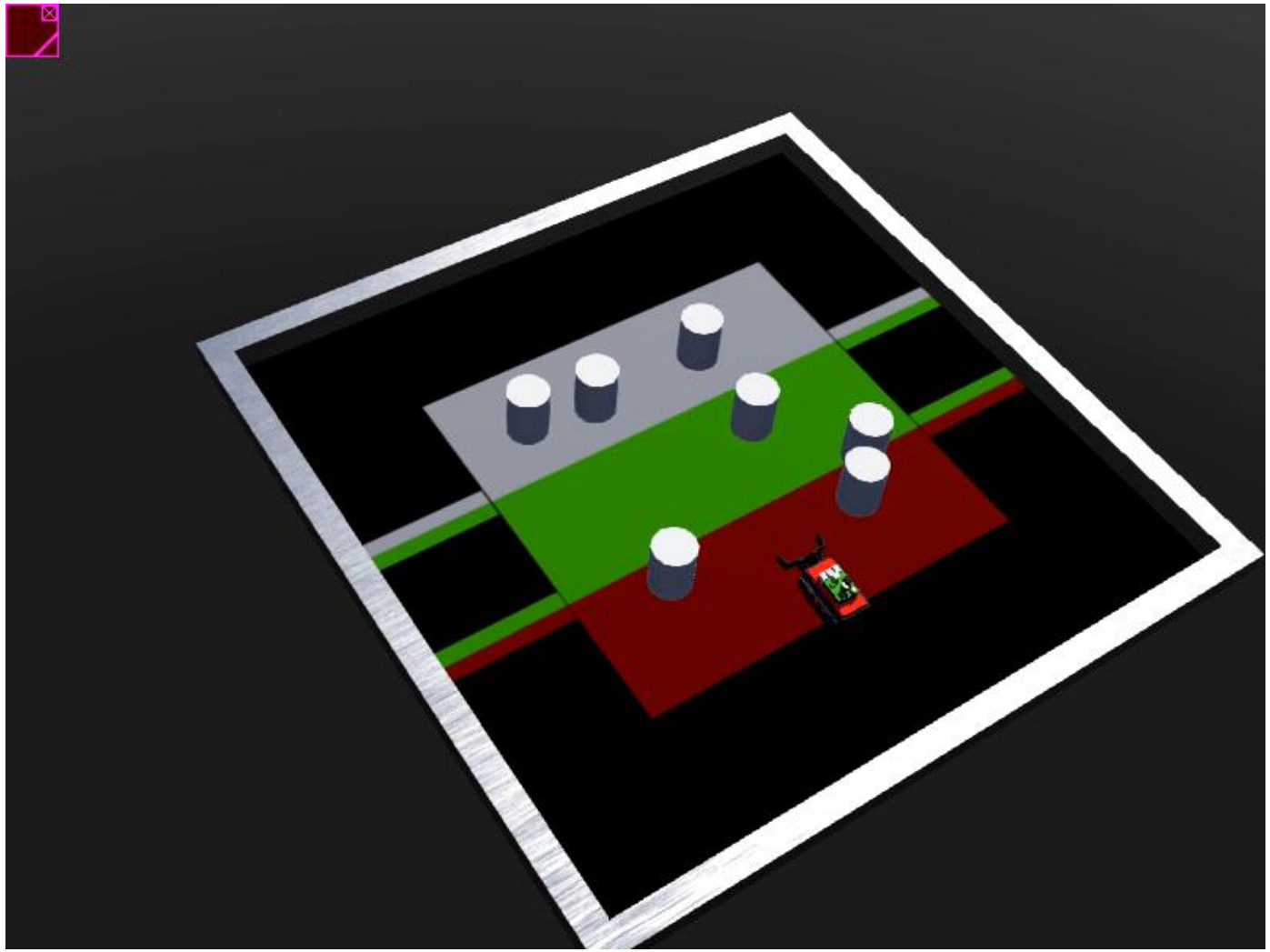
.exe

実行ファイル



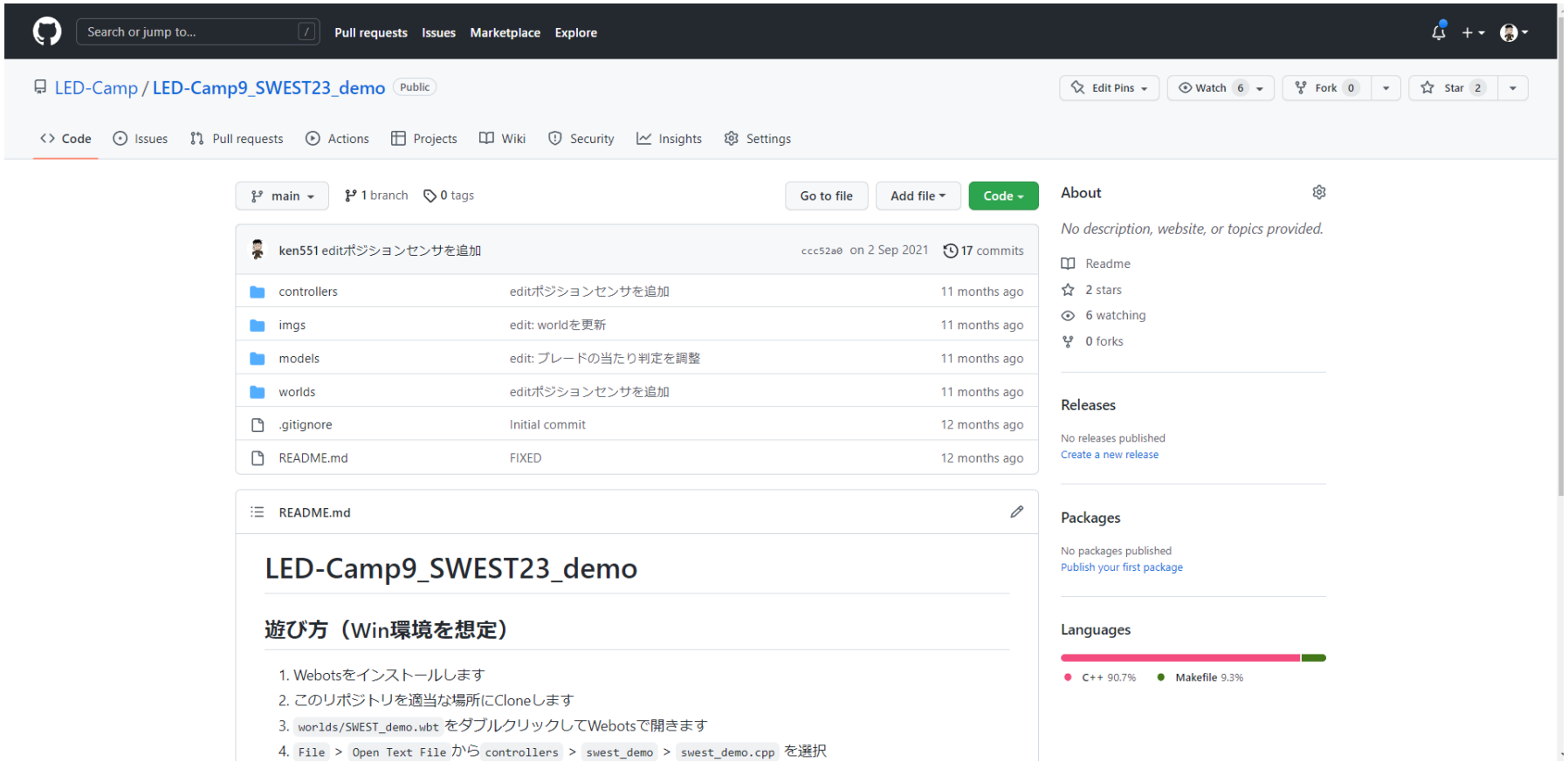
実行！

シミュレータでの競技会（Camp9）

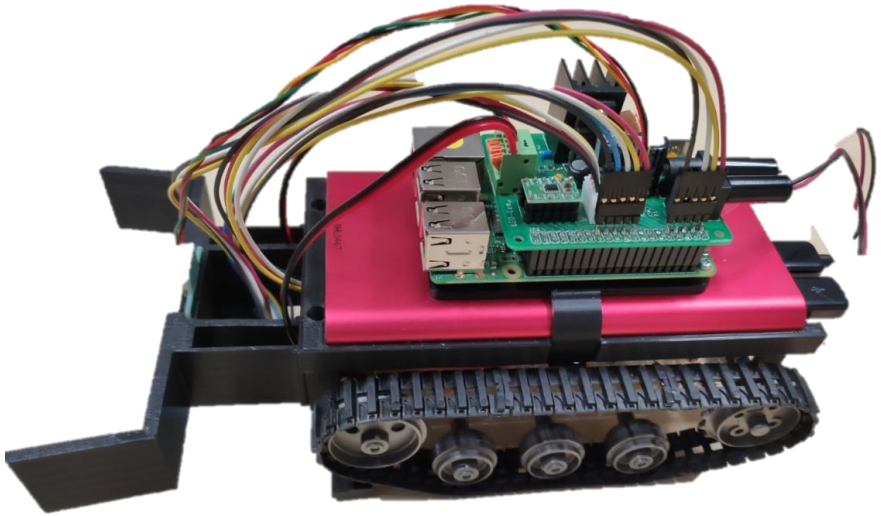


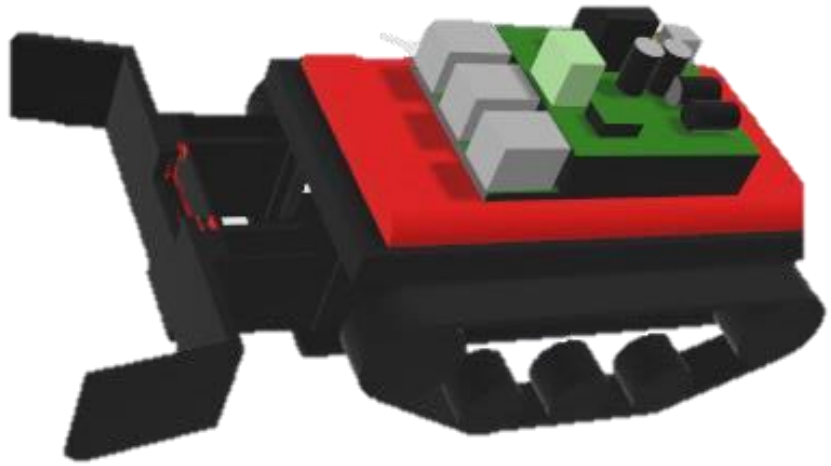
昨年度のデモ、あります！

https://github.com/LED-Camp/LED-Camp9_SWEST23_demo



じゃあ今年は・・・？





- 組み合わせてみました

発表目次

1. LED-Campってどんなイベント？
2. 実機TankとシミュレータTank
3. **“なぜ”併用したの？**
4. “どうやって”併用したの？
5. 併用して”どうだった”の？
6. 終わりに

なんでハイブリッド形式に？

ハイブリッド形式の主な目的・狙いは以下

- 開発実習中のボトルネック解消
- 「シミュレータでのソフト検証は大事だよね」と伝えたい
- 「でもそれに加えて実機検証も大事だよね」と伝えたい
- 実行委員の興味

開発実習中のボトルネック

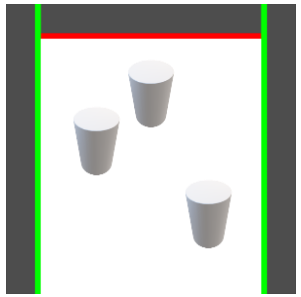
- 筐体の制約
- 場所の制約

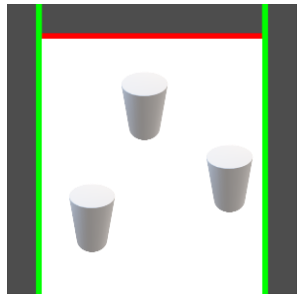
開発実習中のボトルネック（筐体）

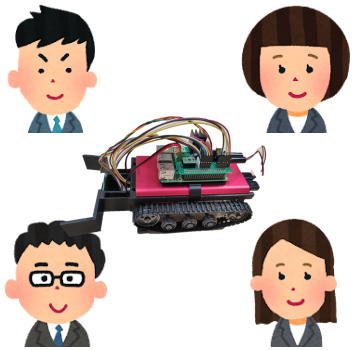
- 実機はチーム（3-4人）に１台
  →分散的な開発が難しい
  - コンパイルにラズパイが必要
    →実機検証の手前で時間がかかる
  - プログラム検証で実機を占有
    →狙ったプログラムの完成に
    時間がかかる

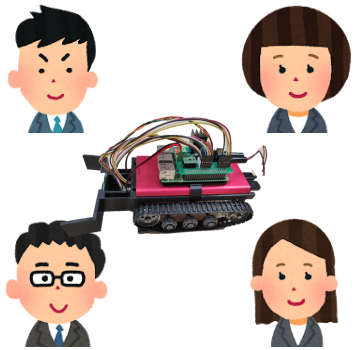
開発実習中のボトルネック（場所）

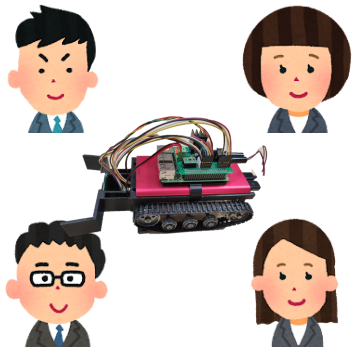
- コースは全チームに対し1-2セット（タイムスライス制）

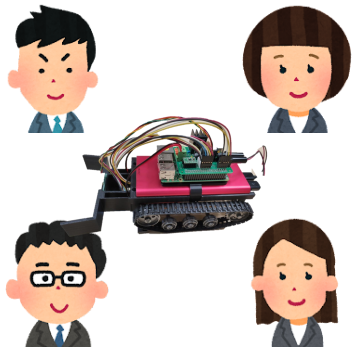






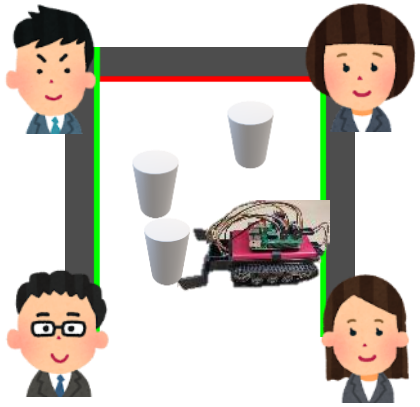


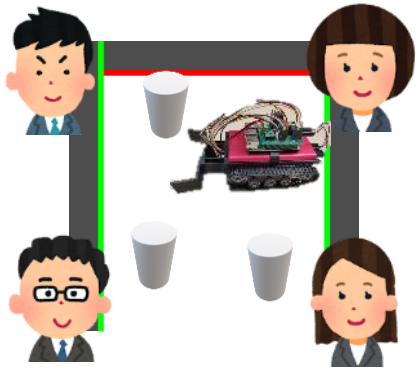




開発実習中のボトルネック（場所）

- コースは全チームに対し1-2セット（タイムスライス制）



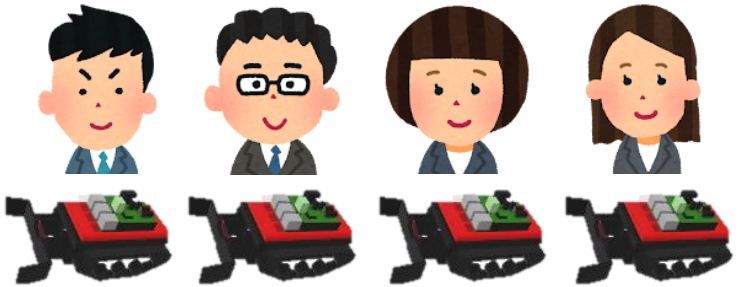


・チームの作戦が検証できない

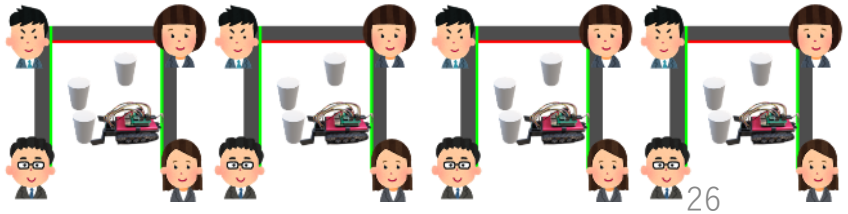
・実機での動作確認ができない

シミュレータの利用によるボトルネック解消

- 筐体のボトルネック：個人がシミュレータを利用して解決
  - シミュレータ上でビルド可否を検証可能
    →コンパイルエラーを個人作業に
  - シミュレータ上である程度の振舞は把握可能
    →イージーミス修正を個人作業に

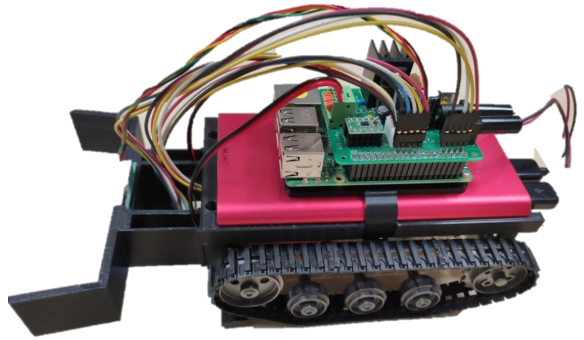


- 場所のボトルネック：チームがシミュレータを利用して解決
  - 作戦の検証を全チーム並列にできるように
  - 実機動作確認の代わりは狙わず



「なんだかんだ実物を使う検証が大事だよね」と伝えたい

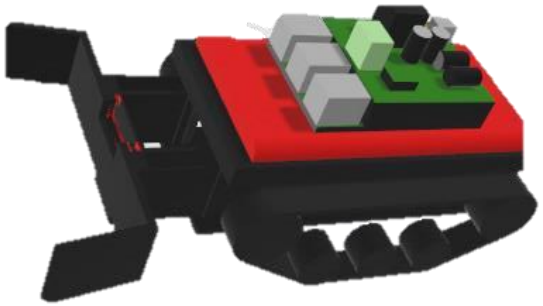
- 実機ーシミュレータ の完全な一致は**不可能**



→実際のシミュレータ・エミュレータ使用時には、どこまでできるか？の理解が大事

シミュレータと実機を同時に触ることで

- 「こういうところの振る舞いが違う」という気付き
- 「こういう理由で振る舞いが違う」という分析
- 「振る舞いが違うのでこう使おう」という工夫



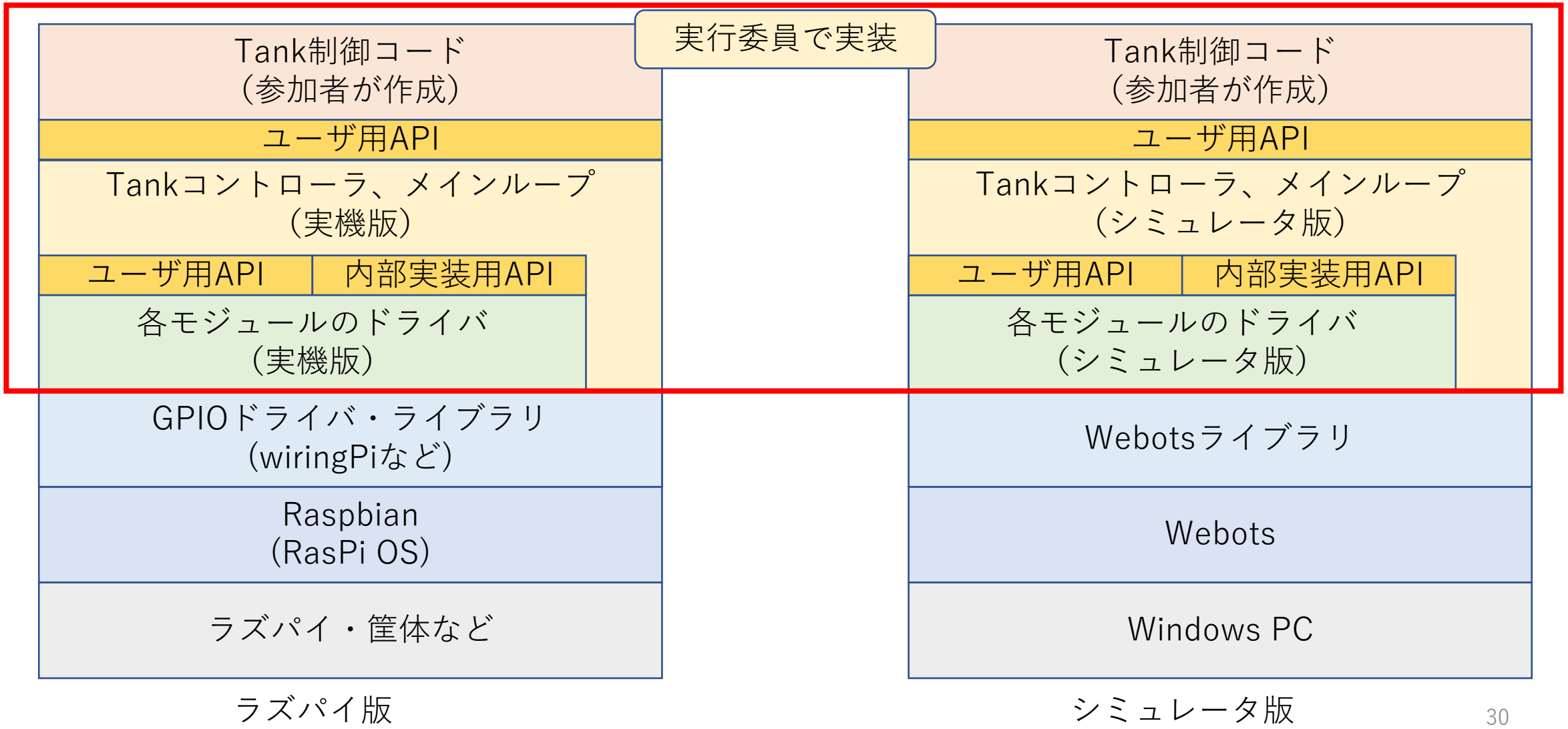
実行委員の興味

- おもしろいかなとおもったから・・・・

発表目次

1. LED-Campってどんなイベント？
2. 実機TankとシミュレータTank
3. “なぜ”併用したの？
4. **“どうやって”併用したの？**
5. 併用して”どうだった”の？
6. 終わりに

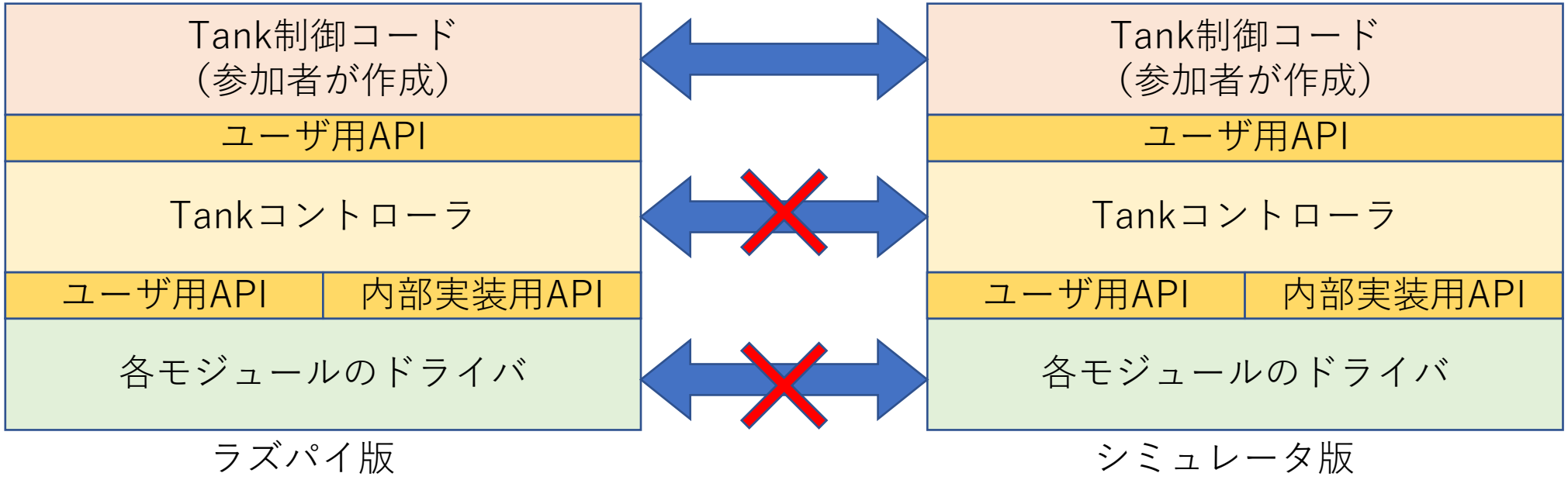
実機版・シミュレータ版のソフト構造



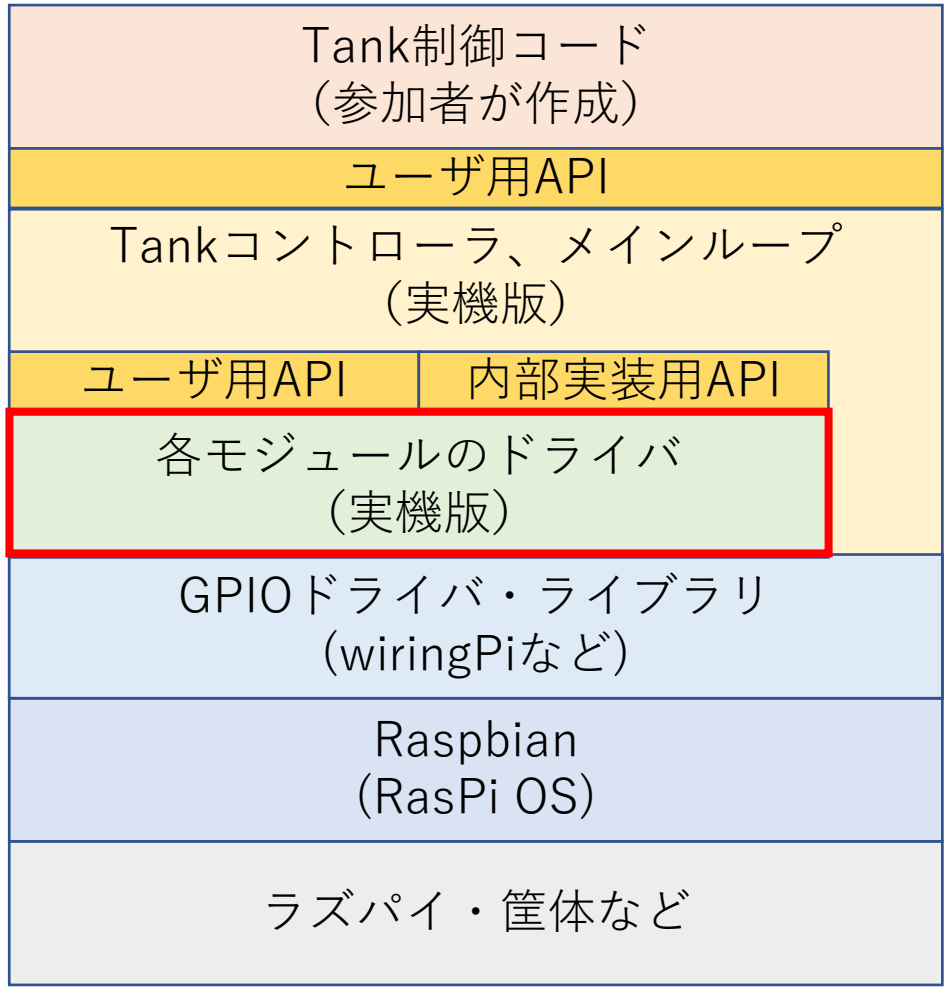
どれくらい使いまわしたの？

Q.ほぼ同じ構造だけど、まるっと使いまわしたの？

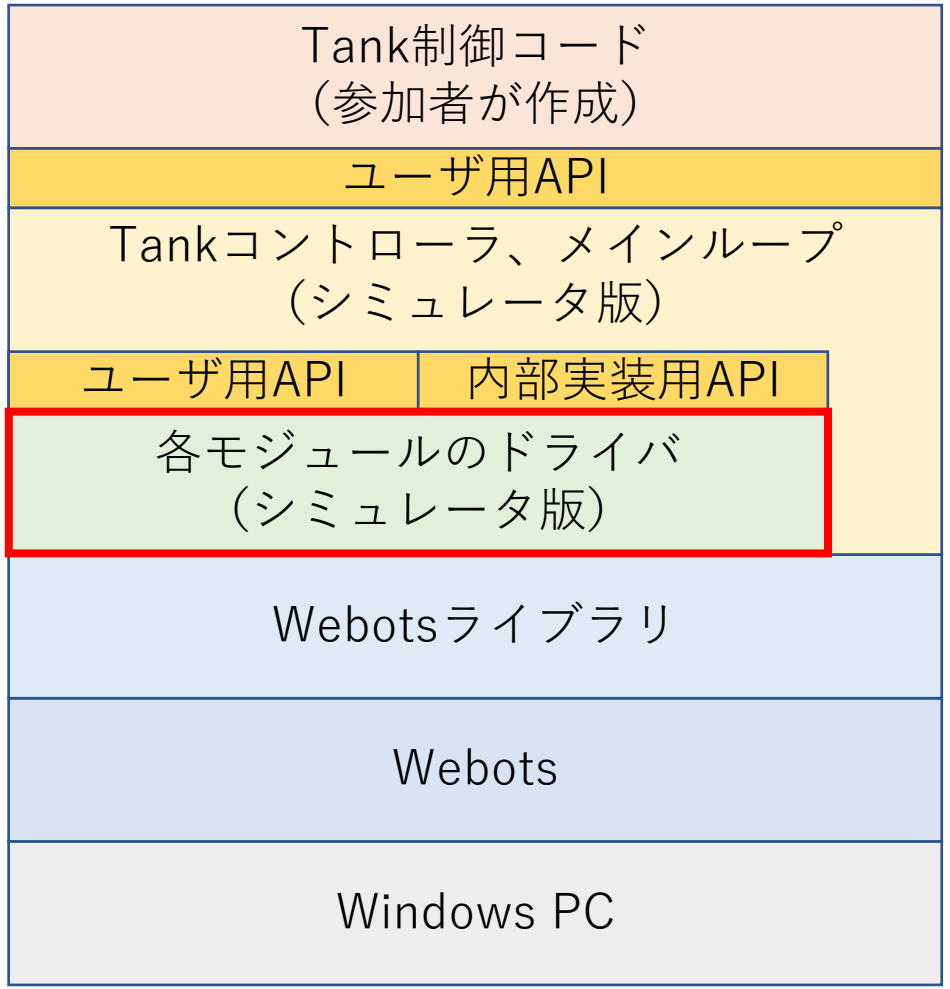
A.**Tank制御コードとその下のAPI以外は基本的に作り直しました**



各モジュールのドライバの中身は？



ラズパイ版



シミュレータ版

各モジュールのドライバ実装例

ラインセンサの値を取得する getLineValue()の中身は・・・

Webotsのセンサから
連続値を取得→スレッシュで区切って
0/1にしてリターン

```
LineValue
    LineVa

    lineValue.left = digitalRead(this->pinLeft);
    lineValue.center = digitalRead(this->pinCenter);
    lineValue.right = digitalRead(this->pinRight);

    return lineValue;
}
```

ラインセンサからのLo/Hi値を
ピンのinputで取得

ラズパイ版

```
LineValue LineSensor::getLineValue(){
    LineValue lineValue;
    if(this->sensorElementLeft->getValue() > THRESHOLD_LINE_VALUE)
        lineValue.left = LINE_SENSOR_VALUE_BLACK;
    } else {
        lineValue.left = LINE_SENSOR_VALUE_WHITE;
    }
    if(this->sensorElementCenter->getValue() > THRESHOLD_LINE_VALUE
```

シミュレータ版

各モジュールのドライバ実装例

同名のAPIに
異なるインタフェース

```
ColorSensor* ColorSensor::getInstance()
{
    if (_instance == 0)
    {
        _instance = new ColorSensor();
    }

    return _instance;
}
```

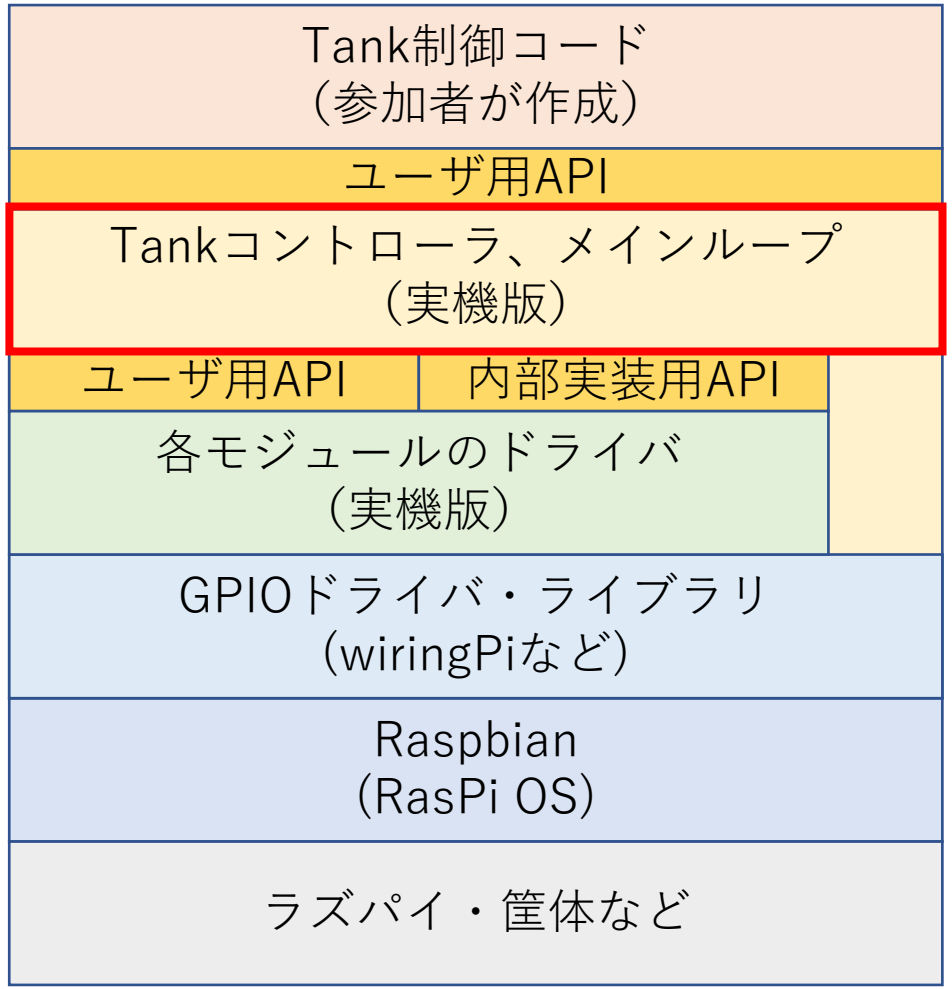
```
ColorSensor* ColorSensor::getInstance(Supervisor* supervisor, std::string sensorName, int timeStep) {
    if (_instance == 0) {
        _instance = new ColorSensor(supervisor, sensorName, timeStep);
    }

    return _instance;
}
```

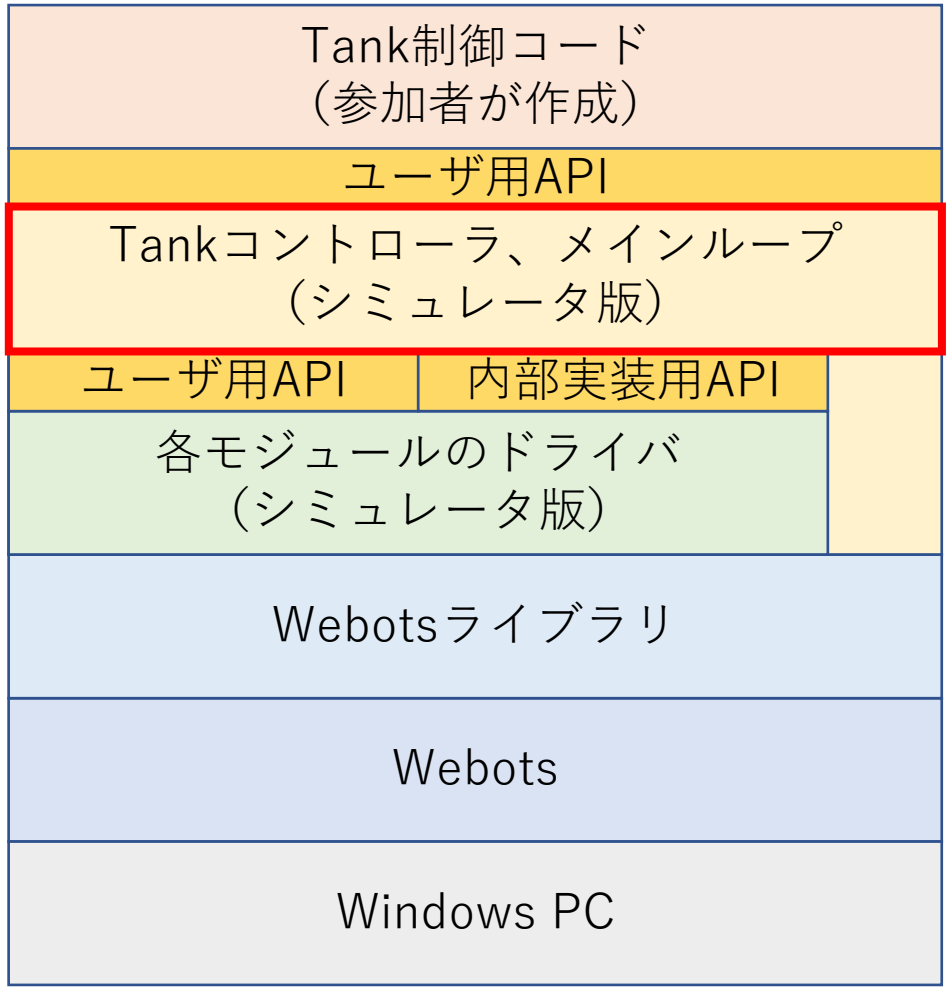
ラズパイ版

シミュレータ版

各モジュールのドライバの中身は？



ラズパイ版



シミュレータ版

メイン関数 初期化処理は？

```
int main(void){
  struct timeval now;
  struct timeval old;

  Controller *controller;
  Event *event;

  LEDTank *lEDTank;

  if( wiringPiSetupGpio() < 0){ //initialize failed
    return 1;
  }
```

ラズパイ版

```
int main(int argc, char **argv) {
  Controller *controller;
  LEDTank *lEDTank;
  Event *event;

  controller = Controller::getInstance();
  lEDTank = new LEDTank(controller);
  event = new Event(controller);
```

シミュレータ版

初期化処理は
メイン関数にべた書き

メインループの中身は？

プラットフォーム依存の部分が
やや多くなっている

```
while(true){
  while((now.tv_sec - old.tv_sec) + (now.tv_
    gettimeofday(&now, NULL);
  }
  old = now;
  if(event->updateEvent() < 0){
      printf("STOP\n");
      controller->changeDriveMode(STOP, 0);
      break;
   }
```

ラズパイ版

```
while (controller->clockForward())
  if(event->updateEvent() < 0 || co
      controller->changeDriveMode(S
      controller->clockForward();
      break;
   }
  lEDTank->execState();
  lEDTank->doTransition(event->getE
}
```

シミュレータ版

コントローラ部分の中身は？

```
Controller::Controller(void) {
    position = Position::getInstance(17, 27);
    lineSensor = LineSensor::getInstance(10, 9, 11);

    colorSensor = ColorSensor::getInstance();
    colorSensor->Initialize();

    rangingSensor = RangingSensor::getInstance();
    rangingSensor->Initialize();

    twinWheelDriver = TwinWheelDriver::getI

}

Controller::~Controller(void) {
}

void Controller::positionReset(void) {
    position->reset();
}

PositionValue Controller::getPosition() {
    return position->getPosition();
}
```

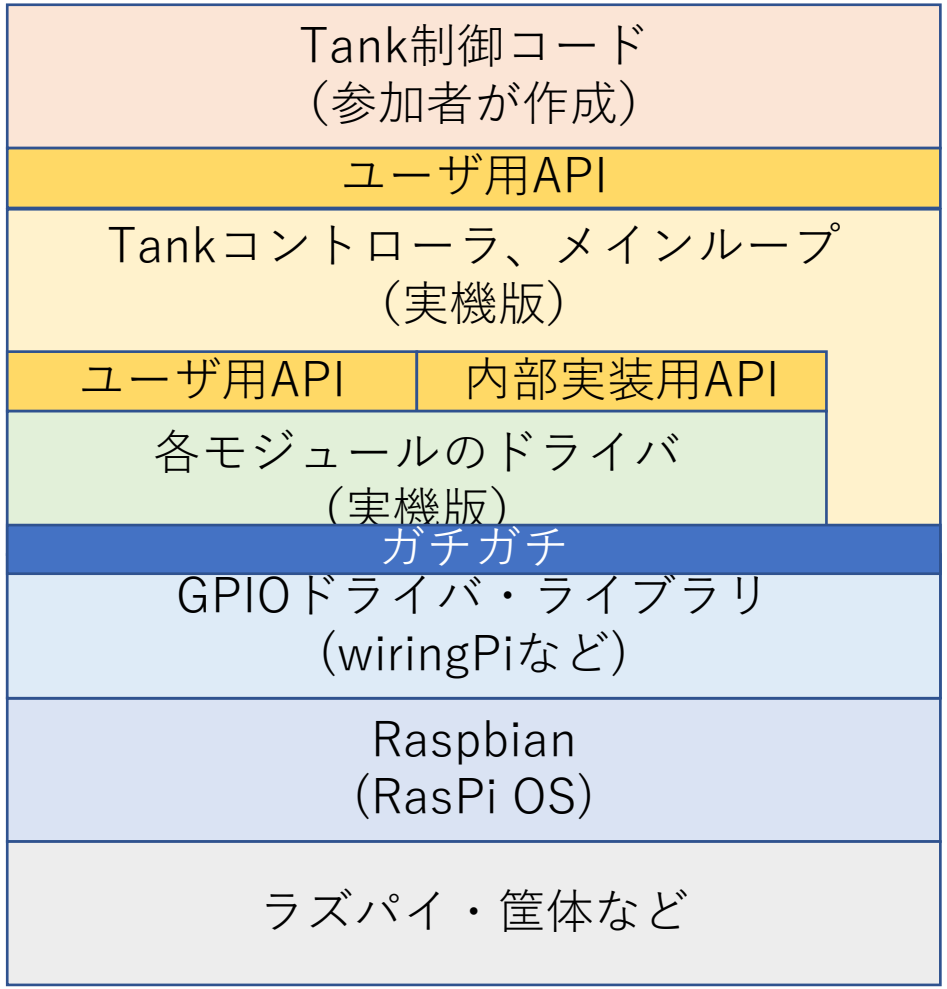
ラズパイ版

```
Controller::Controller() {
    this->supervisor = new Supervisor();
    // 測距センサ初期化
    this->rangeSensor = RangeSensor::getInstance(supervisor, "RangeSensor", TIME_STEP);
    // カラーセンサ初期化
    this->colorSensor = ColorSensor::getInstance(supervisor, "cam", TIME_STEP);
    // ラインセンサ初期化
    this->lineSensor = LineSens          nstance(
        supervisor
    position = Position::getInstance(supervisor, "positionSensorL", "positionSensorR");
    // モータドライバ初期化
    twinWheelDriver = TwinWheelDriver::getInstance(supervisor, "motorL", "motorR");
}
```

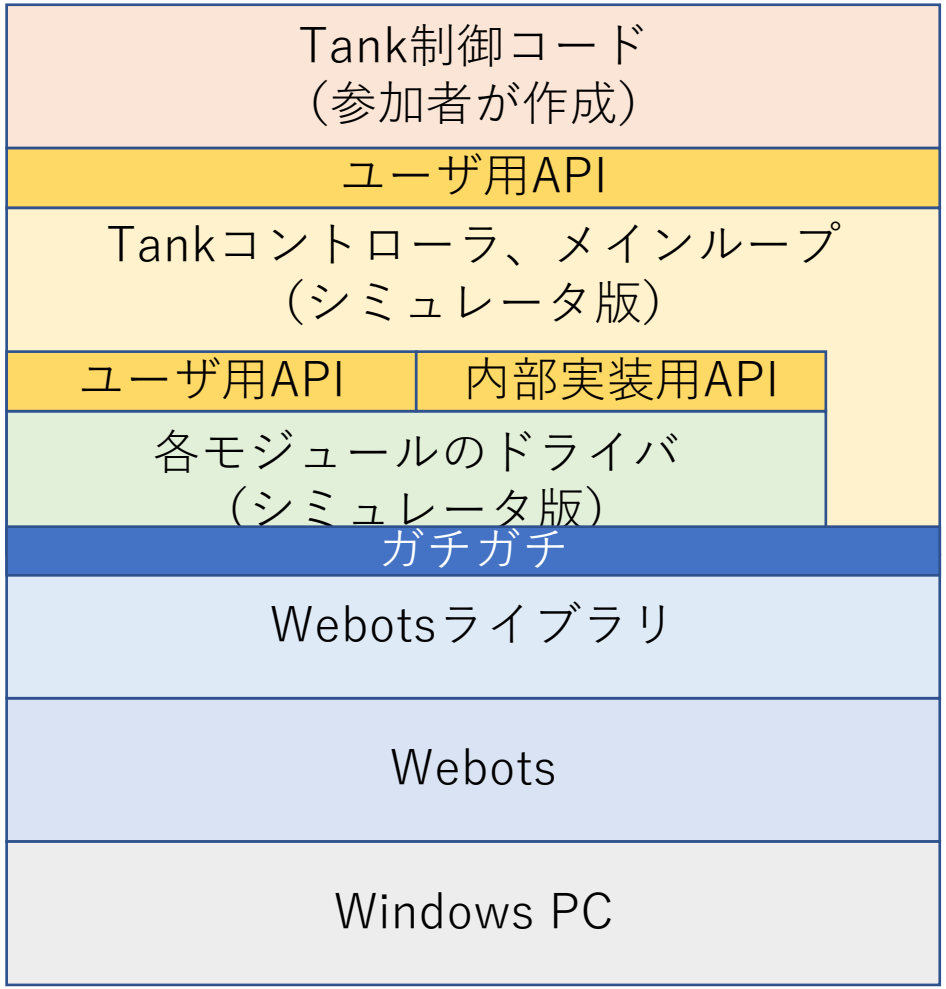
シミュレータ版

インターフェースの違いにより
呼び出し側も使いまわし不可に

各モジュールのドライバの中身は？



ラズパイ版



シミュレータ版

Tank制御コードについて

Q. じゃあ、制御コードはどれくらい共通化してるの？

A. ほぼほぼ手を加えていません！

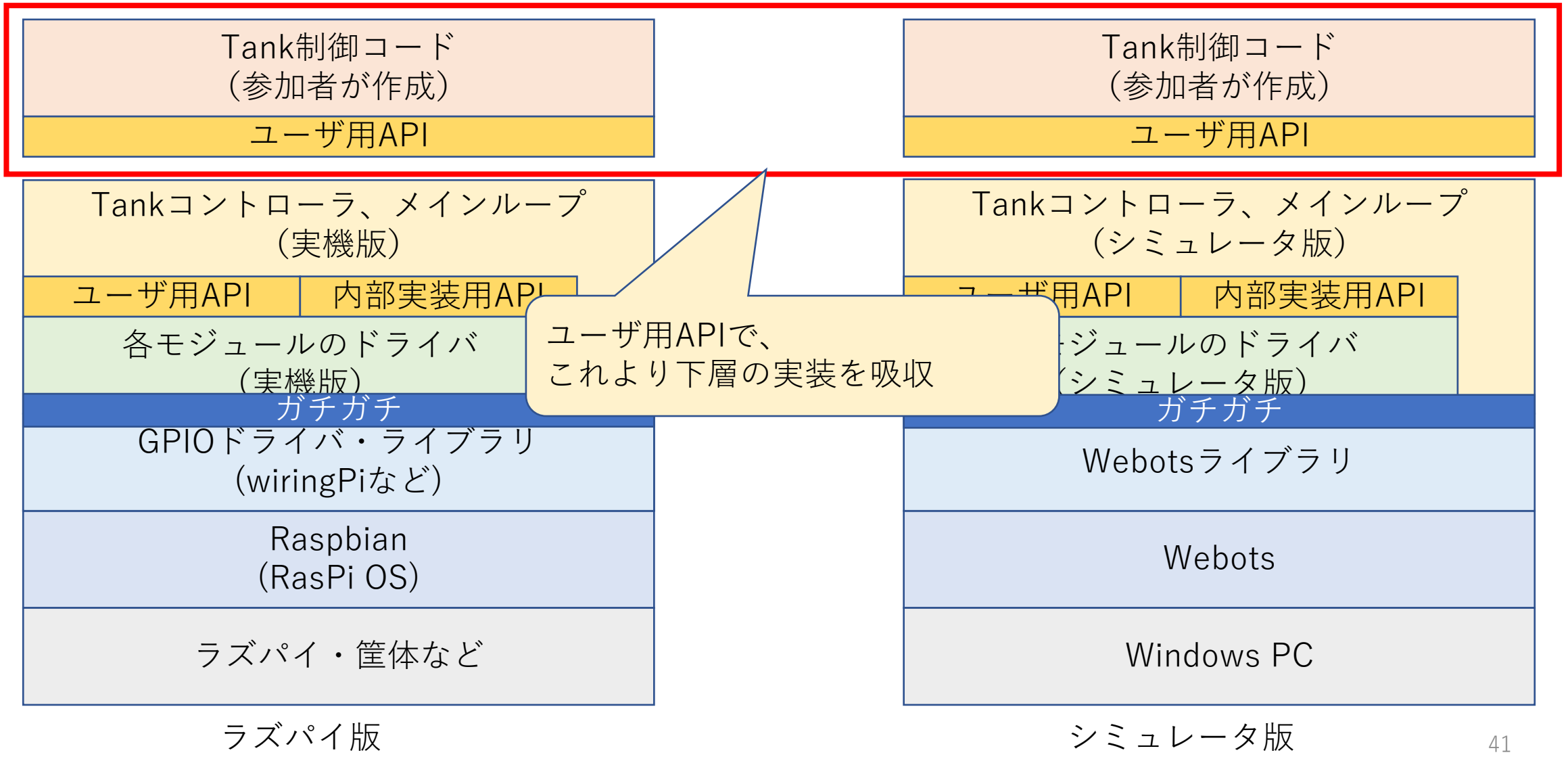
（そもそもシミュレータ開発時に変更ないことを目指した）

参加者のコード（Tank制御コントローラ）を、

- UMLクラス図・ステートマシン図からの生成
- 内製APIの仕様

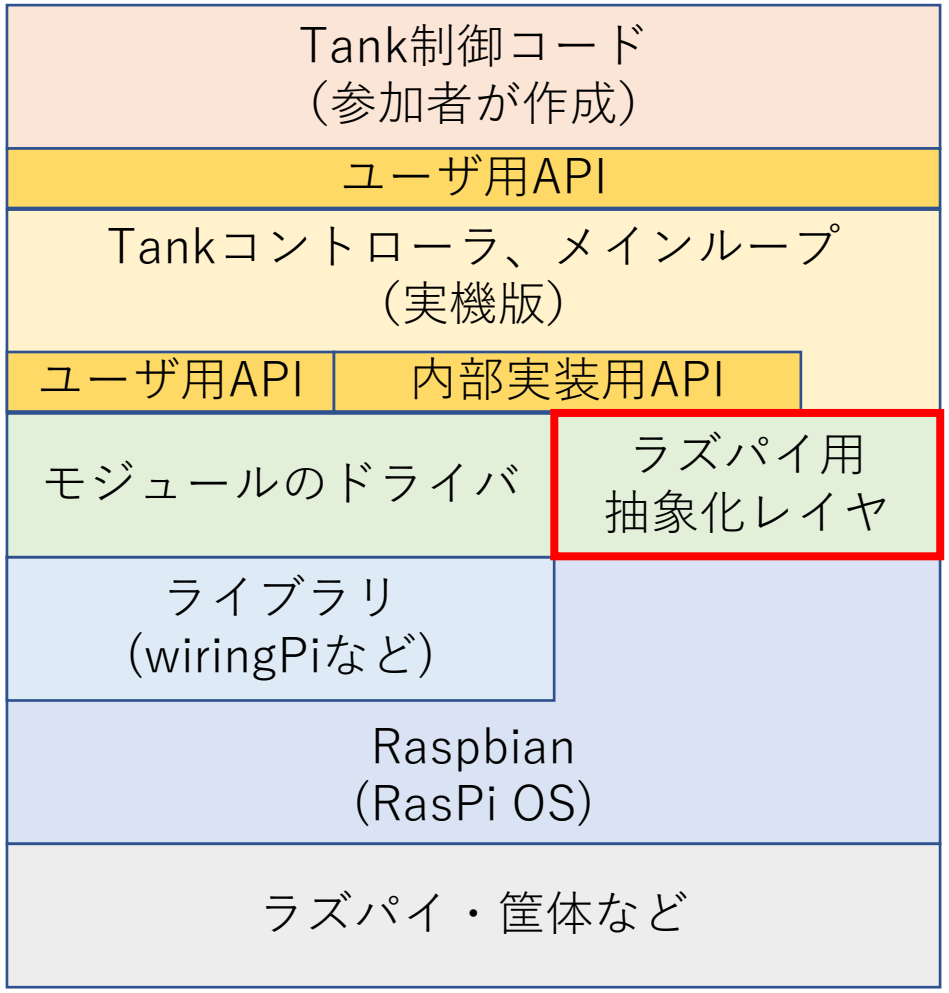
によって、ある程度**定型化**していたため

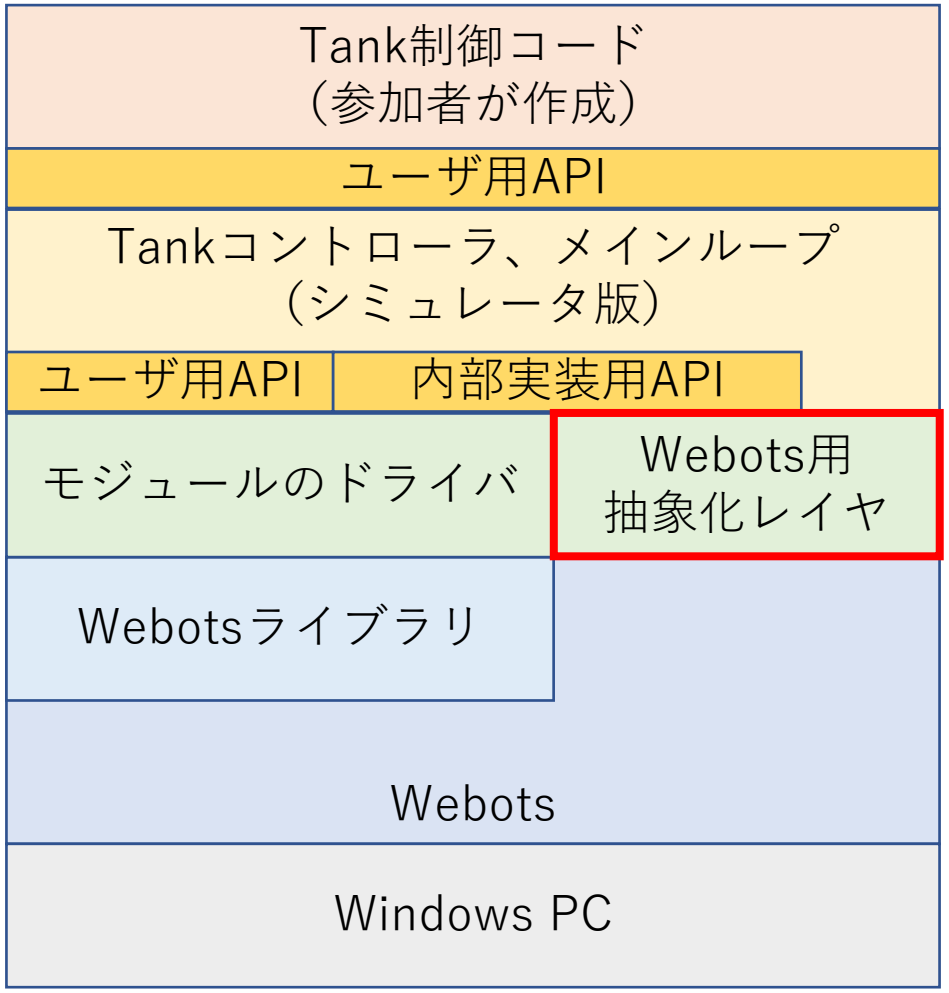
各モジュールのドライバの中身は？



余裕があればこうしたかった1

- コントローラからプラットフォーム依存の実装を分離

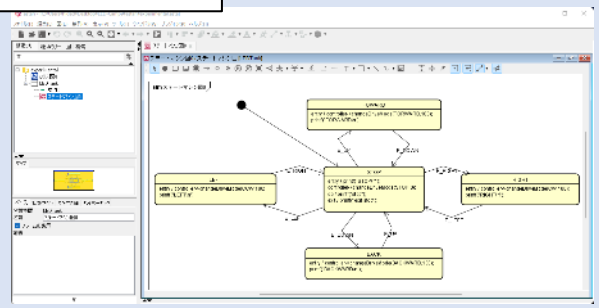




余裕があればこうしたかった2

ラズパイ用バイナリのクロスコンパイル化&オプションによる選択

参加者PC



UML図（astah*上）

コード生成

C++

Tank制御コード
（実機向け）

.exe

実行ファイル

実行！

Tank上RasPi

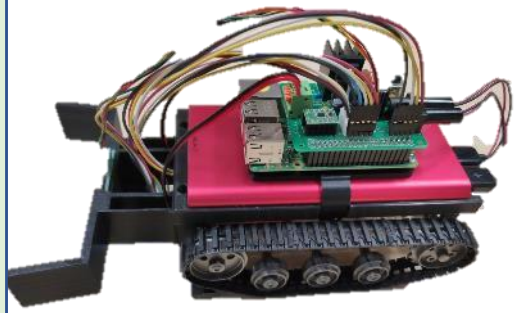
C++

Tankドライバ

コンパイル・リンク

a.out

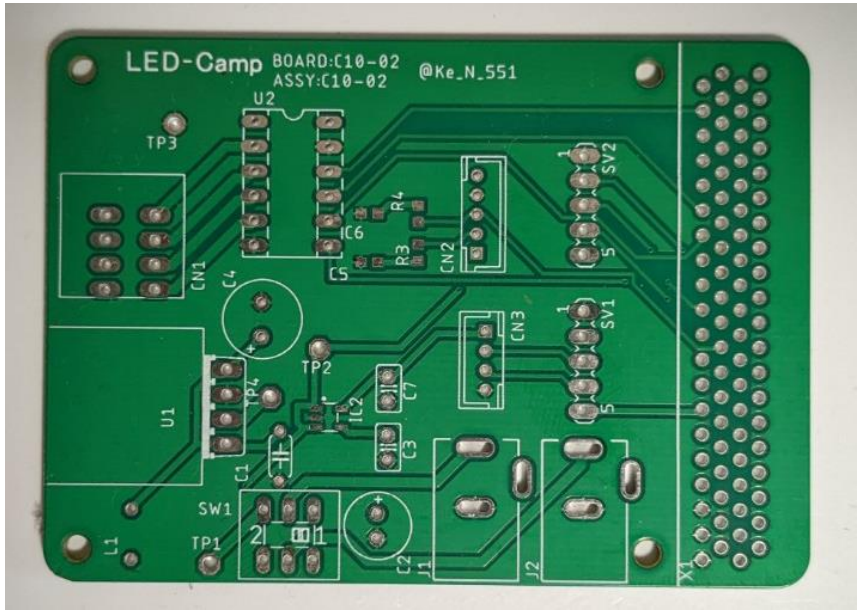
実行ファイル

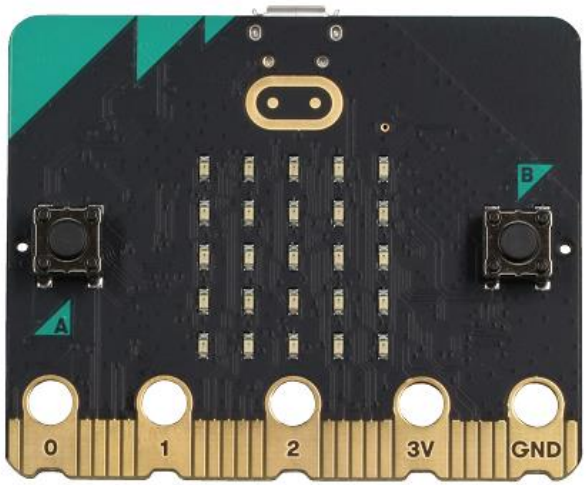


実行！

おまけ：micro:bitの導入について

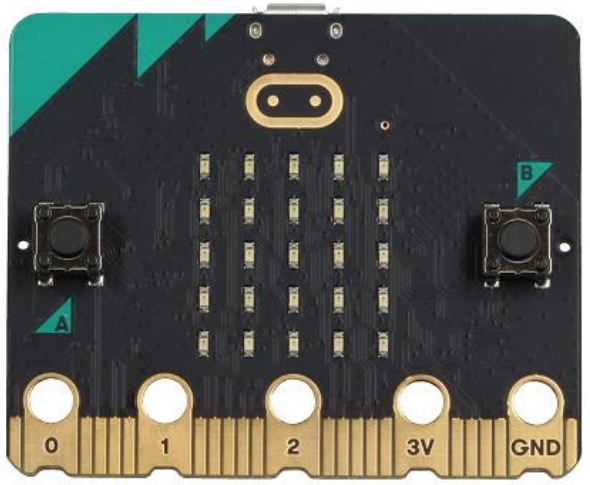
- 今年度前半は、制御部分をRaspberry Piから micro:bit（v2.2）に変更しようとしていた！
- 基板も~~勝手に~~製作（ラズパイ用基板から流用）

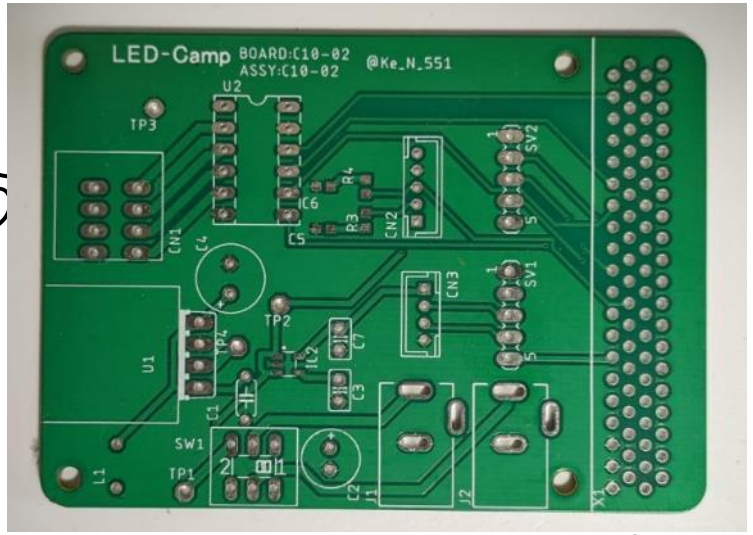




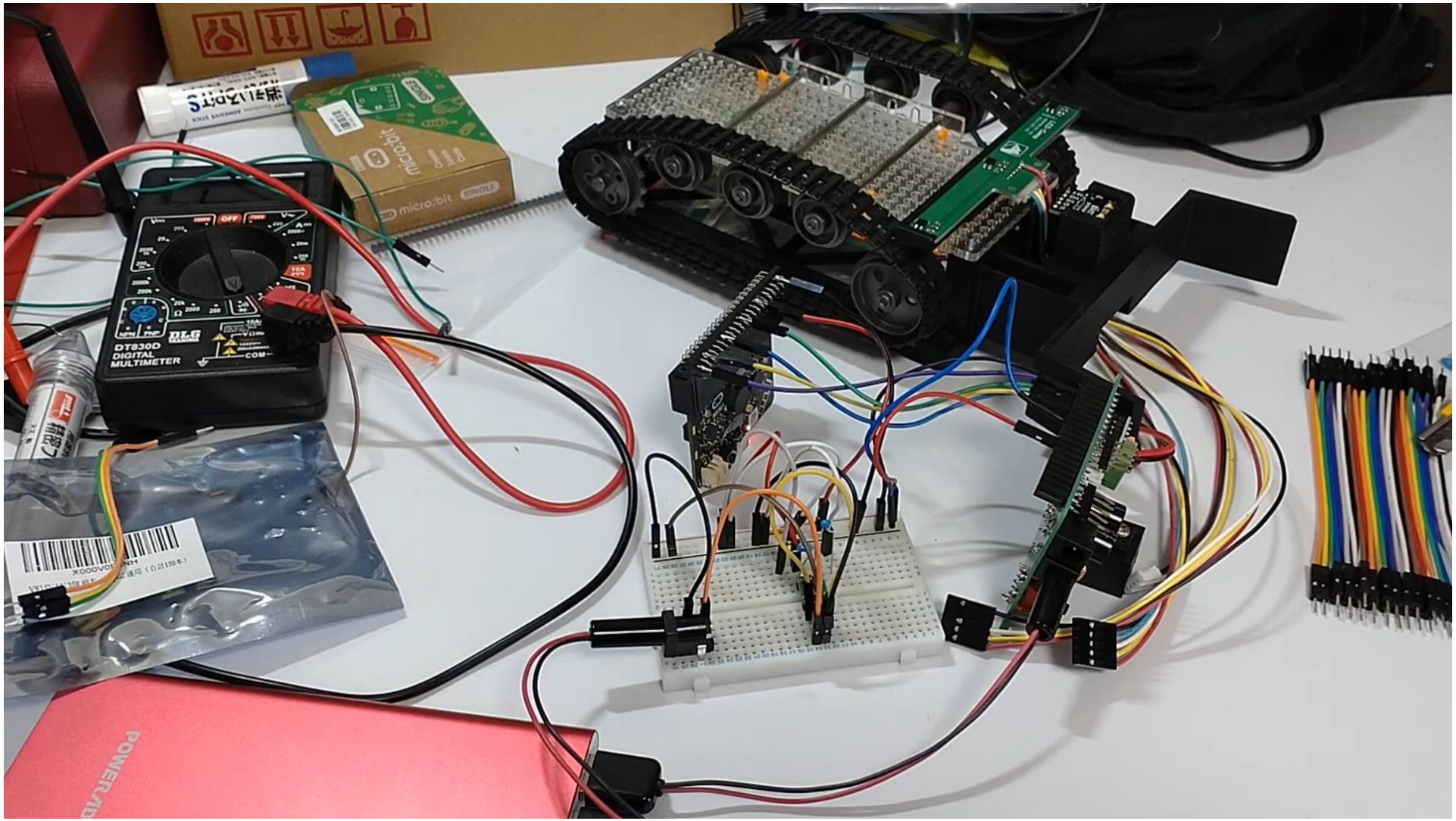
なぜmicro:bitに？

- クロス開発必要でより”組込み”ぽい
- 加速度センサ・コンパスを搭載
- Windowsで比較的簡単に環境構築できそう
- C++で開発できる
- BLEモジュール搭載でprintデバッグできそう
- 調達ができた

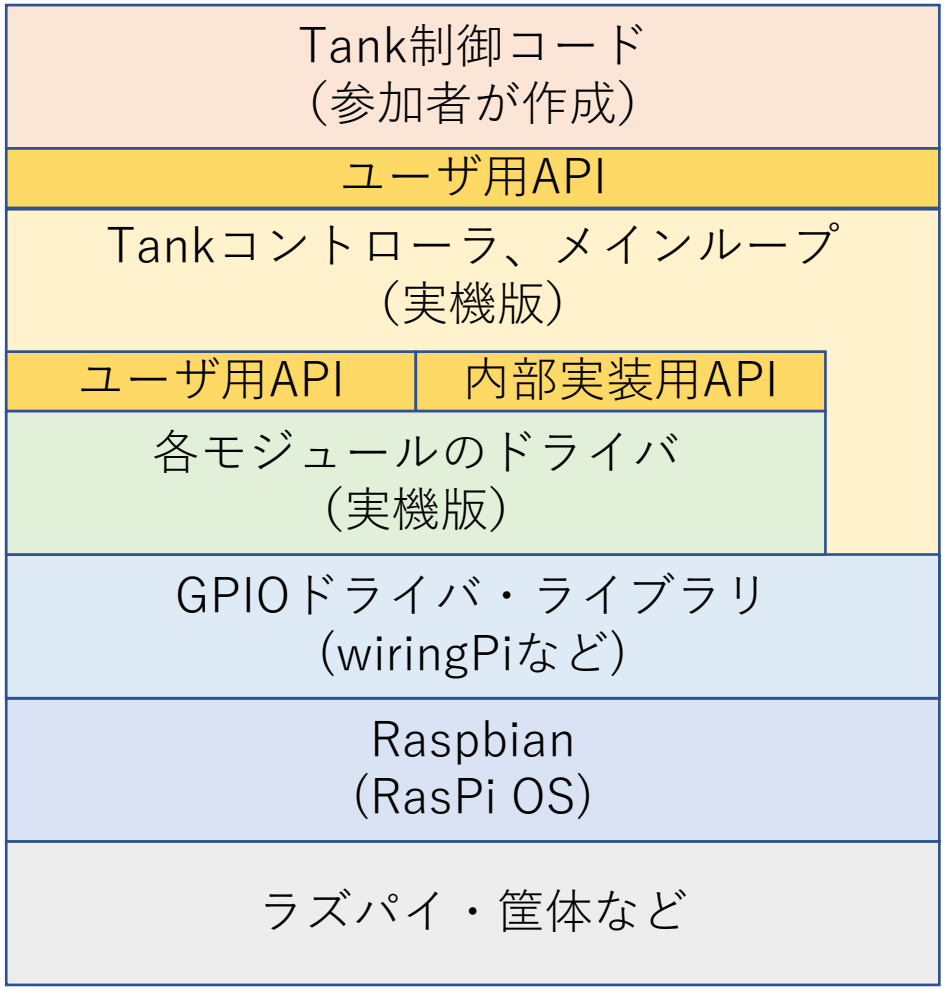




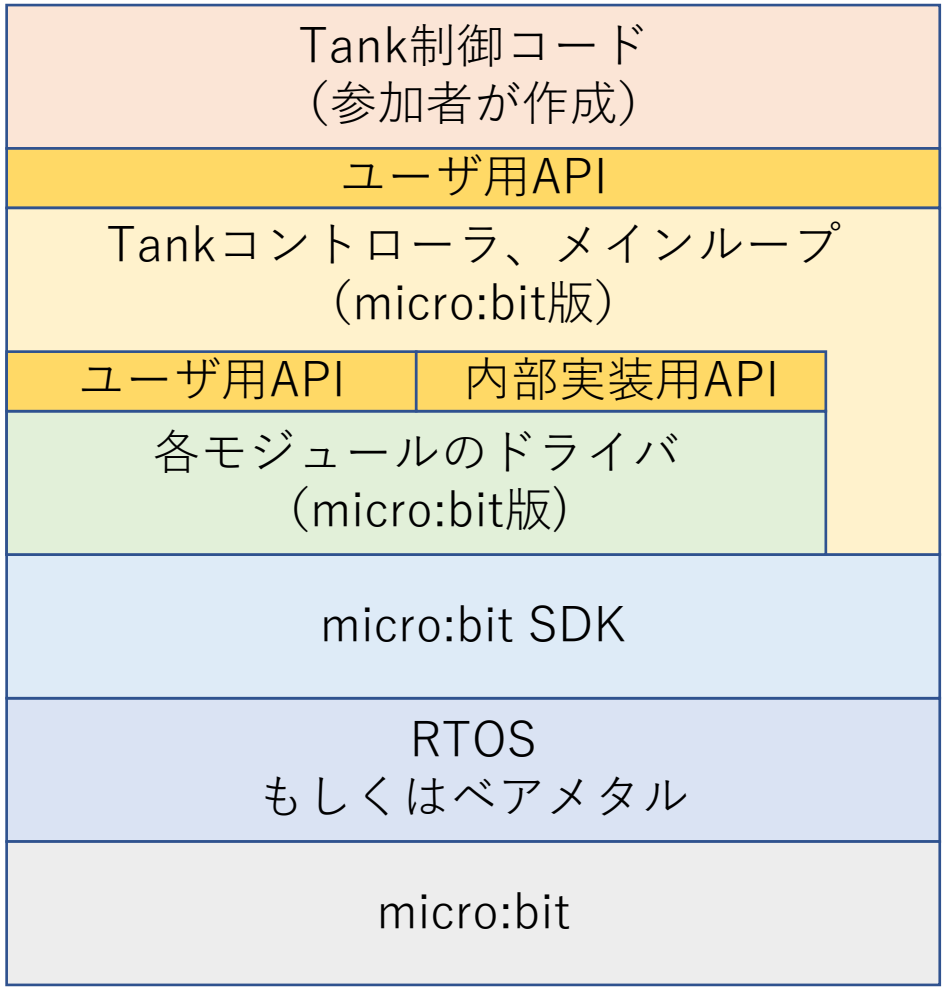
micro:bitでの開発



開発時の計画



ラズパイ版



micro:bit版

課題点

- printfの扱いが要検討
  - micro:bit→BLE シリアルで出力予定だった
  - シリアル出力用のAPIを作成→下回りを変更、が妥当？
- 処理性能が足りるか不安
  - ラズパイ（Cortex-A53 1.2GHz）→micro:bit（Cortex-M4F 64MHz）

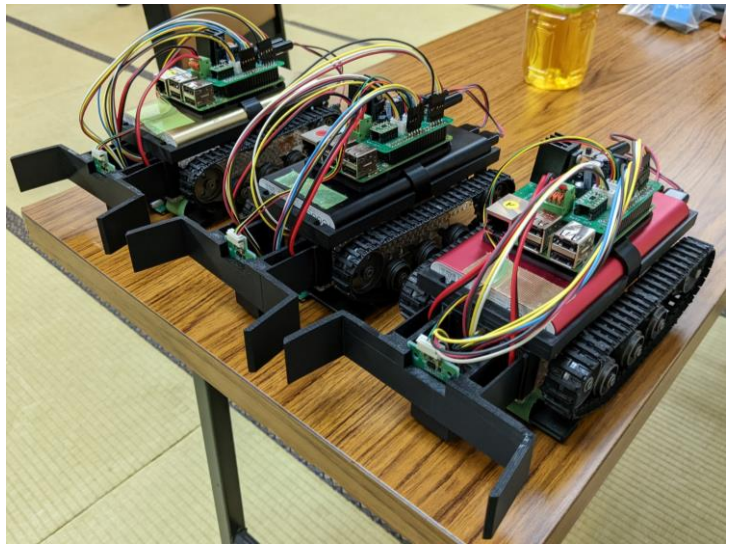
試作→検証が必要な状況

発表目次

1. LED-Campってどんなイベント？
2. 実機TankとシミュレータTank
3. “なぜ”併用したの？
4. “どうやって”併用したの？
5. **併用して”どうだった”の？**
6. 終わりに

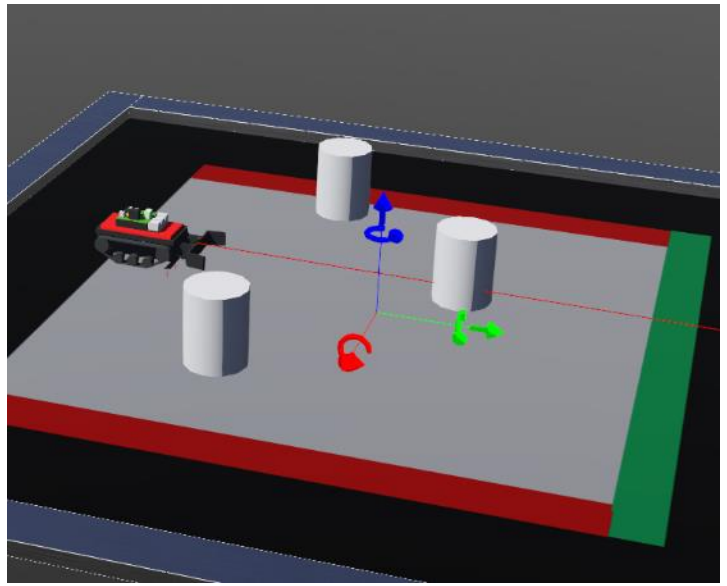
LED-Camp10概要

- 8/29~9/1, 下呂温泉木曽屋で実施
- 参加者は9名
  - 社会人2名、大学院生2名、学部生1名、高専生4名
  - 開発チームは3人/チーム×3チーム
- 開発時間は約9時間（90分のスプリント×6）
- Tankで円柱を押し出す競技内容



開発してみてどうだった？(1/2)

- 実機→シミュレータ は結構大変
  - 実機にあってシミュレータにない機能の
    代替実現手法など
- 実機・シミュレータのコード共通化はすぐ
  - もともと同一生成コードを使うようにしていたため
- パラメータ調整はかなり大変
  - シミュレータ・実機でとれる値を大体そろえなければならない
  - 筐体の個体差を減らすため、全個体で調整

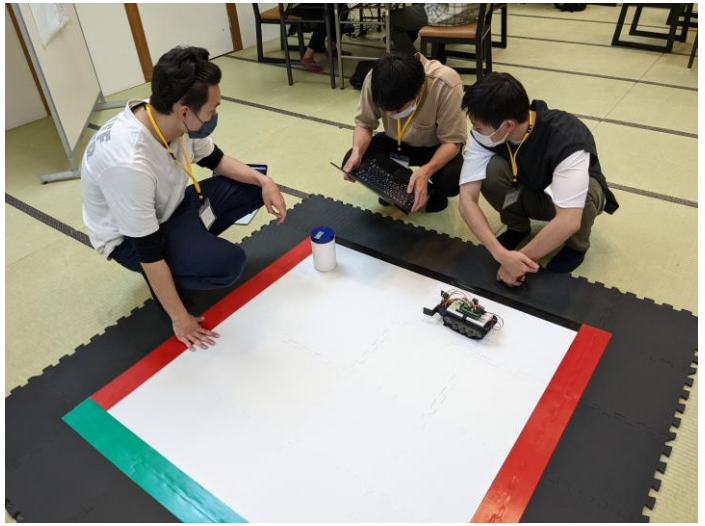


開発してみてどうだった？(2/2)

- 教材開発のため、シミュレータ固有の
  使いかたを習得する必要あり
  （他シミュレータに転用しにくい知識）
  - シミュレータ内のオブジェクト・API
  - 設定パラメータの触り方など
- 「単一のプラットフォーム上で動けばよし」
  から1ステップ進むので、勉強になる
- シミュレータの動作が参加者PC依存なので
  少しハラハラする

実行委員から見た参加者の様子(1/5)

- 実機・シミュレータの使いかたについてトラブルなし
  - 生成コードを書き換える必要がなかったため
  - シミュレータでの実行手順が易しかったため
  - 参加者いわく、情報過多にはなっていなかった
- シミュレータの動作もあまりもっさりせず
  - もともと、比較的軽量
  - V2022aでめっちゃ軽くなった？



実行委員から見た参加者の様子(2/5)

- 序盤、中盤、終盤で活用度合いが異なった
  - 序盤：ほとんど使わず（実機の特性把握に注力）
  - 中盤：かなり使ってもらっていた
    （各人の要素的な開発、
    コンパイル・動作検証など）
  - 終盤：やや使ってもらっていた
    （同上、ただ実機直接も）
    終盤のコンパイルエラーはない様子

実行委員から見た参加者の様子(3/5)

ボトルネック解消は**ある程度**できていたといえそう

- 筐体の側面
  - それぞれがそれぞれの開発
  - 1人がシミュレータでコード整備、2人が実機で動作検証
- 場所の側面
  - フィールド待ちの間にシミュレータで動かしつつ全員で議論
  - とはいえ終盤はフィールドが予約でいっぱいに

実行委員から見た参加者の様子(4/5)

- シミュレータは割り切って使ってもらっていた（はず）
  - 序盤・中盤は“シミュレータで実装、のちに実機で検証”の方針も
  - 実機に比べてシミュレータは精度がよすぎる、という理解
  - 「待ち時間中の作戦会議＆理想的な動作の正当性の検証」としてのシミュレータ

実行委員から見た参加者の様子(5/5)

シミュレータ併用による効果？

例年に比べて

- 「終盤でコンパイルエラーに焦る」が少ない
- イージーミスによる想定外の動作が少ない
- 競技会はどのチームも高得点
- 情報共有会が盛り上がっていた

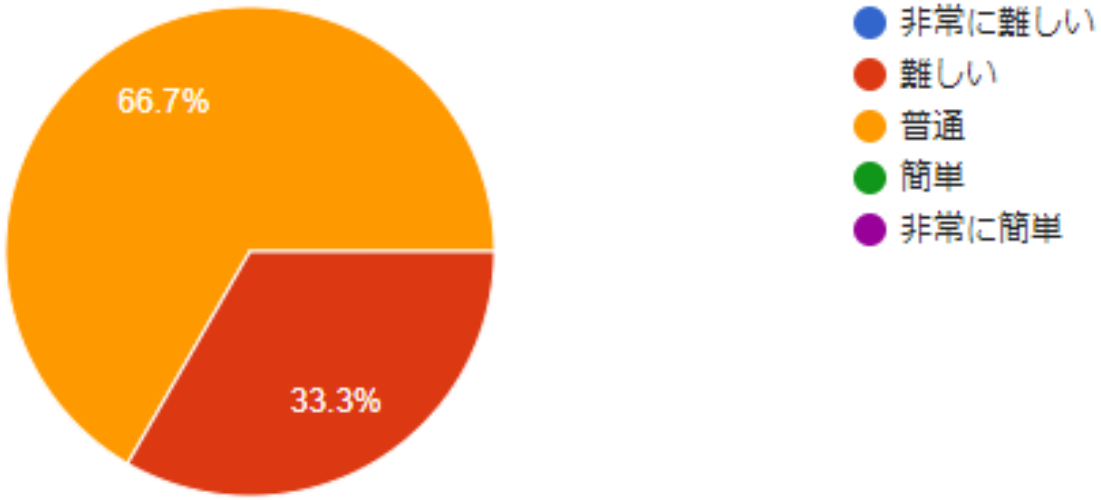

チーム開発実習　タイムテーブル&役割
Doing

参加者の反応は？

「チーム開発実習」の開発教材 、開発課題の難易度はいかがでしたか？

コピー

9 件の回答

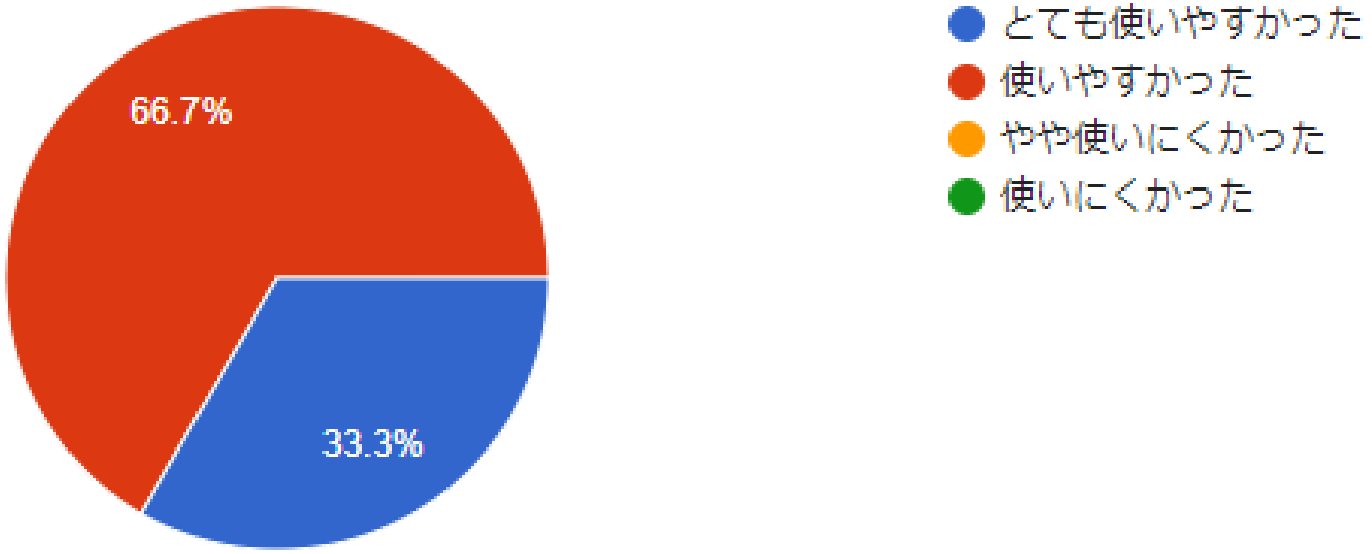


参加者の反応は？

2日間の「チーム開発実習」のなかでLED-Tankは扱いやすかったですか？

コピー

9 件の回答

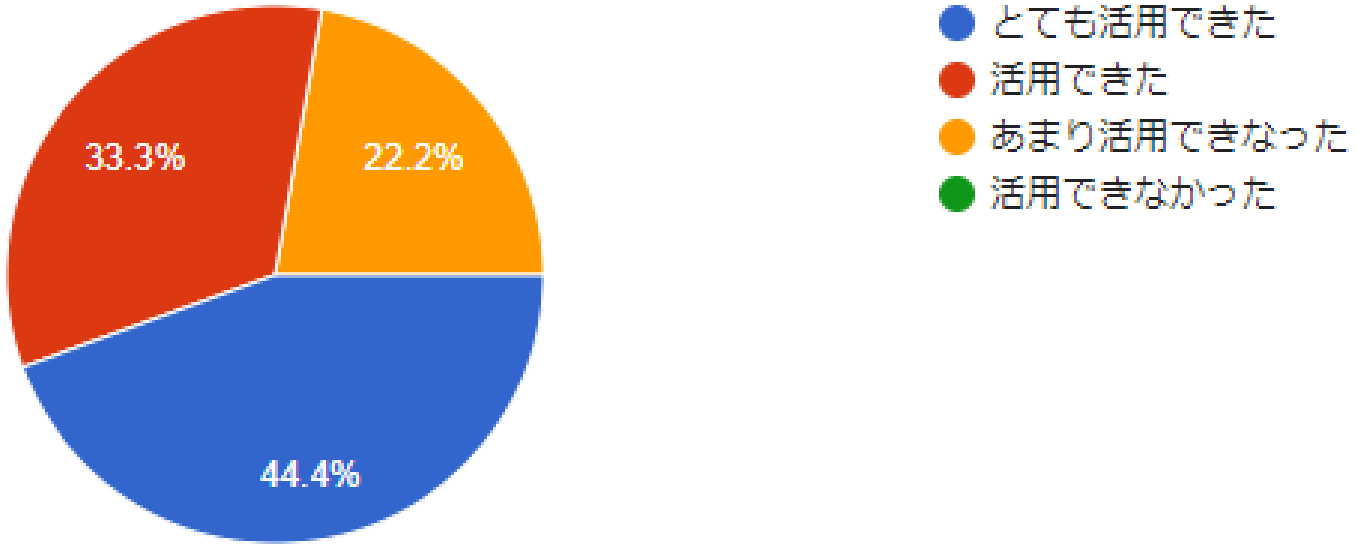


参加者の反応は？

2日間の 「チーム開発実習」のなかでシミュレータは活用できましたか？

コピー

9 件の回答



参加者の反応は？

シミュレータについて良かった点、改善点があればご記入ください。

2 件の回答

実機とシミュレーションでは条件により挙動が違ったため、シミュレーションを上手く使うのが難しかった。

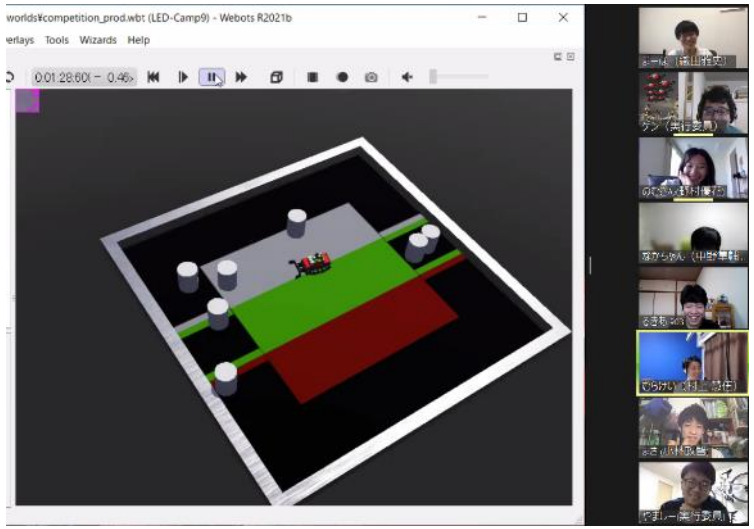
シミュレータがあることで作戦自体の検証が容易にできるおかげで、実機での確認の時は実機とシミュレータの違いに要点を絞って確認がすることができたところがよかった。

発表目次

1. LED-Campってどんなイベント？
2. 実機TankとシミュレータTank
3. “なぜ”併用したの？
4. “どうやって”併用したの？
5. 併用して”どうだった”の？
6. 終わりに

発表まとめ

- ロボットシミュレータと実機を併用したハイブリッド形式の開発研修やってみた
- 開発の加速と気づきの獲得（と、実行委員の興味）が目的
- 内製API以下の差し替えにより、UML→コード生成部分を流用
- 概ね好評だった一方、改善の余地はアリ
- **実行委員大募集中！！！！**

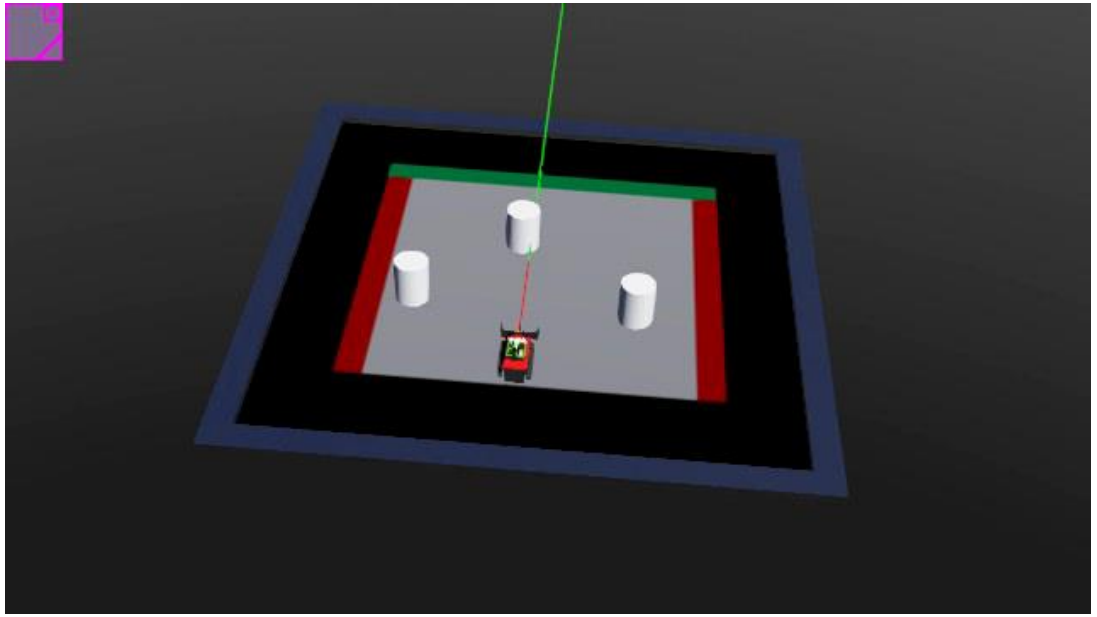


ここからは・・・？

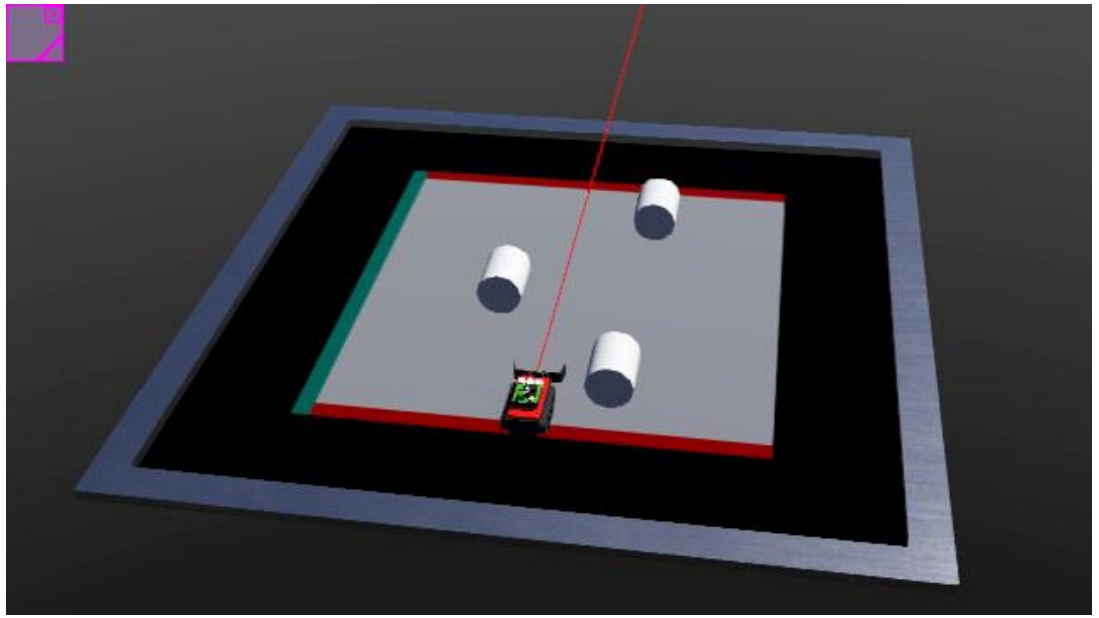
開発形式の研修（ハイブリッド形式含む）について
語りあいましょう！

- デモ・ハンズオンの延長戦
- LED-Camp10開催してみてどうだったの？聞きたい！
- ハイブリッド形式の研修をやってみようと思うけど・・・
- ハイブリッド形式の研修やってみたらこうだった！
- こんなハイブリッド形式の研修があったよ！

おまけ

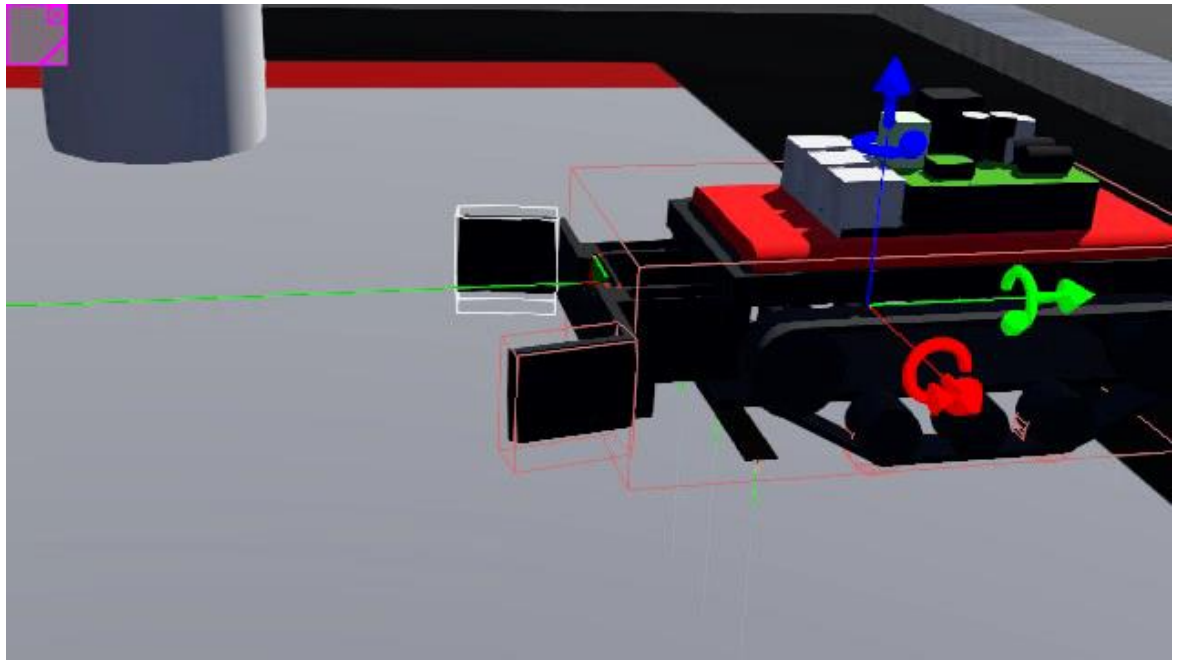


Ver. 2021.b

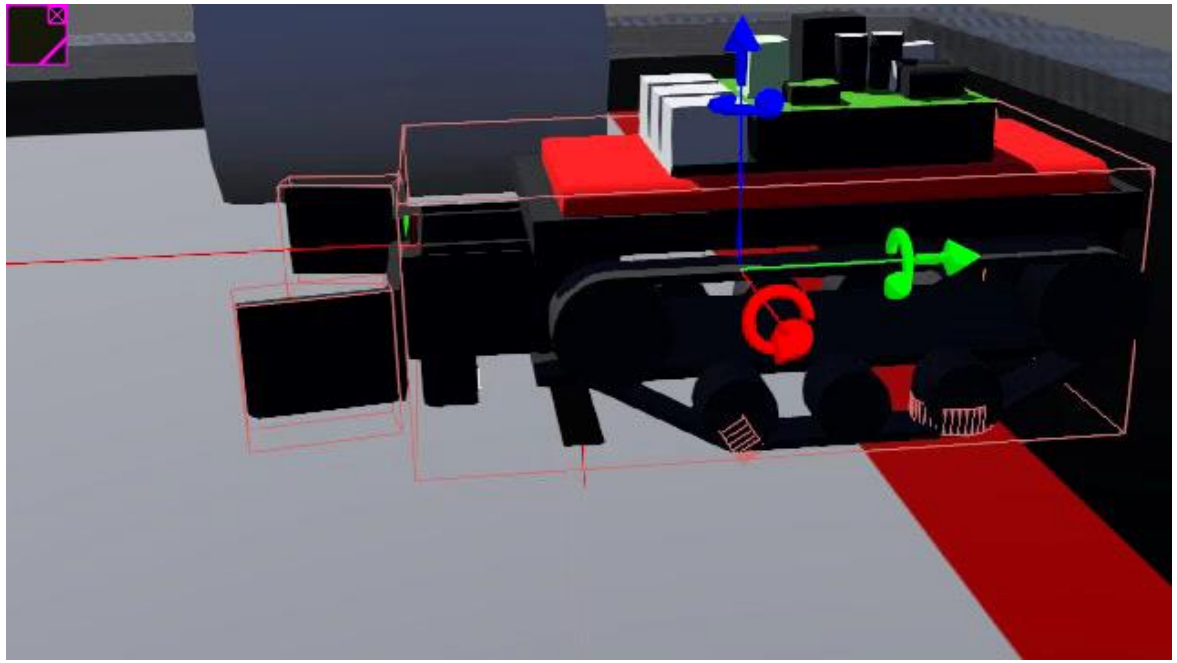


Ver. 2022.a

原因は・・・？



Ver. 2021.b



Ver. 2022.a