TIER IV

09 / 25 / 2024

ROSCon JP 2024

自動運転AIチャレンジの紹介とマルチコンテナ開発・運用の知見公開

近藤 豊 <yutaka.kondo@tier4.jp>
田中 大貴　浅部 佑　高木 勇武

近藤 豊 aka @youtalk

仕事

- 2024/04~ TIER IVエンジニア
- 2021/11~ Preferred Robotics EM & PjM
- 2018/05~ Preferred Networksエンジニア
- 2013/04~ カワダロボティクス主任
- 2013/03 博士（工学）

個人活動

- 改訂新版「ROS 2ではじめよう
  次世代ロボットプログラミング」
- ROSConプログラム委員
- ROSCon JP実行委員
- ROS Japan Users Group元主宰
- NVIDIA Jetson Influencer
- 多摩美TCL5期修了



自動運転ソフトウェア
Autoware

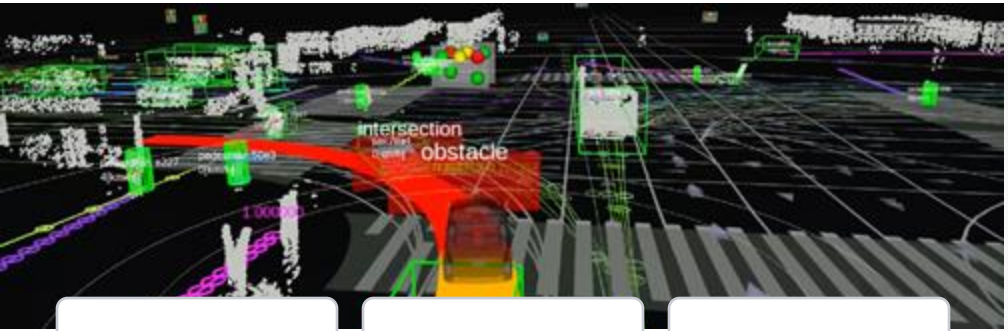
01

Autoware

- AutowareはLinuxとROS 2をベースとした世界初の自動運転OSS
- グローバル規模の業界団体The Autoware Foundationが開発主導

https://github.com/autowarefoundation/autoware

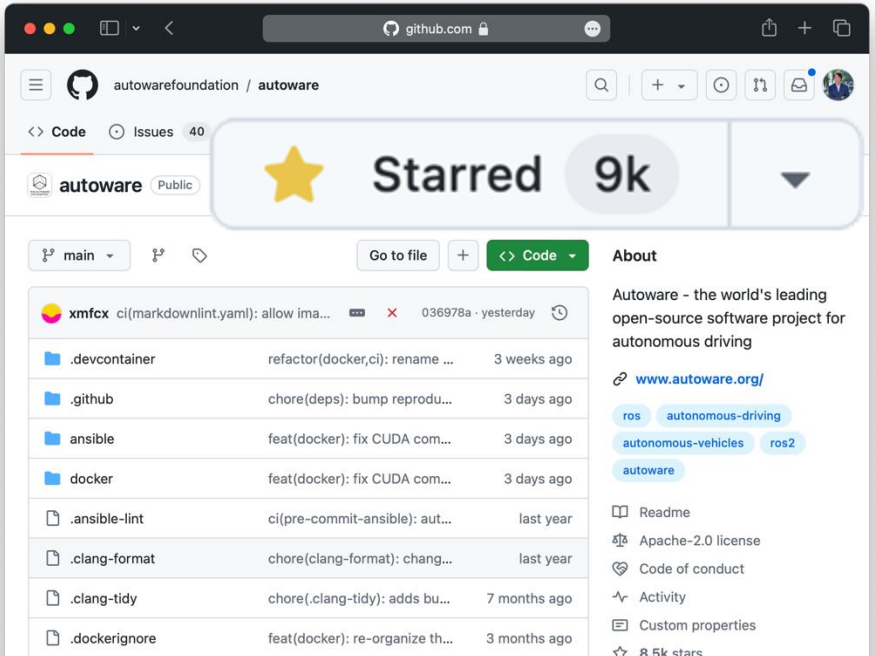
2024年9月現在



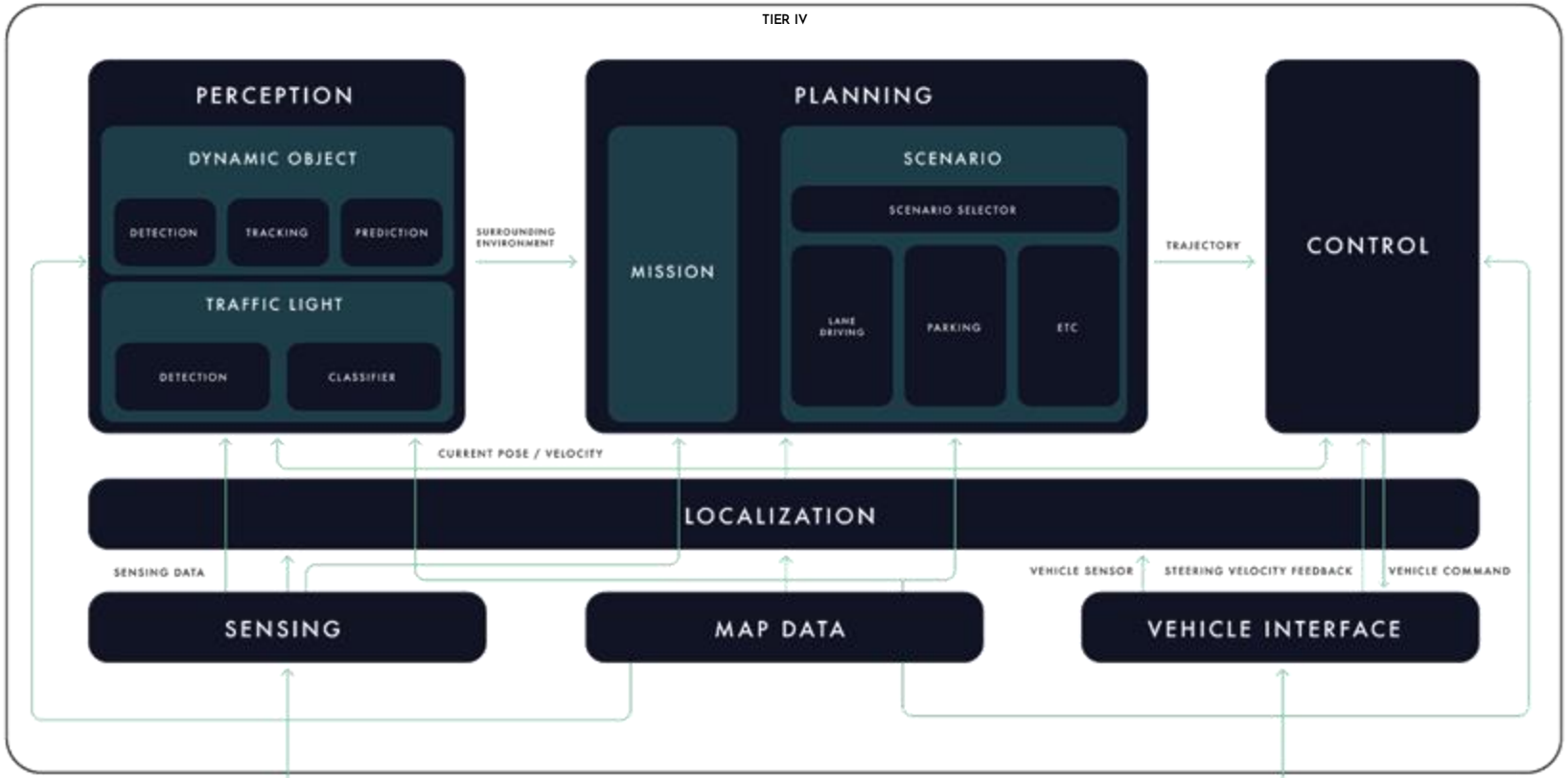
30+
VEHICLES

20+
COUNTRIES

500+
COMPANIES



石川県小松市で自動運転バスを通年運行中



マイクロオートノミー
アーキテクチャ

Autowareの特殊性

10万行以上
総ソースコード行数

400以上
総パッケージ数

AMD64 / ARM64
CPUアーキテクチャ

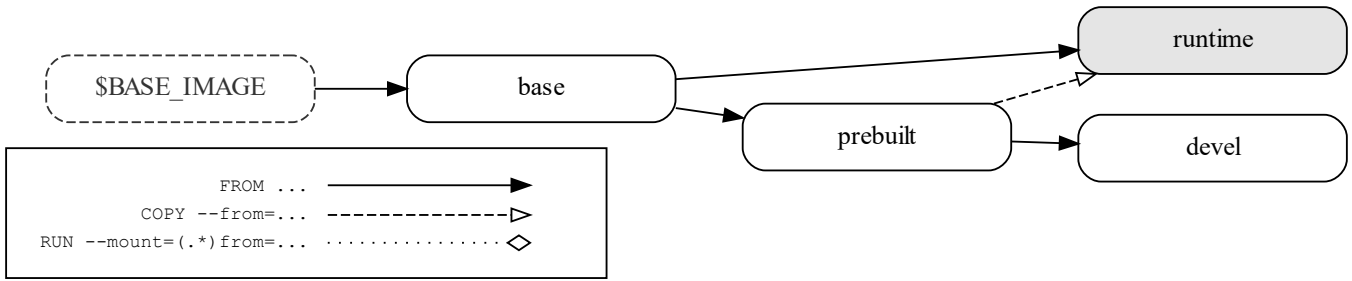
CUDA有無
GPU構成

Apache 2
ソフトウェアライセンス

Autowareの
マルチコンテナ開発・運用

02

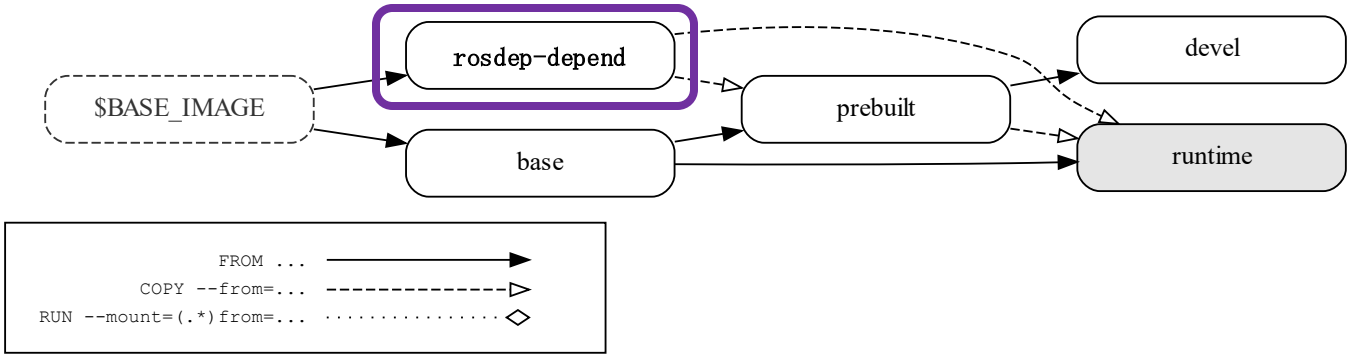
シングルステージビルドからマルチステージビルドへ



https://github.com/autowarefoundation/autoware/blob/main/docker/README.md

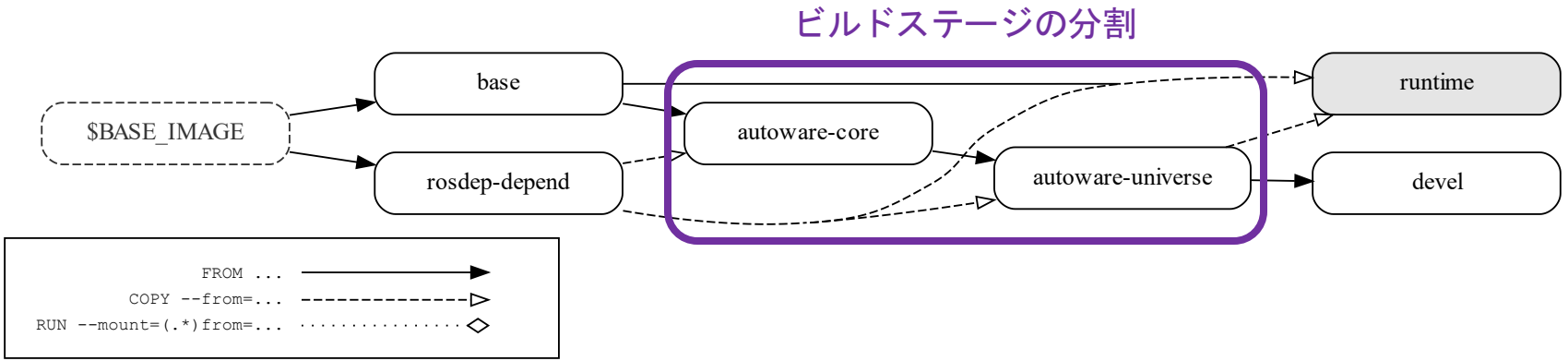
シングルステージビルドからマルチステージビルドへ

rosdep installの
依存パッケージリスト生成



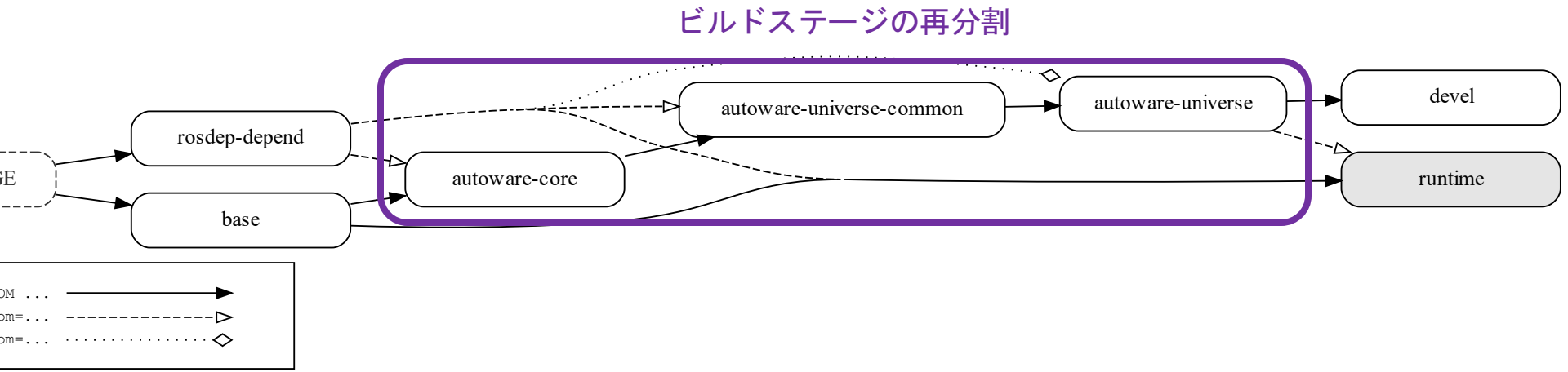
https://github.com/autowarefoundation/autoware/blob/main/docker/README.md

シングルステージビルドからマルチステージビルドへ



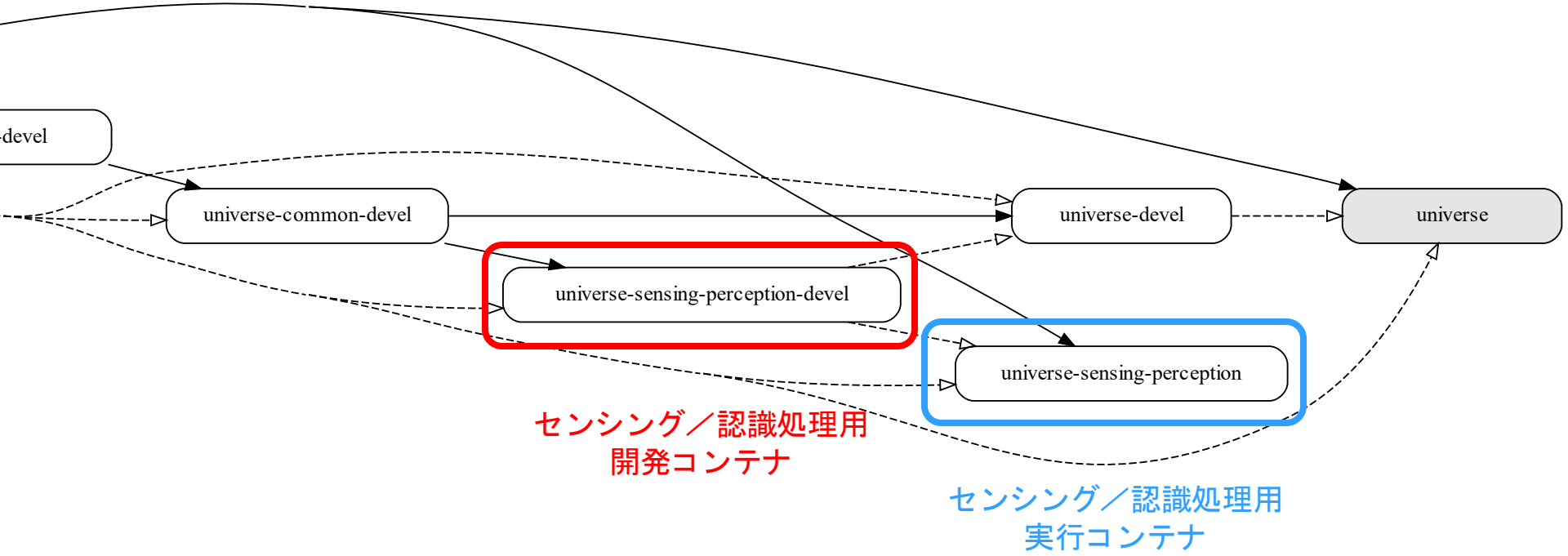
https://github.com/autowarefoundation/autoware/blob/main/docker/README.md

シングルステージビルドからマルチステージビルドへ



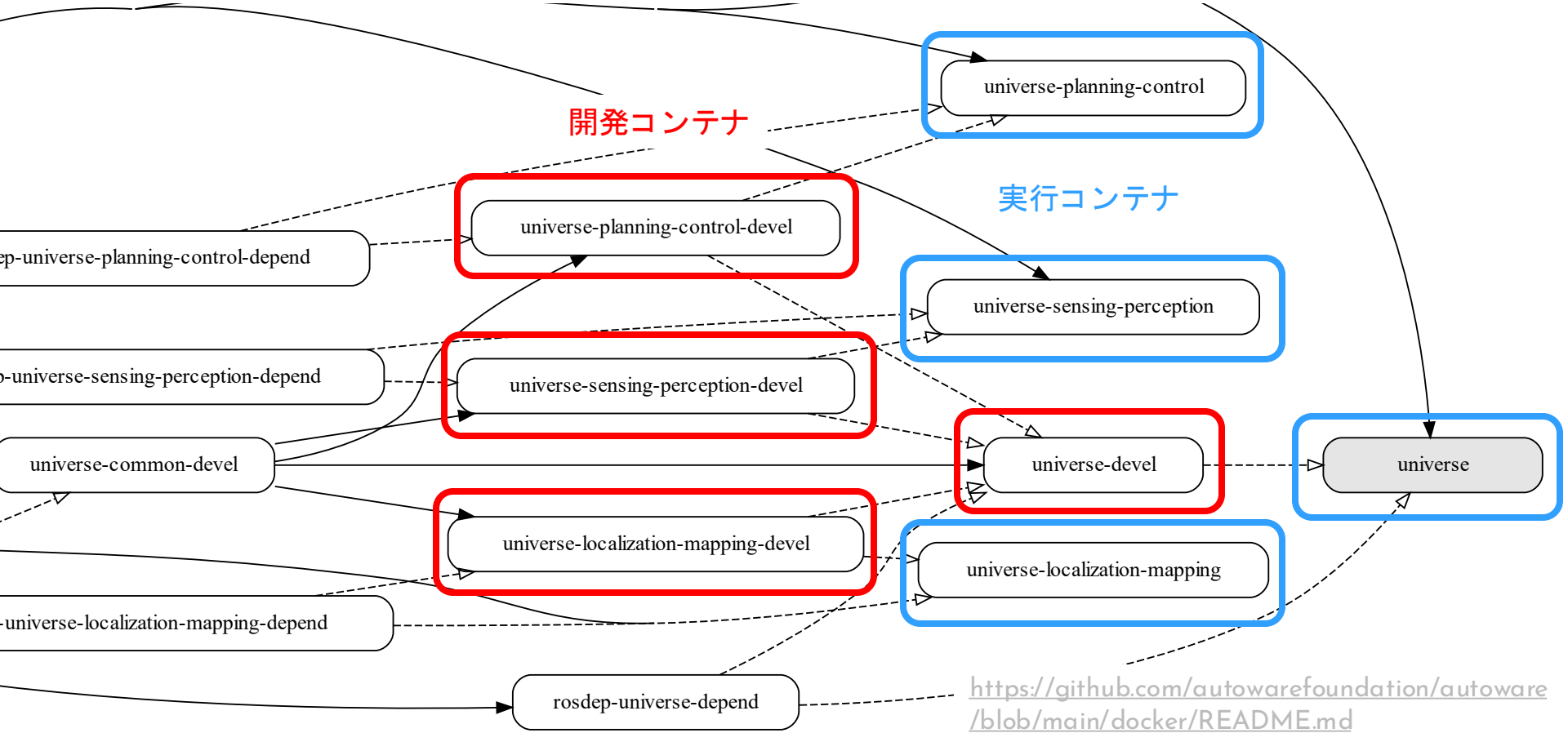
https://github.com/autowarefoundation/autoware/blob/main/docker/README.md

シングルステージビルドからマルチステージビルドへ



https://github.com/autowarefoundation/autoware/blob/main/docker/README.md

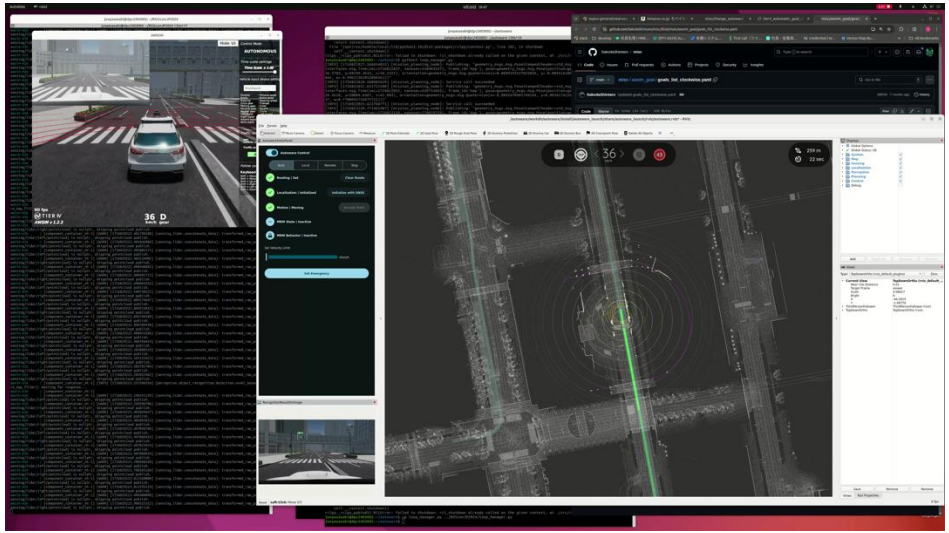
シングルステージビルドからマルチステージビルドへ



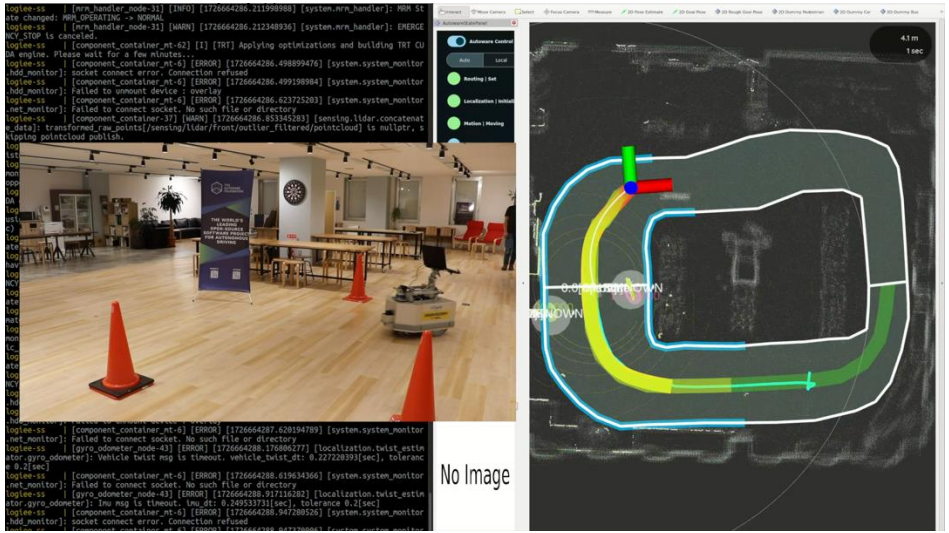
https://github.com/autowarefoundation/autoware/blob/main/docker/README.md

開発コンテナと実行コンテナ

詳しくはティアフォー展示ブースへGO!



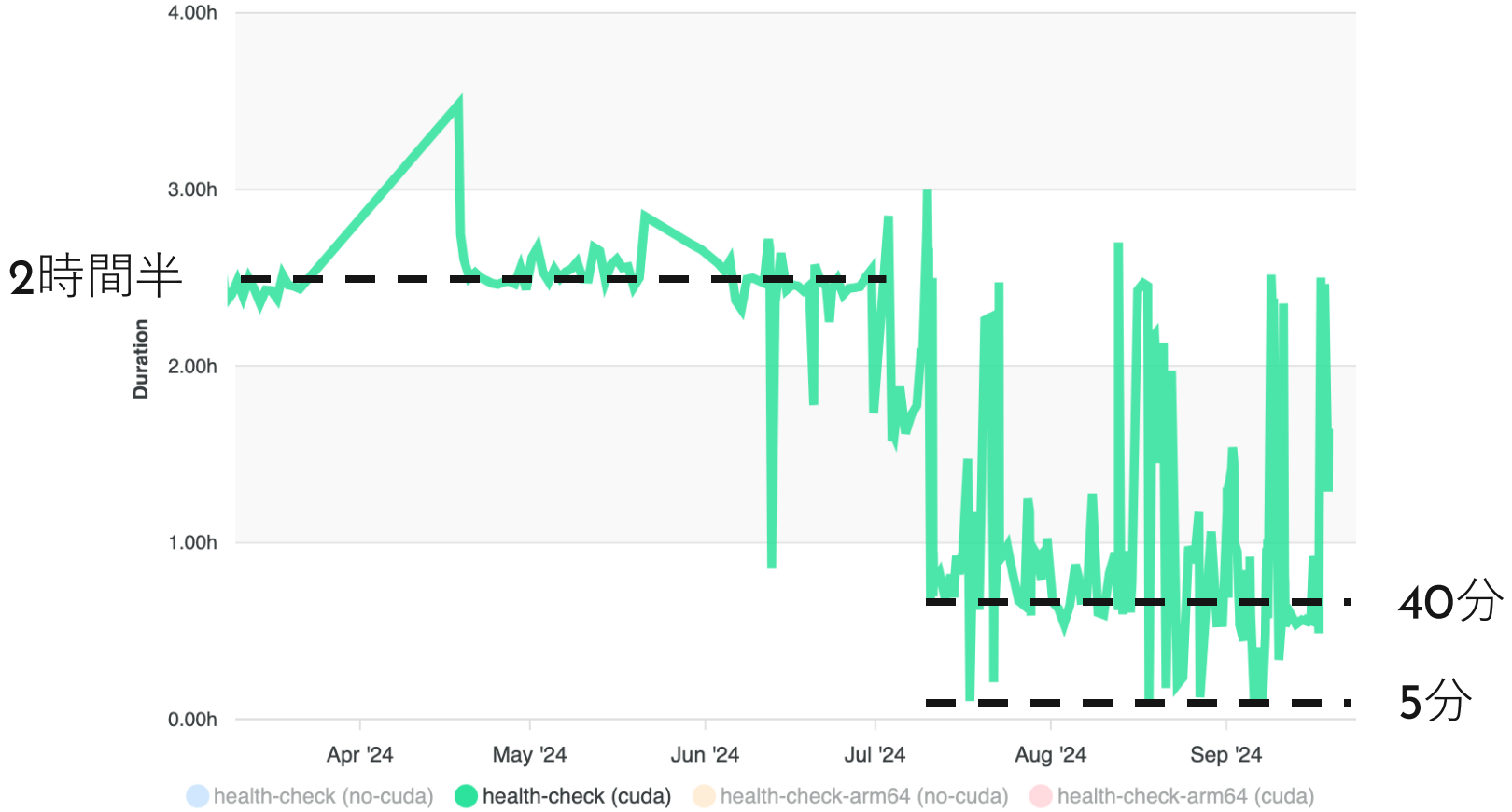
開発コンテナによるデバッグ



実行コンテナによる実車テスト

https://autowarefoundation.github.io/autoware-ci-metrics/

GitHub Actionsでのコンテナビルド時間



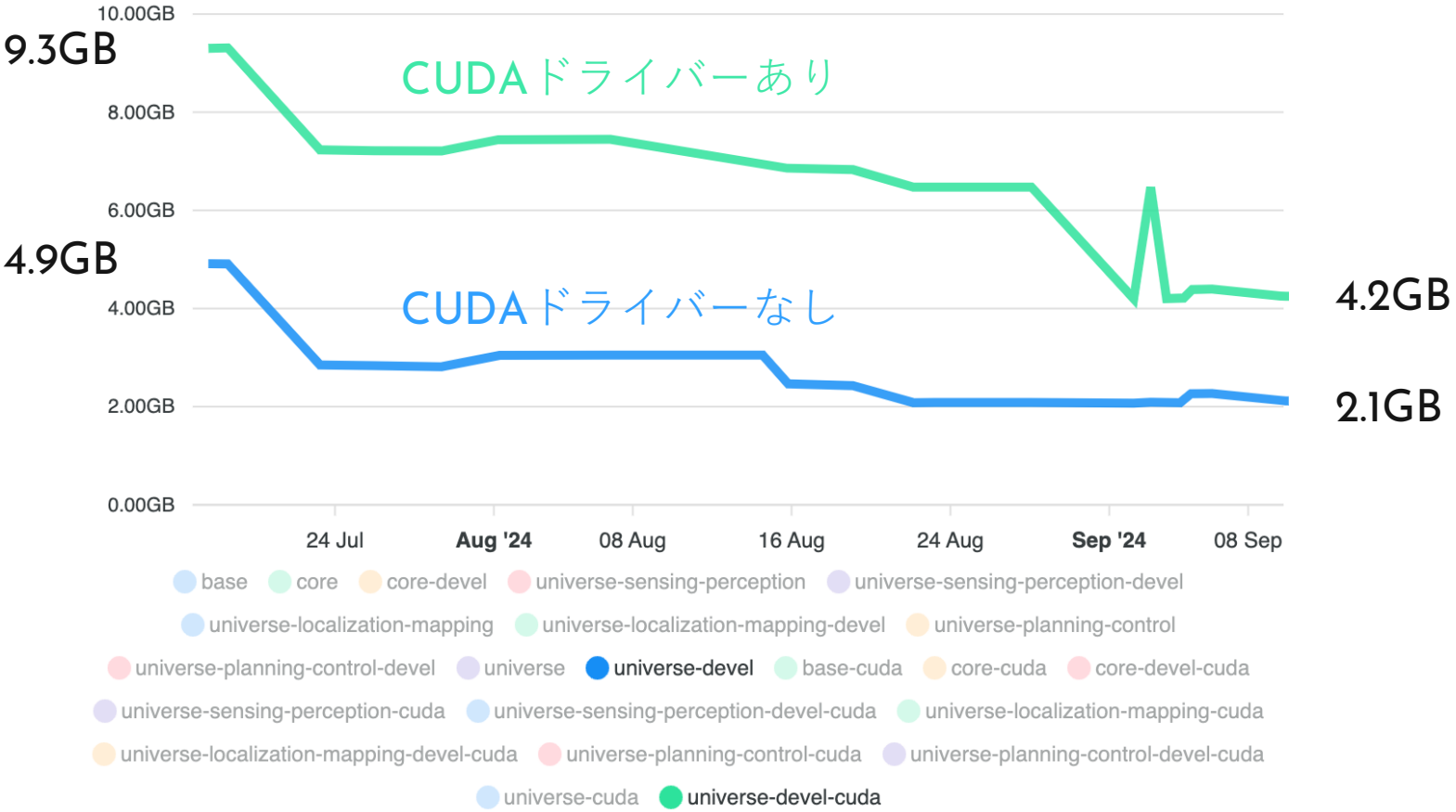
2時間半

40分

5分

https://autowarefoundation.github.io/autoware-ci-metrics/

Autowareコンテナイメージサイズ

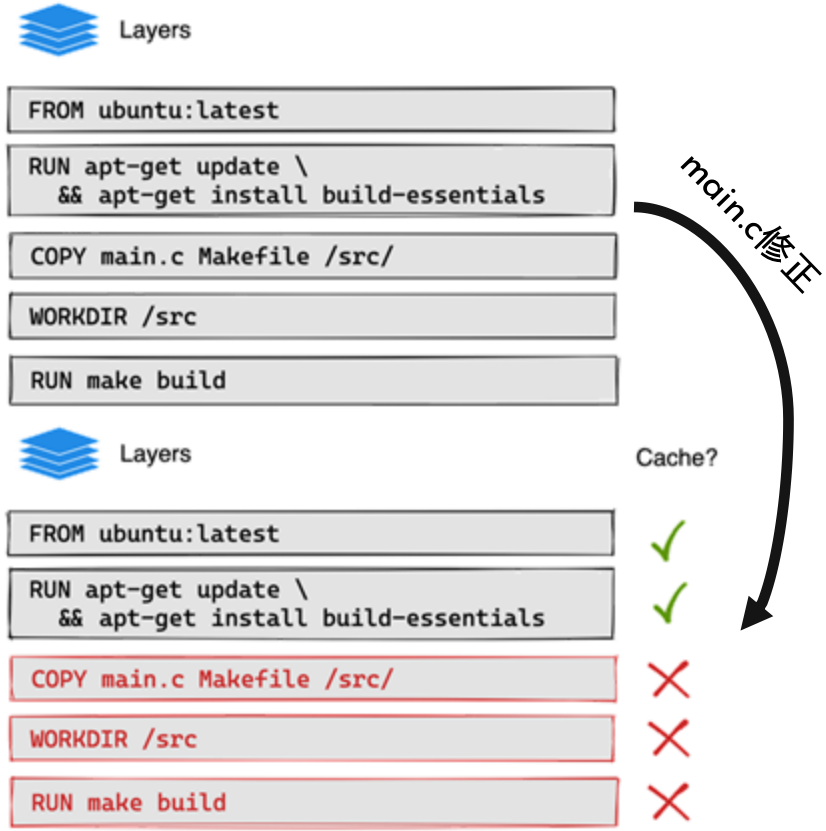


GitHub Actionsを使ったコンテナビルドTips

03

GitHub Actions上でのDocker Build Cacheの有効化

- https://docs.docker.com/build/cache/backends/registry/
- コンテナビルドのビルドキャッシュをレジストリにキャッシュ
- cache-fromはmainブランチとPRブランチの両方を指定可能
- cache-toはmode=maxを指定して全ステージキャッシュ



https://docs.docker.com/build/cache/ より転載

GitHub Actions CacheへのCcache保存

- https://github.com/reproducible-containers/buildkit-cache-dance
- GitHub Runnerでイメージ内のファイルをキャッシュ抽出・復元
- Autowareビルドのように RUNコマンド単体で長時間実行が必要な場合に有用

```yaml
- name: Cache ccache
  uses: actions/cache@v4
  if: ${{ inputs.name == 'no-cuda' && github.ref == 'refs/heads/main' }}
  id: cache-ccache
  with:
    path: |
      root-ccache
    key: ccache-${{ inputs.platform }}-${{ inputs.name }}-${{ hashFiles('src/**/*.cpp') }}
    restore-keys: |
      ccache-${{ inputs.platform }}-${{ inputs.name }}-
      ccache-${{ inputs.platform }}-
```

```yaml
- name: Inject cache into docker
  uses: reproducible-containers/buildkit-cache-dance@v3.1.2
  with:
    cache-map: |
      {
        "root-ccache": "/root/.ccache",
        "var-cache-apt": "/var/cache/apt"
      }
    skip-extraction: true
```

https://github.com/autowarefoundation/autoware/blob/main/.github/actions/docker-build-and-push/action.yaml

rosdepのキャッシュ保持力向上

- 独立したステージでrosdep installする依存パッケージリストをあらかじめ生成
- colcon buildステージで依存パッケージリストのみコピーしてインストール
- さらにcacheマウントでAPTのダウンロード済みパッケージを再利用

依存パッケージリスト
生成ステージ

```
COPY src/core /autoware/src/core
RUN rosdep update && rosdep keys --ignore-src --from-paths src \
    | xargs rosdep resolve --rosdistro ${ROS_DISTRO} \
    | grep -v '^#' \
    | sed 's/ \+/\n/g'\
    | sort \
    > /rosdep-core-depend-packages.txt \
    && cat /rosdep-core-depend-packages.txt
```

colcon buildステージ

```
COPY --from=rosdep-depend /rosdep-core-depend-packages.txt /tmp/rosdep-core-depend-packages.txt
# hadolint ignore=SC2002
RUN --mount=type=cache,target=/var/cache/apt,sharing=locked \
  apt-get update \
  && cat /tmp/rosdep-core-depend-packages.txt | xargs apt-get install -y --no-install-recommends \
  && apt-get autoremove -y && rm -rf "$HOME"/.cache
```

https://github.com/autowarefoundation/autoware/blob/main/docker/Dockerfile

ソースコードのbindマウントとbuildディレクトリ削除

- ソースコードは各ステージに必要な分だけbindマウント
- インストール先を/opt/autowareに変更し、実行コンテナにコピー
- 中間生成物（/autoware/build）はその場で削除

https://github.com/autowarefoundation/autoware/blob/main/docker/Dockerfile

開発コンテナのビルドステージ

```
RUN --mount=type=cache,target=${CCACHE_DIR} \
  --mount=type=bind,from=rosdep-depend,source=/autoware/src/core,target=/autoware/src/core \
  source /opt/ros/"$ROS_DISTRO"/setup.bash \
  && du -sh ${CCACHE_DIR} && ccache -s \
  && colcon build --cmake-args \
    " -Wno-dev" \
    " --no-warn-unused-cli" \
    --merge-install \
    --install-base /opt/autoware \
    --mixin release compile-commands ccache \
  && du -sh ${CCACHE_DIR} && ccache -s \
  && rm -rf /autoware/build
```

実行コンテナのビルドステージ

```
COPY --from=universe-sensing-perception-devel /opt/autoware /opt/autoware
COPY --from=universe-localization-mapping-devel /opt/autoware /opt/autoware
COPY --from=universe-planning-control-devel /opt/autoware /opt/autoware
```

最小サイズの実行コンテナ化

- exec_dependの依存パッケージのみインストール

```
RUN rosdep keys --dependency-types=exec --ignore-src --from-paths src \
    | xargs rosdep resolve --rosdistro ${ROS_DISTRO} \
    | grep -v '^#' \
    | sed 's/ \+/\n/g'\
    | sort \
    > /rosdep-exec-depend-packages.txt \
    && cat /rosdep-exec-depend-packages.txt
```

- ヘッダーファイルや不要な開発用ファイルを根こそぎ削除

```
&& find /usr/lib/$LIB_DIR-linux-gnu -name "*.a" -type f -delete \
&& find / -name "*.o" -type f -delete \
&& find / -name "*.h" -type f -delete \
&& find / -name "*.hpp" -type f -delete \
&& rm -rf /autoware/ansible /autoware/ansible-galaxy-requirements.yaml /autoware/setup-dev
  /root/.local/pipx /opt/ros/"$ROS_DISTRO"/include /etc/apt/sources.list.d/cuda*.list \
  /etc/apt/sources.list.d/docker.list /etc/apt/sources.list.d/nvidia-docker.list \
  /usr/include /usr/share/doc /usr/lib/gcc /usr/lib/jvm /usr/lib/llvm*
```

- 開発コンテナのビルドステージから最終成果物のみコピー

```
COPY --from=universe-devel /opt/autoware /opt/autoware
```

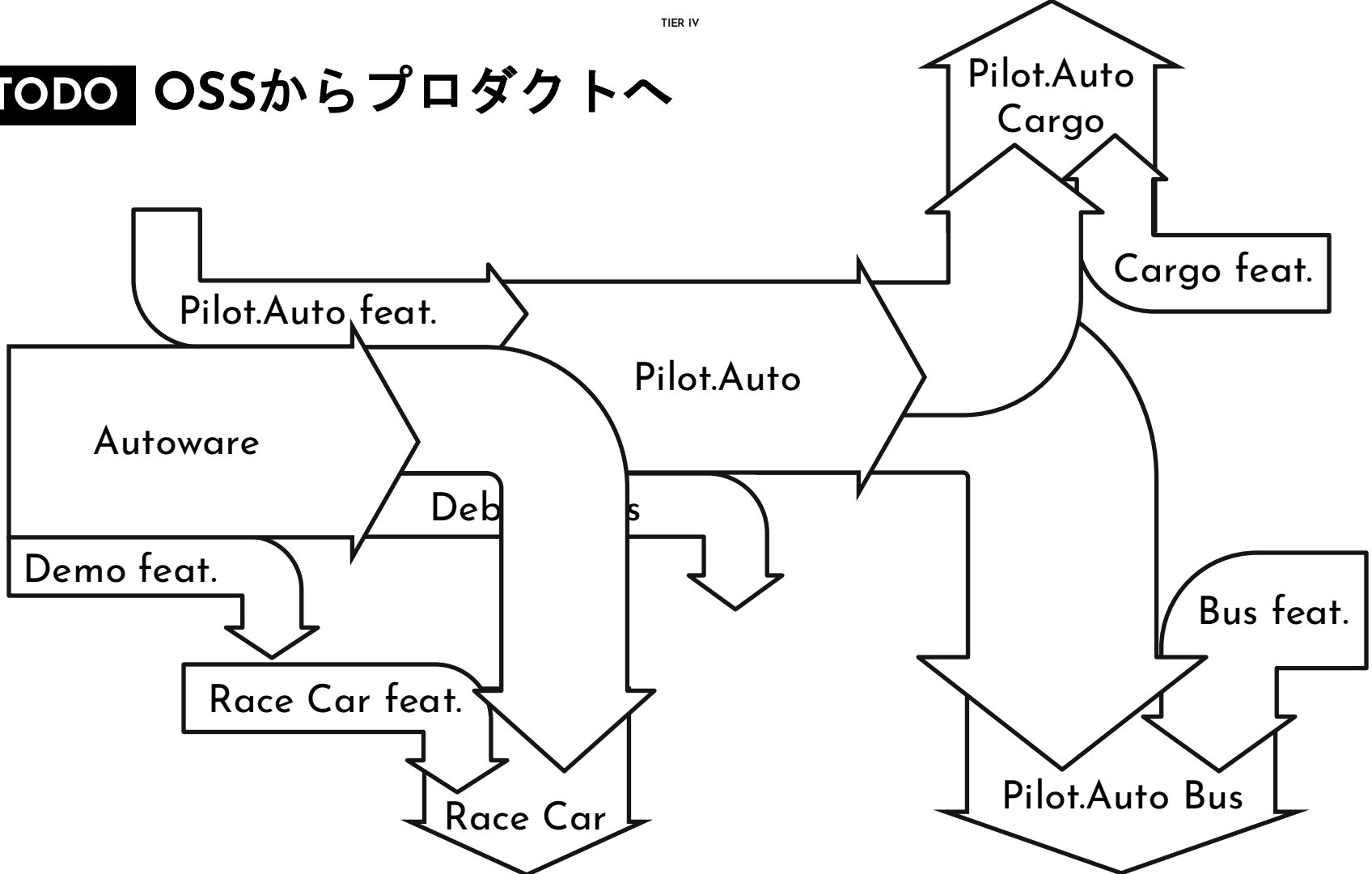
開発コンテナを使ったAutowareパッケージ開発

- Dockerコマンドを使った方法

```
$ git clone git@github.com:autowarefoundation/autoware.git
$ cd autoware
$ vcs import src < autoware.repos
$ docker run -it --rm \
        -v $PWD/src/universe/autoware.universe/XXX:/autoware/src \
        ghcr.io/autowarefoundation/autoware:universe-devel-cuda
$ colcon build --mixin debug compile-commands
$ source install/setup.bash
$ ros2 run --prefix "gdb -ex run --args" autoware_YYY ZZZ
```

- VSCode/devcontainerを使った方法：TBA

TODO OSSからプロダクトへ



自動運転AIチャレンジ2024

04

自動運転AIチャレンジ2024

今年度は約半年にわたる期間にて実施
技術のコンペティションとエンターテインメント性の双方を重視した大会を開催

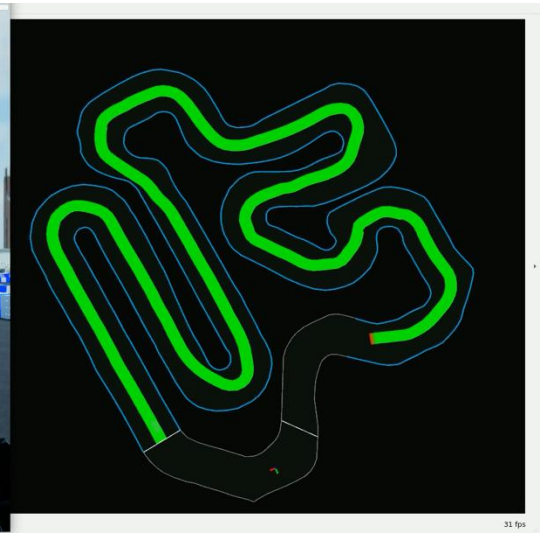
- **決勝会場：シティサーキット東京ベイ（運営：株式会社トムス）**
- **ハードウェア：EVレーシングカート**
- **参加方式：チャレンジクラスとアドバンストクラス**

https://youtu.be/GsuCUoNrMDM より転載

予選（チャレンジクラスのみ）

デジタルツイン指向のコースを用いたシミュレーションで実施

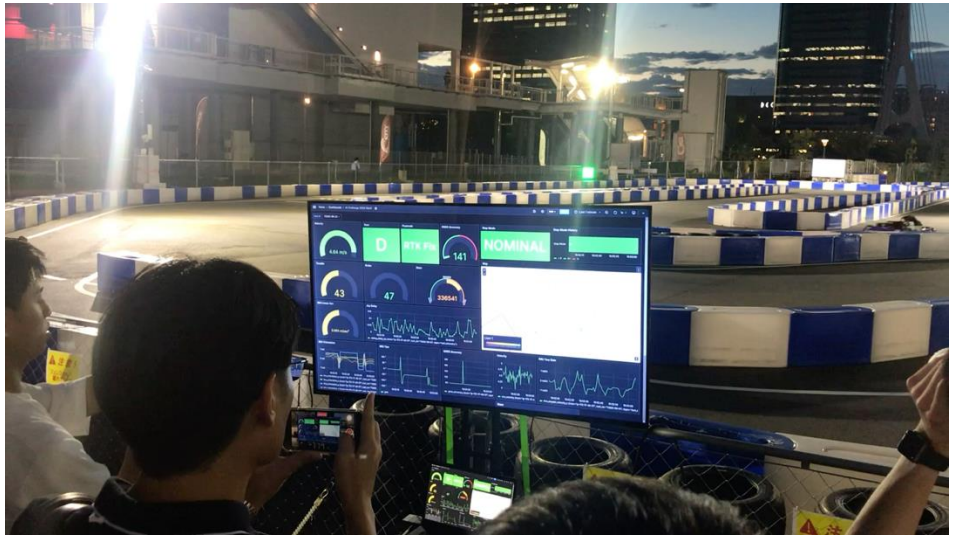
https://www.youtube.com/watch?v=K-N-2nLXXF8 より転載



準決勝・決勝

大人向け屋外EVレーシングカートを自動運転対応に特別改造した実車両で実施

https://youtu.be/COZDHMm4E_8 より転載



TIER IV

09 / 25 / 2024

ROSCon JP 2024

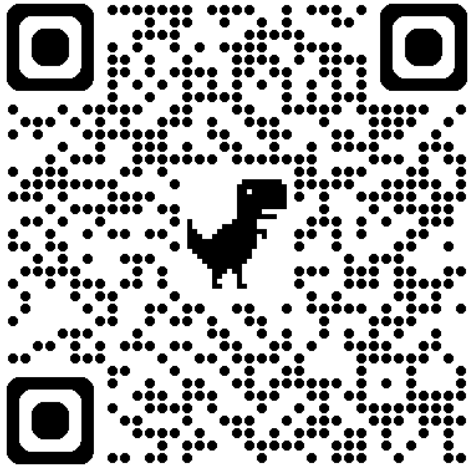
自動運転AIチャレンジの紹介とマルチコンテナ開発・運用の知見公開

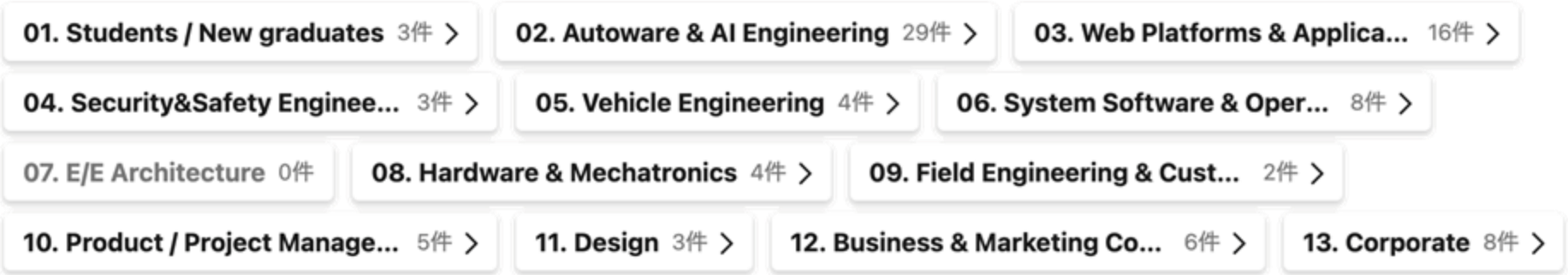
近藤 豊 <yutaka.kondo@tier4.jp>
田中 大貴　浅部 佑　高木 勇武

We Are Hiring!

- 自動運転の民主化はあなたの手で
- 特に自動運転ソフトウェア、Web、クラウド、シミュレーションエンジニア求む

https://herp.careers/v1/tier4





2024年9月現在

改訂新版ROS 2ではじめよう 次世代ロボットプログラミング

- 9/19発売
- 初版から5年ぶりの全面改訂新版
- 5年サポートのROS 2 Jazzy完全対応
- ROS 1紹介、ROS 2移行の章を削除
- ROS 2ツール／パッケージ章の完全書き換え
- ROS 2エコシステム章の追加
- 実践ROS 2プログラミング章の追加