POLITECHNIKA RZESZOWSKA
im. IGNACEGO ŁUKASIEWICZA

KATEDRA INFORMATYKI I AUTOMATYKI
POLITECHNIKI RZESZOWSKIEJ

Cyfrowy bliźniak, ROS 2 i teleoperacja XR

WiRR · Wykład 07 · Automatyka i Robotyka · 3 ECTS · 2026/2027 · v1.3.2

dr inż. Mateusz Pomianek

Katedra Informatyki i Automatyki, Politechnika Rzeszowska

27.10.2026

Cel: pokazać aktualny stan, nie tylko ruchomy model



Model XR pokazuje tylko poprawnie zweryfikowany stan

Rozpatrujemy podgląd przegubów robota w XR, bez sterowania sprzętem.

- Łączymy węzły i wiadomości z jawną semantyką.
- Rozróżniamy czas pomiaru, odbioru i prezentacji.
- Oznaczamy brak danych oraz blokujemy użycie stanu STALE.

Węzły i tematy ROS 2

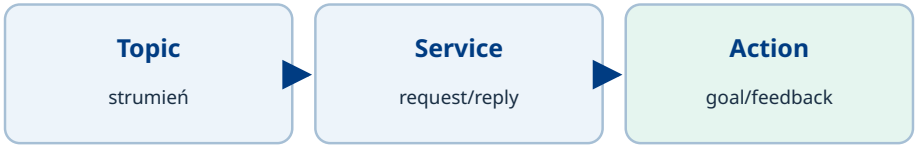


Nadawca i odbiorca pozostają niezależnymi węzłami

Węzły publikują lub subskrybują wiadomości na tematach. Model publish-subscribe rozdziela wytwórcę danych od odbiorcy.

Kompozycja: wiele węzłów może działać w jednym procesie. Węzeł jest jednostką logiczną grafu, nie musi odpowiadać odrębnemu procesowi.

Temat, usługa i akcja mają różną semantykę



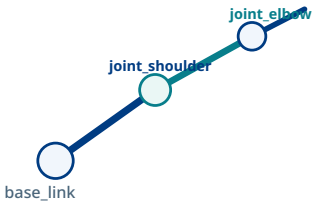
Strumień stanu nie jest poleceniem ruchu

Topic: strumień pomiarów, np. `/joint_states`. **Service:** żądanie i odpowiedź. **Action:** dłuższa operacja z postępem i wynikiem.

Nie traktuj podglądu jako komendy ani przyjęcia komendy jako dowodu wykonania.

JointState: nazwa wiąże wartość z przegubem

JointState → konfiguracja przegubów



Wiadomość: JointState

name[i]	position[i]
joint_shoulder	q_1 [rad]
joint_elbow	q_2 [rad]

Mapowanie po nazwie, nie po indeksie modelu

$$\text{name}[i] \leftrightarrow \text{position}[i]$$

Jednostki: przegub obrotowy [rad], przesuwny [m].

Tablice `position`, `velocity`, `effort` mogą być puste. Do animowania konfiguracji wymagamy **pozycji dla wszystkich wyświetlanych przegubów**.

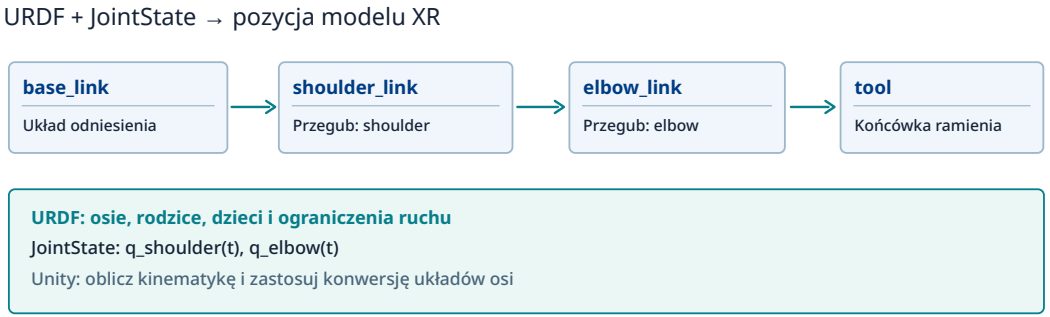
Walidacja JointState przed aktualizacją modelu

Rozdziel poprawność wiadomości od przydatności do wizualizacji.

```
if not joint_arrays_valid(msg.name, msg.position,
                          msg.velocity, msg.effort):
    return  # INVALID
if not has_joint_positions(msg.name, msg.position):
    return  # NO_POSITION
```

Sprawdzamy również zgodność zegara, wiek próbki i obecność przegubów w URDF.

Model robota: URDF i JointState



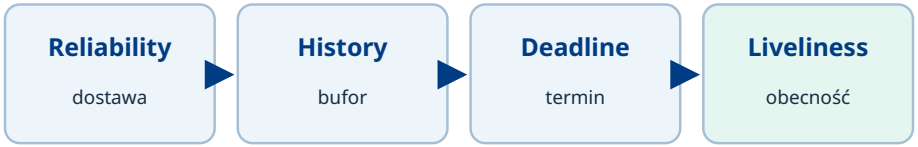
URDF: struktura członów, typy i osie przegubów oraz geometria.

JointState: bieżące wartości `q(t)` identyfikowane nazwami.

$$\mathrm{URDF} + \{q_i(t)\} \rightarrow T_{\mathrm{link}}(t)$$

Przenoszenie do Unity wymaga dodatkowo uzgodnienia układów współrzędnych.

QoS zależy od rodzaju danych



Zgodność QoS poprzedza analizę aktualności

Obrazy, odczyty czujników i polecenia sterujące mogą wymagać różnych gwarancji niezawodności, trwałości i opóźnienia.

Zgodność QoS jest warunkiem połączenia

QoS: zgodność Requested–Offered

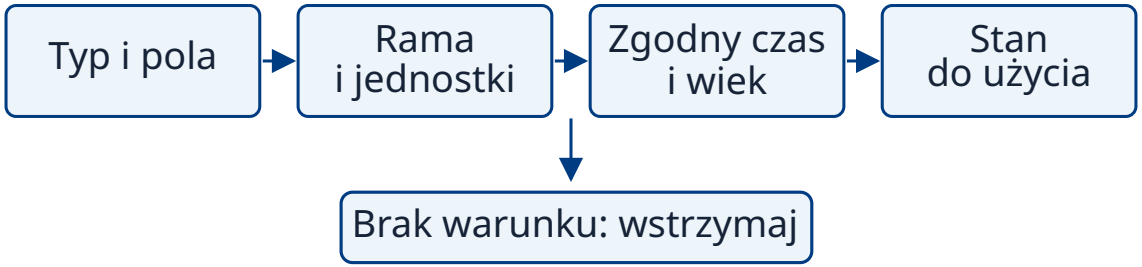
Publisher oferuje ↓	Subscriber: BEST_EFFORT	Subscriber: RELIABLE
BEST_EFFORT	ZGODNE	NIEZGODNE
RELIABLE	ZGODNE	ZGODNE

Zasada Requested–Offered:

- Publisher **BEST_EFFORT** → subscriber **RELIABLE**: niezgodne.
- Publisher **RELIABLE** → subscriber **BEST_EFFORT**: zgodne pod względem reliability.

Sprawdź `ros2 topic info -v /joint_states`; pozostałe polityki QoS również muszą być zgodne.

Reliable nie oznacza aktualny

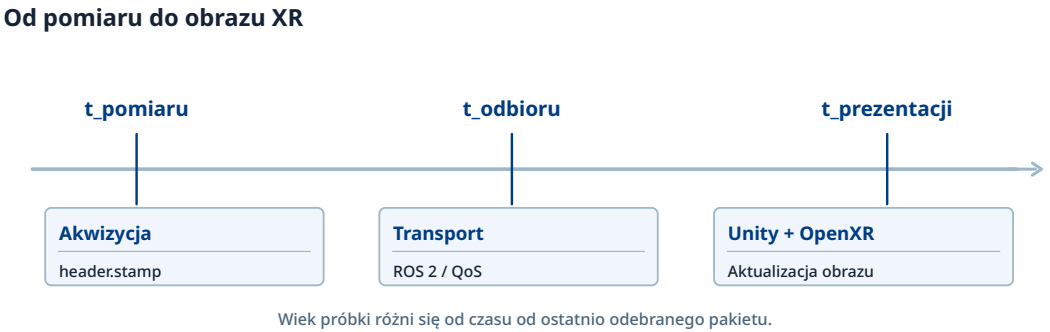


Dostarczenie ≠ aktualność. Kolejka i retransmisja mogą dostarczyć historyczny pomiar.

$$a = t_{\mathrm{ROS\,now}} - t_{\mathrm{header.stamp}}$$

Brak wiadomości przez dłuższy czas wymaga niezależnego **watchdoga**, nawet gdy QoS nie zgłasza błędu.

Znacznik czasu

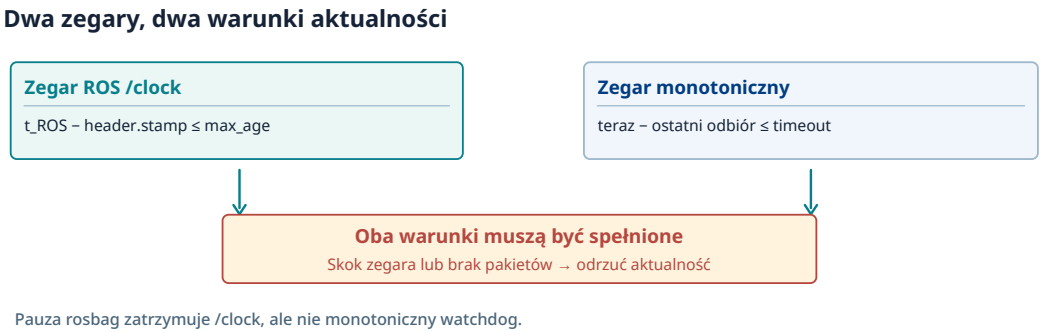


Trzy chwile tego samego przebiegu:

`t_pomiaru` ⟶ `t_odbioru` ⟶ `t_prezentacji_XR`.

Wiek pomiaru liczony od `header.stamp` nie jest tym samym co czas od ostatniego odebrania pakietu.

Jeden zegar przed odejmowaniem znaczników

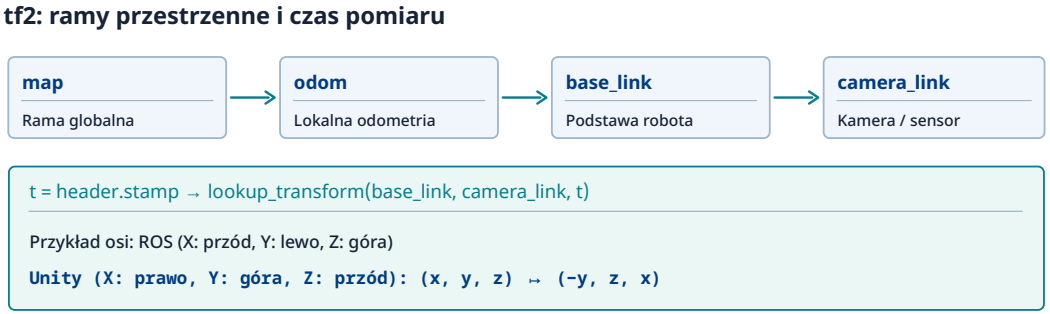


$$a = t_{\text{now}} - t_{\text{stamp}}$$

`use_sim_time` oraz `/clock` muszą określać **ten sam zegar ROS**, w którym zapisano `header.stamp`.

Dodatkowy watchdog: zegar monotoniczny kontroluje czas od nadejścia próbki, także podczas pauzy rosbag.

tf2: transformacje w czasie



Ramy: `map → odom → base_link → camera_link`.

Transformacja `T_{\mathrm{base,cam}}(t)` odnosi się do wybranej chwili `t`.

ROS 2 → Unity: uzgodnij konwencję osi oraz jednostki; sama zmiana nazw ram nie zmienia orientacji przestrzeni.

Transformację pobieramy dla chwili pomiaru

W tf2 pytamy o transformację dla czasu pomiaru:

```
from rclpy.time import Time

stamp = Time.from_msg(msg.header.stamp)
transform = buffer.lookup_transform(
    "base_link", "camera_link", stamp)
```

Brak historii transformacji lub błąd ramy oznacza **brak pozy do wyświetlenia**.

sensor_msgs



Typ, układ współrzędnych i czas muszą być zgodne

`sensor_msgs/msg/JointState`: nazwy i wartości przegubów.

`sensor_msgs/msg/Image`: piksele, format i `header.stamp`.

`sensor_msgs/msg/PointCloud2`: punkty, pola i rama przestrzenna.

Typ wiadomości, rama, jednostki i chwila pomiaru muszą być interpretowane łącznie.

Kod: predykat świeżości bez ROS 2

```python
import math

def stamp_is_fresh(now_s, stamp_s, max_age_s):
    return (all(math.isfinite(v) for v in
                (now_s, stamp_s, max_age_s))
            and now_s > 0 and stamp_s > 0
            and max_age_s > 0
            and 0 <= now_s - stamp_s <= max_age_s)
```

`stamp_is_fresh(10.0, 9.9, 0.2)` → `True`

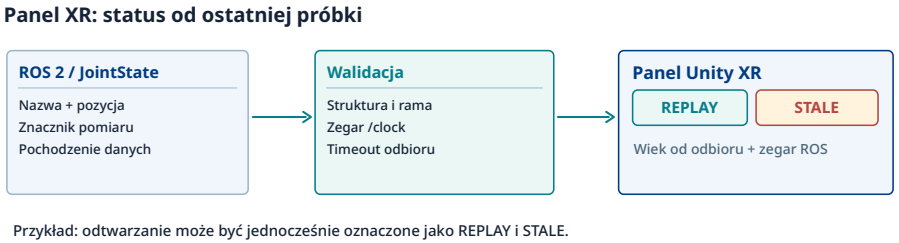
W trybie rosbag porównuj znaczniki w **czasie ROS**. Osobny watchdog wykrywa brak nowych pakietów.

5 minut przerwy

Pytania do samodzielnego powtórzenia

1. Dlaczego QoS `RELIABLE` nie gwarantuje aktualności danych?
2. Dlaczego `tf2` wymaga ramy i znacznika czasu?
3. Jakie dwa kierunki opóźnienia występują w teleoperacji XR?
4. Dlaczego przyjęcie komendy nie oznacza jej wykonania?

Unity: status LIVE, REPLAY i STALE w XR



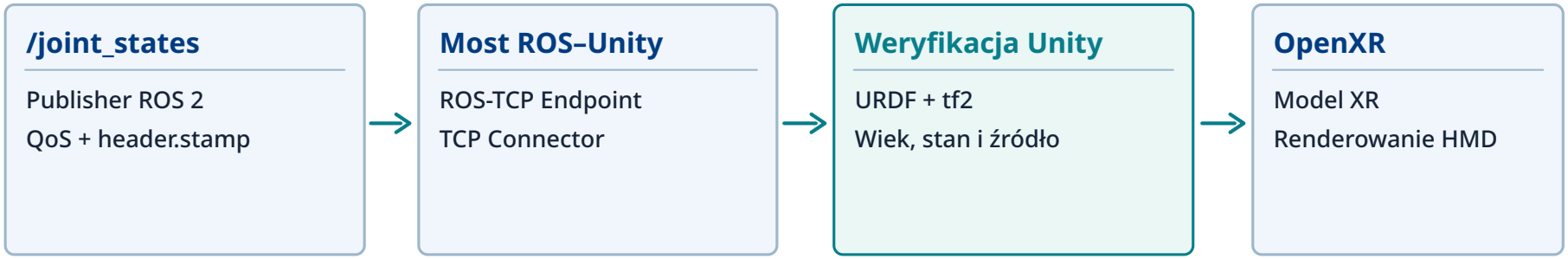
Dwa atrybuty danych:

- **Źródło:** LIVE (strumień) albo REPLAY (odtwarzanie).
- **Jakość:** aktualne, STALE, niepoprawne lub brak danych.

`WiRRW07StateBadge.AcceptObservation(...)` aktualizuje panel i uruchamia kontrolę wieku danych. Metody ręczne `ShowLive()` i `ShowReplay()` działają tylko przy włączonym `allowTeachingPreview`; również wtedy etykieta wygasa po przekroczeniu limitu.

Czujnik – edge – model – XR

Droga danych: ROS 2 → Unity → OpenXR



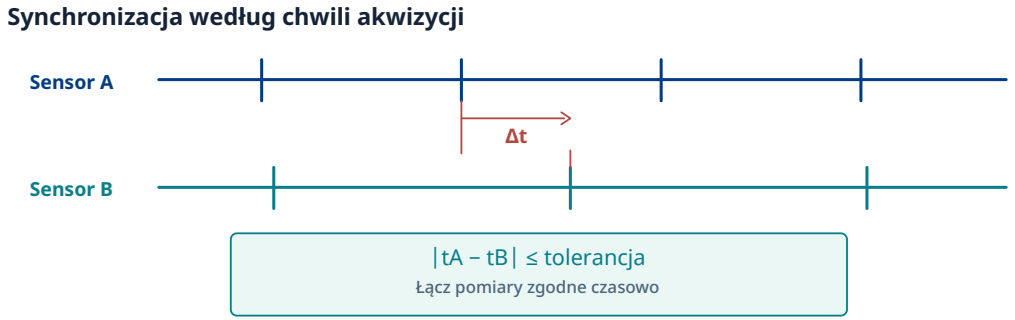
Kierunek sterowania pozostaje oddzielony i zablokowany.

Tor danych: `/joint_states → QoS → walidacja → mapa URDF → model Unity → XR`.

Most ROS–Unity wymaga osobnej konfiguracji komunikacji i kontroli zaufania.

Interaktywny przebieg ROS 2 ⟶ Unity ⟶ OpenXR

Synchronizacja sensorów

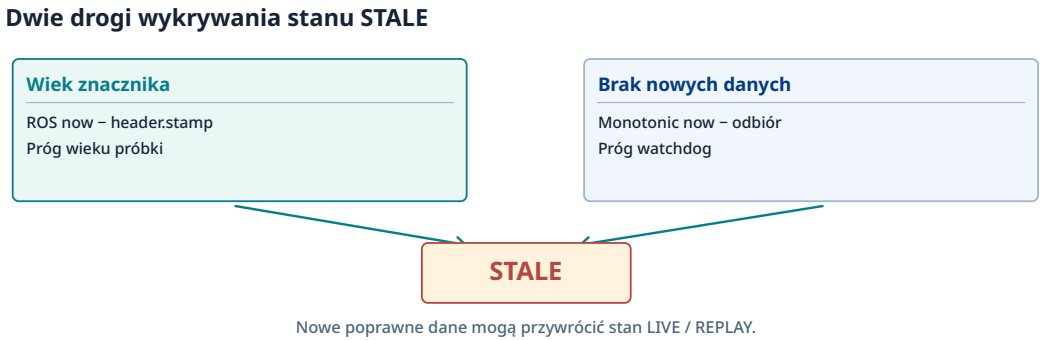


$$\Delta t = |t_A - t_B|$$

Fuzja danych: akceptuj parę próbek, gdy różnica chwil pomiaru mieści się w tolerancji dobranej do ruchu.

Nie utożsamiaj zbliżonego czasu odbioru z równoczesnym pomiarem.

Stale data

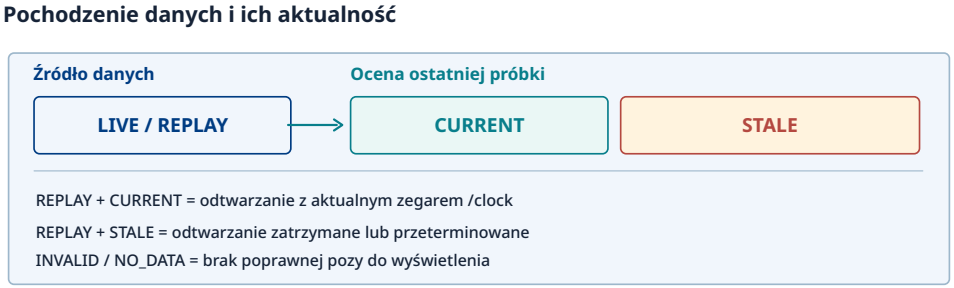


Dwie niezależne przesłanki STALE:

1. `t_ROS_now - header.stamp > max_age`
2. `t_monotonic_now - t_ostatniego_odbioru > timeout`

Watchdog działa także wtedy, gdy callback subskrybenta nie jest wywoływany.

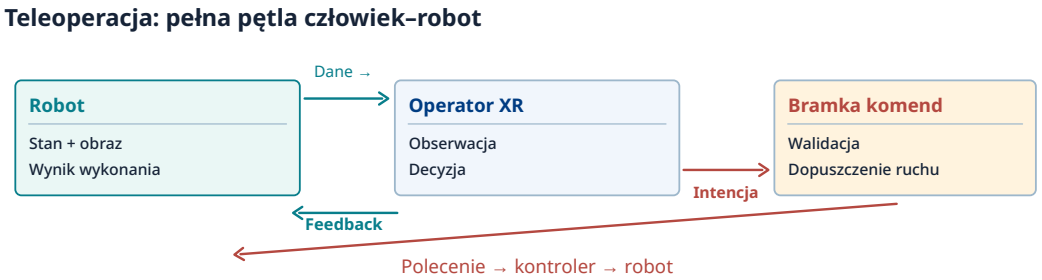
LIVE, REPLAY, STALE i brak danych



Źródło	Jakość	Panel XR
LIVE	aktualne	LIVE
REPLAY	aktualne	REPLAY
LIVE / REPLAY	za stare	STALE (z zachowaniem źródła)
dowolne	brak / błąd	BRAK DANYCH / BŁĘDNE DANE

REPLAY i STALE mogą wystąpić jednocześnie.

Teleoperacja: dwa kierunki opóźnienia

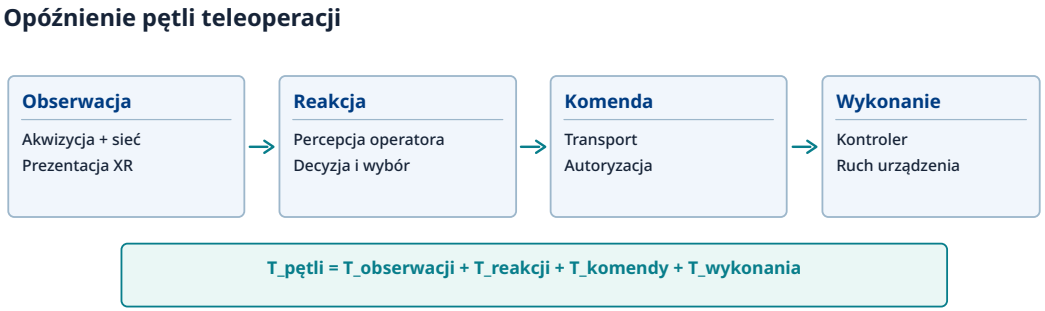


Robot → operator: obraz, konfiguracja i informacja zwrotna.

Operator → robot: intencja, walidacja, żądanie i wykonanie.

Łączna pętla obejmuje także **czas decyzji człowieka**.

Budżet opóźnienia i sterowanie zwrotne



$$T_{\text{pętli}} = T_{\text{obserwacji}} + T_{\text{reakcji}} + T_{\text{komendy}} + T_{\text{wykonania}}$$

Mierz osobno: akwizycję, sieć, renderowanie XR, reakcję operatora, kontroler i urządzenie.

Niestabilne opóźnienie może pogarszać stabilność i jakość teleoperacji.

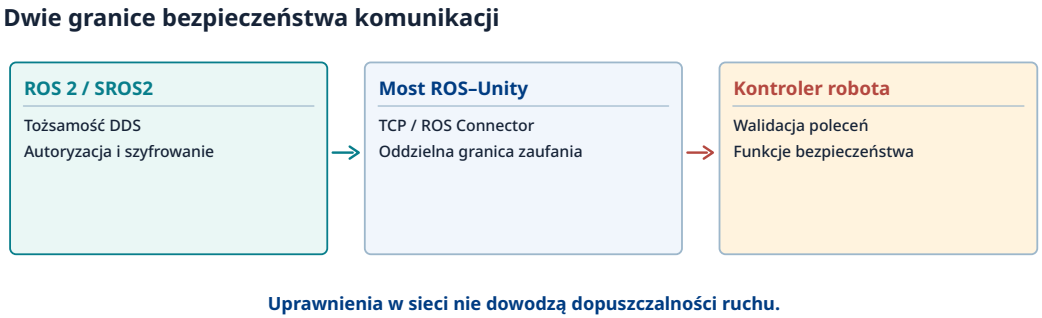
Virtual commissioning a cyfrowy bliźniak



Virtual commissioning: weryfikacja sterowania w środowisku modelowym.

Klasa	Integracja danych
Digital Model	aktualizacja ręczna
Digital Shadow	automatyczny przepływ obiekt ⟶ model
Digital Twin	sprzężenie obiekt ↔ model

Virtual commissioning: weryfikacja sterowania i procedur na modelu przed wdrożeniem.

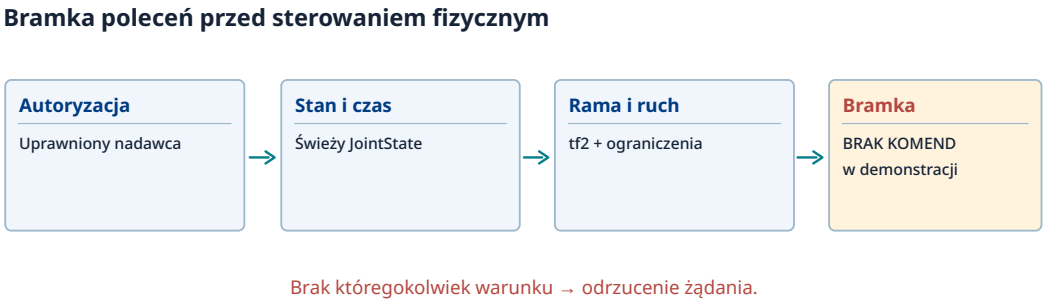


DDS-Security / SROS2: tożsamość uczestników, uprawnienia i ochrona komunikacji DDS.

ROS–Unity przez TCP: odrębna granica zaufania, wymagająca własnej konfiguracji ochrony.

Autoryzacja transmisji ≠ dopuszczenie ruchu robota.

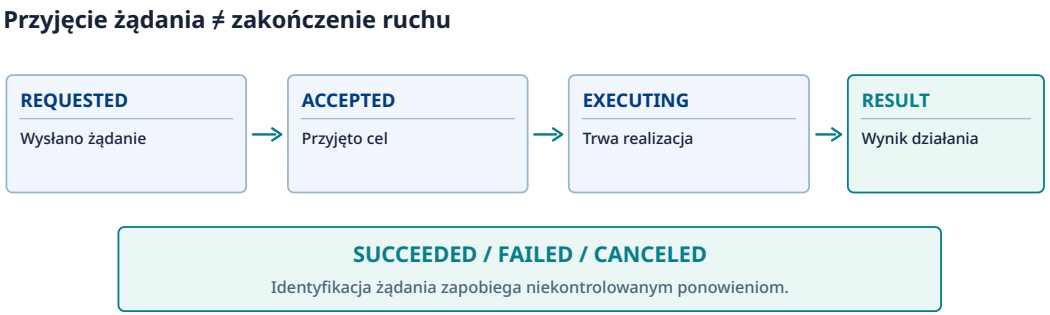
Walidacja komend



Przed przekazaniem polecenia: uprawnienia ⟶ aktualny stan ⟶ zgodna rama ⟶ ograniczenia ruchu ⟶ potwierdzenie operatora.

Każde niespełnione wymaganie oznacza **odrzucenie żądania**. W demonstratorze bramka pozostaje zamknięta.

Oddzielne potwierdzenie przyjęcia i wykonania

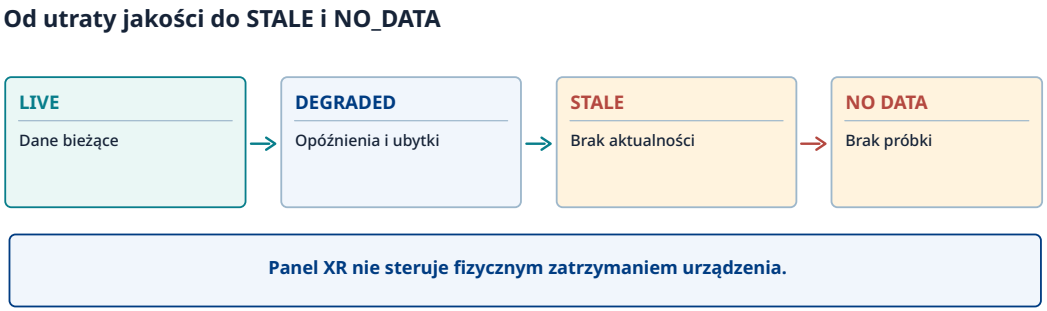


REQUESTED → ACCEPTED → EXECUTING → SUCCEEDED / FAILED / CANCELED

Przyjęcie żądania przez usługę lub akcję nie jest równoznaczne z wykonaniem.

Bez pewności co do wyniku nie powtarzaj ruchu automatycznie.

Utrata komunikacji: stan bezpieczny i degradacja



Podgląd XR: `LIVE → DEGRADED → STALE → NO_DATA` (według przyczyn utraty jakości).

Urządzenie wykonawcze: własna logika ograniczeń, zatrzymania i analizy ryzyka.

Zamrożona animacja XR **nie potwierdza** zatrzymania fizycznego robota.

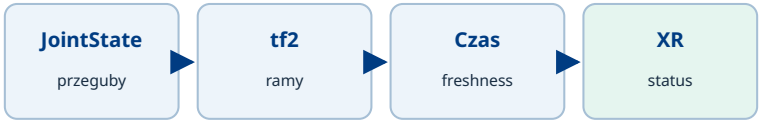
Diagnostyka ROS 2 bez sterowania robotem

```
ros2 topic info -v /joint_states
ros2 topic echo /joint_states --once
python3 joint_state_age.py --ros-args -p max_age_s:=0.2
```

Scenariusz: LIVE → przerwa napływu → STALE → wznowienie → LIVE. Watchdog działa co `min(0.1, max_age_s/4)` s (lub według `watchdog_period_s`), a opóźnienie wykrycia obejmuje okres timera i jitter harmonogramowania. Duplikaty i próbki ze starszym timestampem są odrzucane; przewijanie rosbag wstecz resetuje cache przez `/clock`, przewijanie w przód wymaga `reset_for_seek()` albo restartu obserwatora.

Demonstator etapów komunikacji (offline)

Podsumowanie



Źródło oraz aktualność danych są dwoma kryteriami

Wiarygodny podgląd XR wymaga jednocześnie:

- poprawnego `JointState` i mapowania do URDF;
- zgodnego czasu ROS, tf2 i niezależnego watchdoga;
- jawnego źródła LIVE/REPLAY oraz statusu STALE;
- oddzielenia wizualizacji od wykonania komend.

Bibliografia

- ROS 2, QoS.
- ROS 2, tf2.
- JointState.
- ROS Clock and Time.
- Unity, ROS–Unity Connector.
- Kritzinger i in. (2018), Digital Model / Shadow / Twin.
- Hokayem i Spong (2006), Bilateral teleoperation.