Case Study for an Integrated Digital Twin Composition: Robot Arm and Gripper

Yehoshua Halle - University of Maryland, College Park
Mentor: Guodong Shao

Disclaimer

Identification of commercial systems does not imply recommendation or endorsement by NIST.

Identified commercial systems are not necessarily the best available for the purpose.

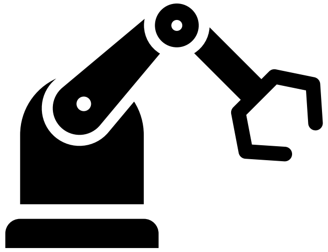
Outline

1. Define digital twin composition
2. Introduce generic procedure – ISO 23247-6 Draft
3. Implement integrated digital twin composition
4. Discuss lessons learned

ISO 23247-6 Draft
Types of Digital Twin Composition (DTC)

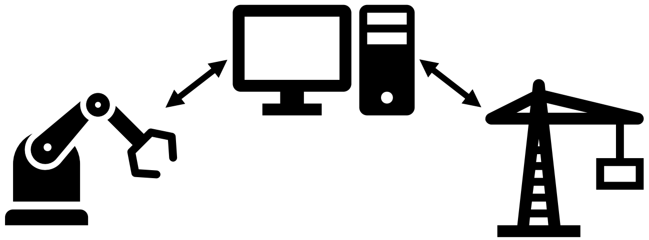
Integrated: Consolidates DTs into a single system with centralized control.

Example: a robot arm with a gripper



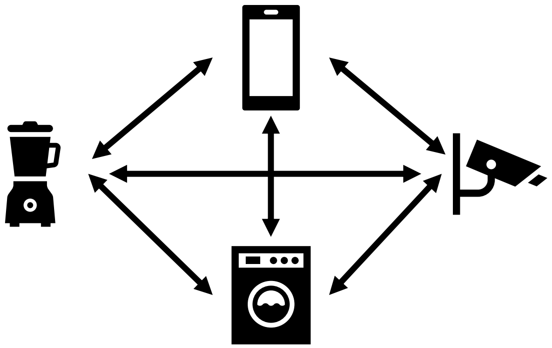
Unified: Independent DTs connected with a central coordinating entity.

Example: a shop floor



Federated: Independent DTs connected without a central coordinating entity.

Example: IoT devices



Why compose digital twins?

- Digital twins are **complicated** systems
- Complicated systems are **difficult to understand and change**
- Single DTs are **limited** in functionality
- Development requires **time and money**

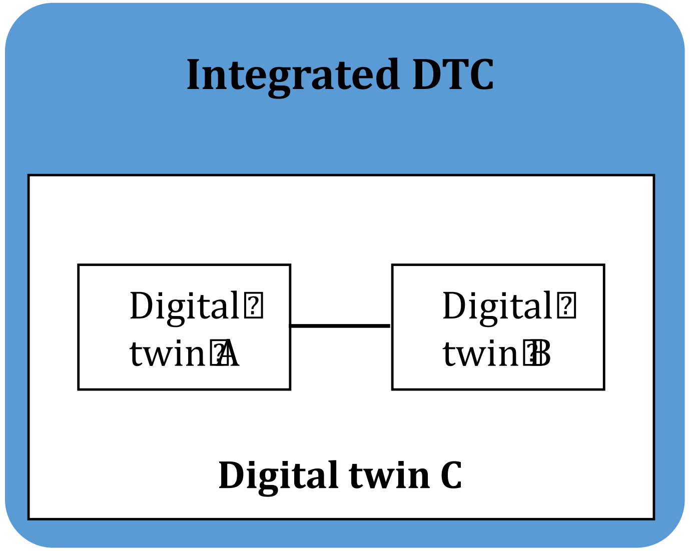
Composing digital twins…

- Reduces needed development with **reuse**
- Reuse is **faster, easier, and cheaper**
- Can increase interoperability

Integrated DTC – ISO 23247-6 Draft

“An Integrated DTC involves creating a single, comprehensive digital twin model that consolidates all data and functionalities from other individual digital twins into one overarching digital twin system.”

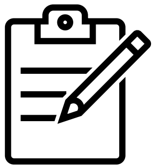
- Centralized control, communication
- Common data representation



ISO 23247-6

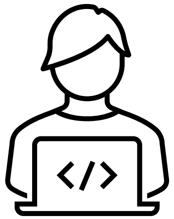
Digital Twin Composition Life Cycle

1. **Requirements**

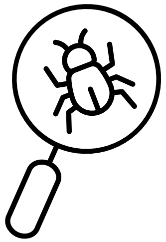


2. **Design**

3. **Development**

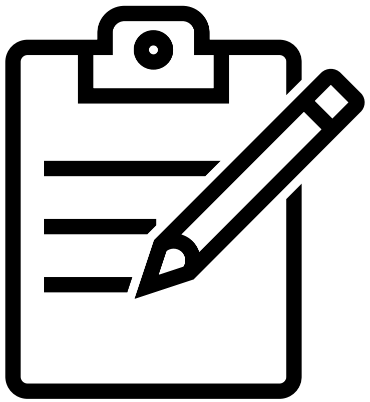


4. **Operation**



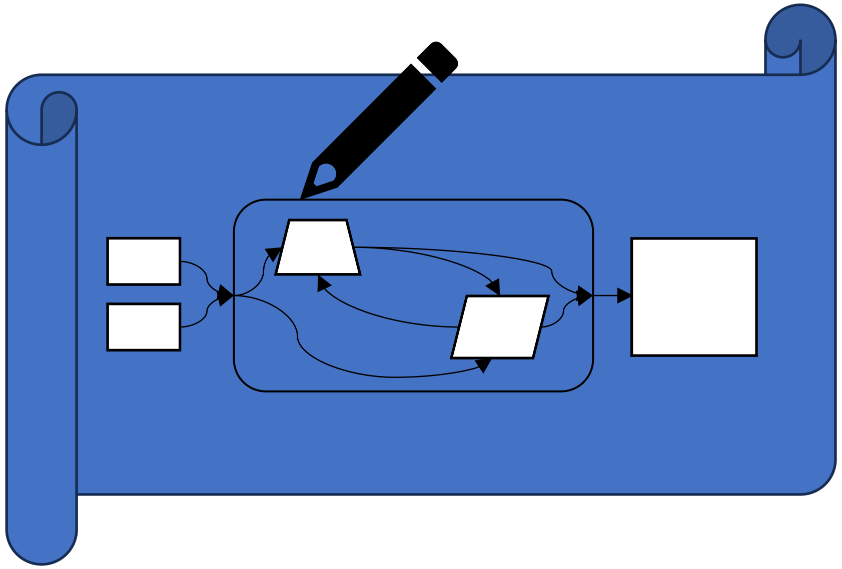
Generic Procedure – Requirements

- Define purpose
- Functional requirements
  - What does it do?
- Non-functional requirements
  - How well does it do?
  - How much does it do?
- Composition type
  - Integrated, unified, or federated?



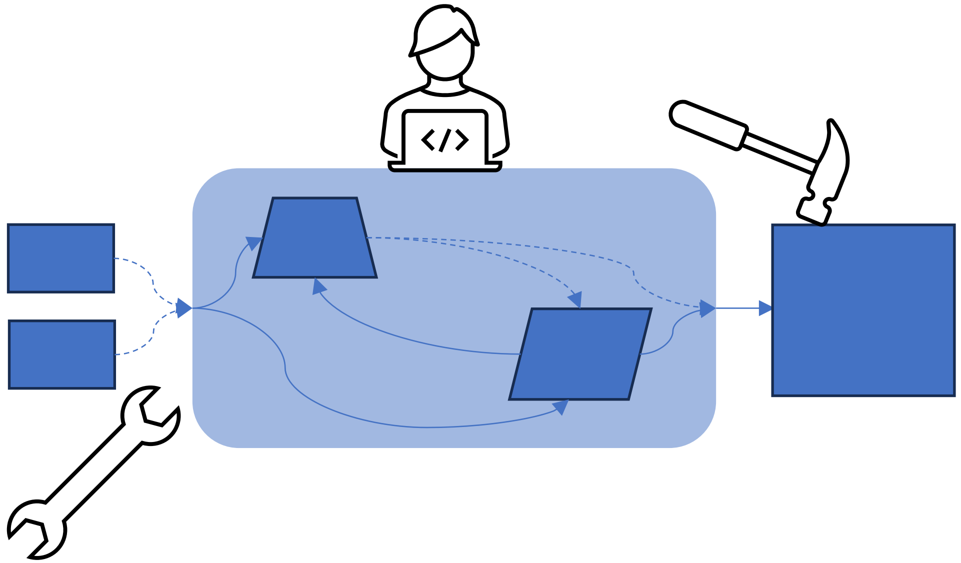
Generic Procedure – Design

- Catalog existing digital twins
- Define data types and their meaning
  - Includes **common data model**
- Design system structure and behavior



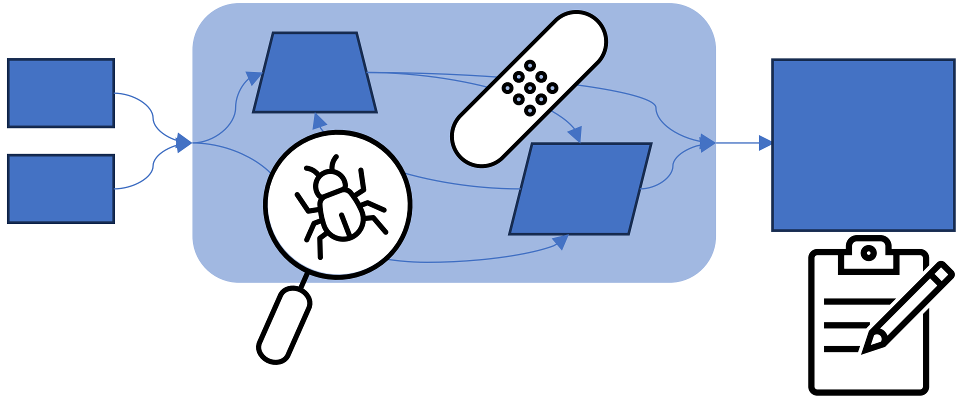
Generic Procedure – Development

- Implement designed system
- Create a UI
- Testing
  - Verification & Validation



Generic Procedure – Operation

- Deploy the system
- Track performance and integrity
- Maintain the system
  - Bugfixes
  - Updates
- Upgrade with new features
  - New requirements



Case Study: Integrated DTC of Robot Arm with Gripper

How **useful and clear** is the generic procedure of the draft ISO 23247-6 digital twin standard?

What **actual process** was followed to create an integrated composition of a robot arm and gripper digital twin?

What **changes** are suggested for the draft standard?

What features of digital twin systems **facilitate composability**?

Existing work relating to DTC in Robotics

Many papers on developing digital twins through composition, but not by **composing DTs together**.

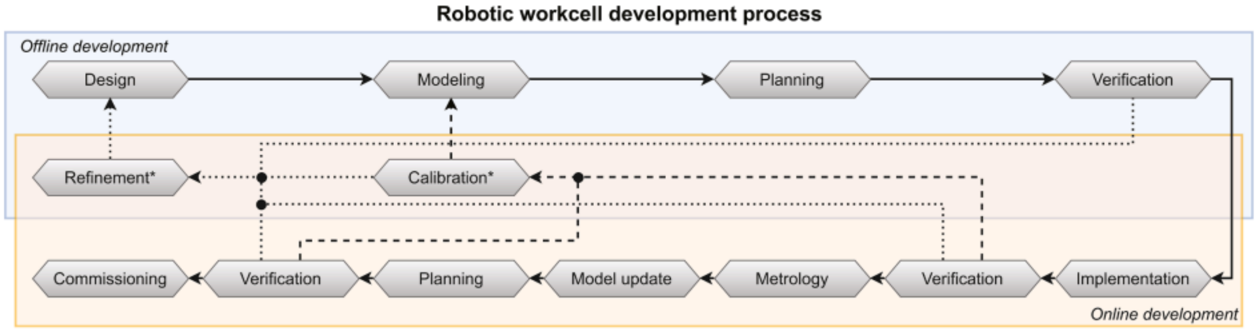
Examples:

Tipary (2021): Generic method for workcell DTs

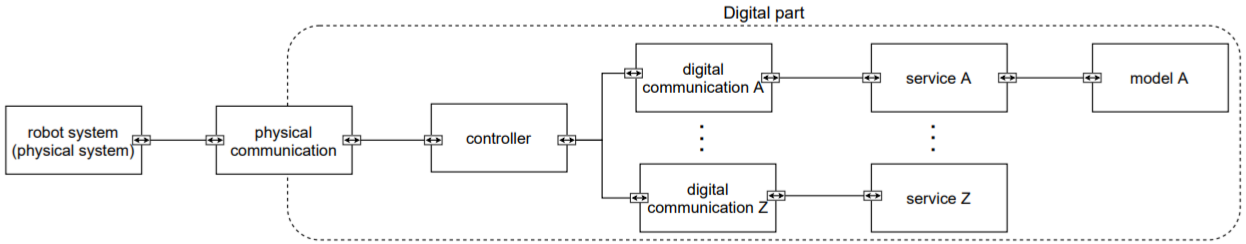
- Procedure for a singular digital twin of a workcell
- Modularity using "models" for different functions

Tola (2022): Modular Robot DTs

- Modularity within a single digital twin using software "services"



Development process for pick & place (Tipary, 2021)



Proposed architecture for modular digital twins of robot systems (Tola, 2022)

Tipary, B., & Erdős, G (2021). Generic development methodology for flexible robotic pick-and-place workcells based on Digital Twin.

Tola. D, et al (2022). Towards Modular Digital Twins of Robot Systems.

Implementation

DTC Progress

- ☐ **Requirements**
  - Define purpose
  - Functional requirements
  - Non-functional requirements
  - Composition type
- ☐ Design
- ☐ Development
- ☐ Operation

Purpose and Requirements

Purpose: Pick and place operations

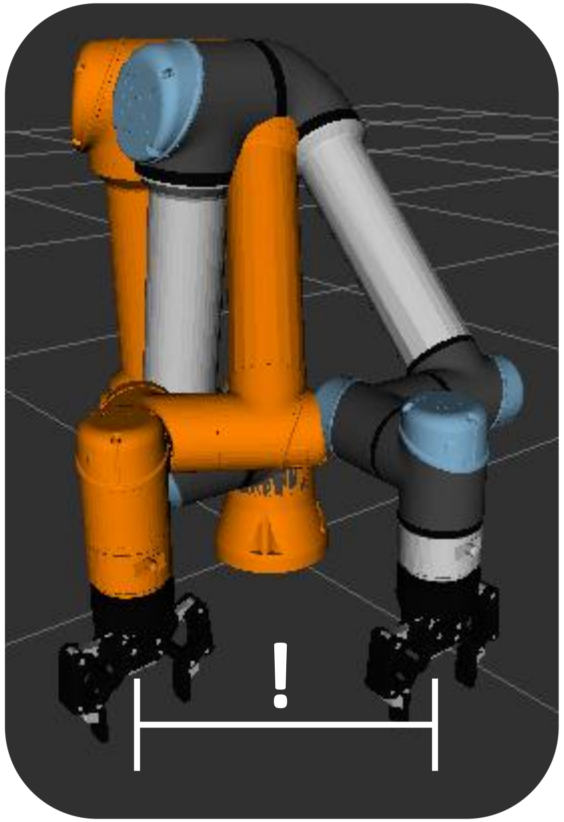
Functional Requirements

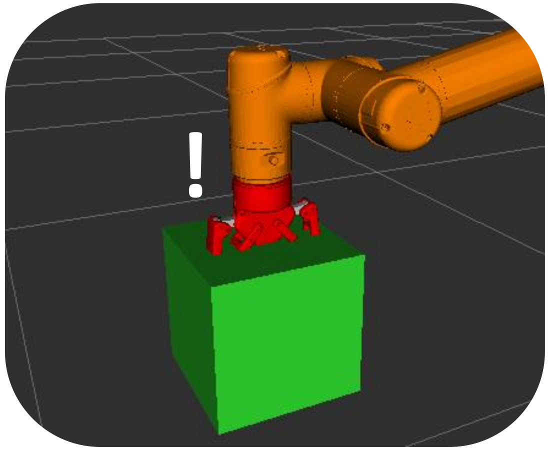
- Grasp and Move objects
- Avoid collisions with known obstacles
- Warn operator when accuracy of physical robot is out of tolerance

Non-Functional Requirements

- Position tolerance: < 1 mm
- Rotation tolerance: < 1 degree

"Tolerance" = difference between physical and digital entities





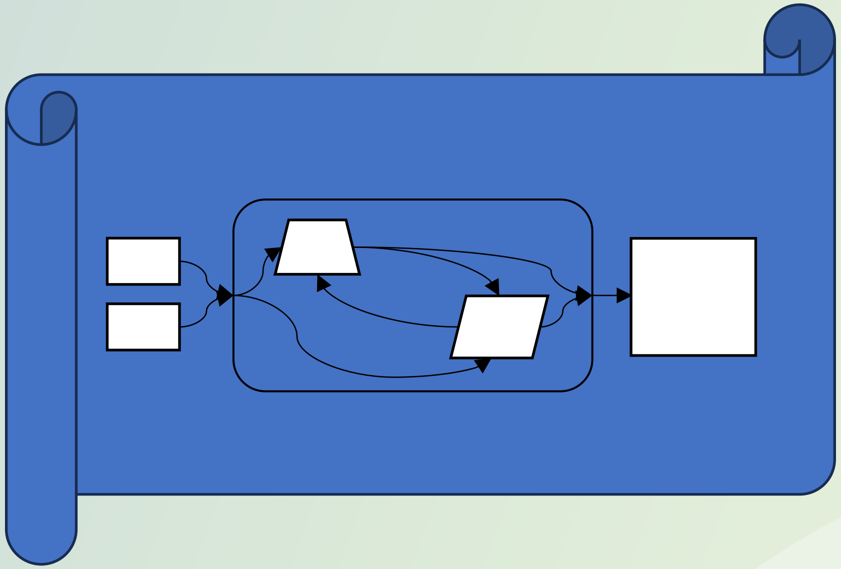
Why Integrated DTC?

Highly applicable:

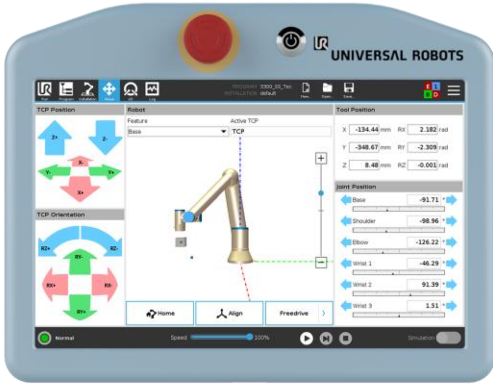
- Control of arm and gripper is **centralized**
- Arm and gripper “combine” into single entity: robot arm **with** gripper
- Individual twins of a robot arm or gripper would **not be useful** on their own.

DTC Progress

- ✓ Requirements
- ☐ **Design**
  - Catalog existing DTs
  - Define data types and their meaning
  - Design system structure and behavior
- ☐ Development
- ☐ Operation



Existing Digital Twins



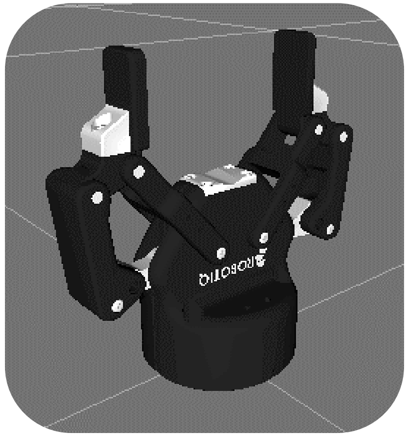
universal-robots.com/products/polyscope

UR5e
- 3D Model + Kinematics
- Visualizer
- Control Software
- URSim
- Synchronizer

2F85
- 3D Model + Kinematics
- Visualizer
- Control Software
- Simulator
- Synchronizer

Custom, Simple

Same commands, independent ideal execution



Common Data Model

What data does each DT use?

What data is for new functionality?

- Types, content, meaning

How is data used/transformed in the system?

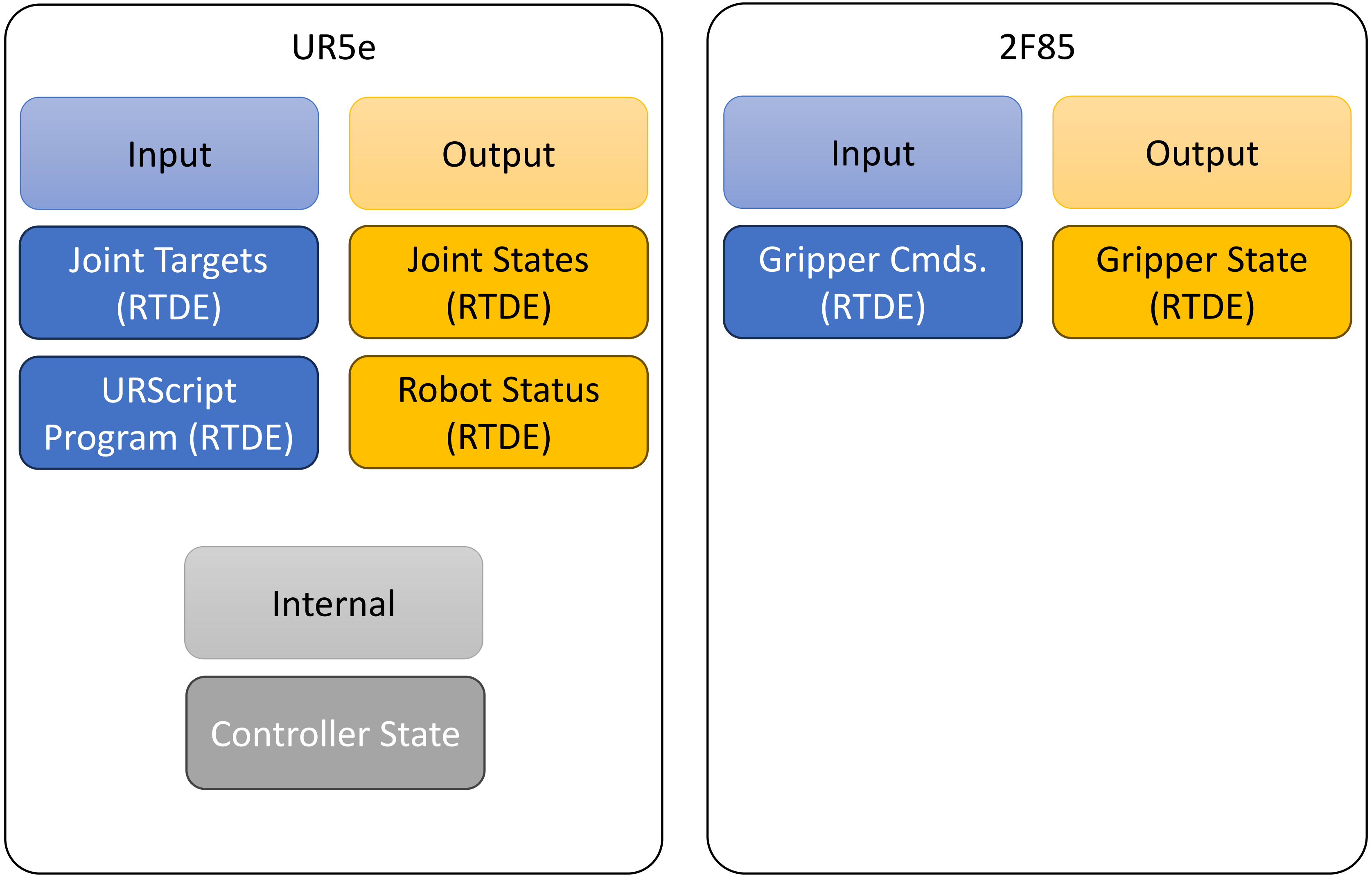
- Relationships between data producers/consumers

Core Framework: ROS 2

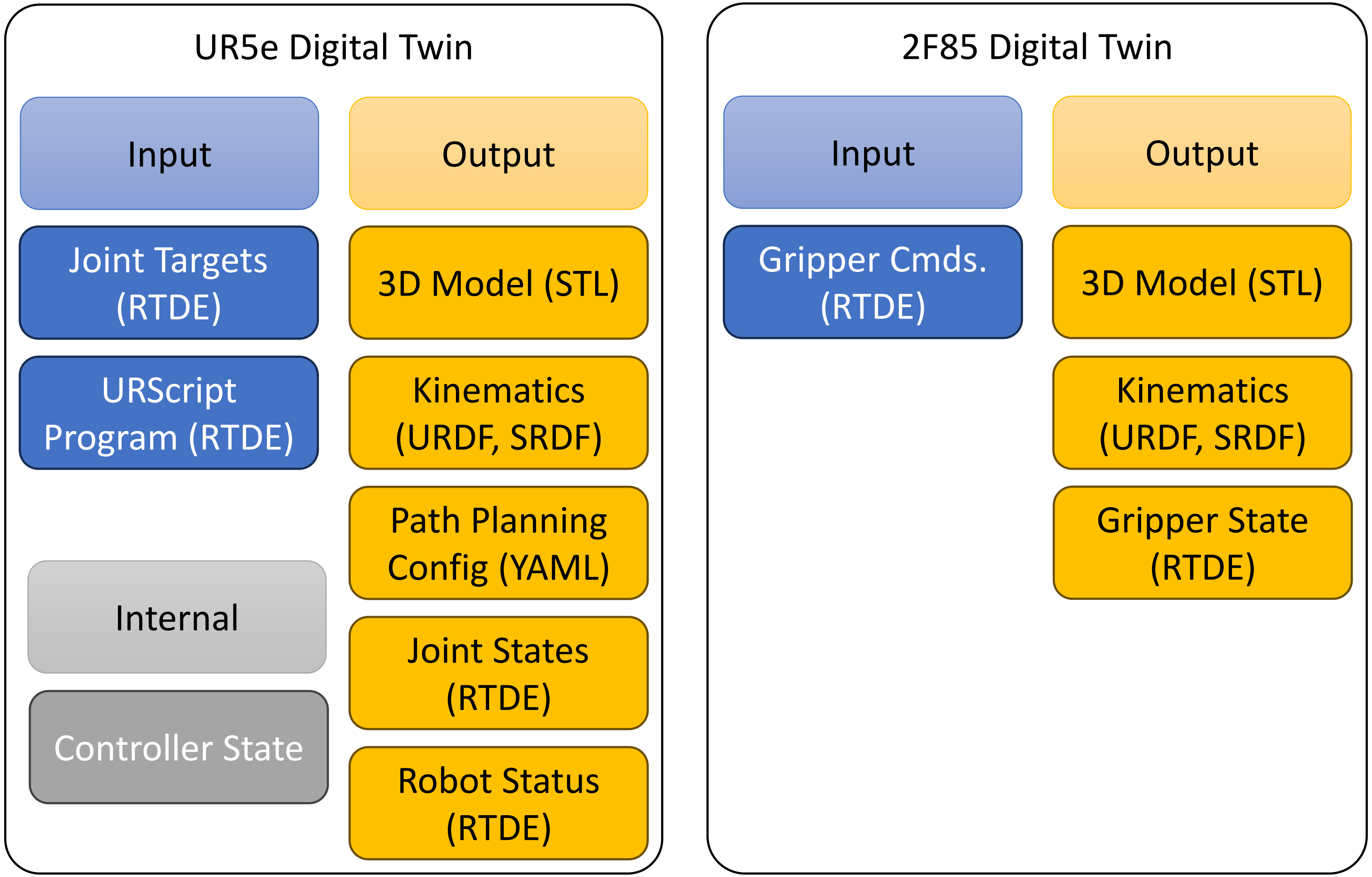
“Robot Operating System”

- Open-source project
- Widely used in robotics community
- Both UR and RobotIQ support ROS
- Interface Definition Language (IDL): shared data types
- Data Delivery System (DDS): secure communication
- Can be extensible and modular

Data for Observable Manufacturing Elements



Data for Individual DTs



Data for Integrated Functionality

Path Planner

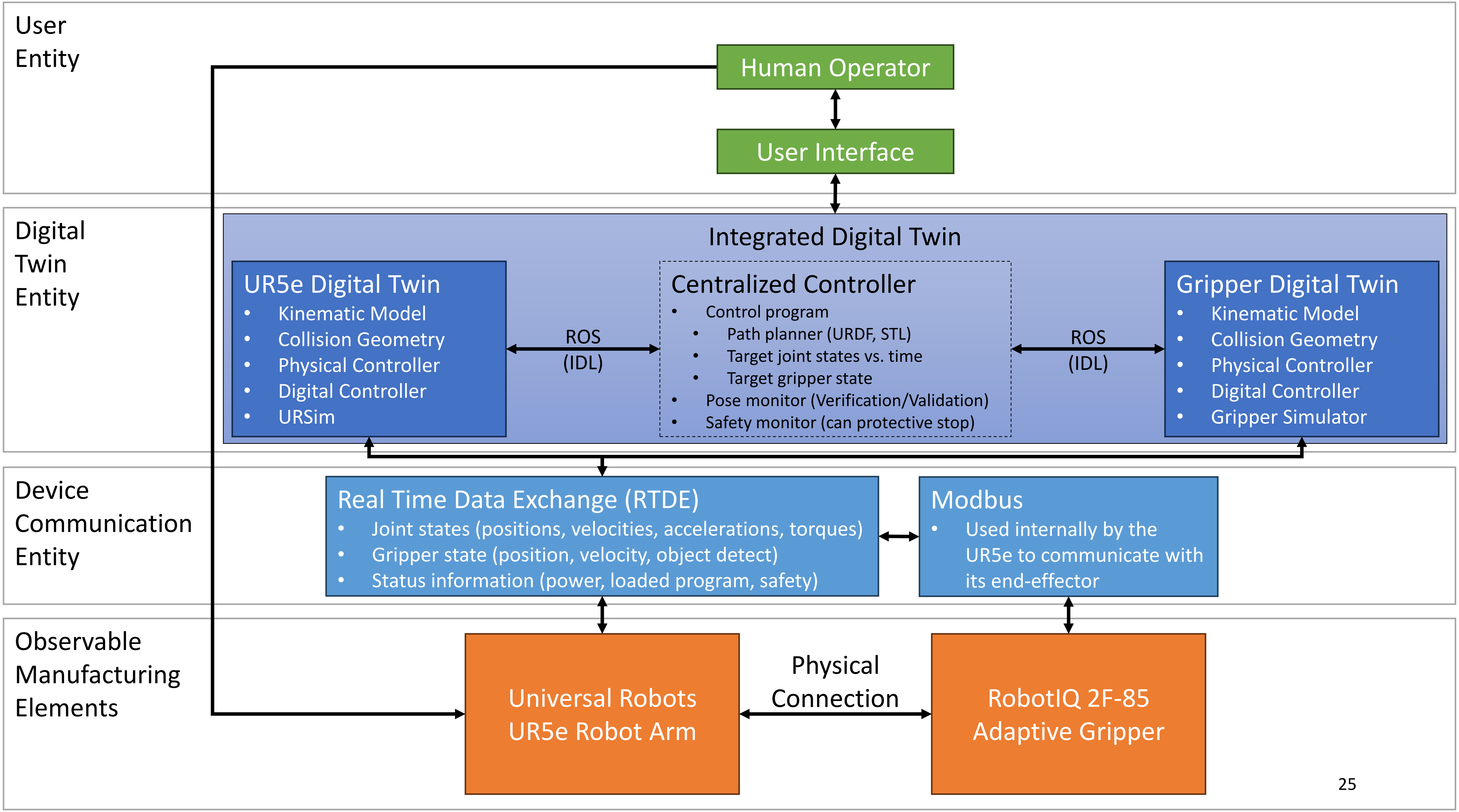
Input	Output
Joint/Pose Targets	Joint Trajectory
Kinematics (URDF, SRDF)	
Path Planning Config (YAML)	
Collision Geometry	

Pose Monitor

Input	Output
Physical Joint States	Physical Pose
Digital Joint States	Digital Pose
	Pose Error

Centralized Controller

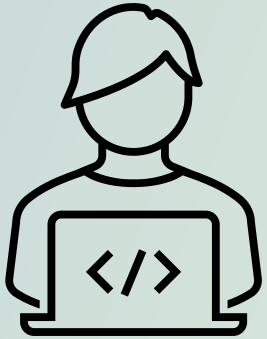
Input	Output
Joint/Pose Targets	Joint Targets
Gripper Targets	URScript Program
	Gripper Commands



User Entity
Human Operator
User Interface
Digital Twin Entity
Integrated Digital Twin
UR5e Digital Twin
• Kinematic Model
• Collision Geometry
• Physical Controller
• Digital Controller
• URSim
ROS (IDL)
Centralized Controller
• Control program
• Path planner (URDF, STL)
• Target joint states vs. time
• Target gripper state
• Pose monitor (Verification/Validation)
• Safety monitor (can protective stop)
ROS (IDL)
Gripper Digital Twin
• Kinematic Model
• Collision Geometry
• Physical Controller
• Digital Controller
• Gripper Simulator
Device Communication Entity
Real Time Data Exchange (RTDE)
• Joint states (positions, velocities, accelerations, torques)
• Gripper state (position, velocity, object detect)
• Status information (power, loaded program, safety)
Modbus
• Used internally by the UR5e to communicate with its end-effector
Observable Manufacturing Elements
Universal Robots UR5e Robot Arm
Physical Connection
RobotIQ 2F-85 Adaptive Gripper

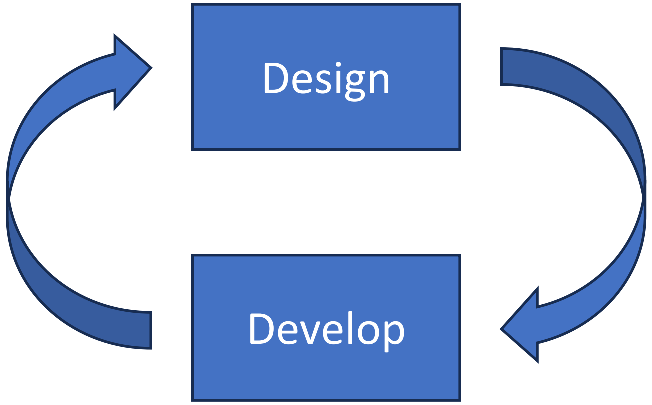
DTC Progress

- ✓ Requirements
- ✓ Design
- ☐ **Development**
  - Implement designed system
  - Create a UI
  - Testing
  - Verification & Validation
- ☐ Operation



Not straightforward

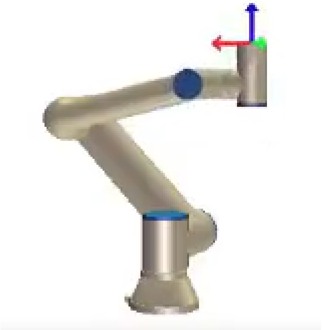
- Data types and behavior were initially unknown
  - Make progress
  - Test
  - Break things
  - Learn of new behavior/data
  - Update common data model/system structure
  - Try again

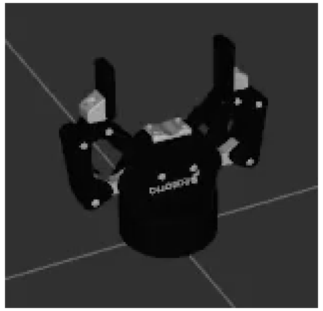


User Interfaces

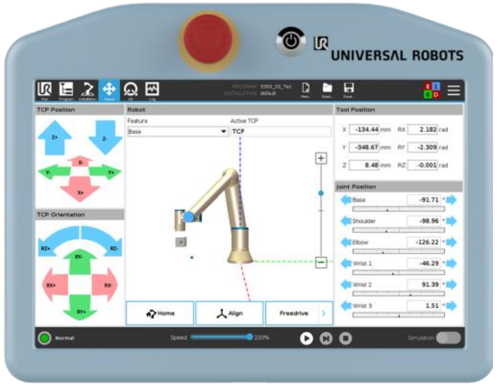
Existing UI

- Visualizing UR5e and 2F85 state





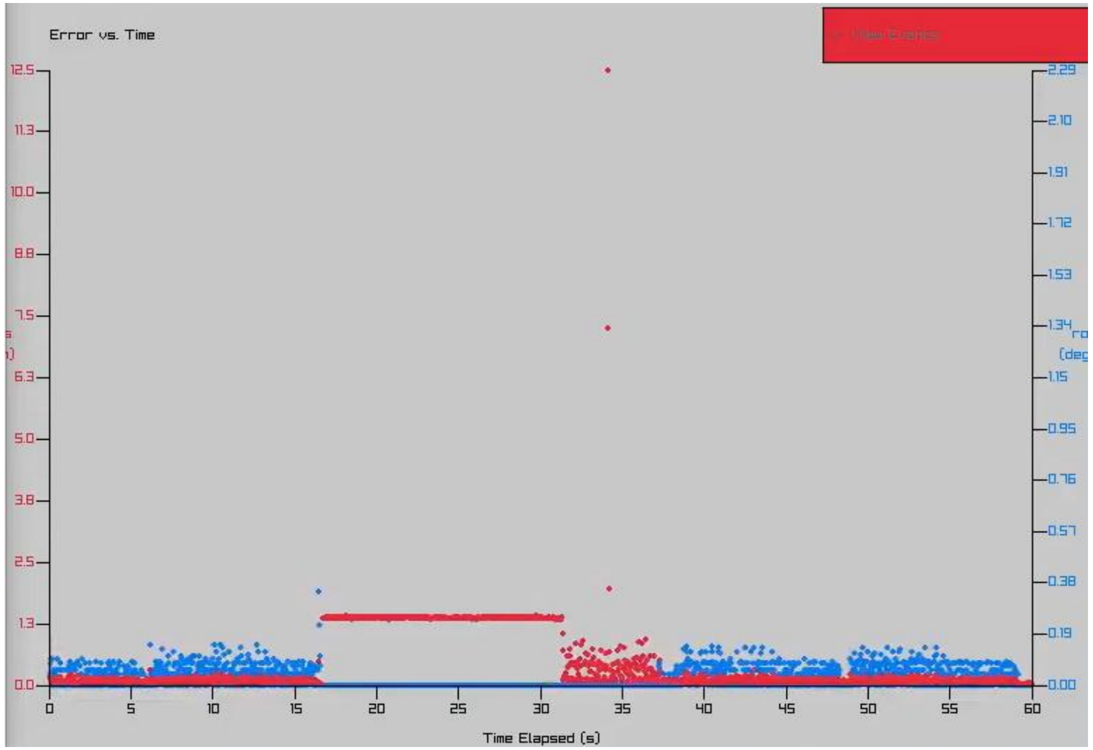
- Teach Pendant



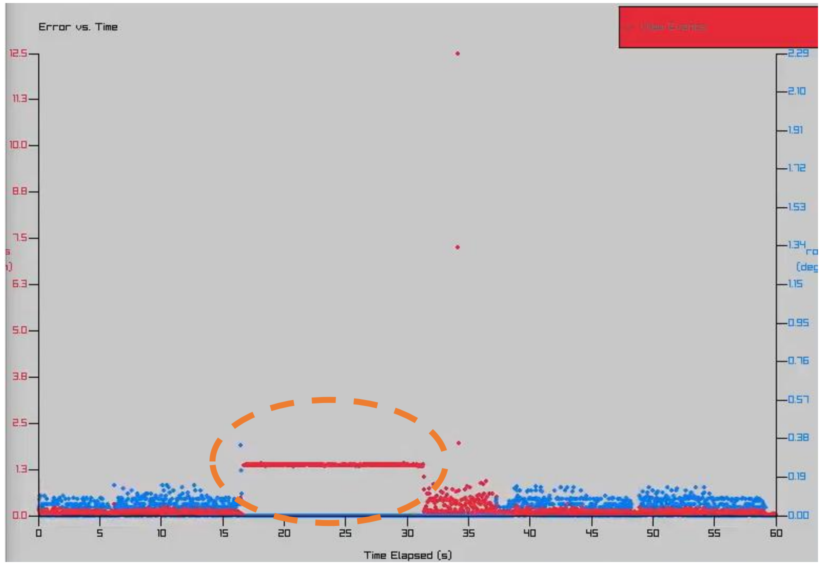
universal-robots.com/products/polyscope

Developed UI

- Graph of error vs time, & alerts



Verification and Validation



Performance goals:

- 1 mm position tolerance
- 1 deg rotation tolerance

Used error graph to reduce sources of position error

1. Consistent error > 1 mm when not moving
2. Path planner tolerance too high, decreased from 1mm to 0.1mm
3. Robot execution tolerance decreased from 1mm to minimum
4. Movement now stays within tolerance except for abrupt jerks

DTC Progress

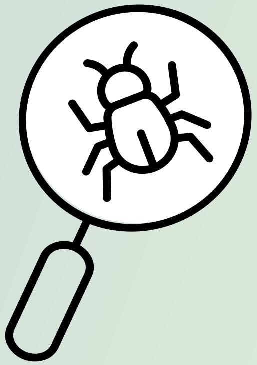
✓ Requirements

✓ Design

✓ Development

☐ **Operation**

- Deploy the system
- Track performance and integrity
- Maintain the system
- Upgrade with new features (if needed)



Operation

Deploy the system

- Already deployed in Digital Twins Lab

Track performance and integrity

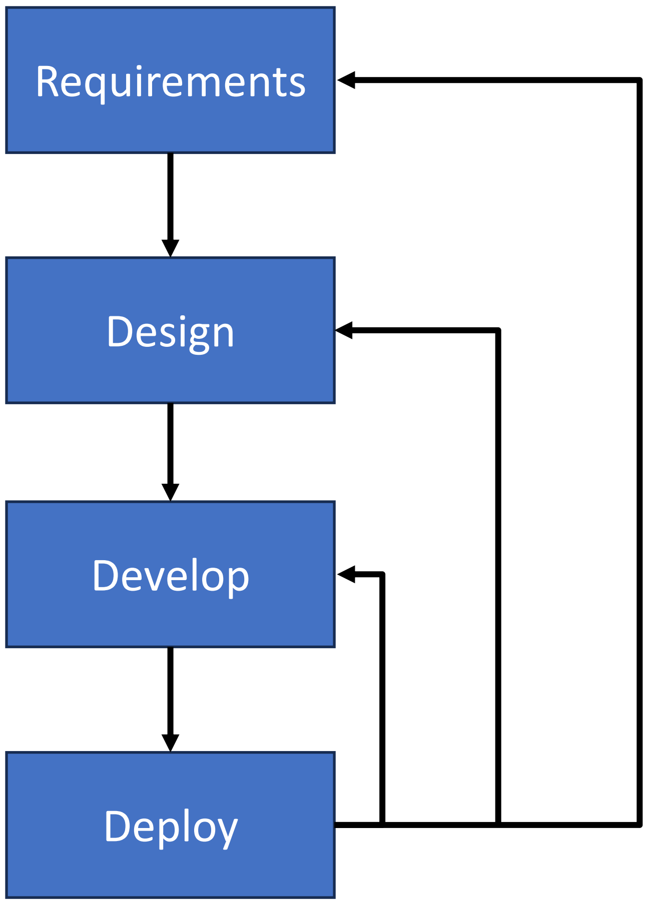
- All tolerance violations are saved in log files

Maintain the system

- Fix bugs as they are discovered

Upgrade with new features

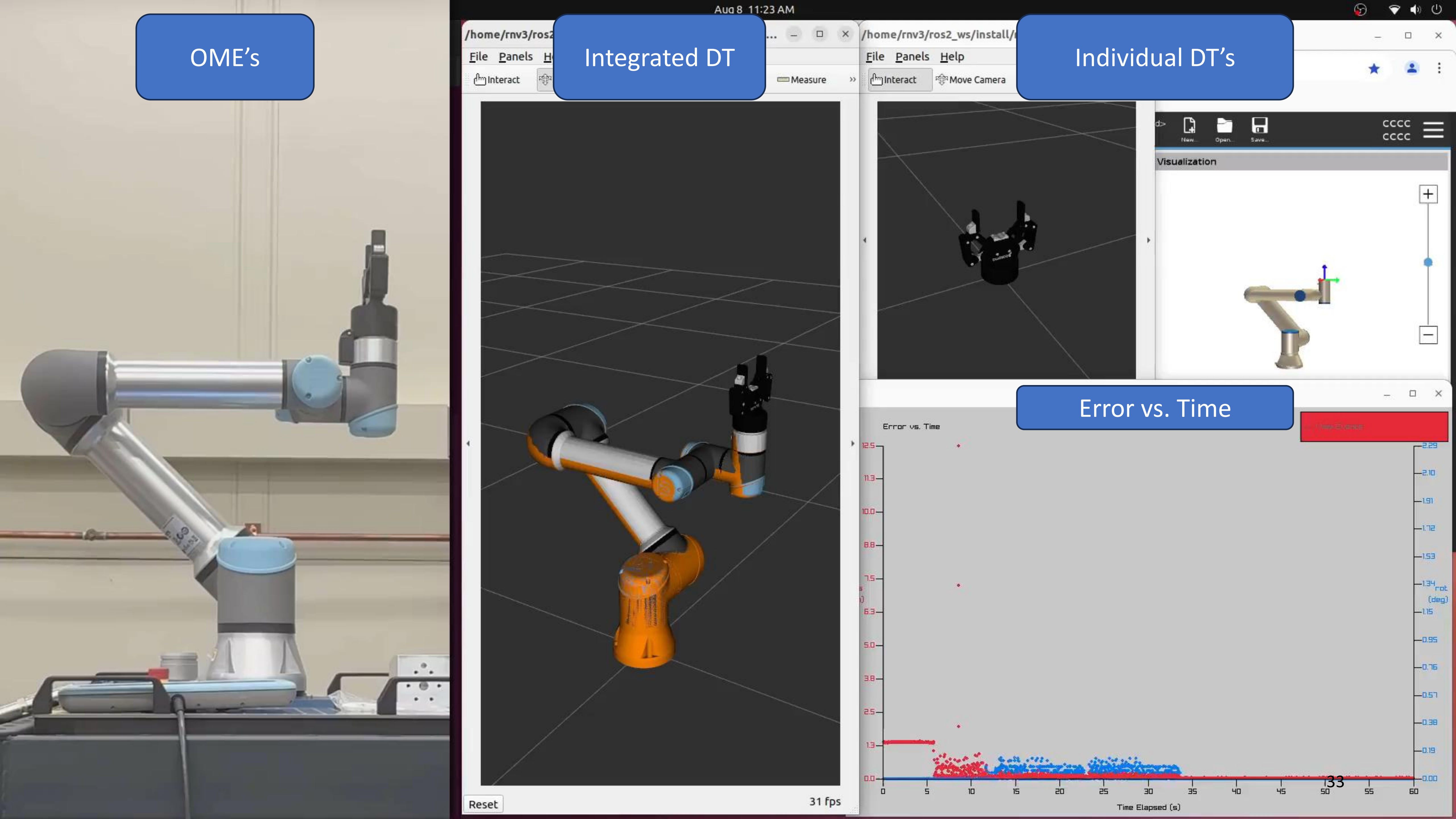
- When requirements/components change



DTC Progress – Complete!

- ✓ Requirements
- ✓ Design
- ✓ Development
- ✓ Operation

DEMO!


OME's
Integrated DT
Individual DT's
Error vs. Time
Aug 8 11:23 AM
File Panels Help
Interact
Move Camera
Measure
Visualization
Reset
31 fps
Error vs. Time
rot (deg)
Time Elapsed (s)

Lessons Learned

How **useful and clear** is the generic procedure of the draft ISO 23247-6?

- Supported implementation
- A common data model is essential
- Steps are generic, require interpretation
  - Purpose, requirements, and situation inform the interpretation
  - More use cases can provide example interpretations

Any changes?

- Cyclic nature of procedure should be indicated
  - Common in design processes

What features of digital twin systems **facilitate composability**?

- Common data format should be **expressive** and **extensible**
  - Expressive: can inherently **communicate its meaning**
    - **Example: "**Point" vs. "Vector3"
  - Extensible: can be **extended easily** to meet the needs of new problems
    - **Example:** Interface Definition Language
  - Both features make the system **easier to understand and maintain**
- Clearly defined data and behavior
- Minimal added complexity

Naming Everything Is Less Composable

ROS entities must be referred to by name, names exist **globally**

- Naming conflicts (not **extensible**)
- Name doesn’t always capture meaning (not **expressive**)
- Named references must be manually updated
    - Solution: references without names or managed automatically

Key Takeaways

- Generic procedure results in valid DTC of robot arm and gripper
- Design of composable systems: common data model, naming

Future Work

- Resilience of composable systems: change DTs in existing DTCs
- Case studies for Unified and Federated DTC

Thank You!

Questions?