

MOSIM Framework

Dr. Klaus Fischer

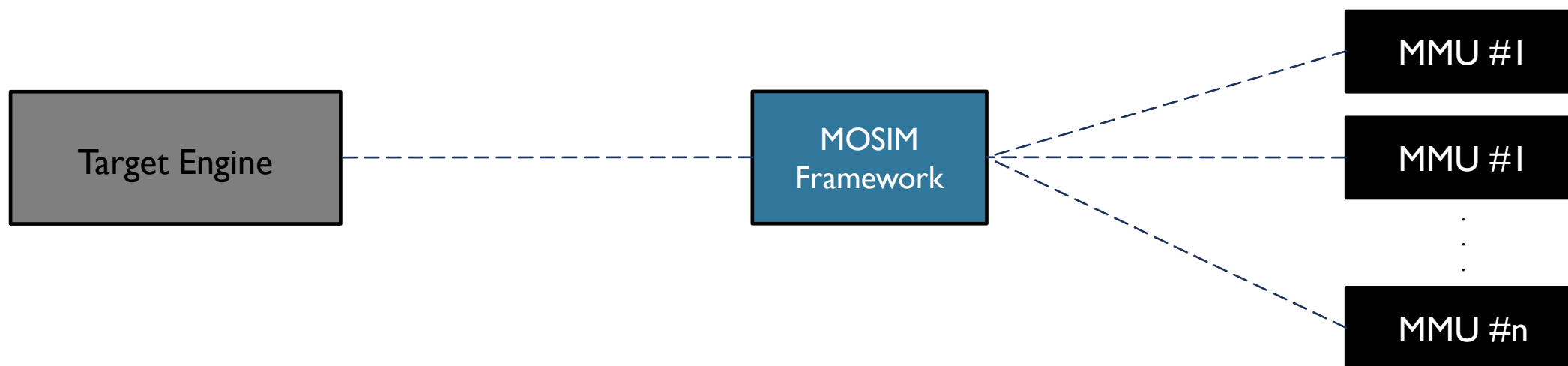
Webinar, 29th of November 2021



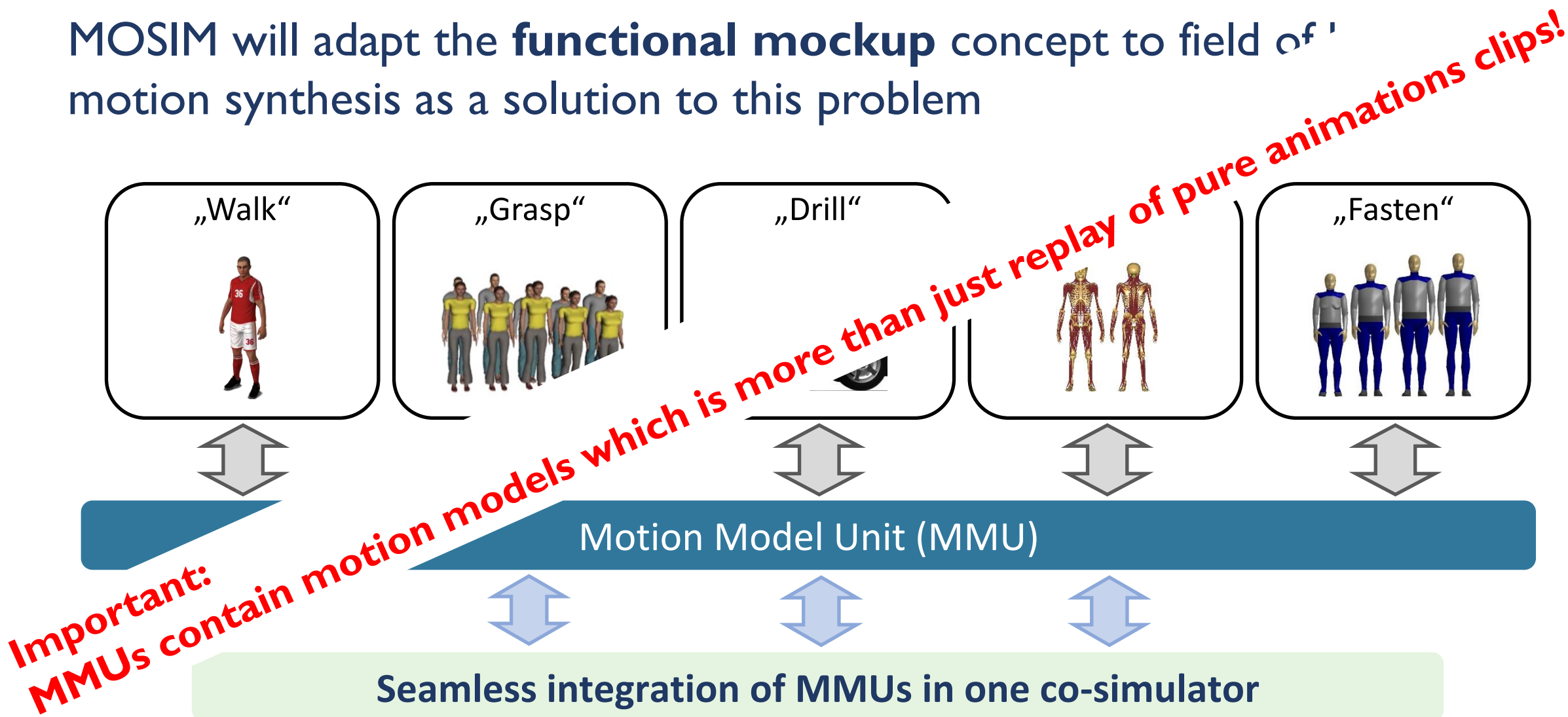
End-to-end Digital Integration based on Modular Simulation of Natural Human Motions

Goal Description: Integration of different Motion Synthesis Approaches

- **MMU** (Modular Motion Unit): (black-box) motion synthesis approach
- **Target Engine**: (game-) engine, in which the visualization is displayed
- **Goal**: Combine multiple sequential and parallel MMUs to display a complex motion sequence in the target engine.



MOSIM will adapt the **functional mockup** concept to field of motion synthesis as a solution to this problem

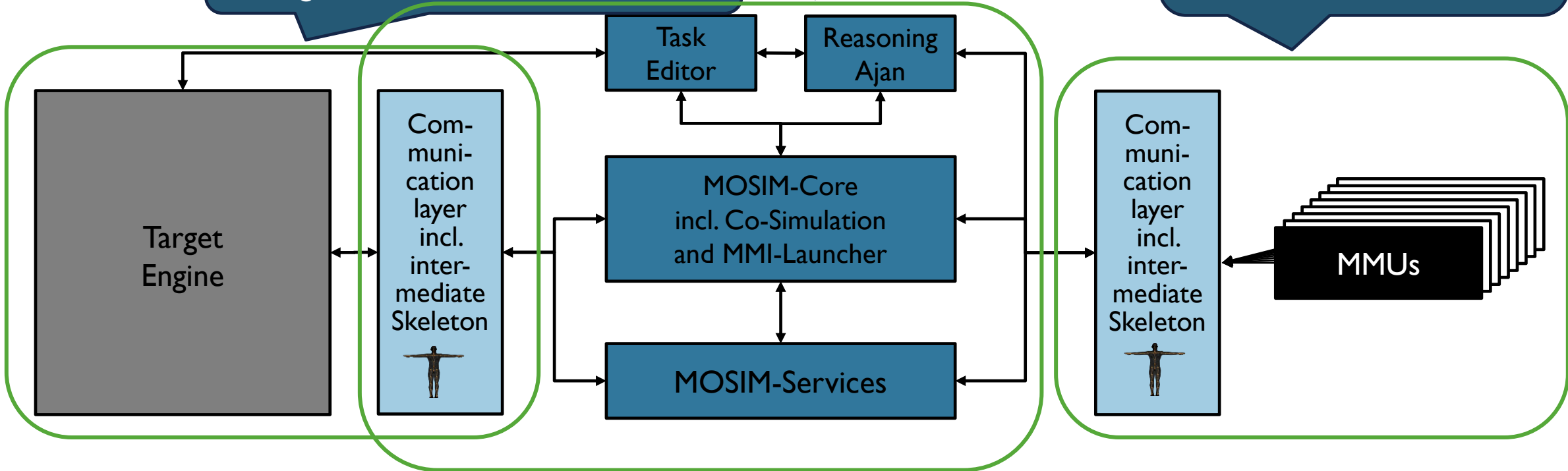


MOSIM Architecture

Flexibility of using existing target engines via communication API's

Assembly of Motion Model Units together into a co-simulator

Library of Motion Model Units for user specific applications



legend



Standardized Interfaces



Tools & Services

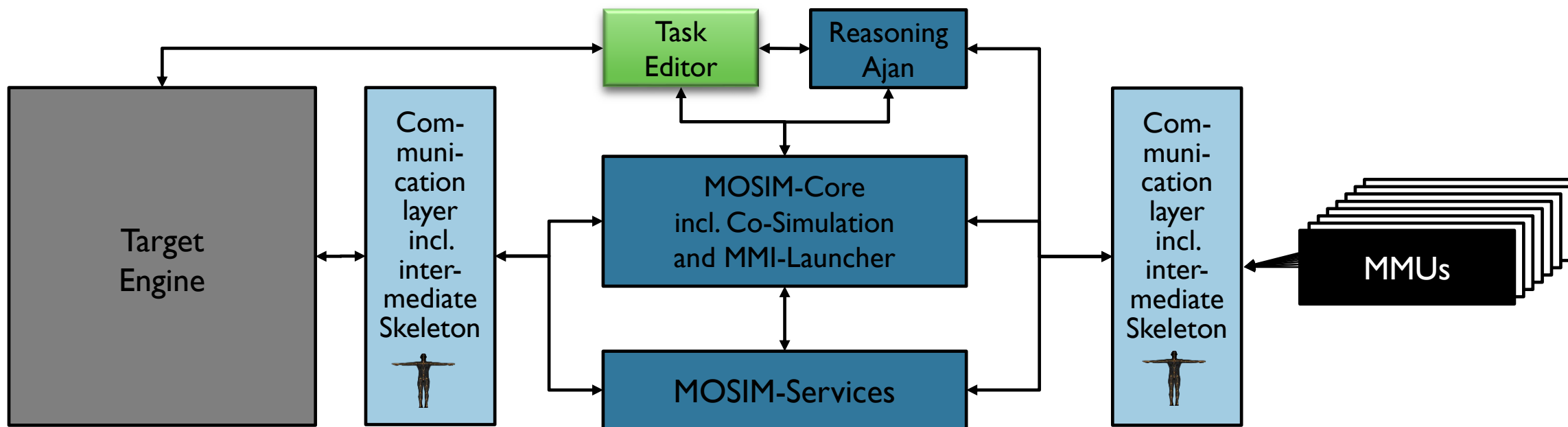


Professional applications



Library of MMUs

Architecture



legend



Standardized Interfaces



Tools & Services



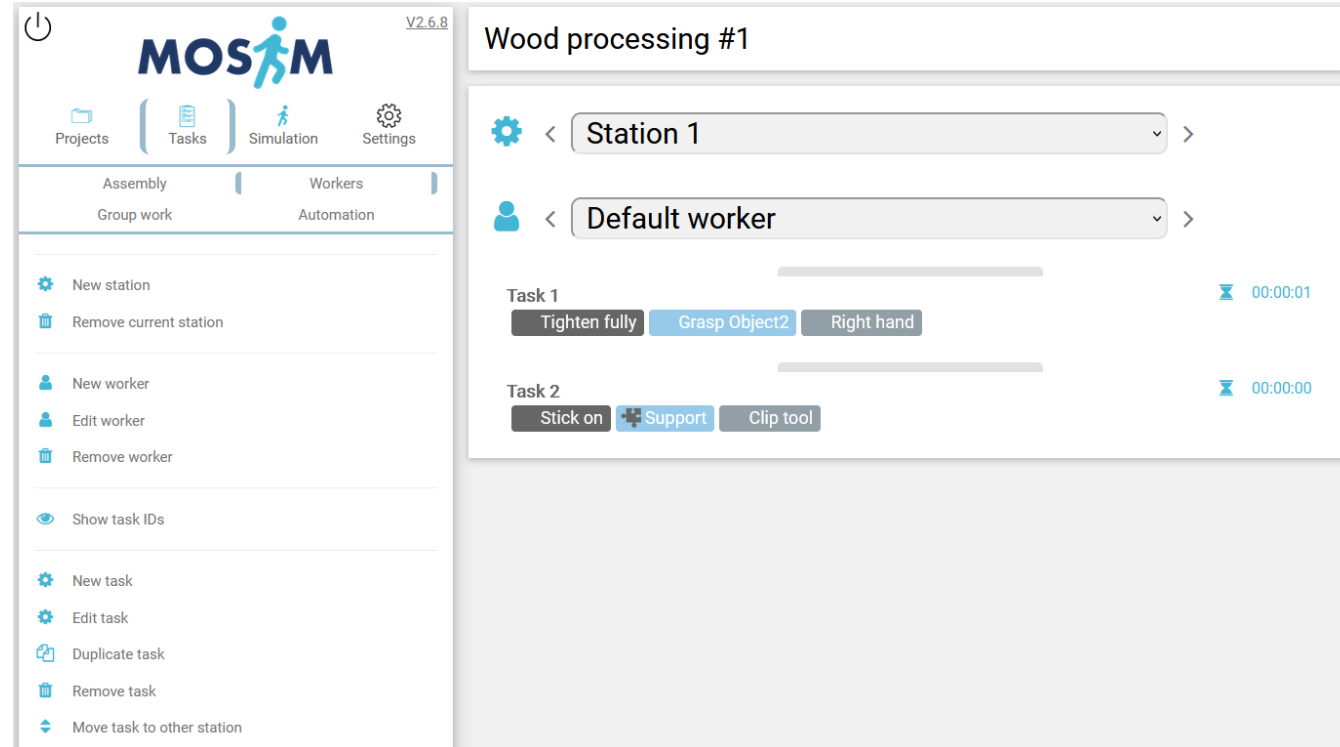
Professional applications



Library of MMUs

Task editor functions

- High-level task editing (tasks are independent from the scene layout and state)
- Station/location definition
- Tool types editing
- Parts classification and grouping
- Definition of avatars and their parameters
- Project user management
- Project related MMU library (sharing MMUs within group)



The screenshot displays the MOSIM software interface for editing tasks. The top navigation bar includes 'Projects', 'Tasks', 'Simulation', and 'Settings'. Below this, there are tabs for 'Assembly' and 'Workers', with sub-tabs for 'Group work' and 'Automation'. The main interface is titled 'Wood processing #1' and features two dropdown menus: 'Station 1' and 'Default worker'. Two tasks are listed:

- Task 1:** Includes steps 'Tighten fully', 'Grasp Object2', and 'Right hand' with a duration of 00:00:01.
- Task 2:** Includes steps 'Stick on', 'Support', and 'Clip tool' with a duration of 00:00:00.

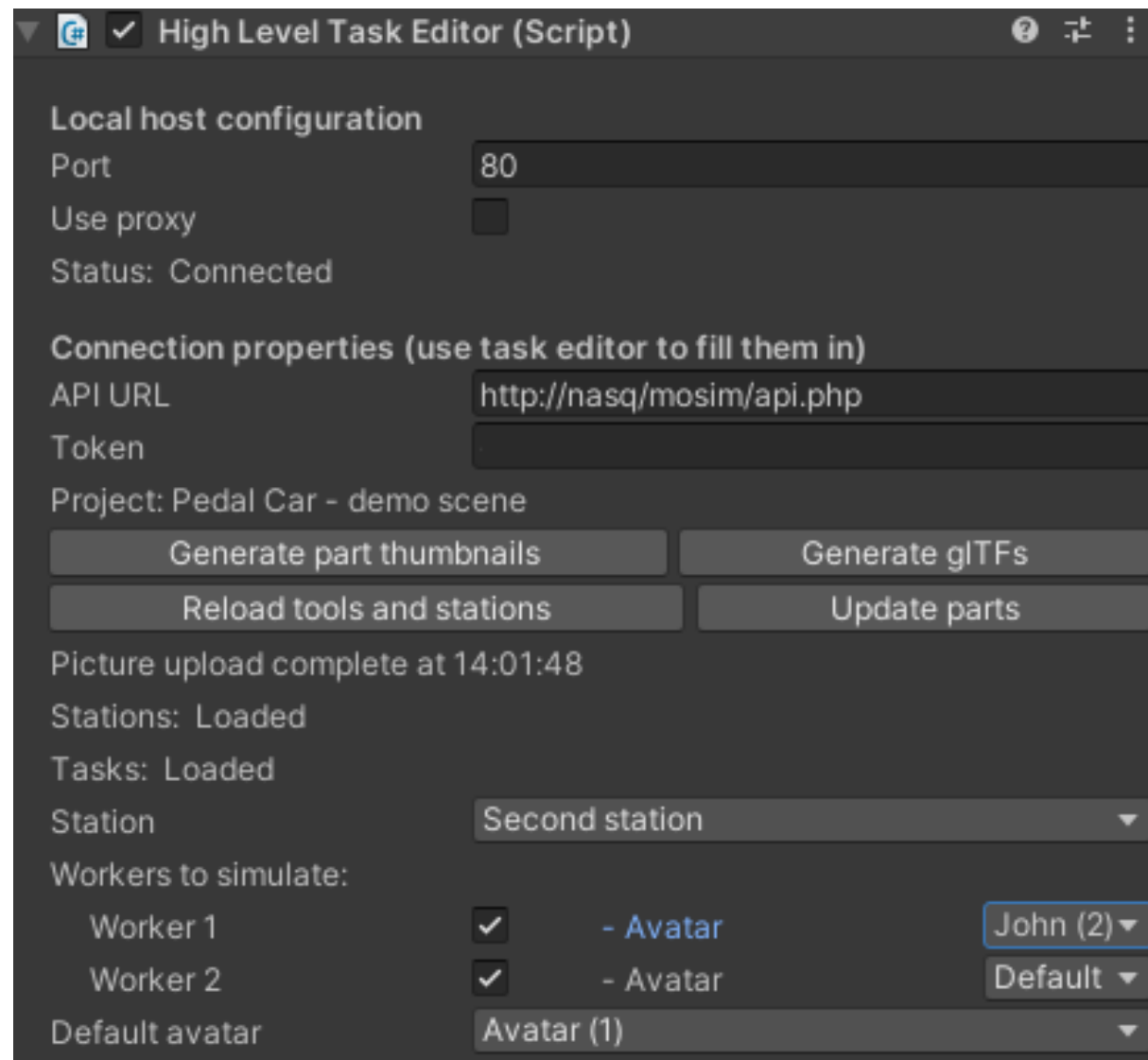
A sidebar on the left provides a list of actions for stations, workers, and tasks, such as 'New station', 'Remove current station', 'New worker', 'Edit worker', 'Remove worker', 'Show task IDs', 'New task', 'Edit task', 'Duplicate task', 'Remove task', and 'Move task to other station'.

Task editor integration to Unity

- Synchronization of parts between Unity and task editor web service (names, place in hierarchy, 3D model, thumbnail)
- List of tool types sync from task editor web service
- Station synchronization
- Avatar to task list assignment
- Task list interface for reasoning agent

In preparation:

- Action-based camera control
- Simulation automation task list (interaction of machines, humans, camera, and events)

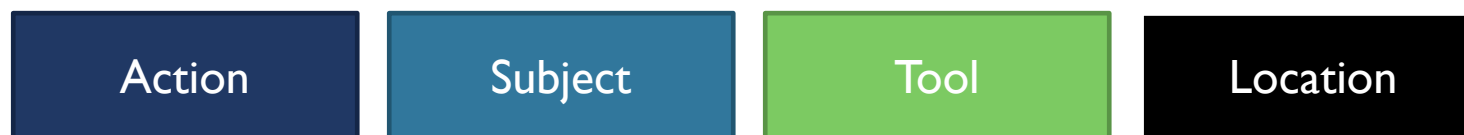


The screenshot shows the 'High Level Task Editor (Script)' window with the following configuration:

- Local host configuration:**
 - Port: 80
 - Use proxy:
 - Status: Connected
- Connection properties (use task editor to fill them in):**
 - API URL: `http://nasq/mosim/api.php`
 - Token: [Redacted]
 - Project: Pedal Car - demo scene
- Buttons:**
 - Generate part thumbnails
 - Generate glTFs
 - Reload tools and stations
 - Update parts
- Status:**
 - Picture upload complete at 14:01:48
 - Stations: Loaded
 - Tasks: Loaded
- Station:** Second station
- Workers to simulate:**
 - Worker 1: - Avatar [John (2)]
 - Worker 2: - Avatar [Default]
- Default avatar:** Avatar (1)

High-level task editing concept

- Task comprises of 4 components



- Examples:

“Position” -> “Engine block” -> “With both hands” -> “Worktable”

“Clean” -> “Large cover” -> “Cloth”

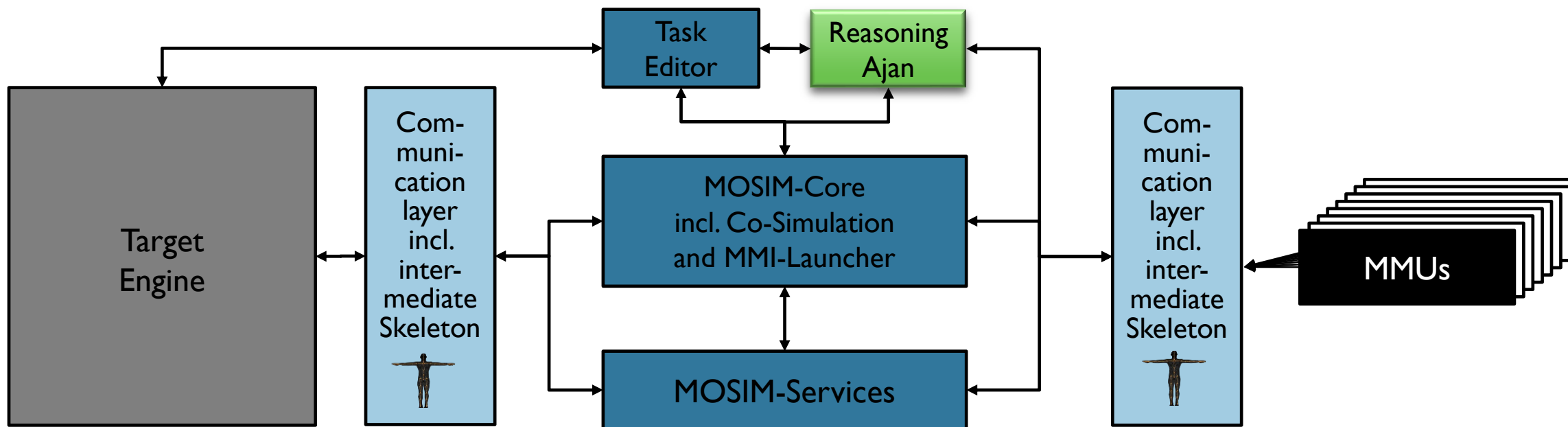
“Tighten fully” -> “M6 screw” -> “EC screwdriver” -> “hole 6”

- Deriving MMU-level tasks (low-level)



- Scene objects are referenced in high-level tasks
- Actual object locations and state are taken from the scene at the simulation execution time
- MMU-level tasks are scene-state specific, while high-level tasks remain unchanged for any scene layout (provided no part required for task has been removed from the scene)

Architecture



legend



Standardized Interfaces



Tools & Services



Professional applications

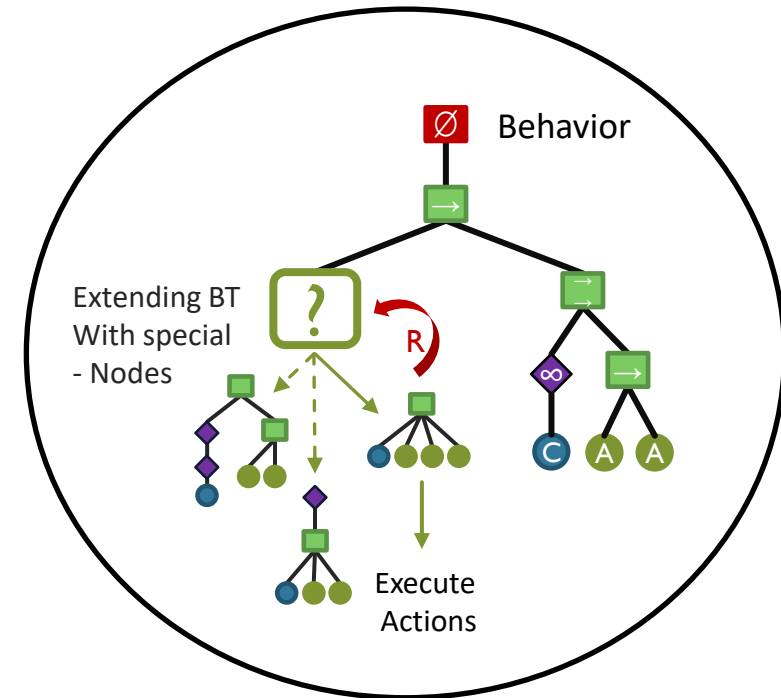


Library of MMUs

Behavior Trees

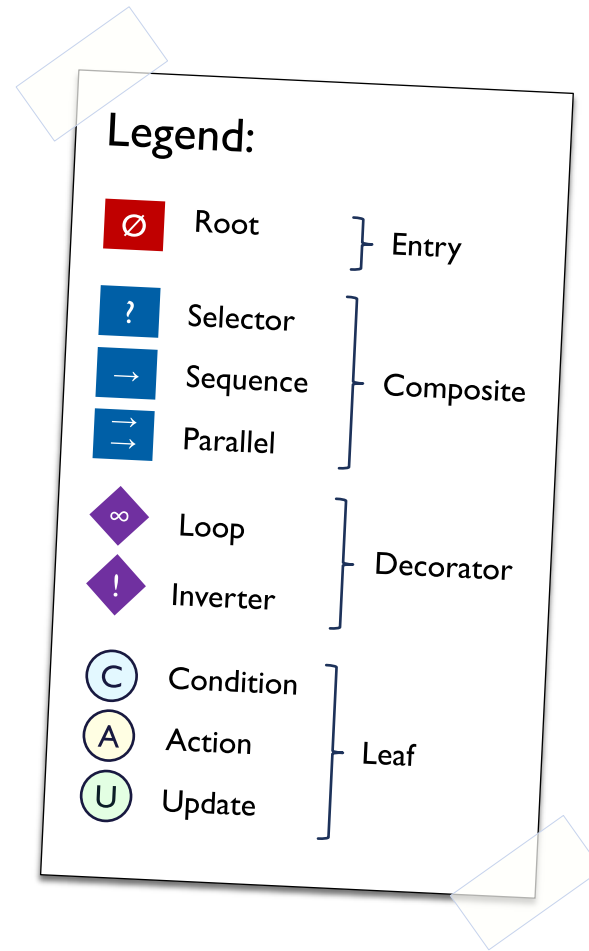
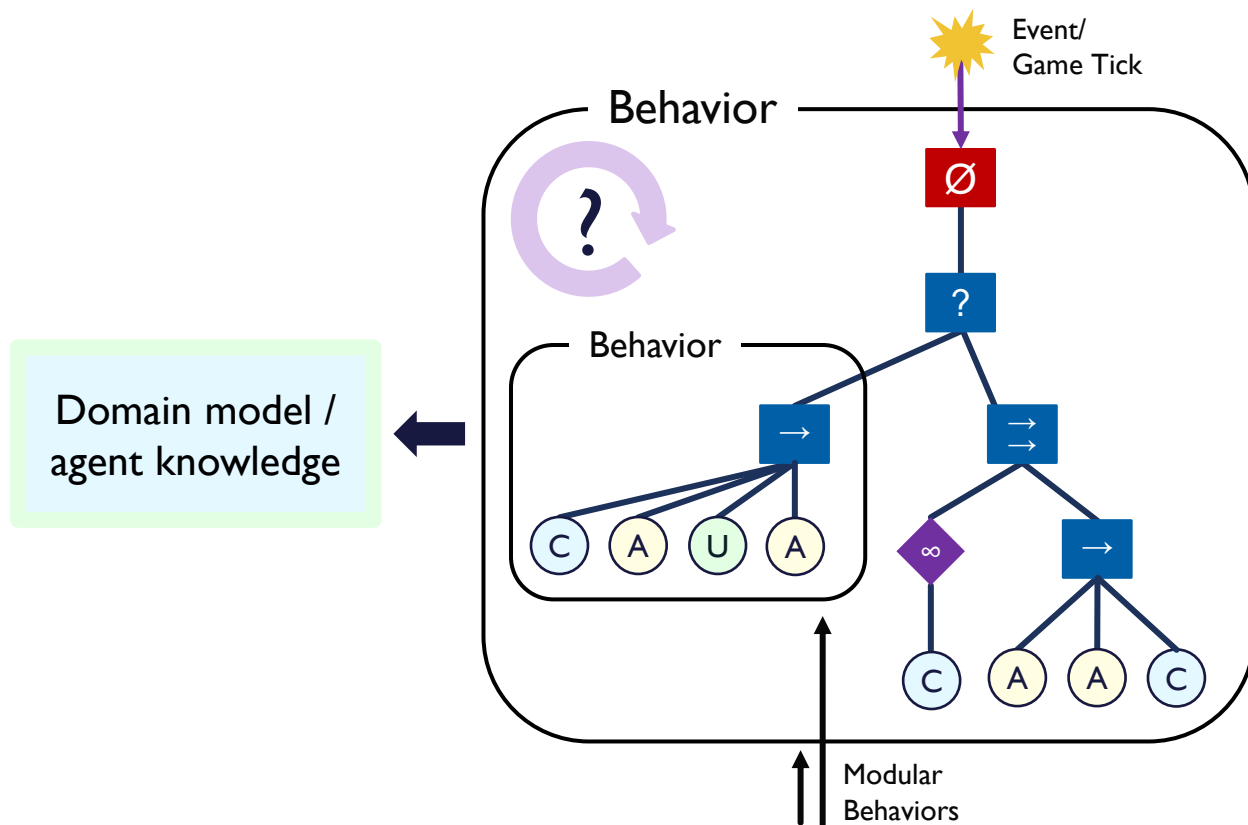


→ BTs are used in various domains



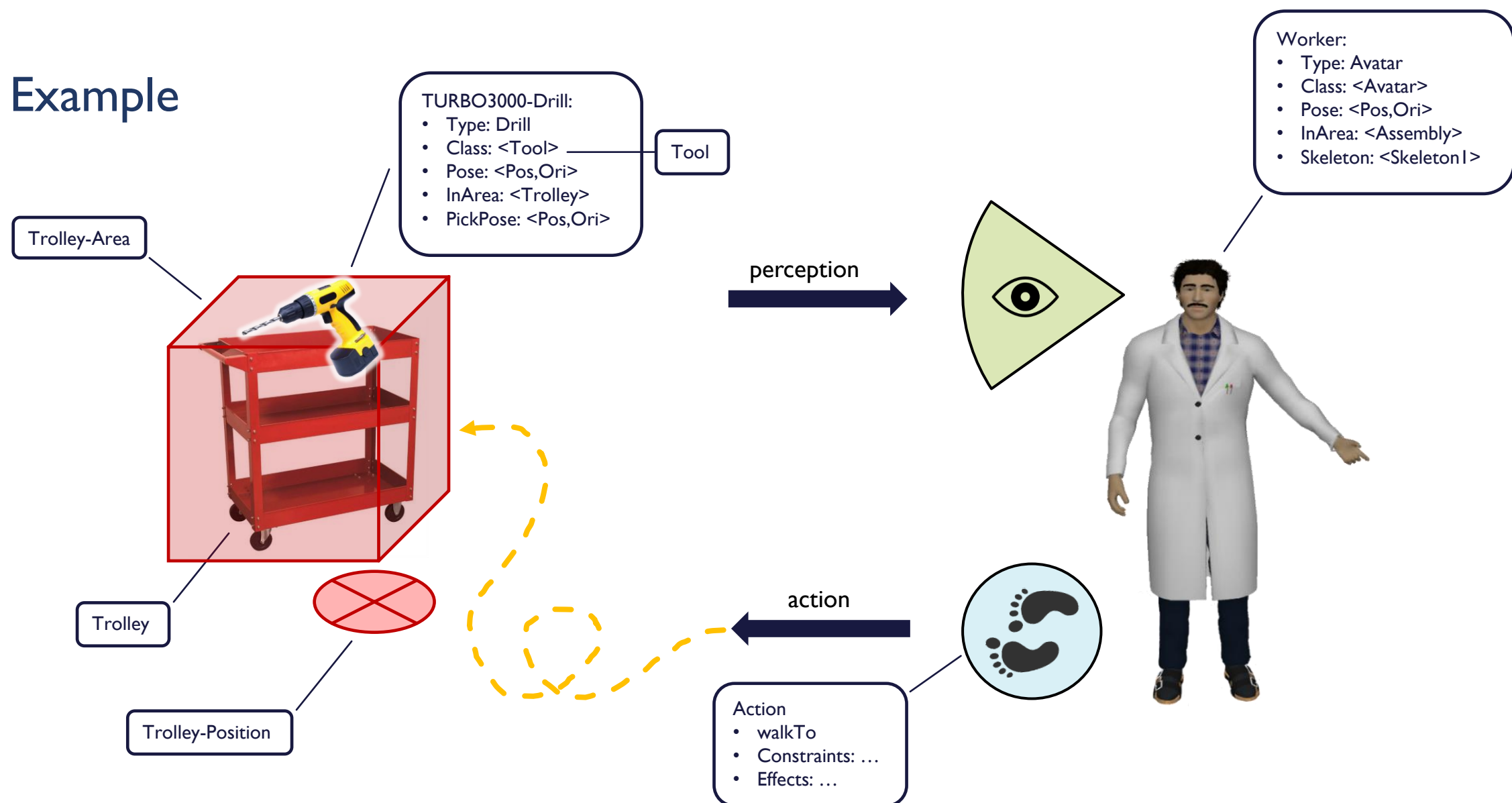
→ BTs are Modular and easy to extend

Behavior Trees

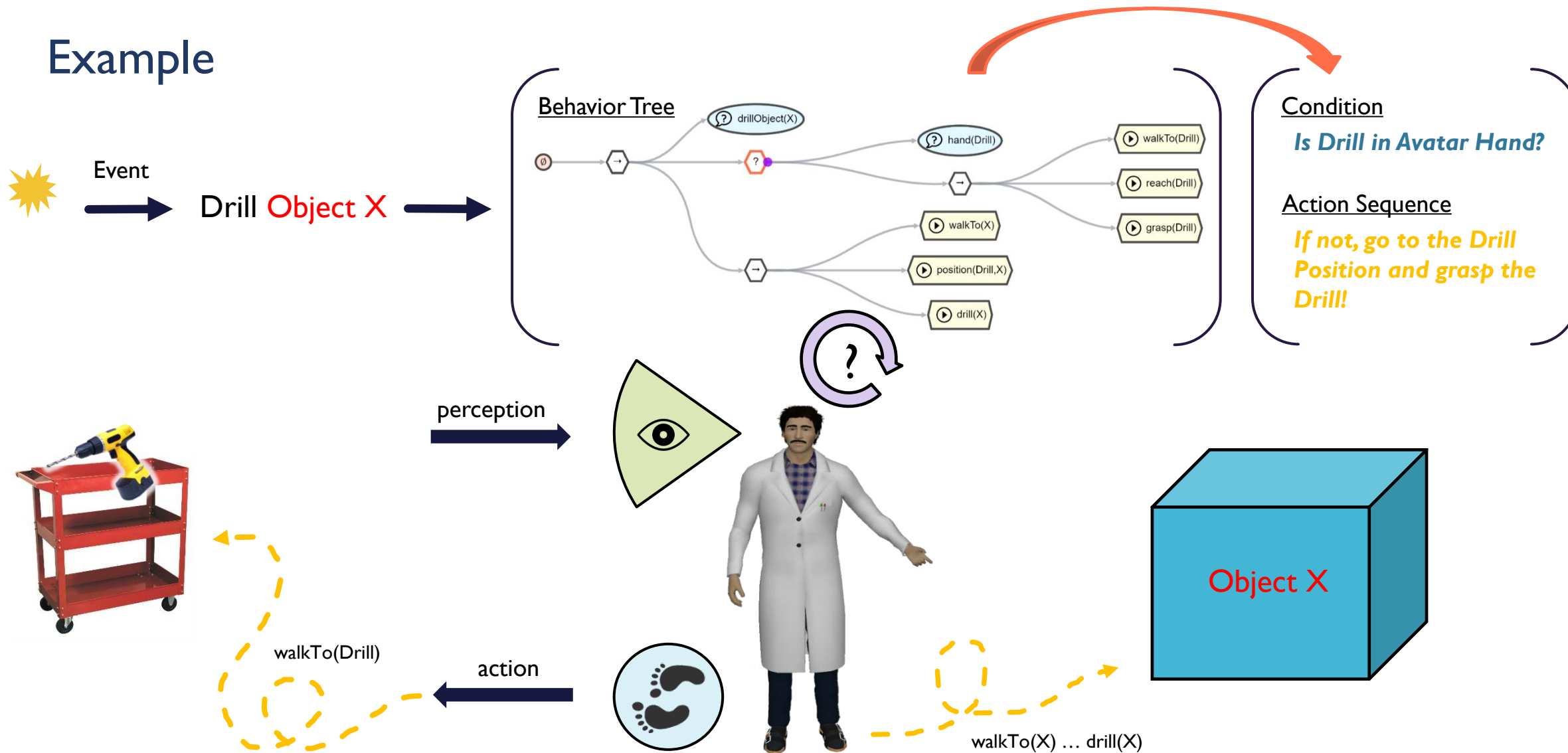


BT = HTN (DT) + FSM

Example



Example



Combination of MMUs in a Co-Simulation Component

- Co-Simulator:
 - Same interface as MMU
 - Central simulation components
 - Forwards Instructions to different MMUs
 - Combines poses of different MMUs

