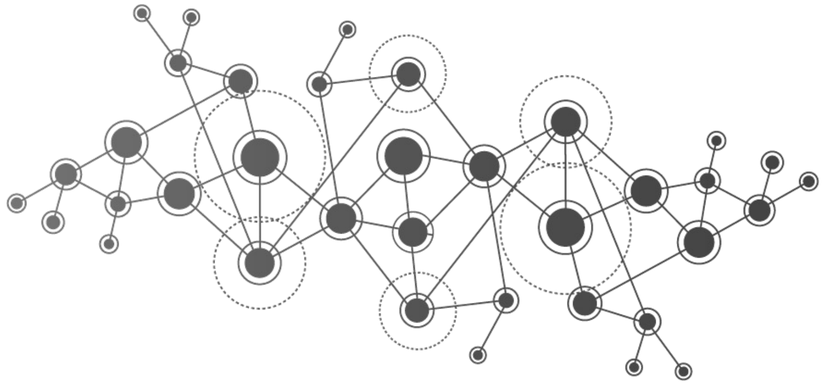
UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA



Intelligent Internet of Things

IoT Digital Twins
HandsOn Session

Prof. Marco Picone

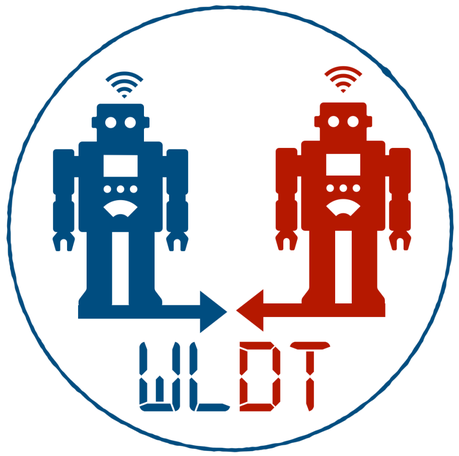
A.A 2023/2024

IoT & Digital Twin HandsOn Session

- Open Source Digital Twins & White Label Digital Twin Library (WLDT)
- DT Software Abstraction
- DT Core Modules
- State Synchronization
- Event Driven Modeling
- Shadowing Process
- Life Cycle
- WLDT Software Architecture
- Demo & Use Cases

2023 - Marco Picone, Ph.D. (marco.picone@unimore.it)

White Label Digital Twin - Java Library



A Java Framework to easily create Digital Twins for your Internet of Things Applications

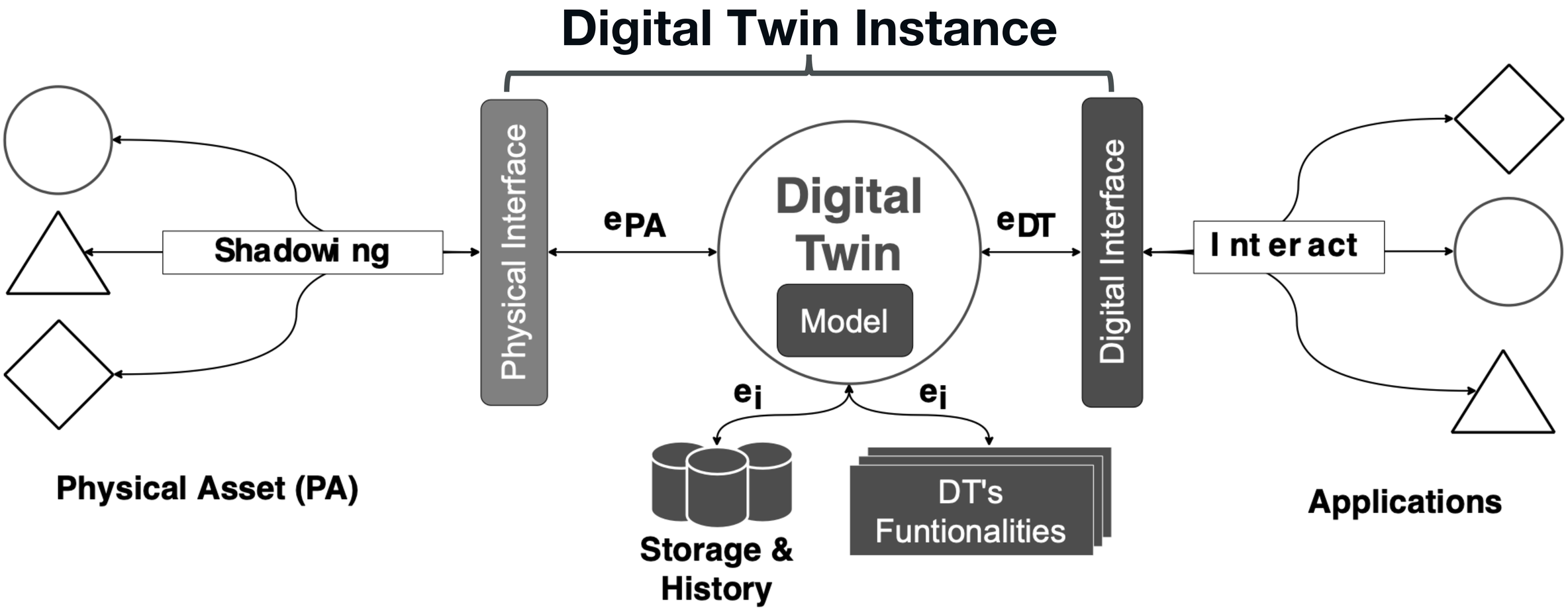
https://github.com/wldt

- The WLDT intends to maximize modularity, re-usability and flexibility in order to effectively mirror physical smart objects in their digital counterparts.
- The proposed library focuses on the simplification of twins design and development aiming to provide a set of core features and functionalities for the widespread adoption of Internet of Things digital twins applications.

White Label Digital Twin - Java Library

- A White Label Digital Twin (WLDT) instance is a general purpose software agent implementing all the features and functionalities of a Digital Twin running in cloud or on the edge
- It has the peculiar characteristic to be generic and ``attachable'' to any physical thing in order to impersonate and maintain its digital replica and extend the provided functionalities for example through the support of additional protocols or a specific translation or normalization for data and formats
- Hereafter, the requirements that led the design and development of the WLDT framework are:
    - **Simplicity** - with WLDT developers must have the possibility to easily create a new instance by using existing modules or customizing the behavior according the need of their application scenario;
    - **Extensibility** - while WLDT must be as simple and light as possible, the API should be also easily extendible in order to let programmers to personalize the configuration and/or to add new features loading and executing multiple modules at the same times;
    - **Portability & Micorservice Readiness** - a digital twin implemented through WLDT must be able to run on any platform without changes and customization. Our goal is to have a simple and light core engine with a strategic set of IoT-oriented features allowing the developer to easily create DT applications modeled as independent software agents and packed as microservices.

Digital Twin Abstraction



Digital Twin Core Modules

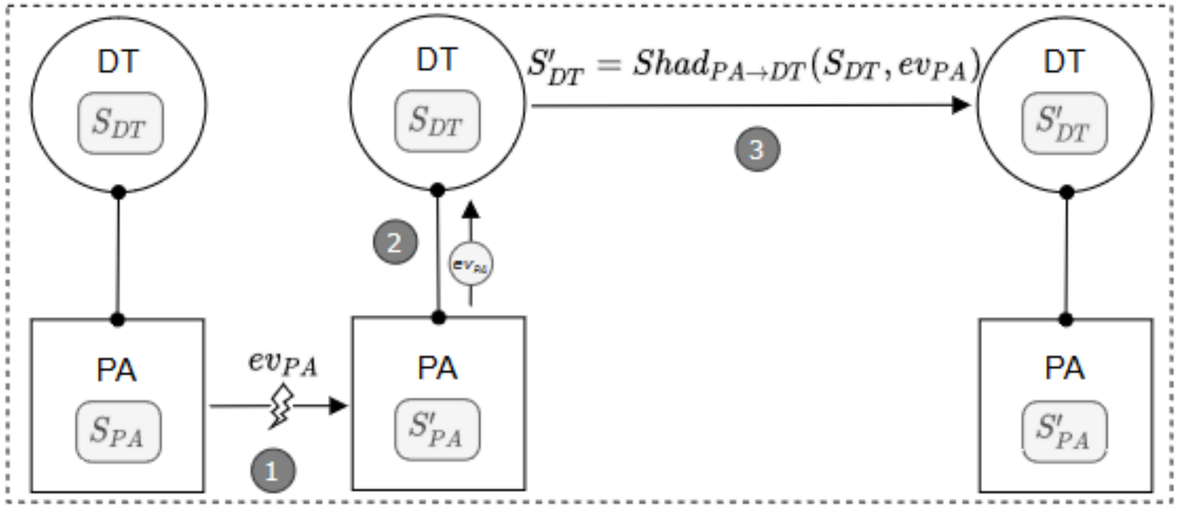
- **Physical Interface:** The entity in charge of both the initial digitalization o shadowing process and the perpetual responsibility to keep the DT and PA in synch during its life cycle. It can execute multiple Physical Asset Adapters to interact with the PA and detect and digitalize the physical event coming from the physical entity according to its nature and the supported protocols and data formats (e.g., through HTTP and JSON).
- **Digital Interface:** The component complementary to the Physical Interface and in charge of handling DT's internal variations and events towards external digital entities and consumers. It executes multiple and reusable Digital Adapters in charge of handling digital interactions and events and responsible for making the DT interoperable with external applications.
- **DT's Model:** The module defining the DT's behaviour and its augmented functionalities. It supports the execution of different configurable and reusable modules and functionalities handling both physical and digial events according to the implemented behaviour. Furthermore, the Model **(M)** is the component responsible to handle and keep updated the Digital Twin State as described in the following sections.

Digital Twin - State

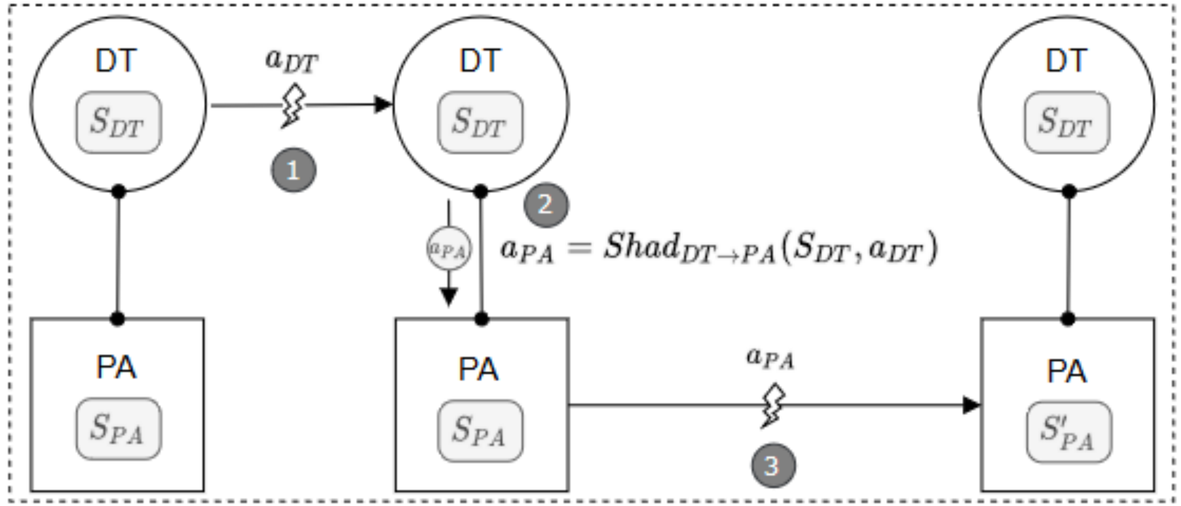
- Each DT is thus equipped with an internal model, which defines how the PA is represented in the digital level
- The DT's representation denoted as **Digital Twin State** supported and defined through **M** is defined in terms of:
  - **Properties:** represent the observable attributes of the corresponding PA as labeled data whose values can dynamically change over time, in accordance with the evolution of the PA's state.
  - **Events:** represent the domain-level events that can be observed in the PA.
  - **Relationships:** represent the links that exist between the modeled PA and other physical assets of the organizations through links to their corresponding Digital Twins. Like properties, relationships can be observed, dynamically created, and change over time, but unlike properties, they are not properly part of the PA's state but of its operational context (e.g., a DT of a robot within a production line).
  - **Actions:** represent the actions that can be invoked on the PA through interaction with the DT or directly on the DT if they are not directly available on the PA (the DT is augmenting the physical capabilities).

Digital Twin - Physical & Digital State Synchronization

- Once the model M is defined, the dynamic state of the DT ($\mathbf{S_{DT}}$) can be defined by through the combination of its:
  - properties
  - events
  - relationships
  - actions
- Each $\mathbf{S_{DT}}$ is associated to the DT timestamp that represents the current time of synchronization between the physical and digital counterparts

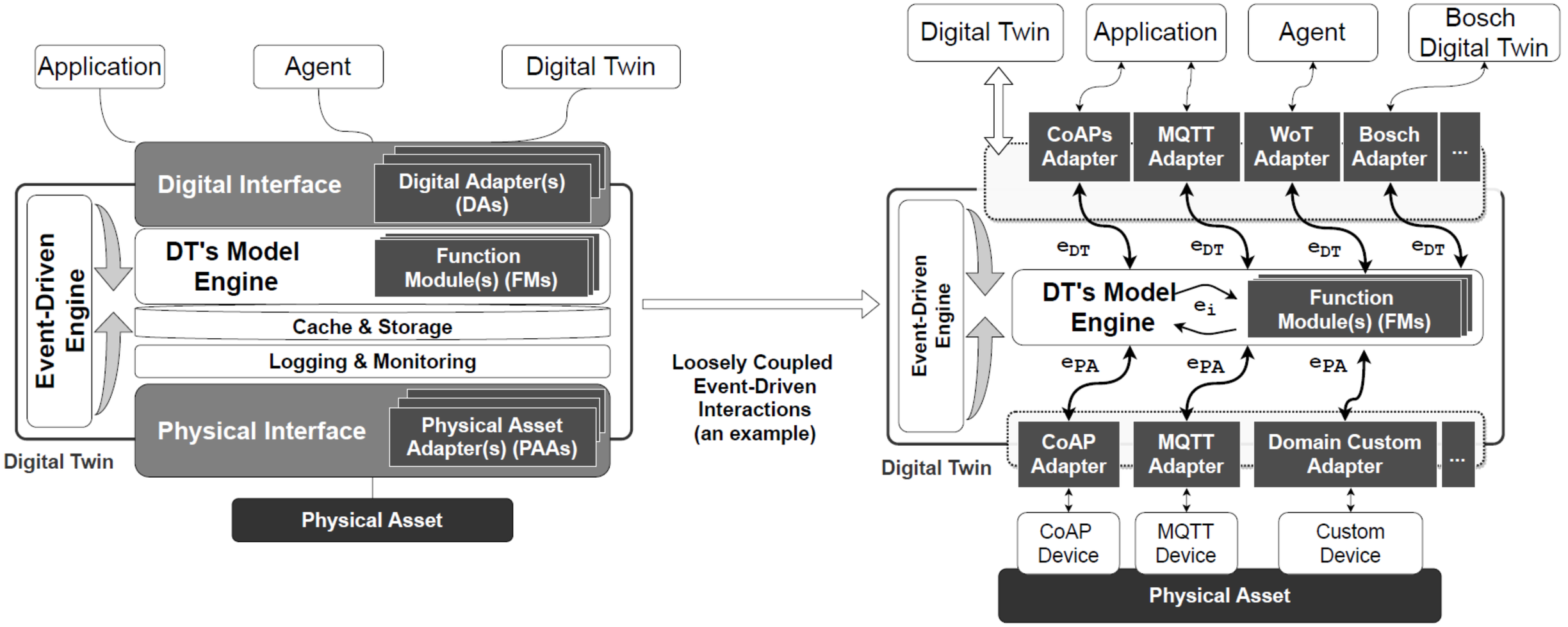


Physical -> Digital

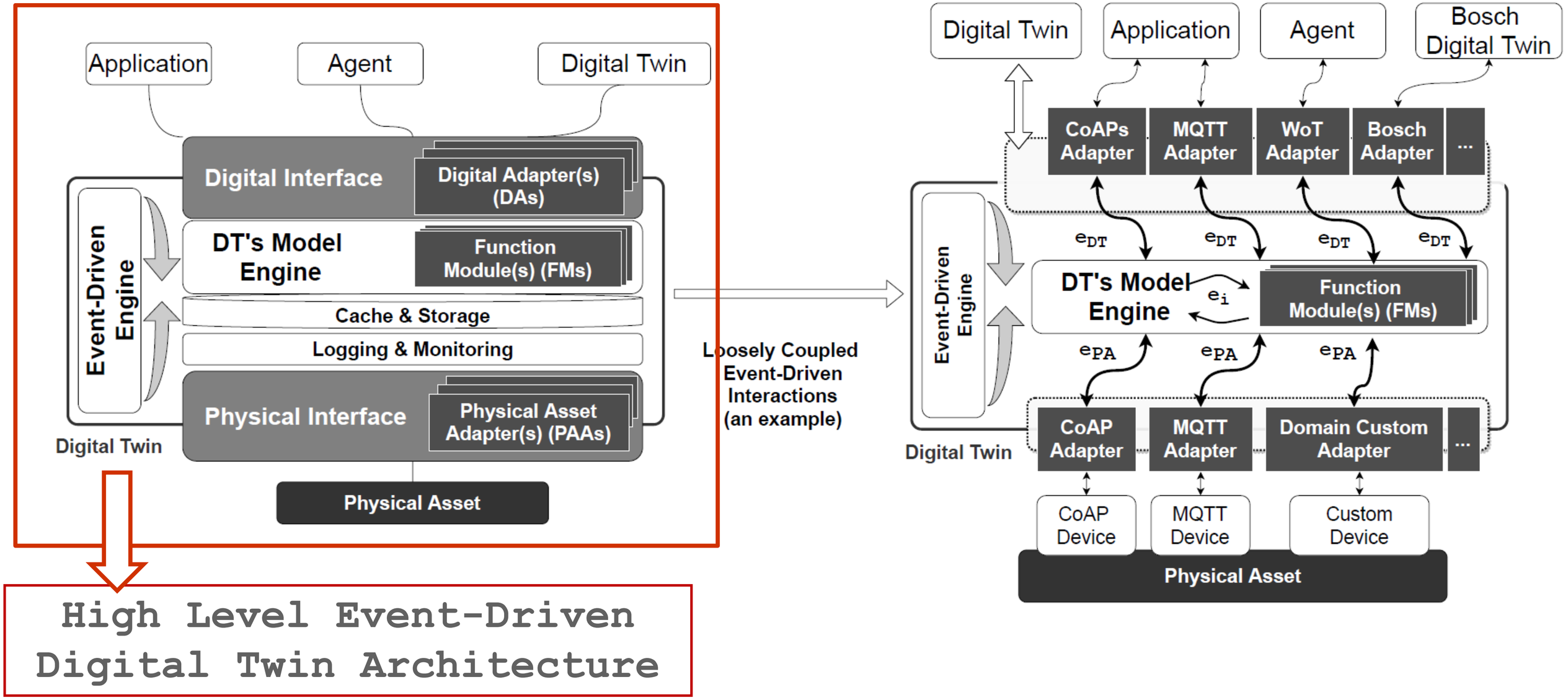


Digital -> Physical

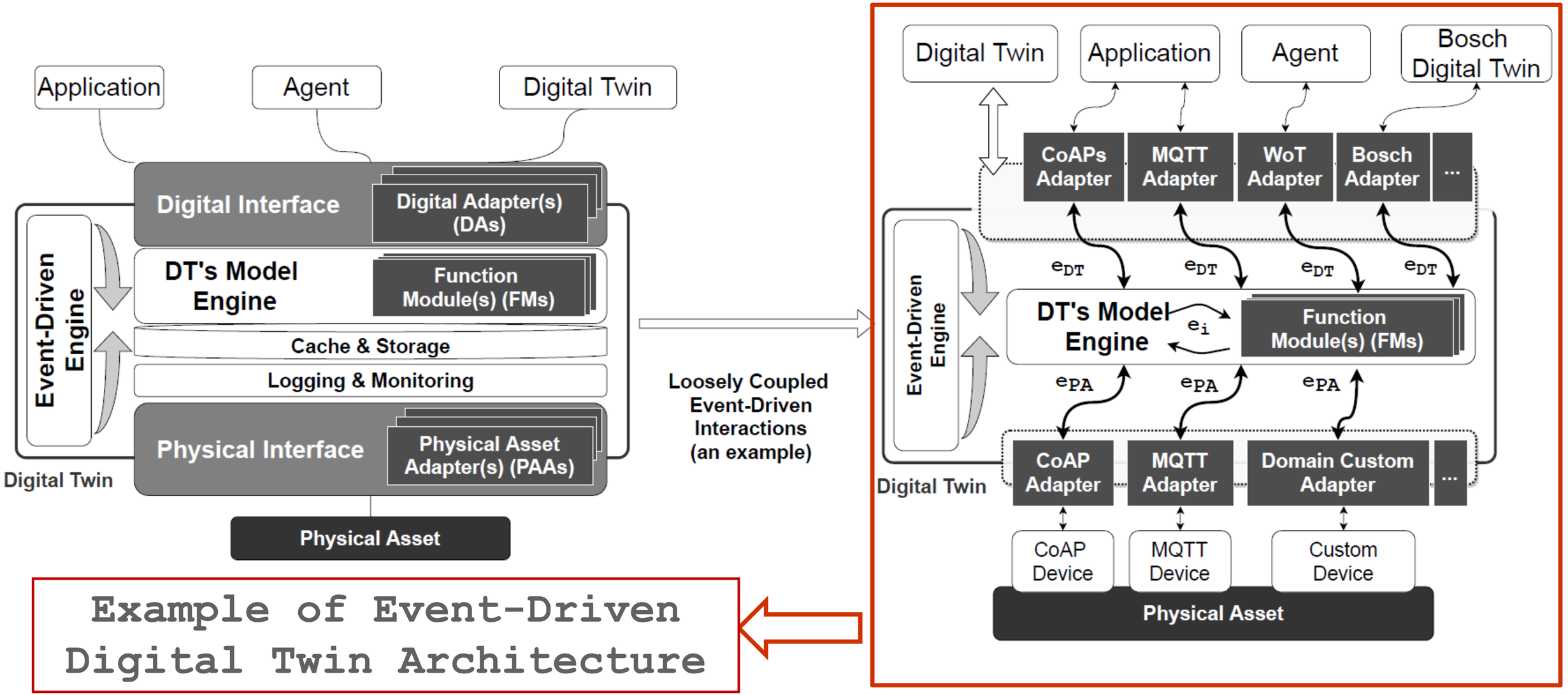
Digital Twin - Event Driven Modeling



Digital Twin - Event Driven Modeling

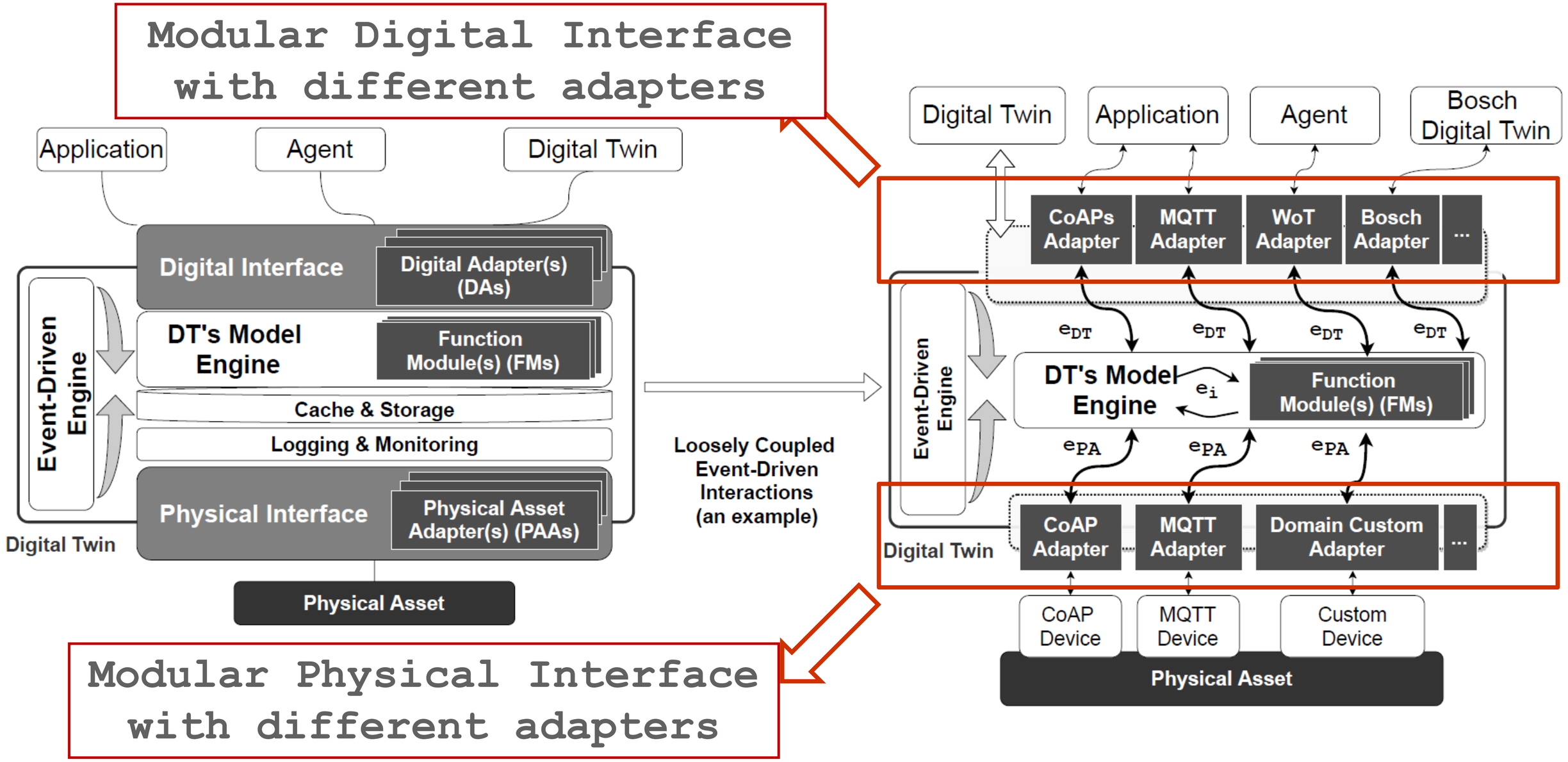


Digital Twin - Event Driven Modeling



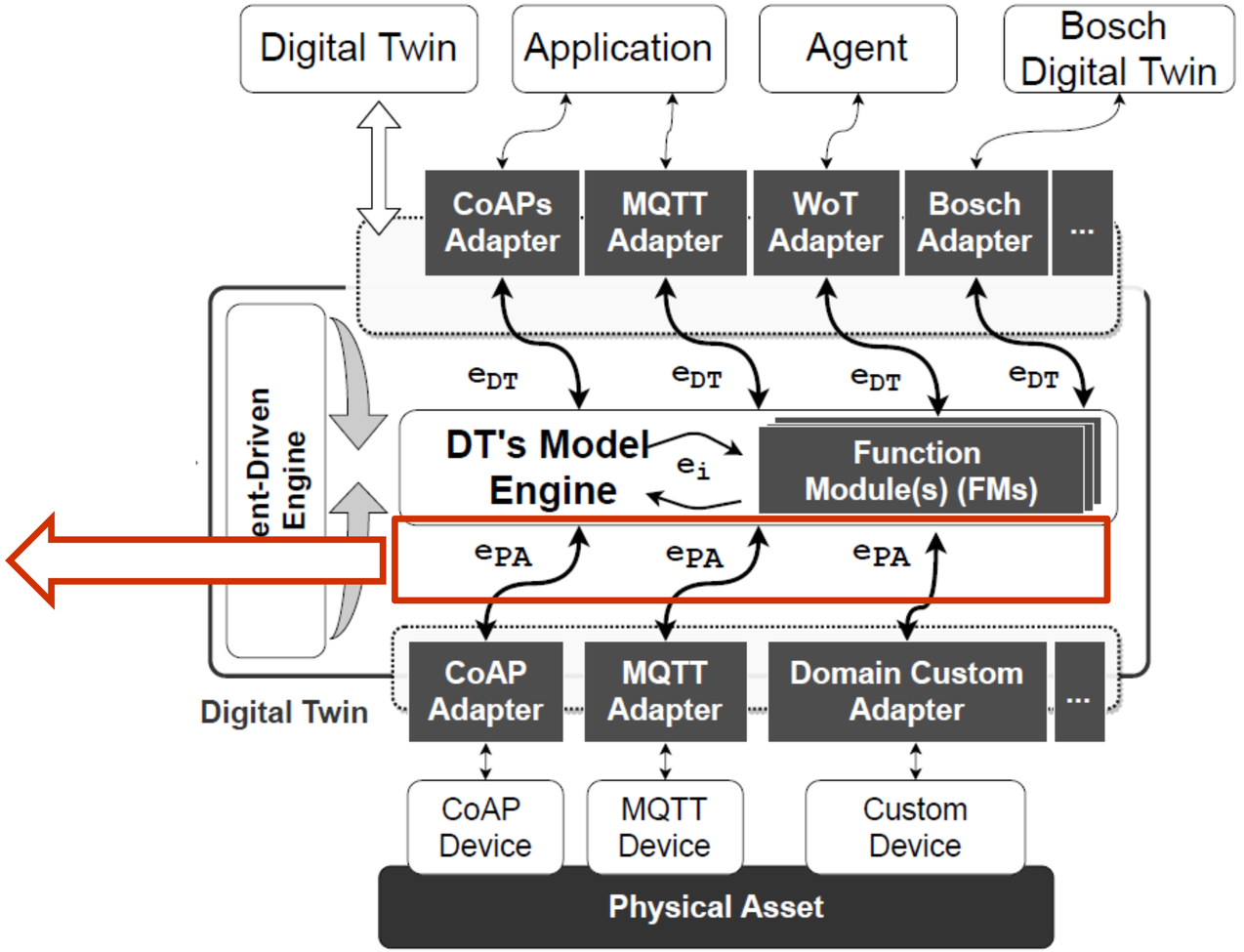
Example of Event-Driven Digital Twin Architecture

Digital Twin - Event Driven Modeling



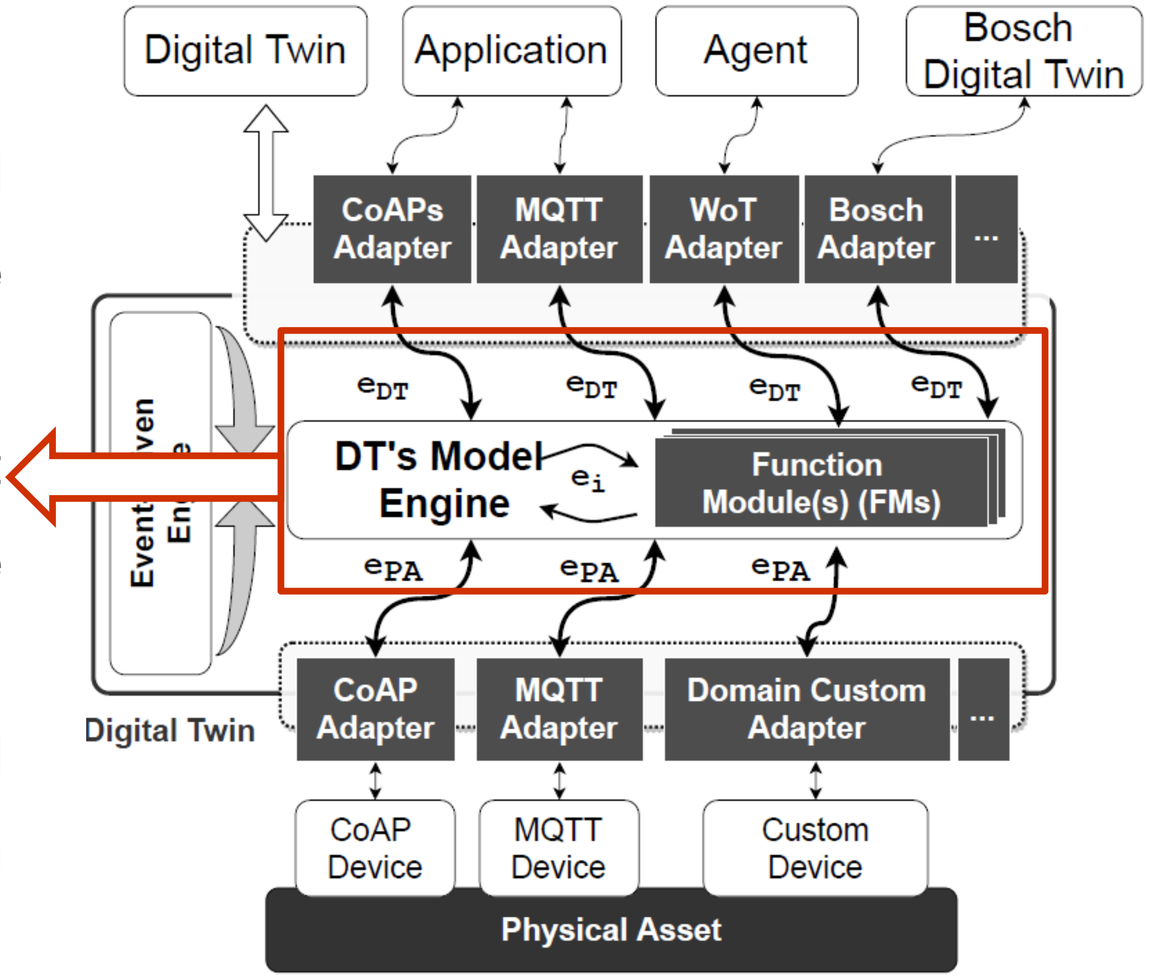
Digital Twin - Event Driven Modeling

Digital Interface with its modules generated e_{PA} (Events of Physical Assets)to notify variations of the Physical State and receive other ePA to invoke actions



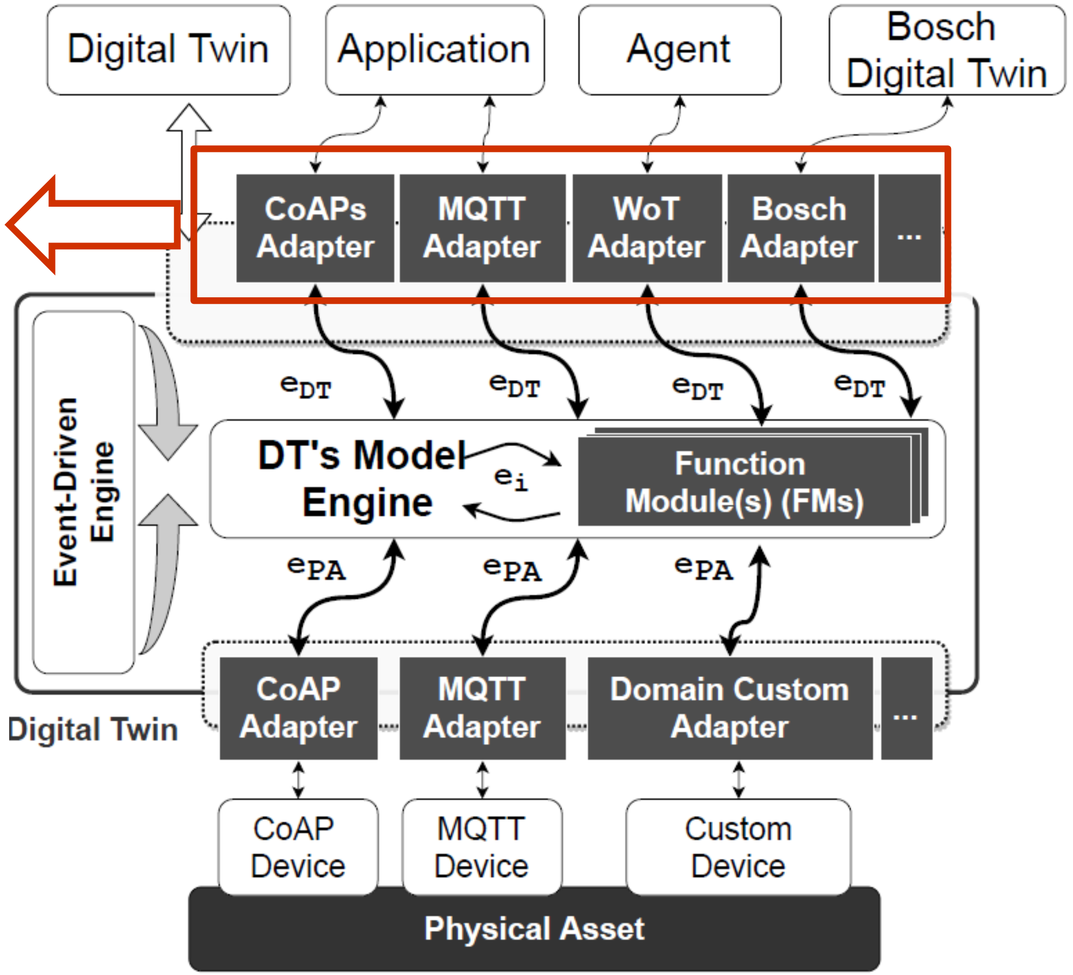
Digital Twin - Event Driven Modeling

- The Core Engine of the Digital Twin with its internal Model and the associated Functions (e.g., Augmentation Functions)receive incoming $\mathbf{e_{PA}}$ and compute (if required by the Model) the new DT State $\mathbf{S_{DT}}$
- When a new SDT has been computed, a DT Event ($\mathbf{e_{DT}}$) will be generated to notify the Digital Interface and its adapters
- The Core Engine can also receive $\mathbf{e_{DT}}$ from the Digital Interface associated to actions requests coming from the digital space



Digital Twin - Event Driven Modeling

- When the Digital Interface receives a new $\mathbf{e_{DT}}$ from the Digital Twin Engine forwards it to available Digital Adapters in order to expose the new variation associated to the $\mathbf{S_{DT}}$ to external digital application through different protocols
- An action request sent through a specific protocol to a Digital Adapter of the Digital Interface will managed by the adapter and a new $\mathbf{e_{DT}}$ will be generated and forwarded to the Digital Twin Engine in order to be processed by the DT's Model and if needed sent to the Physical Interface to propagate the action to the associated Physical Asset



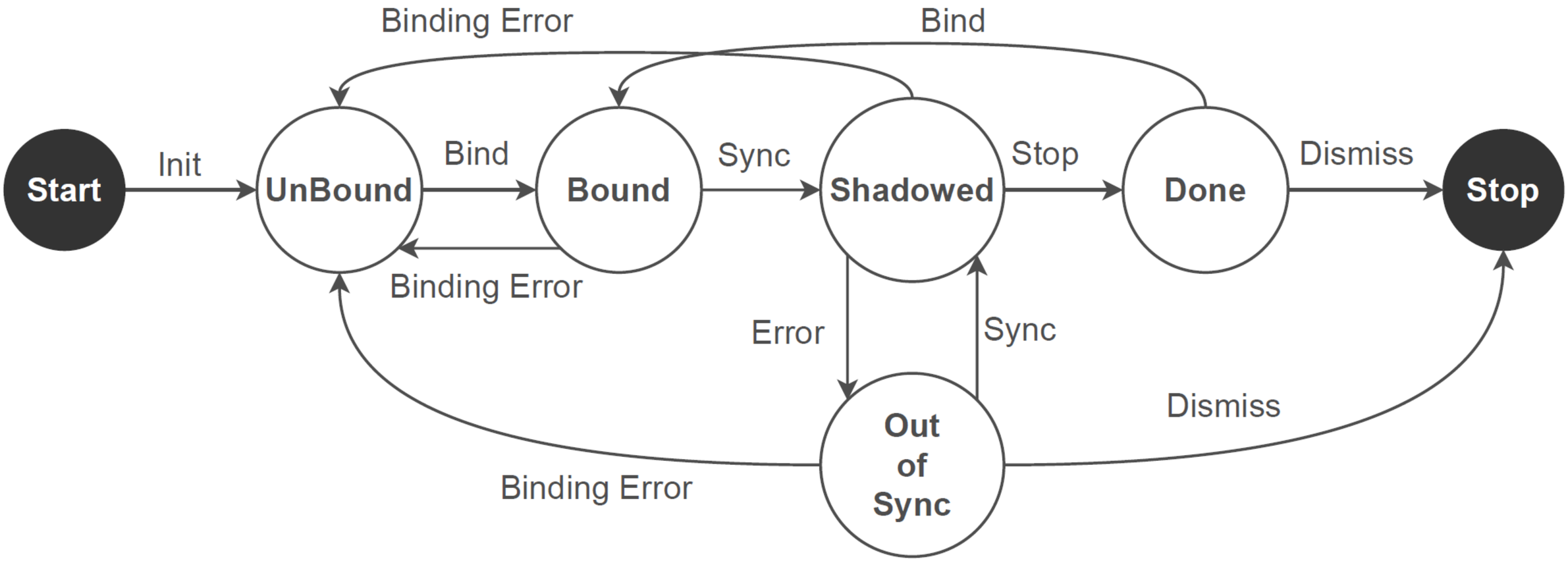
Digital Twin - Shadowing Process

- The **shadowing process (also known as replication of digitalization)** allows to keep the Digital Twin State synchronized with that of the corresponding physical resource according to what is defined by the model M
- Specifically, each relevant update of the PA state ($\mathbf{S_{PA}}$) is translated into a sequence of 3 main steps:
  - each relevant change in physical asset state is modeled by an event denoted $\mathbf{e_{PA}}$;
  - the event is propagated to the DT;
  - given the new the DT's is updated through the application of a shadowing function, which depends on the model **M**
- The shadowing process allows also the DT to reflect and invoke possible actions of the PA. The DT receives an action request on its digital interface generated an $\mathbf{e_{DT}}$, applies the shadowing function to validate it and then propagates the request through its physical interface
- An important aspect to emphasize is that the request for a digital action does not directly change the state of the DT since any changes can only occur as a result of the shadowing function from the PA to the DT, as described earlier

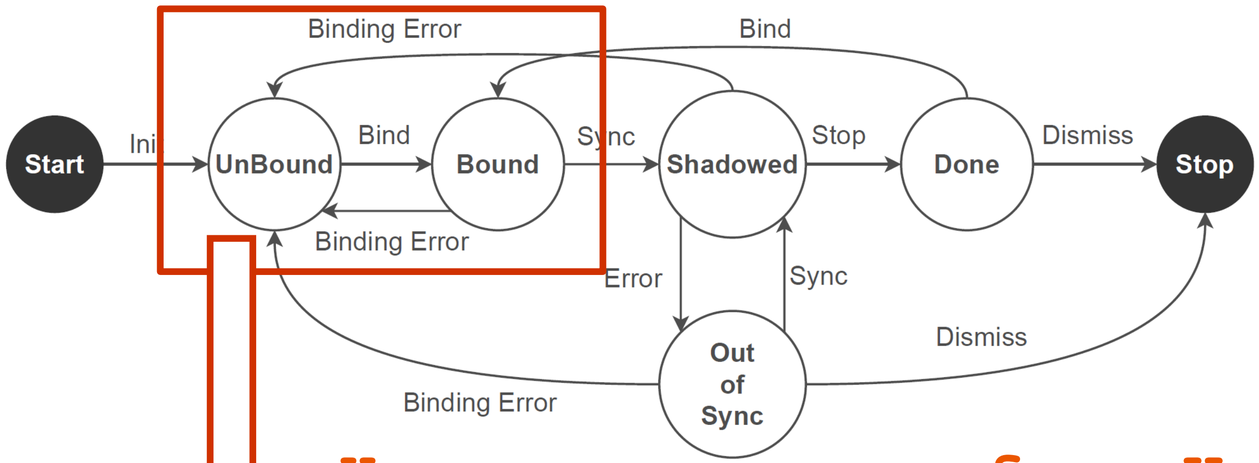
Digital Twin - Life Cycle

- The modeling of the concept of DT includes also the definition and characterization of its life cycle. Based on the scientific literature, we modeled a life cycle with 5 states through which the DT goes from when it is executed to when it is stopped. Involved Steps are the following:
  - **Operating & Not Bound:** this is the state in which the DT is located following the initialization phase, indicating that all internal modules of the DT are active but there is no association yet with the corresponding PA.
  - **Bound:** this is the state in which the DT transitions following the correct execution of the binding procedure. The binding procedure allows to connect the two parts and enables bidirectional flow of events.
  - **Shadowed:** this is the state reached by the DT when the shadowing process begins and its state is correctly synchronized with that of the PA.
  - **Out of Sync:** this is the state that determines the presence of errors in the shadowing process. When in this state, the DT is not able to handle either state alignment events or those generated by the application layer.
  - **Done:** this is the state that the DT reaches when the shadowing process is stopped, but the DT continues to be active to handle requests coming from external applications.

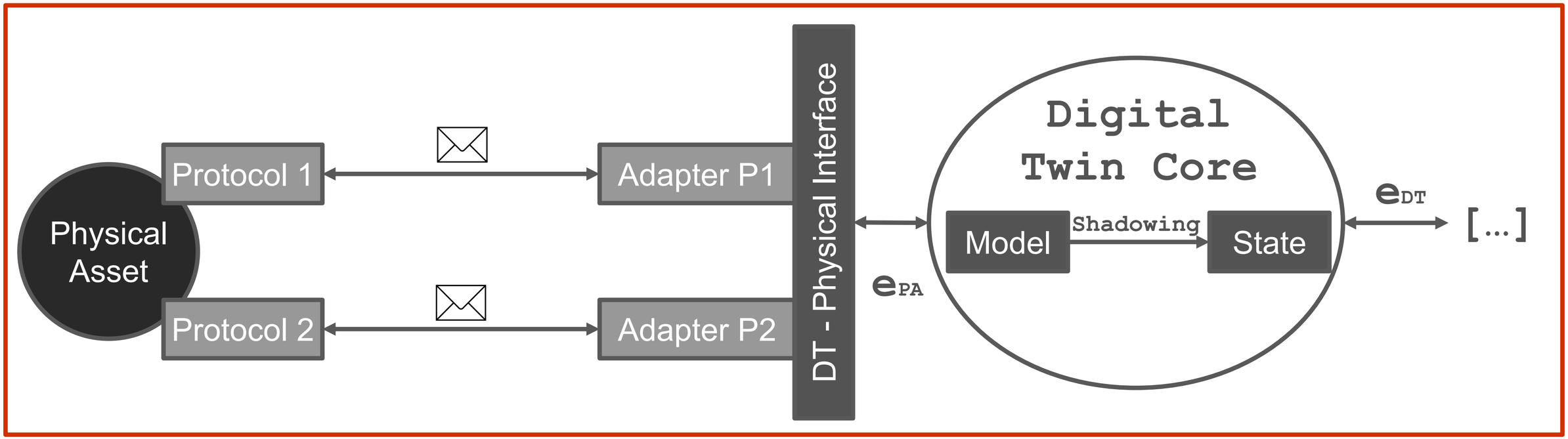
Digital Twin - Life Cycle



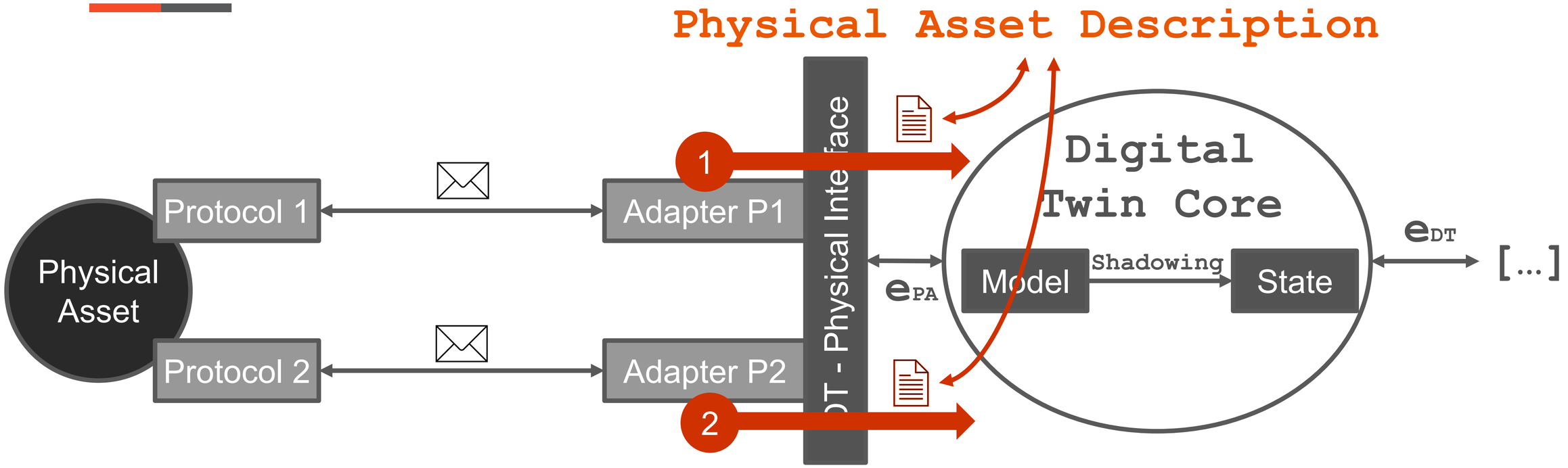
The Physical Binding



How can we move from Unbound to Bound ?



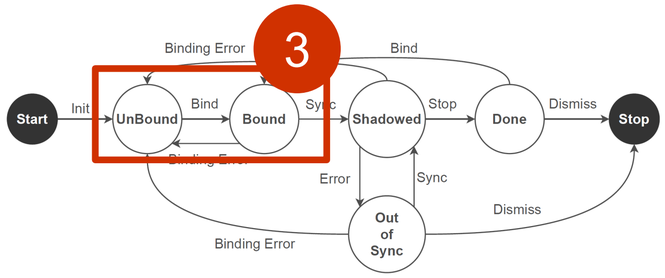
The Physical Binding - Physical Asset Description



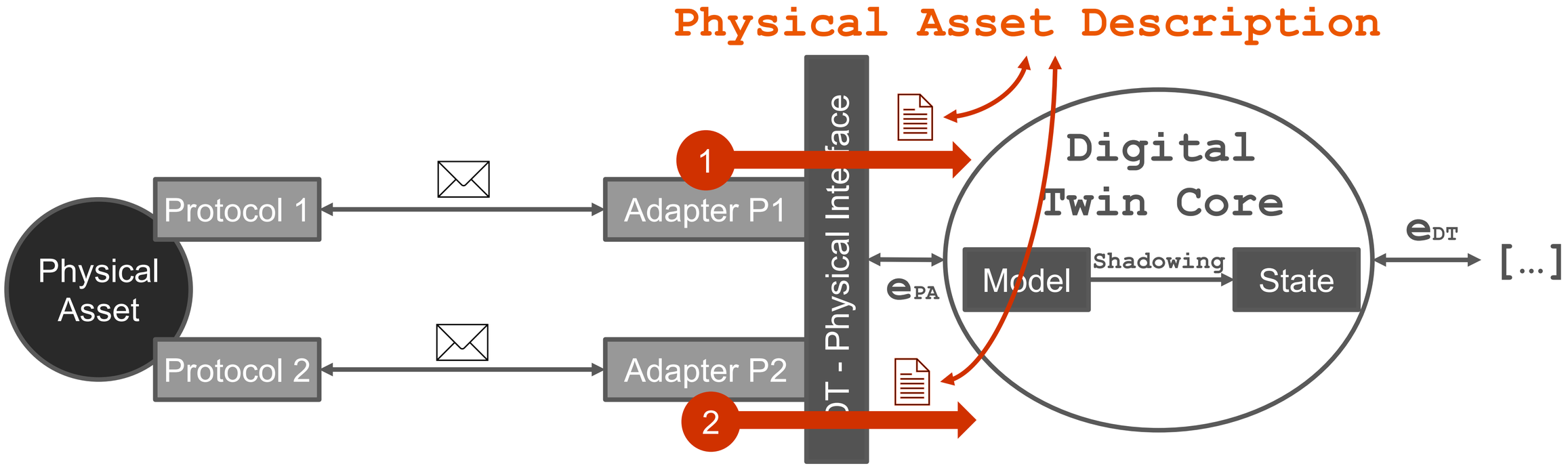
1. The **Adapter P1** communicates with the PA through Protocol 1 and **provides a Description of the Physical Asset** from its perspective

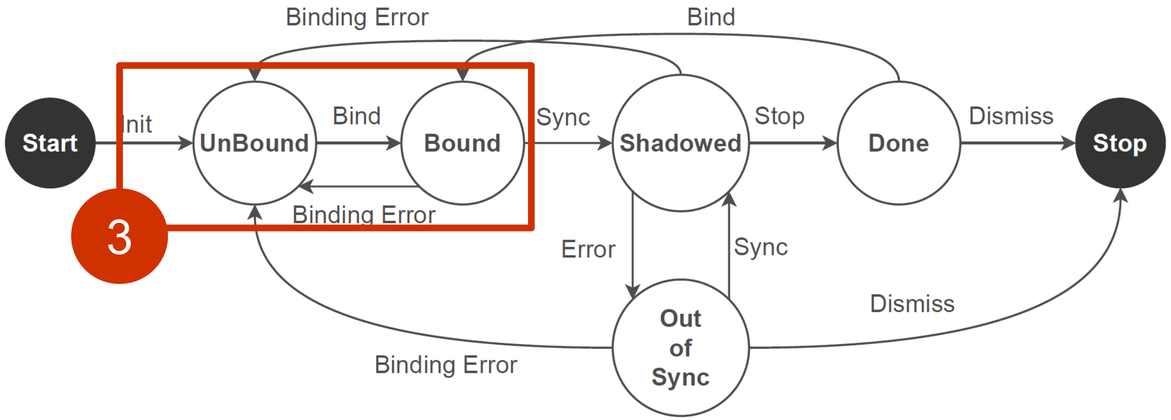
2. The **Adapter P2** communicates with the PA through Protocol 2 and **provides a Description of the Physical Asset** from its perspective

3. **Only when all Physical Adapters have been correctly bound** (it may require time) to the Physical Asset and the associated Physical Asset Descriptions have been generated, **the DT can move from UnBound to Bound**



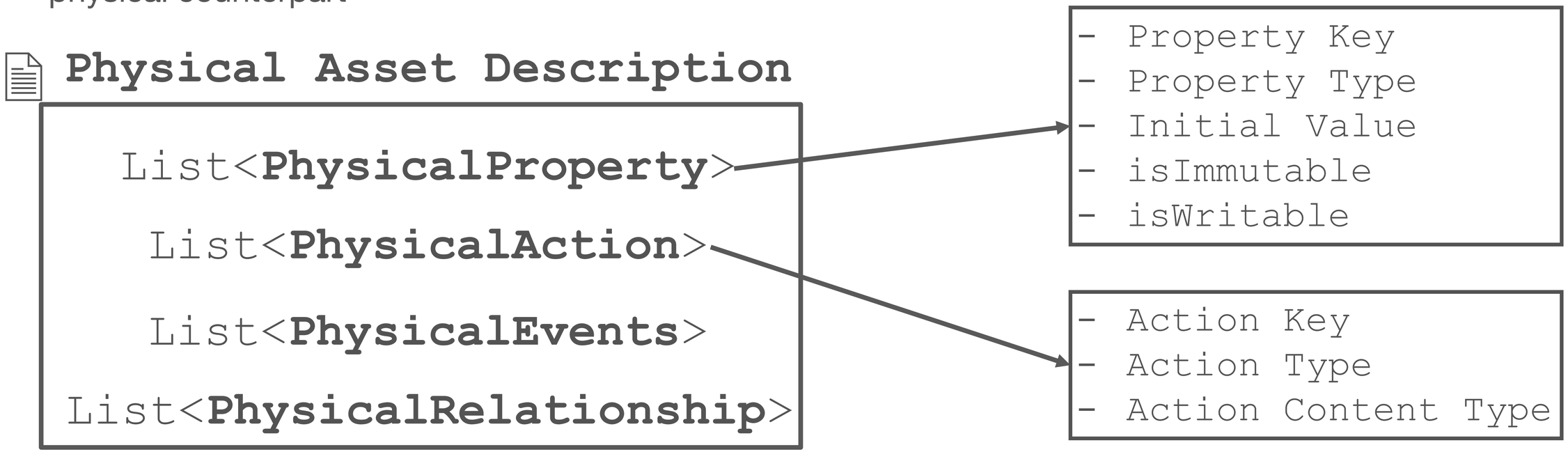
The Physical Binding



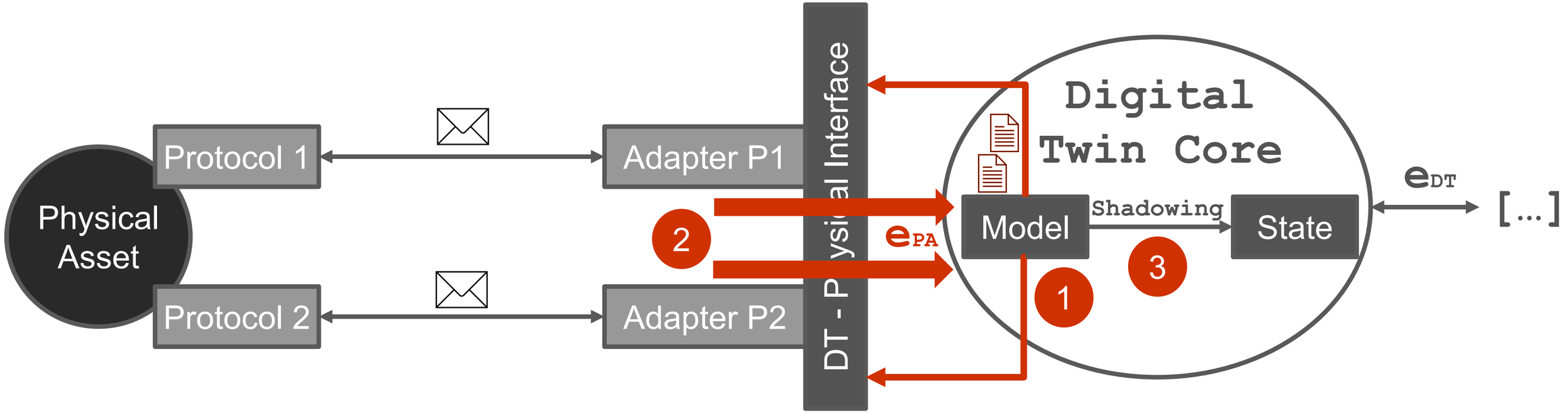


Physical Asset Description

- The **Physical Asset Description (PAD)** is used to describe the list of properties, actions and relationships of a Physical Asset
- Each Physical Adapter generates a dedicated PAD associated to its perspective on the Physical Assets and its capabilities to read data and execute actions
- It is a responsibility of the DT to handle multiple descriptions in order to build the digital replica
- It will be used by the DT to handle the shadowing process and keep the digital replica synchronized with the physical counterpart



Digital Twin Shadowing

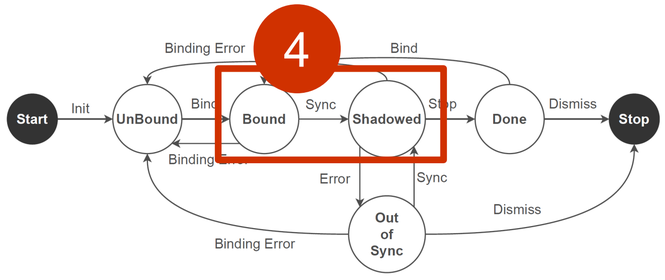


1. The Model defines which properties should be monitored on the Physical Asset and start observing them through the target adapters

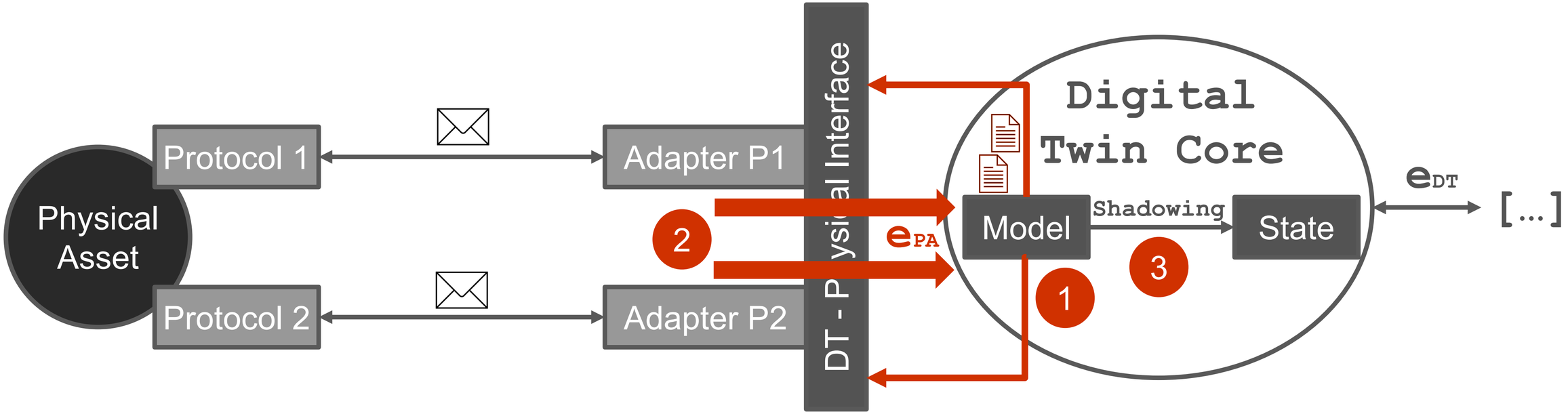
2. Involved Physical Adapters communicate with the Physical Asset, receive data and generate Events (e_{PA}) to notify about physical property changes

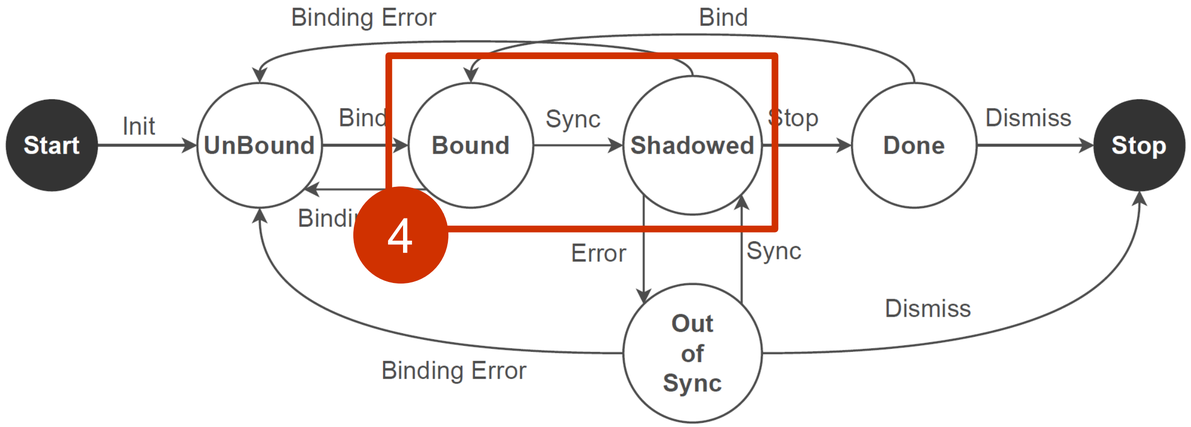
3. Received e_{PA} will be used by the Digital Twin Model in order to run the Shadowing function and compute the new DT State

4. The DT can move from the Bound to Shadowed phase until it is able to maintain a proper synchronization with the physical asset over time



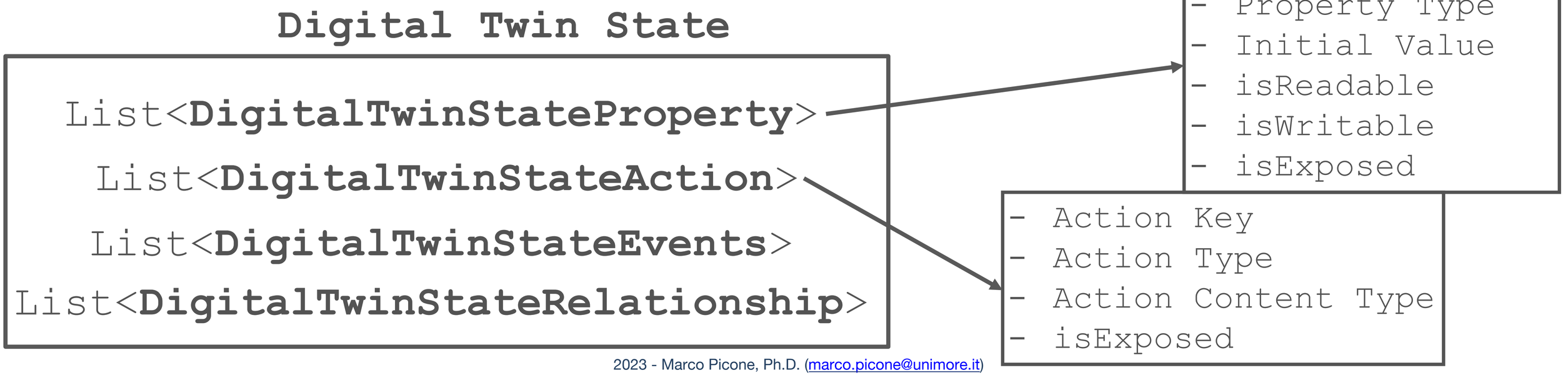
Digital Twin Shadowing





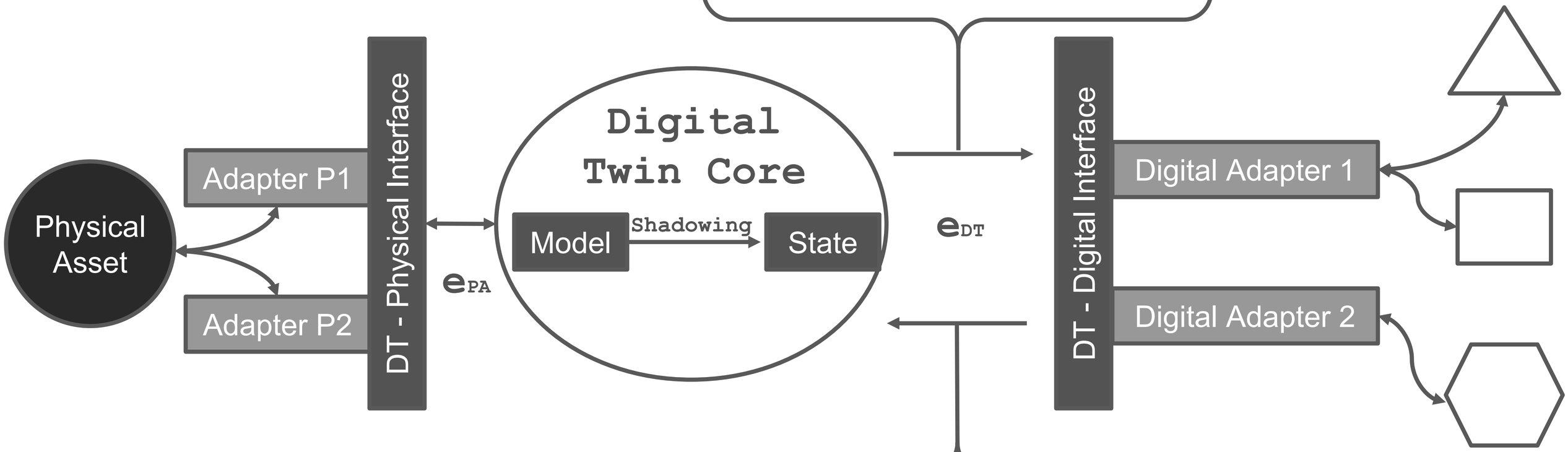
Digital Twin State

- The **Digital Twin State** is structured and characterized by the following elements:
  - A list of properties
  - A list of events
  - A list of actions
  - A list of relationships
- Listed elements can be directly associated to the corresponding element of the Physical Asset or generated by DT Model combining multiple physical properties, actions or relationships at the same time
- The Digital Twin State can be managed through a **ShadowingModelFunction** and exposes a set of methods for its manipulated. **When there is a change in the DT State an event (e_{DT}) will be generated**



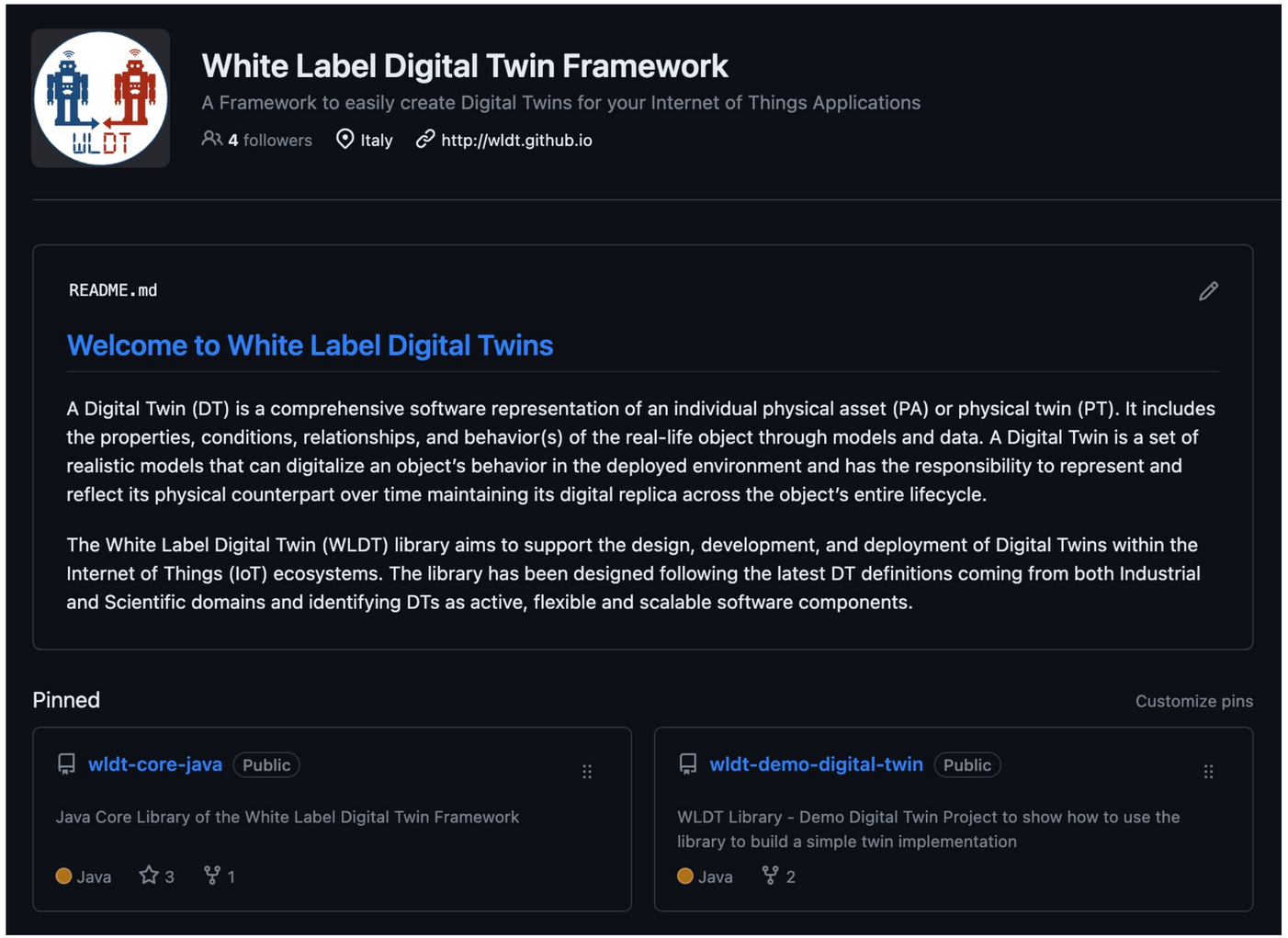
Digital Twin State & Digital Twin Events

- **Property**
  - Created
  - Updated
  - Deleted
- **Action**
  - Enabled
  - Updated
  - Disabled
- **Relationships**
  - Created
  - Updated
  - Deleted



- **Digital Adapters** can invoke exposed Digital Twin Actions generating a target e_{DT}

Digital Twin – Open Source Library



A Java Framework to easily create Digital Twins for your Internet of Things Applications

https://github.com/wldt

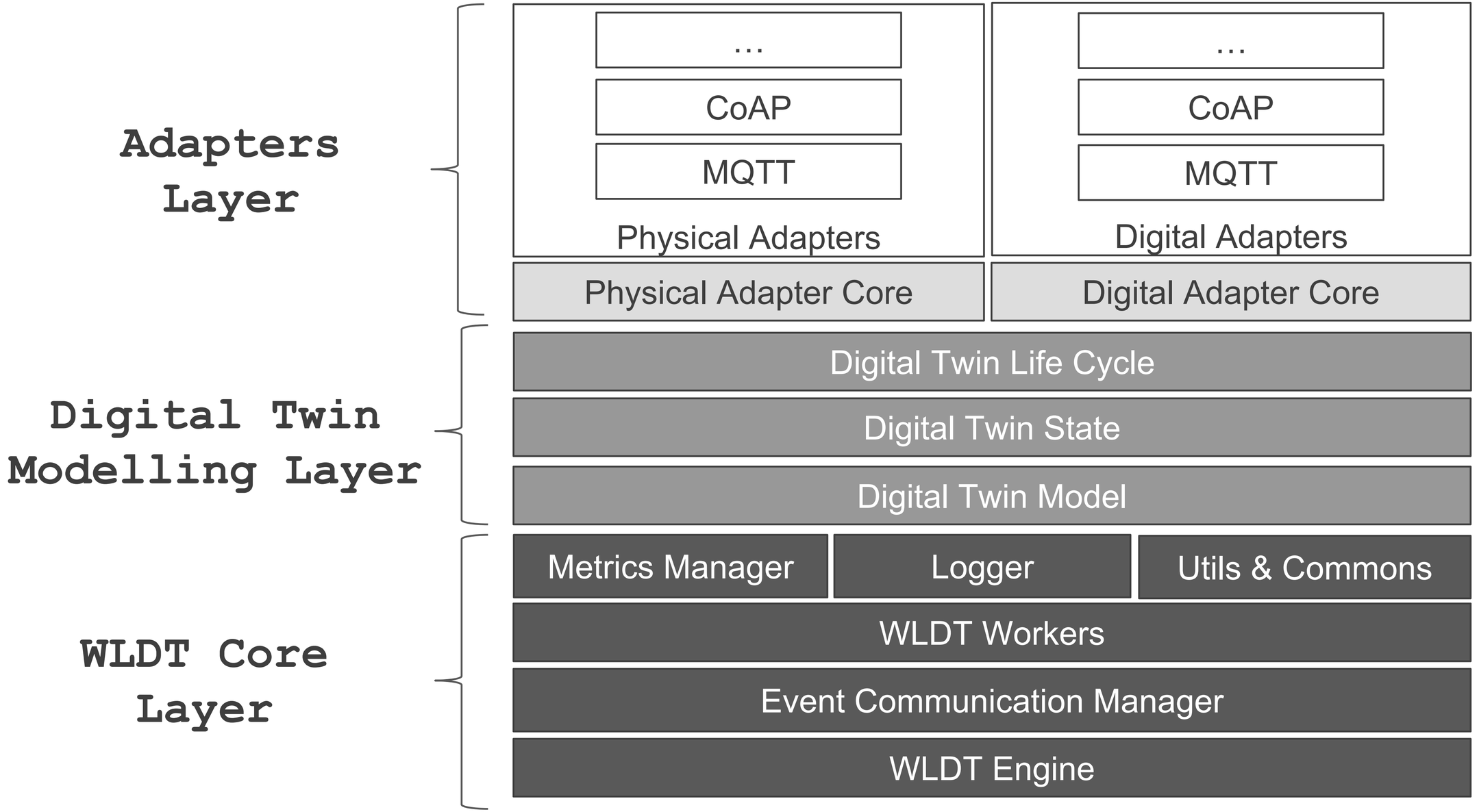
GitHub

Java

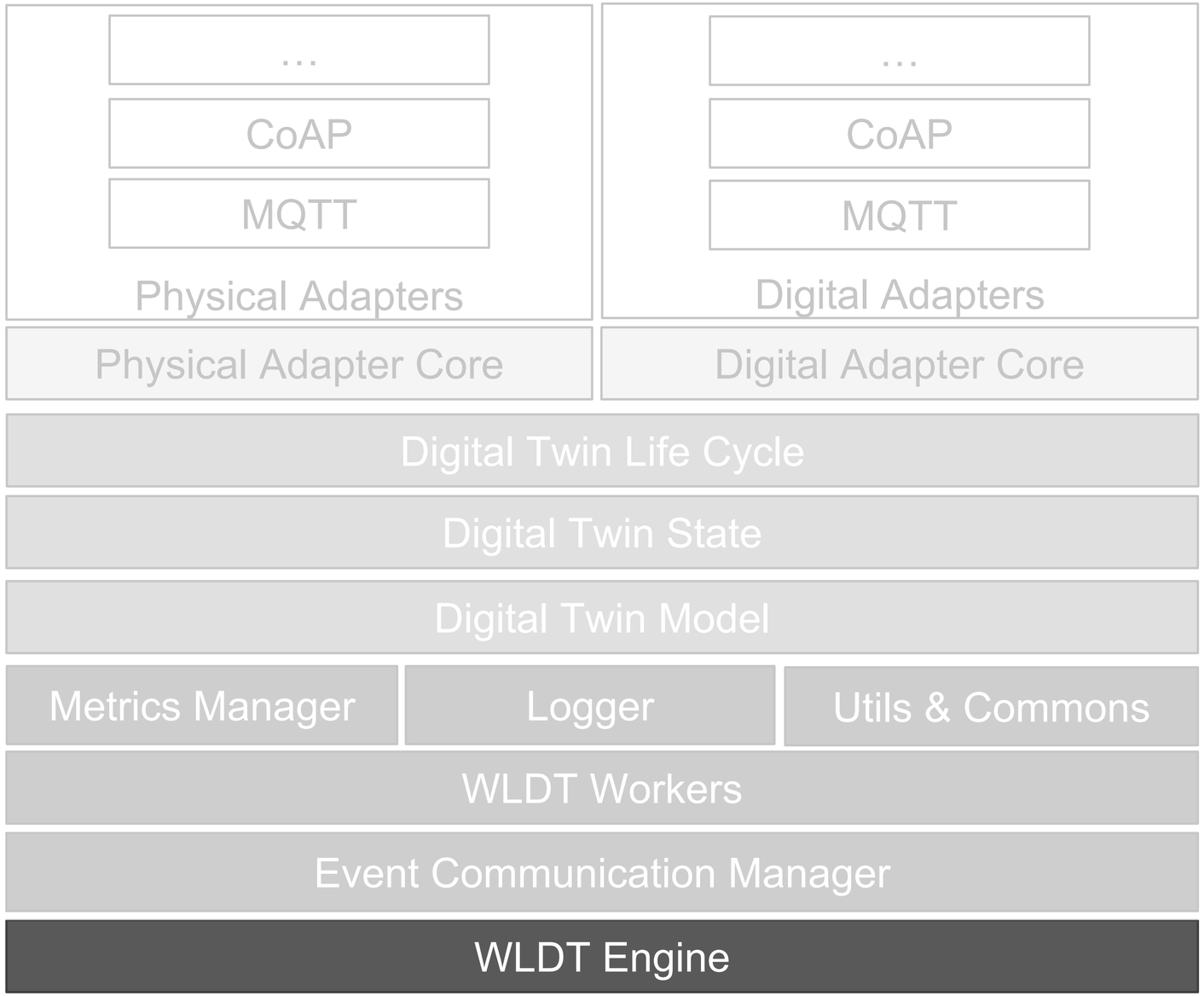
python™

Not Yet ☺ But Planned

White Label Digital Twin – Software Architecture

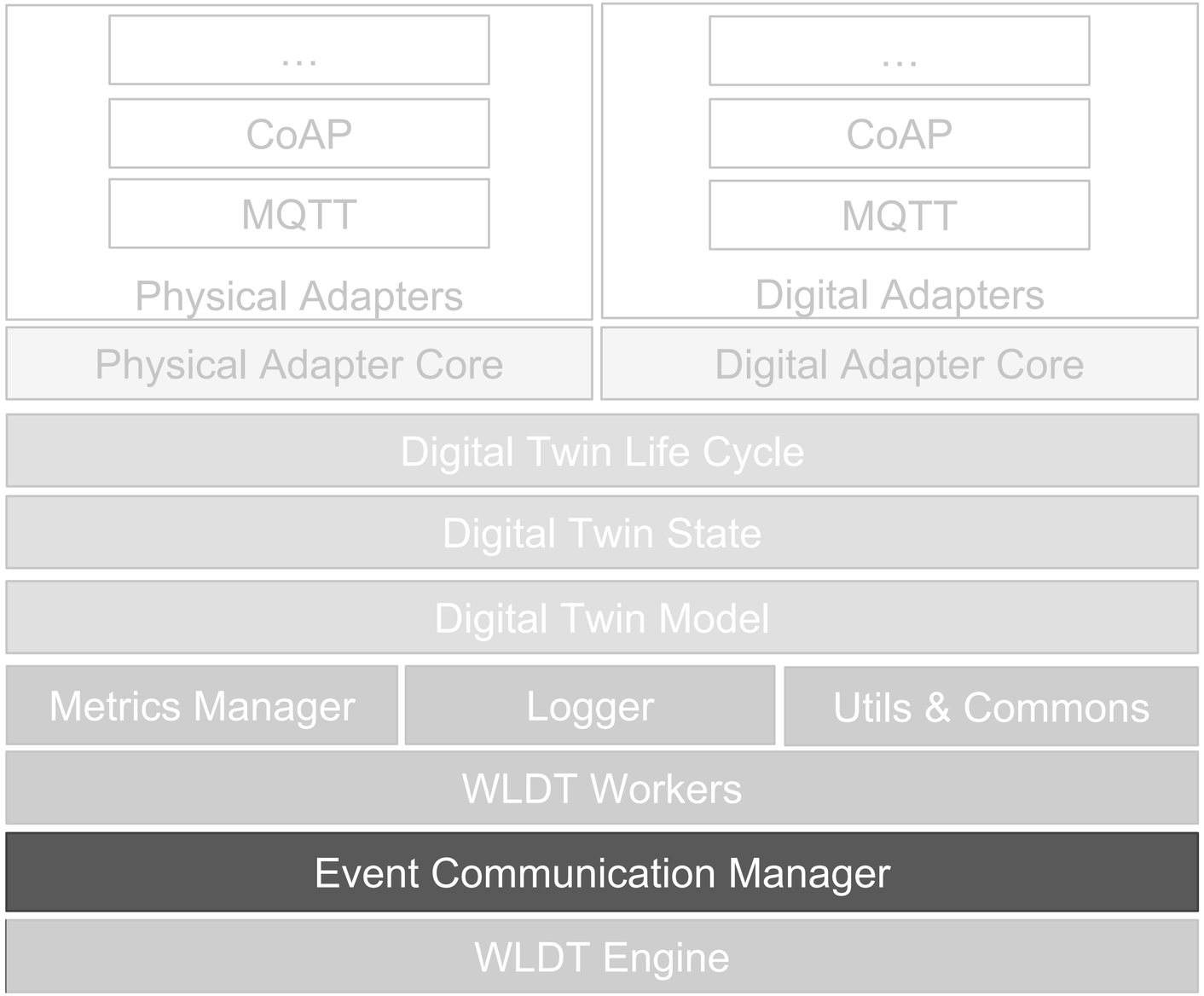


White Label Digital Twin – New Design



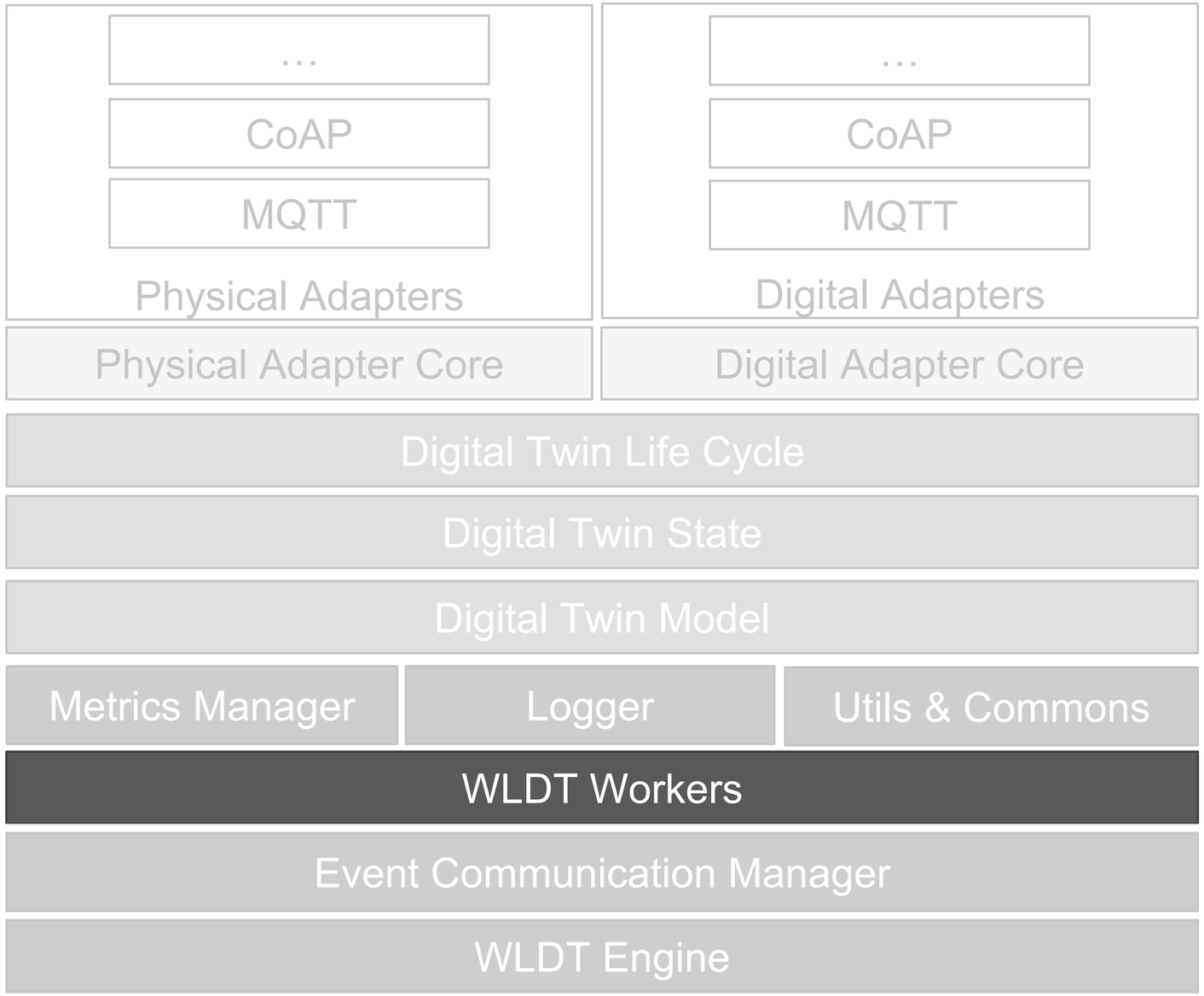
- **Multi-Thread Engine** able to run and monitor multiple Workers at the same time
- Provides a **customizable** WLDT **configuration**
- Keeps references and **orchestrate** all the **internal modules**:
  - Digital Twin Model
  - Digital Twin Life Cycle
  - Digital Twin State
  - Physical Adapters
  - Digital Adapters

White Label Digital Twin – New Design



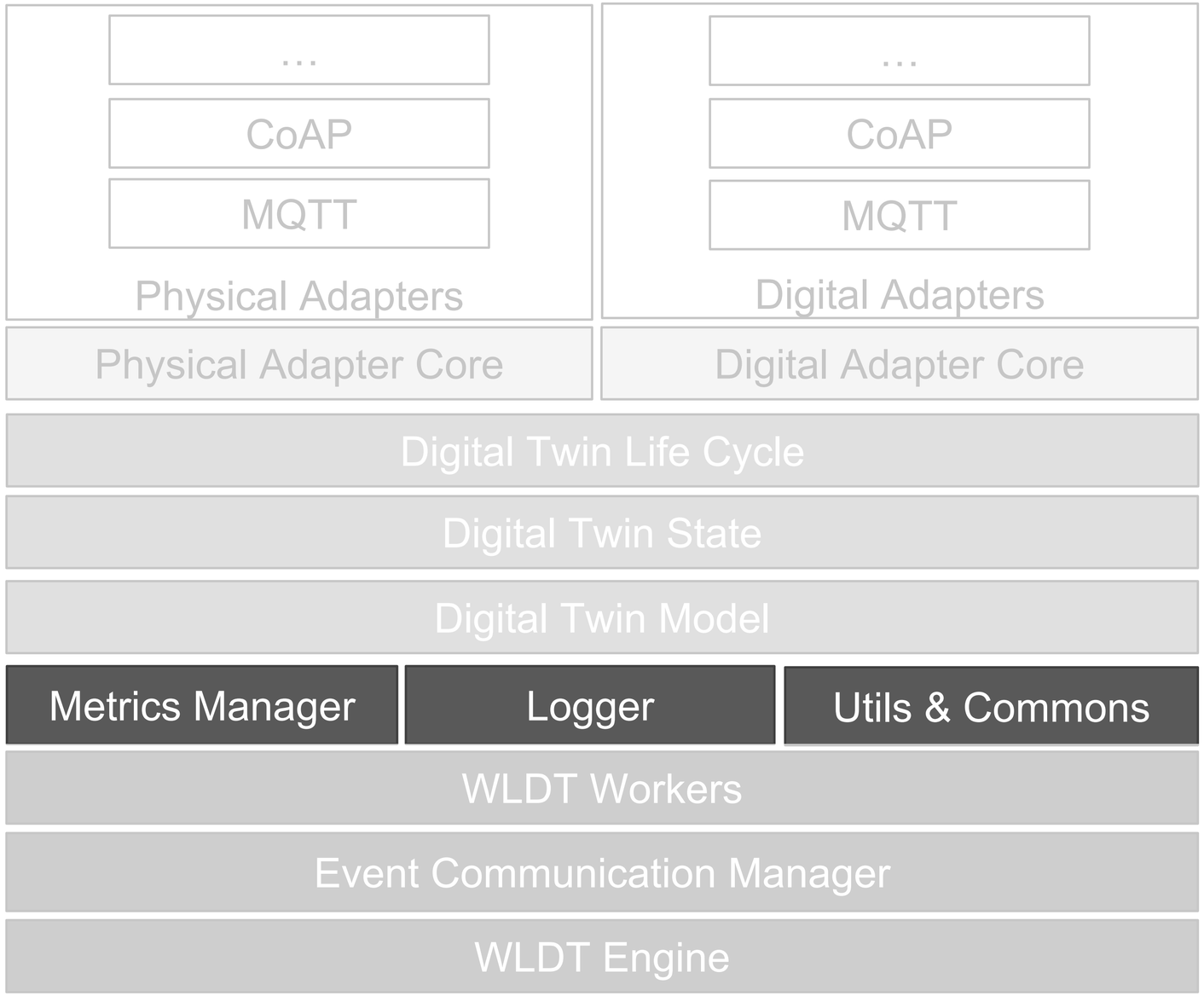
- Internal **Event Bus** designed to support the communication among the modules of a single WLDT Instance
- **Customizable Events** Definition such as Physical and Digital events associated to properties and actions
- **Event Filter** definition to declare interest for target events and messages
- **Listeners and callbacks** to received target events
- The event-driven communication layer is used both by internal modules and third-party components such as digital and physical adapters

White Label Digital Twin – New Design



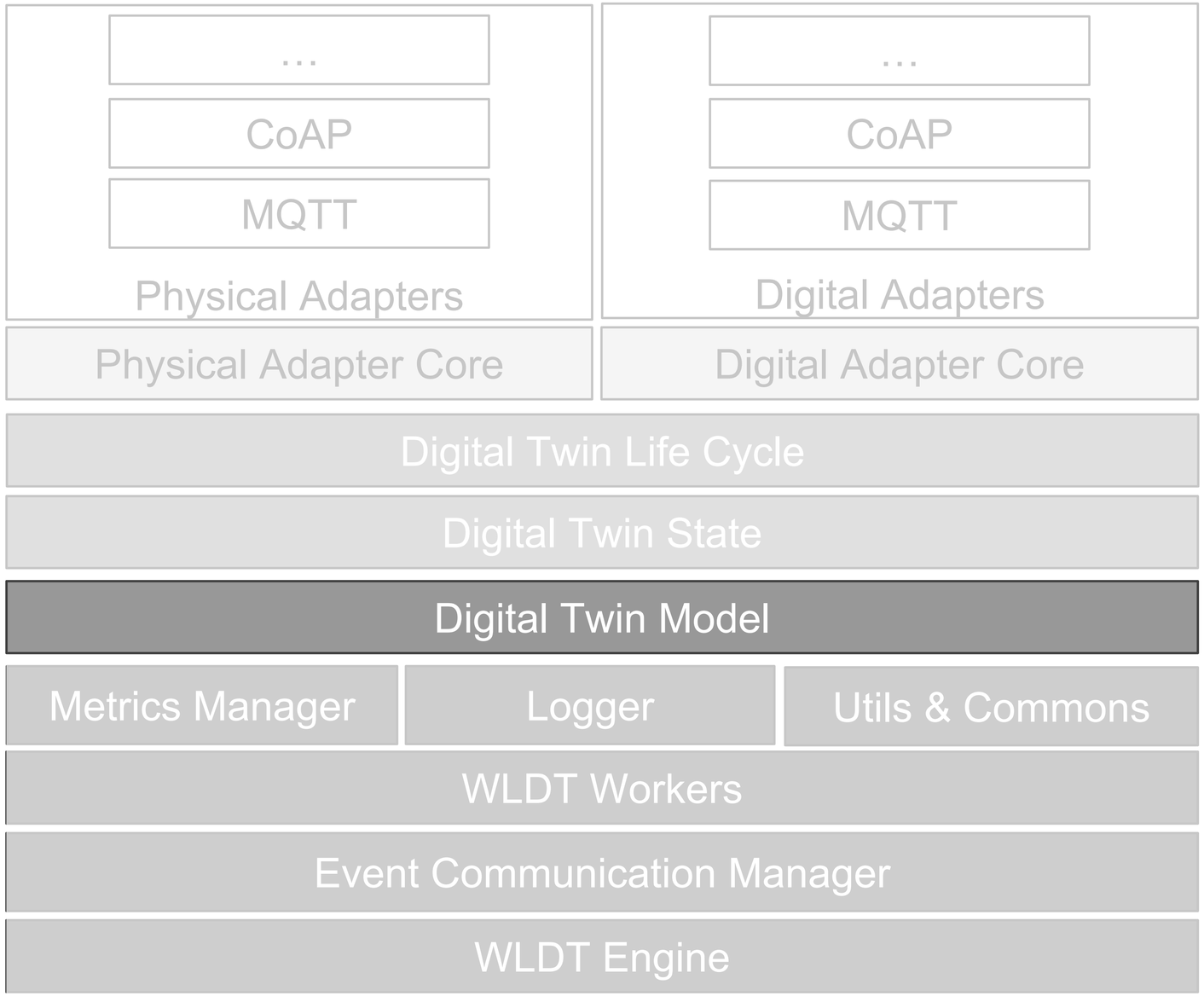
- Basic internal component used to shape an executable element within the WLDT Framework
- A worker is executed by the WLDT Engine and can be associated to multiple definitions and implementation for example DT-Model, Physical and Digital Adapters etc …

White Label Digital Twin – New Design



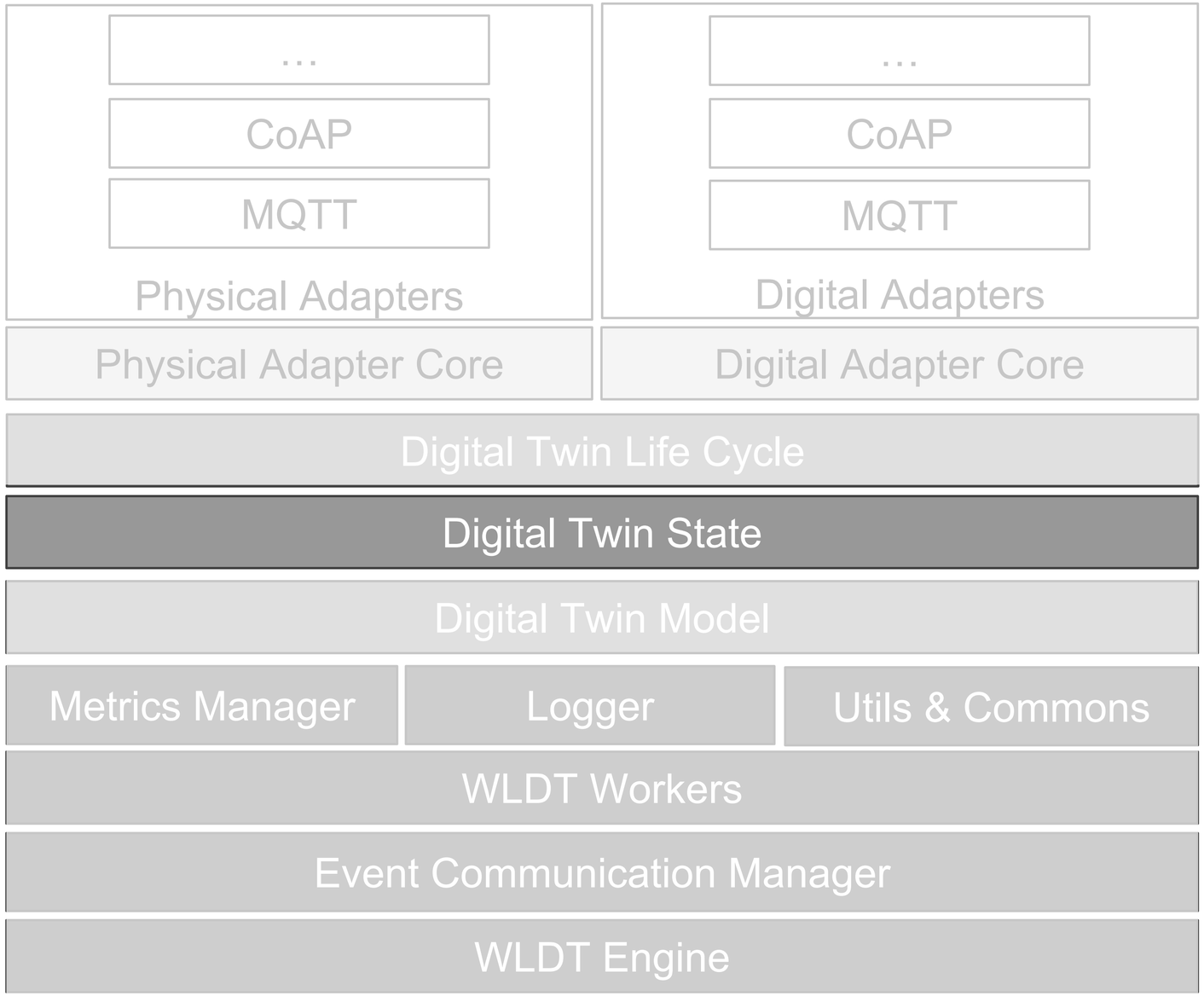
- Internal metric system to support the developer to monitor DT instances (support local CSV based metrics collection and remote collection with Graphite)
- Logger component to monitor and log events passing through the EventBus
- The Logger can be customized by the developer to change the logging type and the destination

White Label Digital Twin – New Design



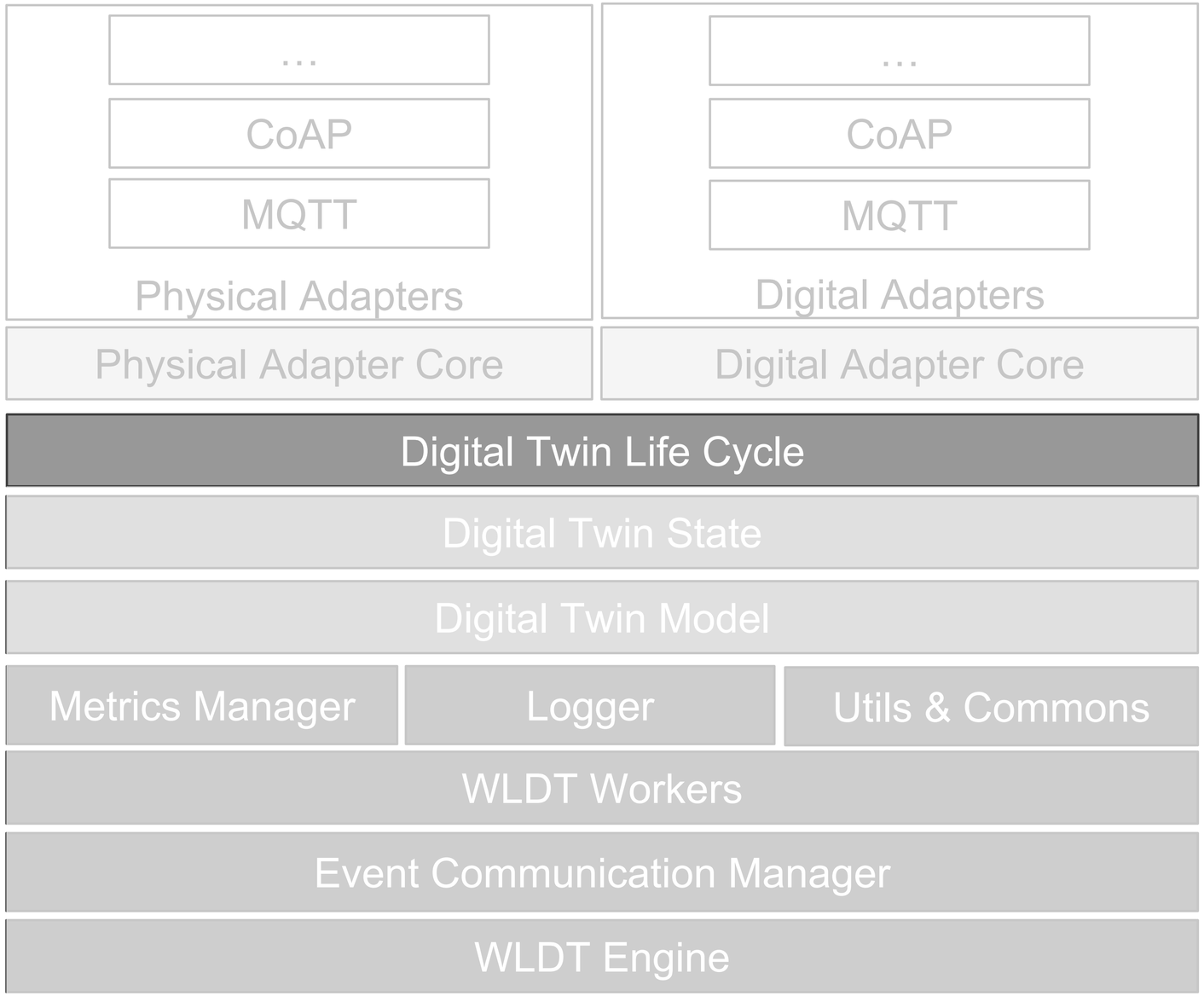
- Has the responsibility to shape the behavior of the Digital Twin interacting with the DT State and through the definition and execution of:
  - **The Shadowing Model Function:** defines the Digital Twin shadowing procedure and it is the component responsible to keep physical and digital part synchronized
  - **[DRAFT] Augmentation Function(s):** Additional customizable functions that can be use do add specific additional behaviors to the DT. They can only interact (read, observe and write) with the Digital Twin State to extend functionalities, properties, actions etc

White Label Digital Twin – New Design



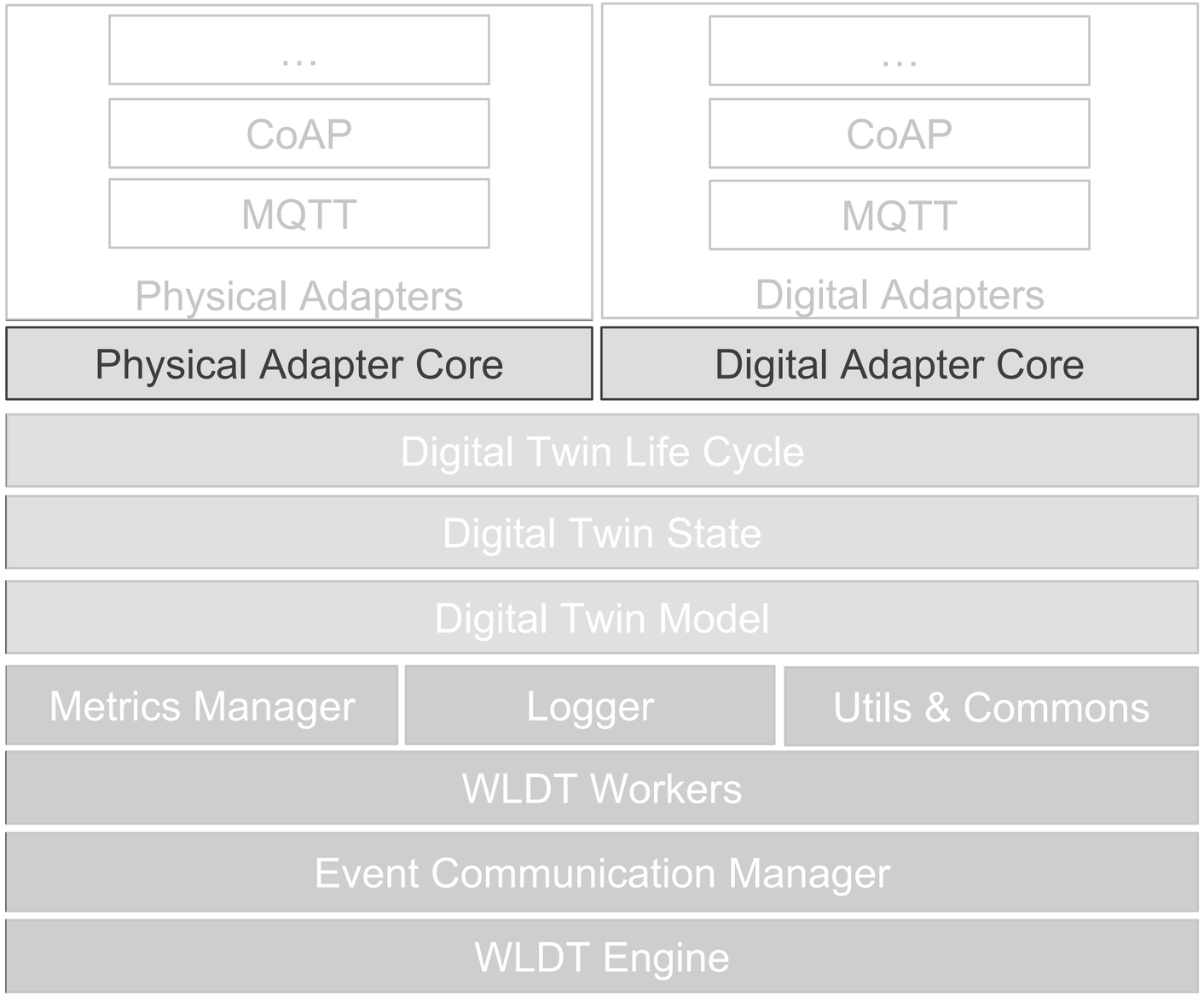
- The Digital Twin State is structured and characterized by the following elements:
  - A list of properties
  - A list of actions
  - A list of relationships **(Not Available Yet)**
- Listed elements can be directly associated to the corresponding element of the Physical Asset or generated by DT Model combining multiple physical properties, actions or relationships at the same time
- The Digital Twin State can be managed through a **ShadowingModelFunction** and exposes a set of methods for its manipulated. **When there is a change in the DT State an event (e_{DT}) will be generated**

White Label Digital Twin – New Design



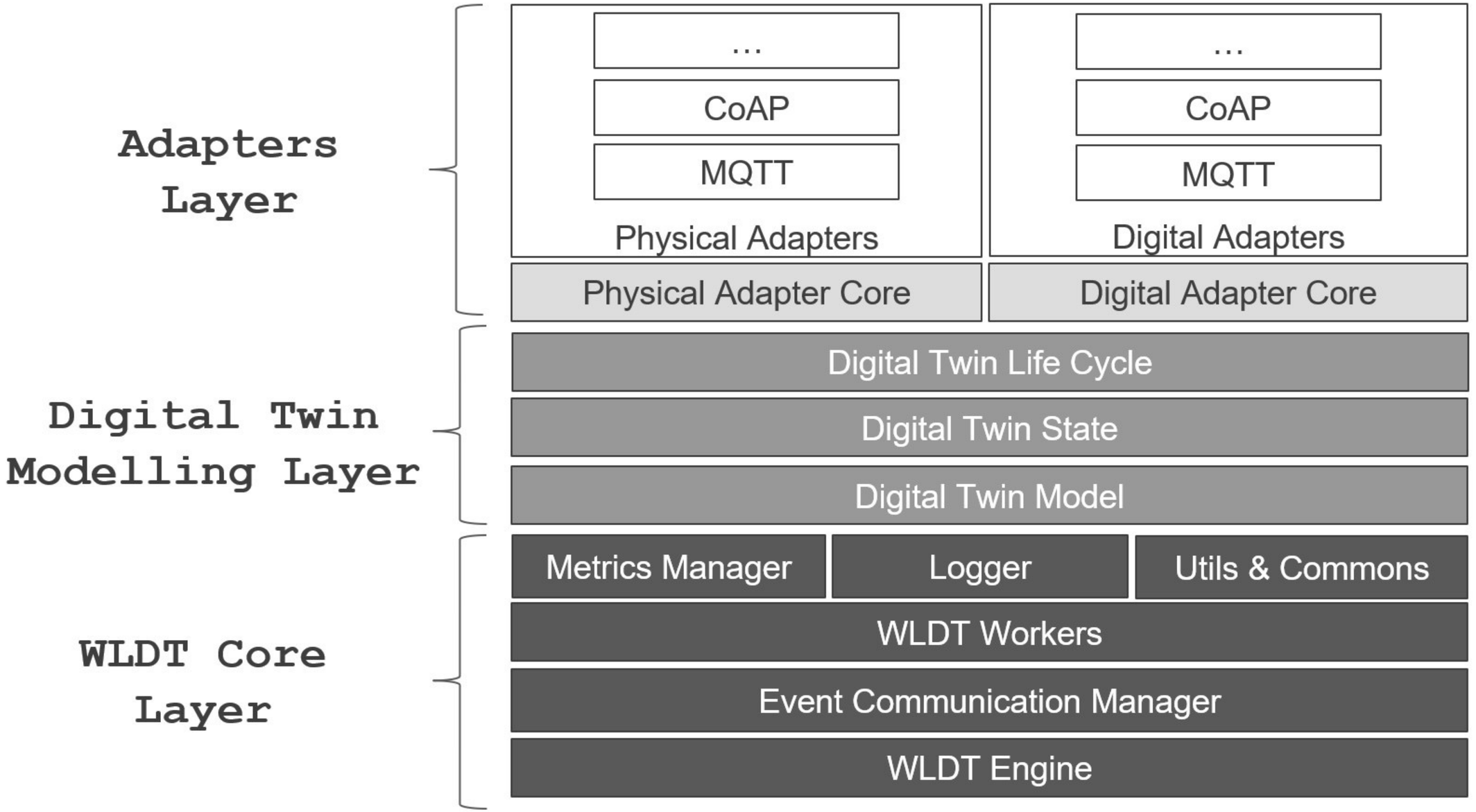
- Defines and control the Digital Twin Life Cycle modelling its core phases:
  - Start
  - UnBound
  - Bound
  - Shadowed/Sync
  - UnSync
- Add some additional operational phases related to:
  - Physical Adapter Bound
  - Physical Adapter UnBound
  - Digital Adapter Bound
  - Digital Adapter UnBound
- Exposes a Listener to allow interested module to receive notifications about life cycle changes

White Label Digital Twin – New Design



- Provide a structured core implementation to support the creation of Physical and Digital Adapter integrated with the WLDT Framework
- They have a dedicated internal life cycle integrated with Digital Twin life
- Support the following main operations:
  - Integration with the Digital Twin Life Cycle
  - Exposition of the Physical Asset Description
  - Generation and propagation of Physical Events
  - Generation and propagation of Digital Twin Events
  - Bidirectional management of actions coming on the Digital Adapter and executed on Physical Adapter passing through the DT Model

Digital Twin – Open Source Library



UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Design & Development of a Digital Twin Based Human-to-Machine Interaction System

Computer Engineering, University of Modena and Reggio Emilia, (Mantova Campus)

Academic Year: 2021/2022

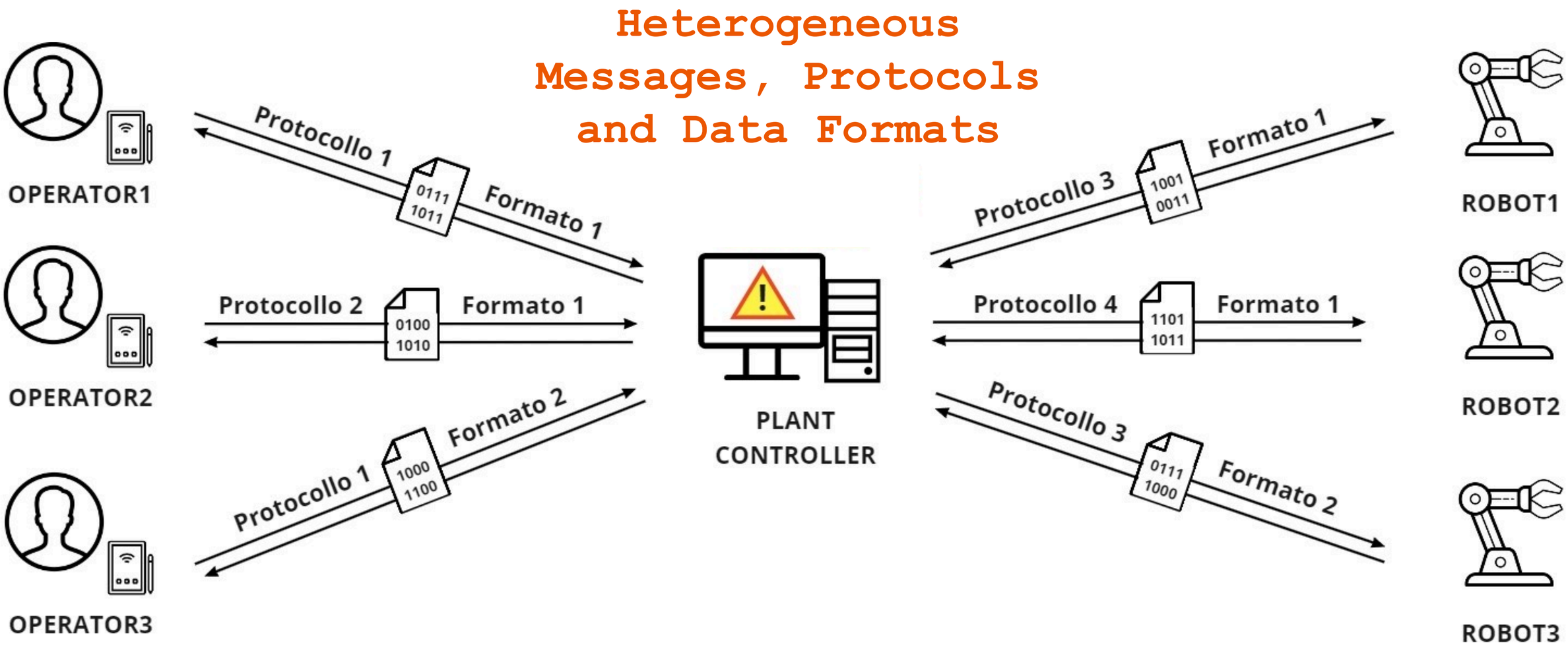
Supervisors: Prof. Marco Picone, Prof. Valeria Villani

Student: Menini Giorgio

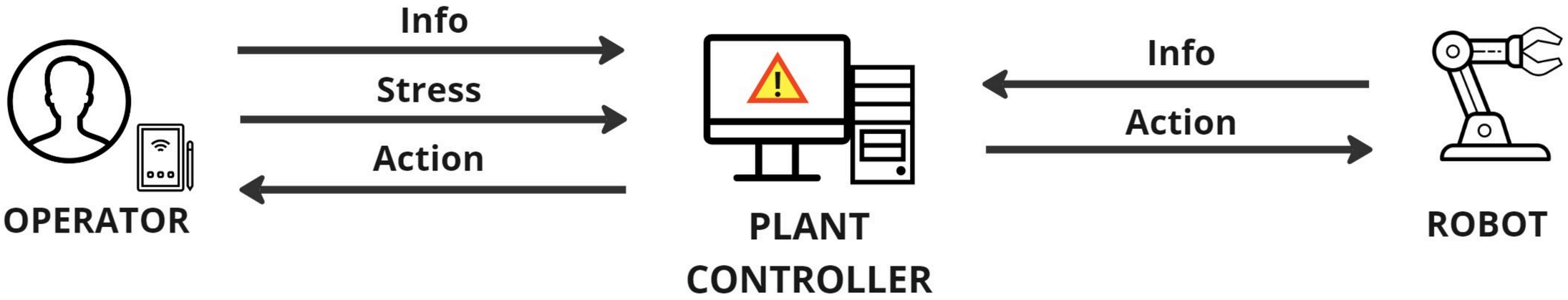
Human-to-Machine Interaction

- In Industry 4.0 and in particular 5.0, Human-to-Machine Interaction is vital. It enables:
  - Quality control
  - Process optimization
  - Maintenance and repair
  - Safety compliance
  - Training and skill development
- This interaction enhances productivity and safety by leveraging human expertise alongside advanced machinery

Open Challenges -> Heterogeneity



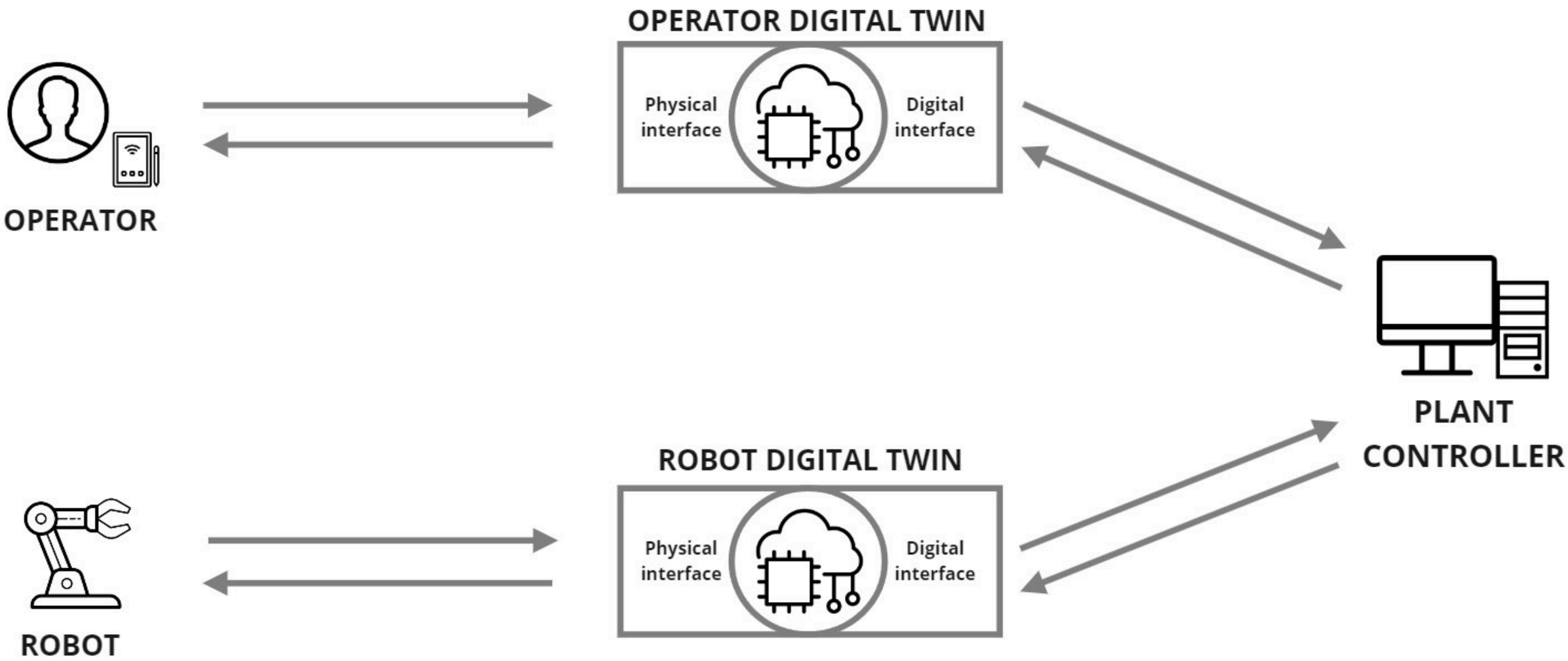
Use Case



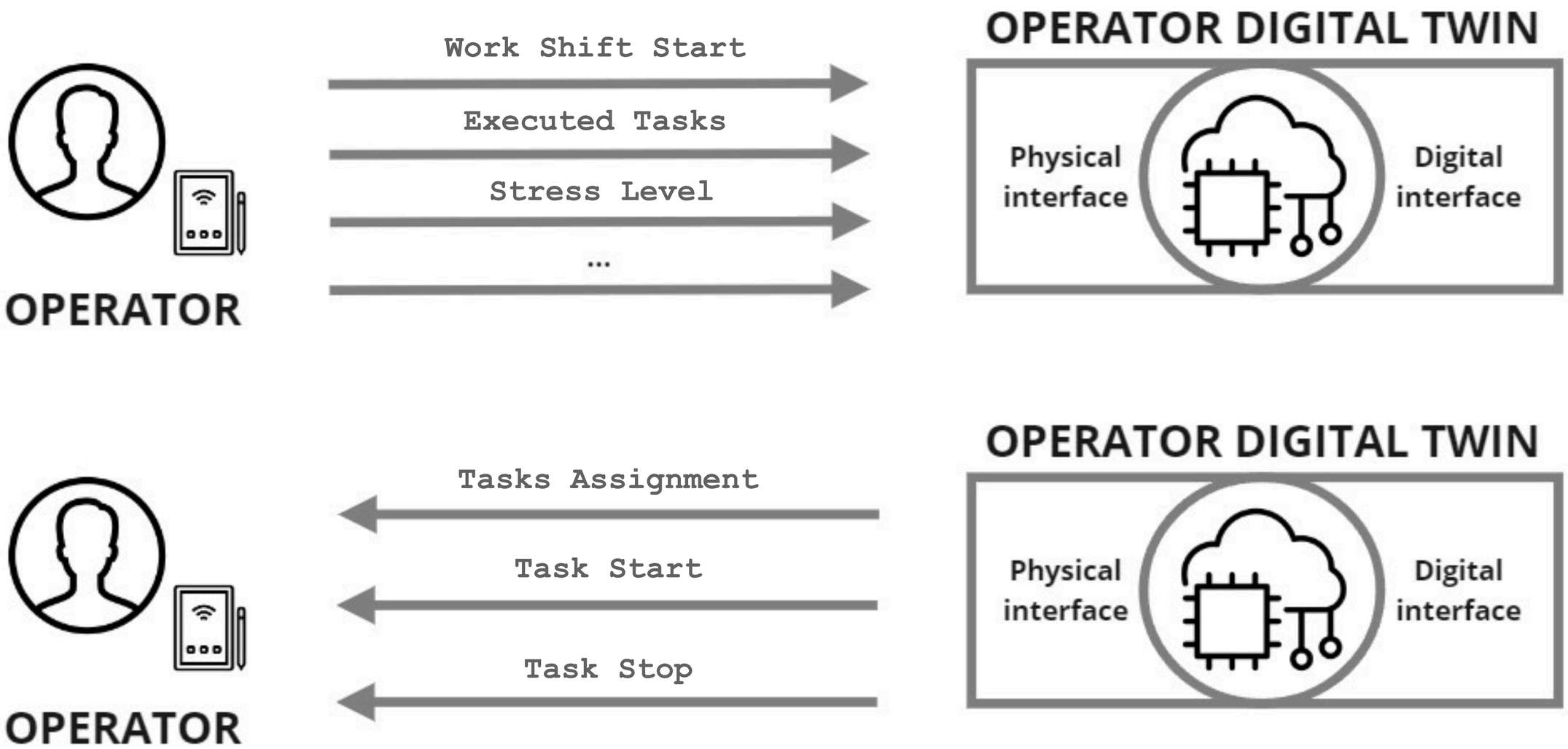
In the example use case, there will be an operator and a robot who want to communicate with the Plant Controller through their respective protocols. The need for seamless and fast communication gives rise to a problem.

Solution: Digital Twins

Digital Twin Based Solution



Operator Digital Twin



Robot Digital Twin

ROBOT

Robot's joints Position

Gripper Position

Speed

Executed Tasks

ROBOT DIGITAL TWIN

Physical interface

Digital interface

ROBOT

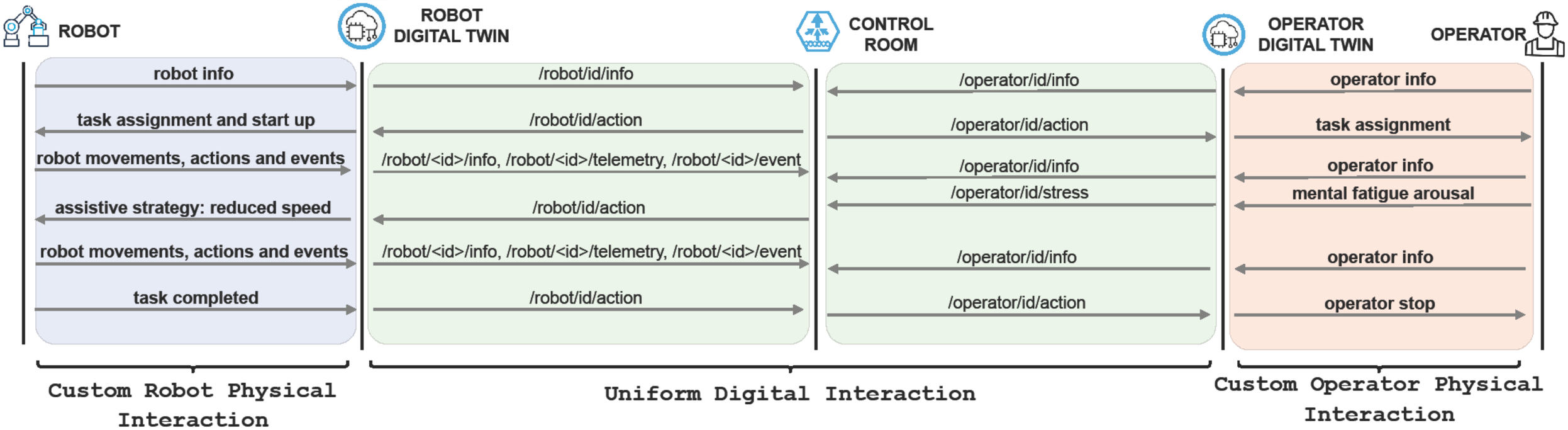
- Operator
- Assignment
- Target Coordinates
- Home Coordinates
- Start/Stop Mission
- Speed Change

ROBOT DIGITAL TWIN

Physical interface

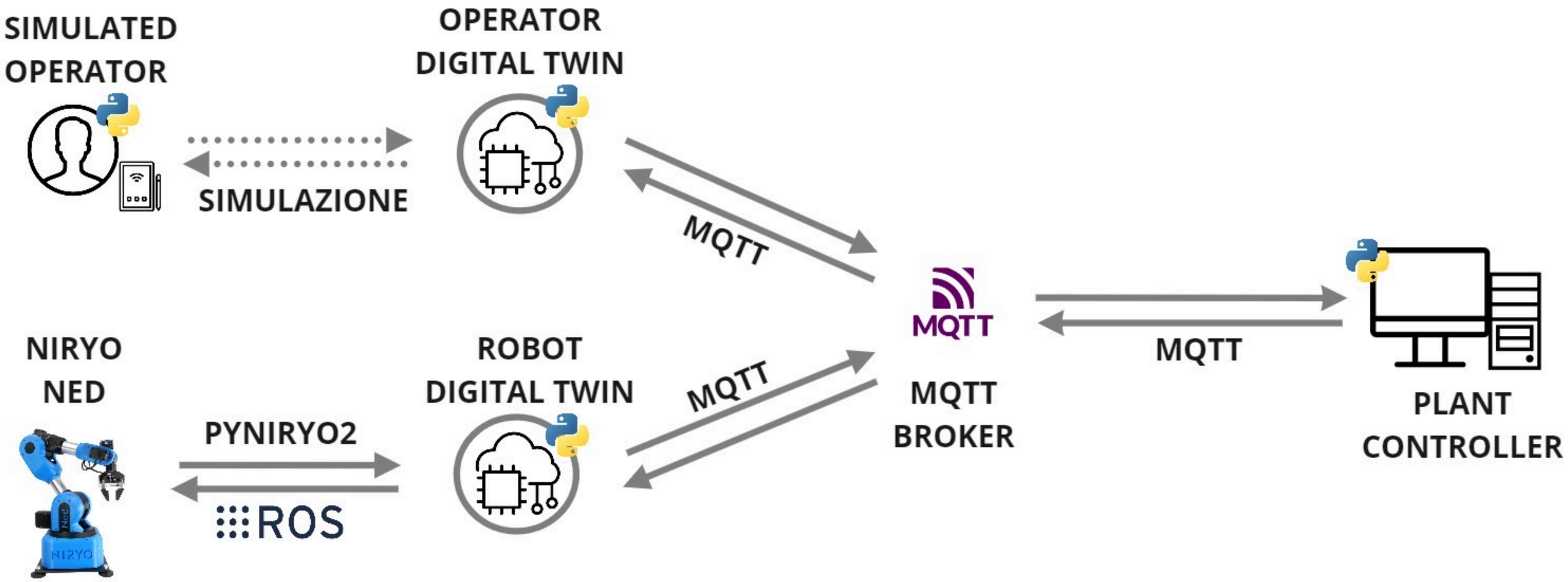
Digital interface

Plant Controller

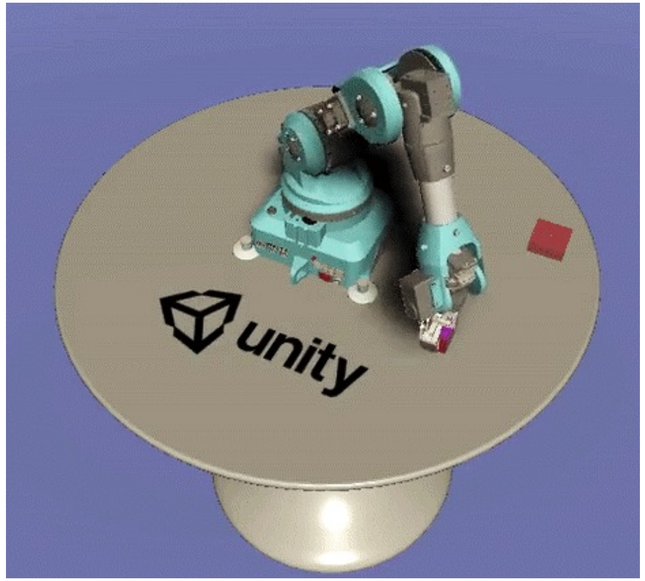


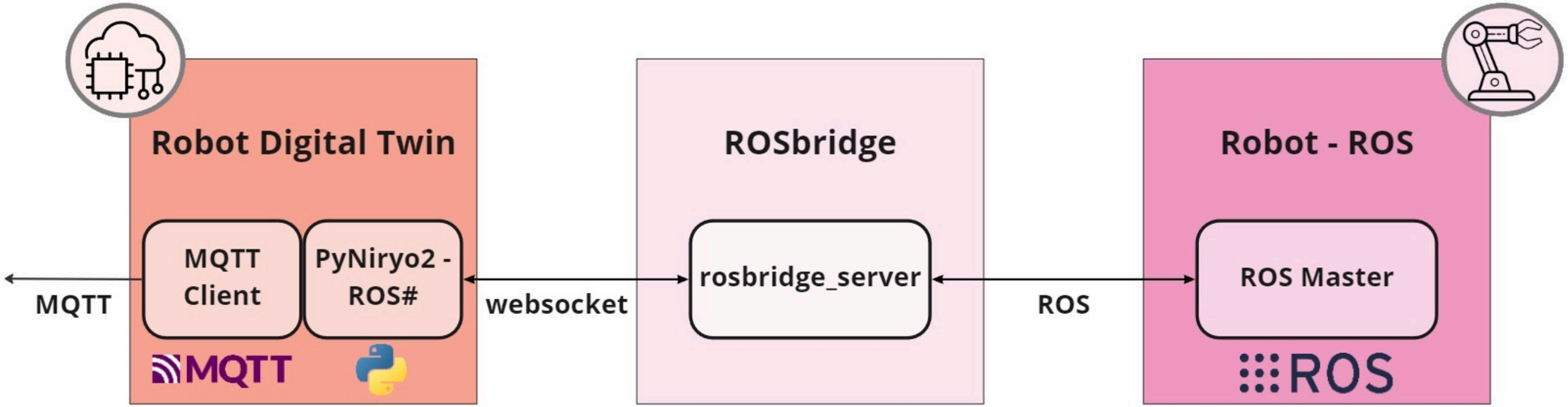
2023 - Marco Picone, Ph.D. (marco.picone@unimore.it)

Implementation

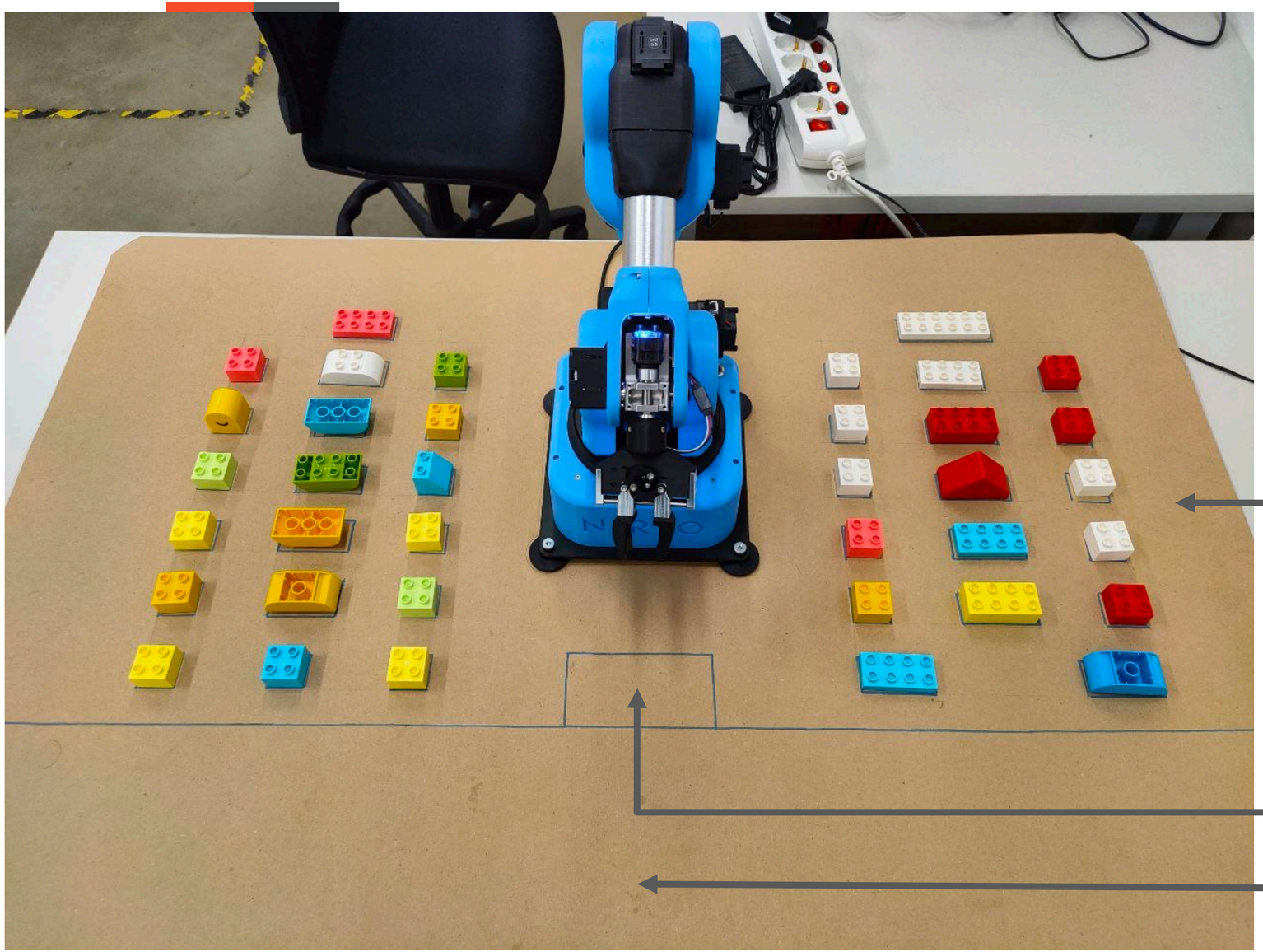


Robot Digital Twin





Experimental Evaluation



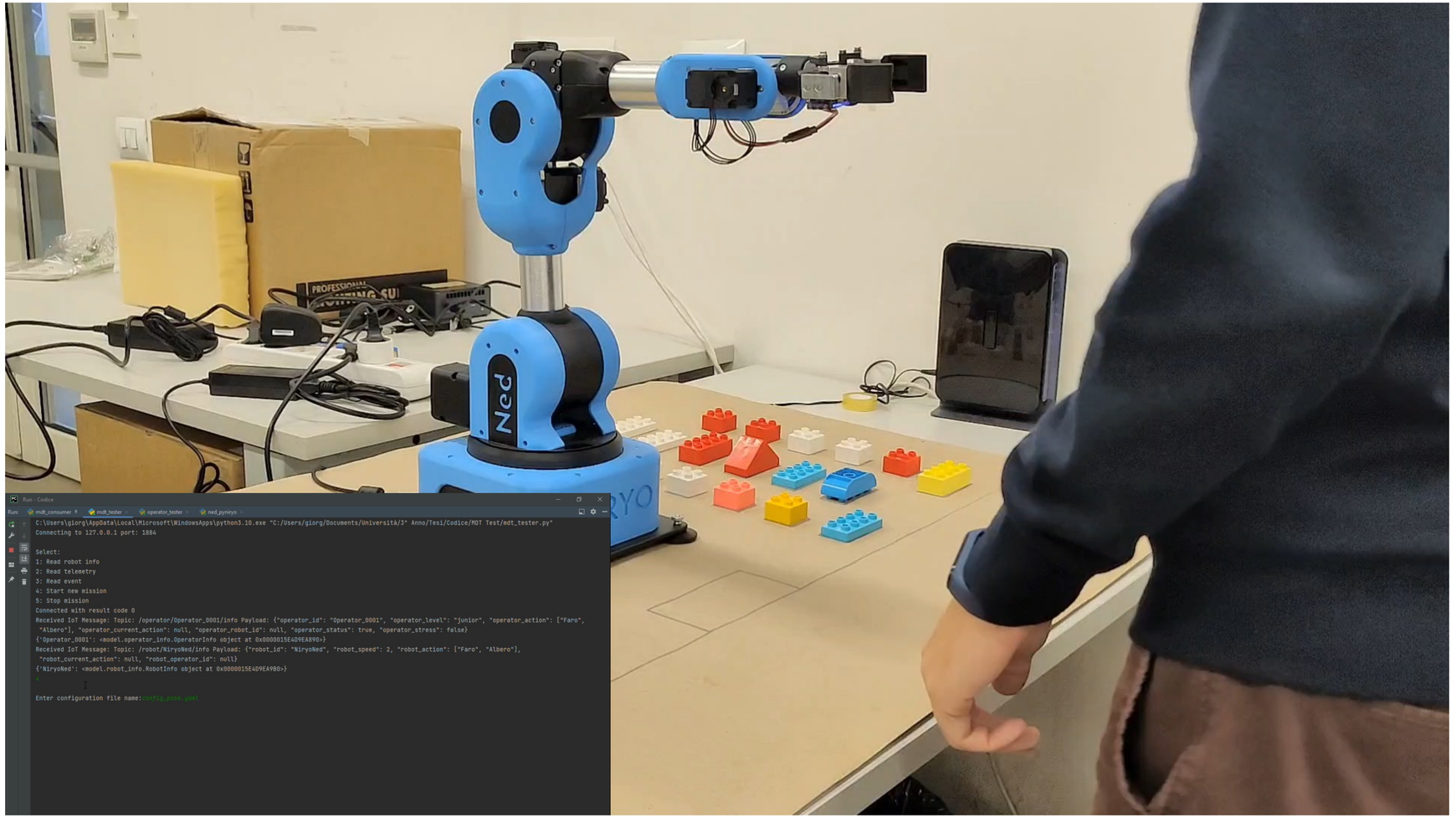
Target Figures

Components Pickup Area

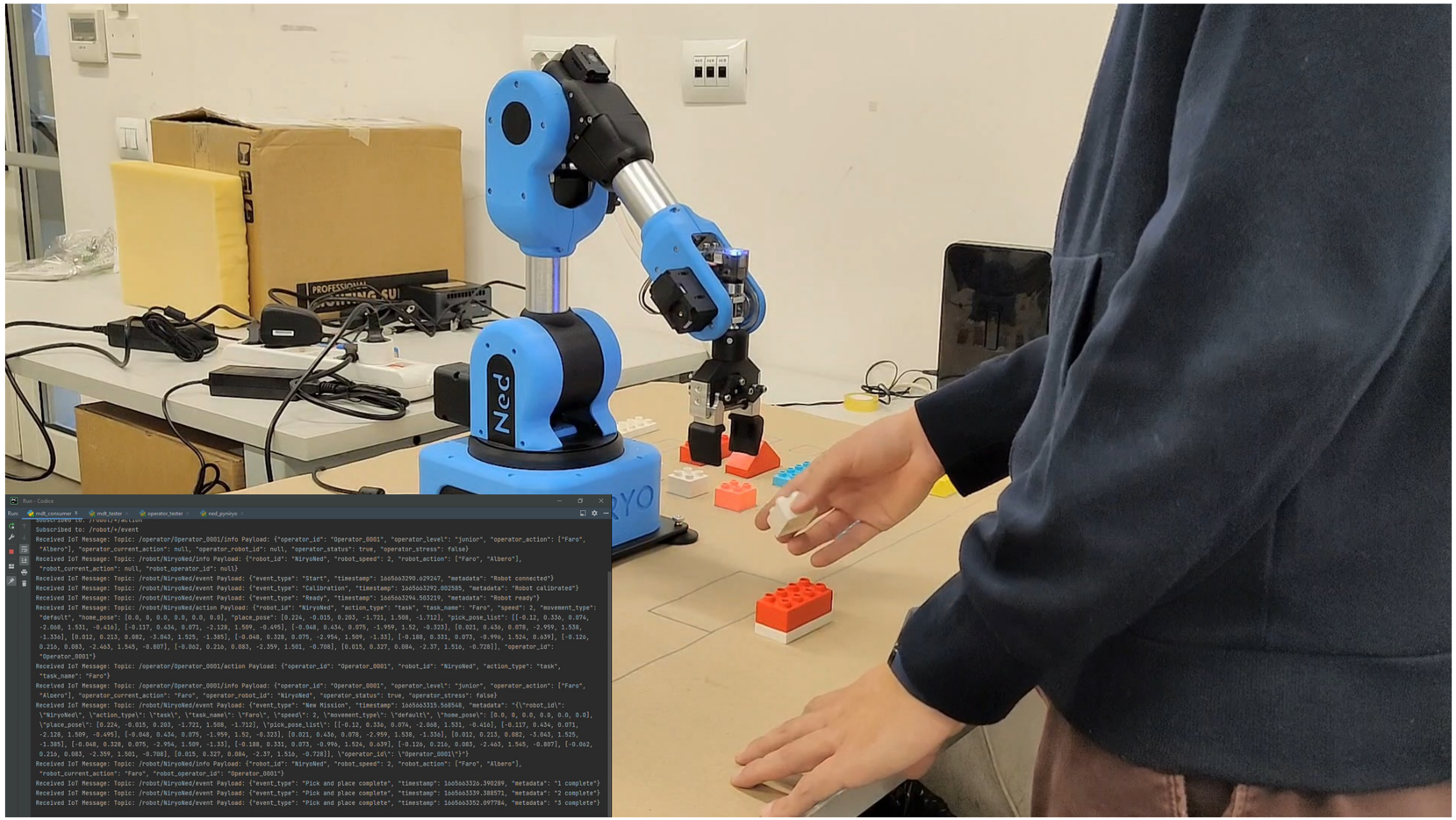
Delivery Area

Operator’s Working Area

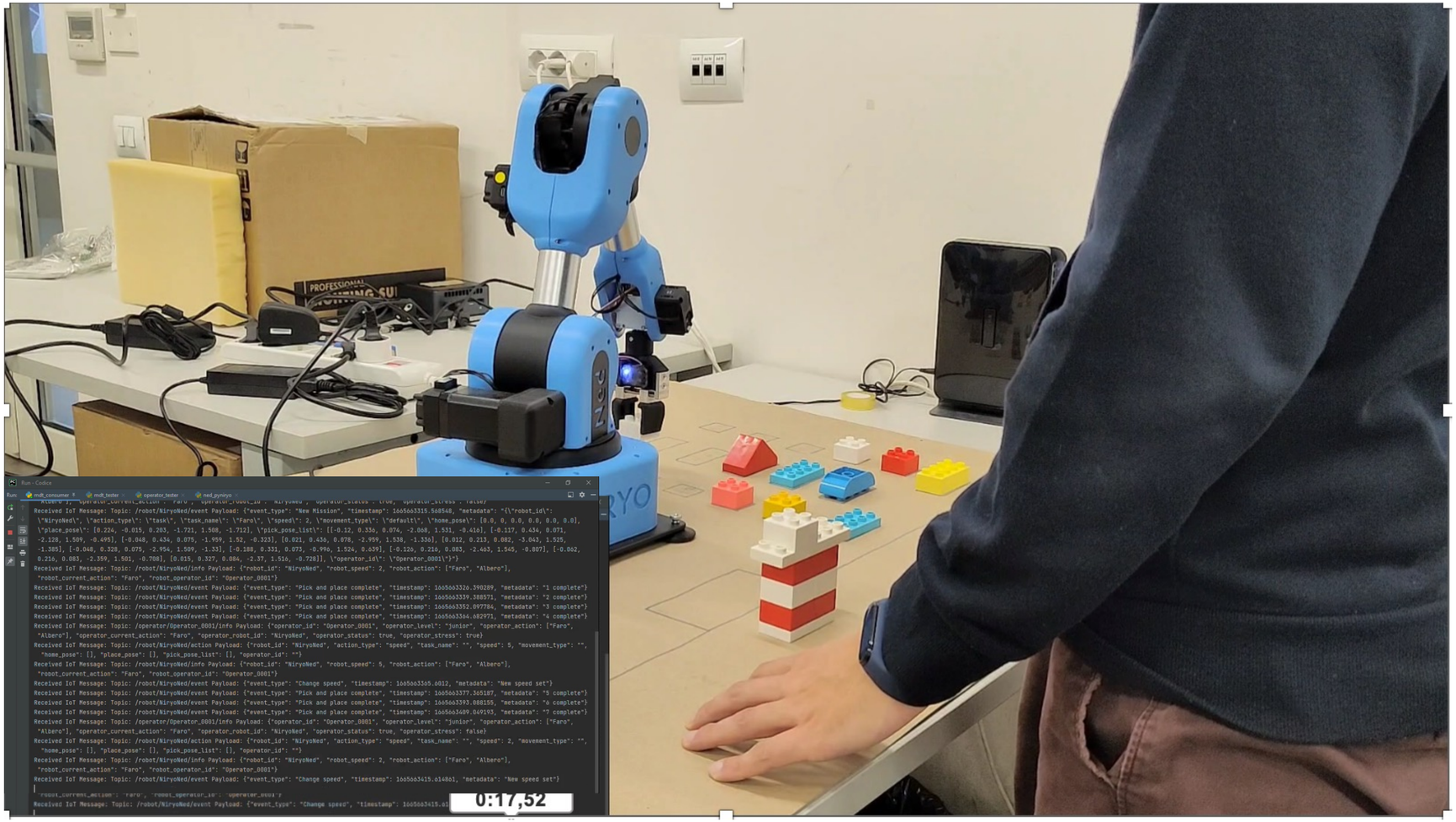
Experimental Evaluation



Experimental Evaluation



Experimental Evaluation



Normal Execution

Assisted Strategy

Normal Execution

Start

Stress Condition Detected

Stress Condition Ended

End

UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Design & Development of Digital Twin Based Applications with Connected Drones

Computer Engineering, University of Modena and Reggio Emilia, (Mantova Campus)

Academic Year: 2022/2023

Supervisor: Prof. Marco Picone

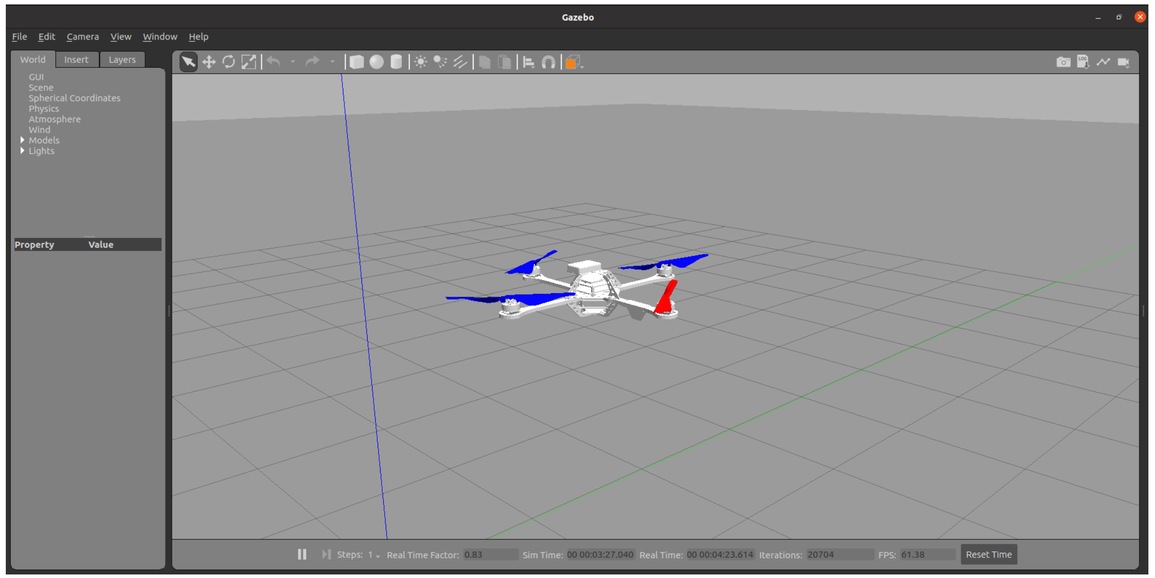
Student: Nicolò Zanoni

Digital Twins & Drones

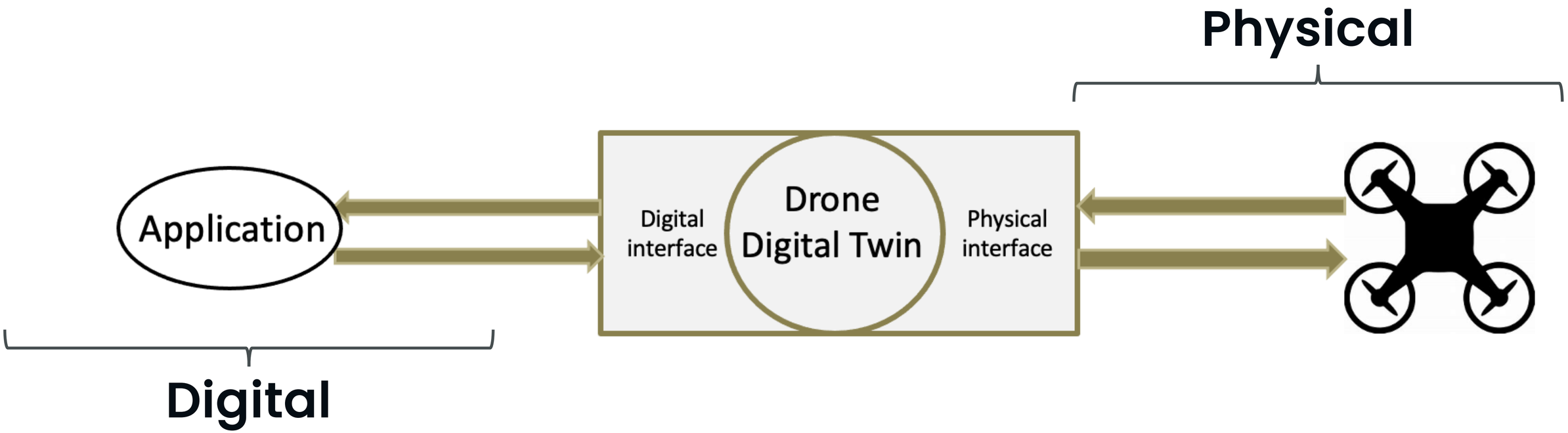
- Drones are Smart Objects that are increasingly widespread and used in the IoT field. They find multiple applications, such as:
  - Surveillance
  - Aerial image acquisition
  - Logistics
  - Transportation
  - Emergency management support

Fonte: IDS

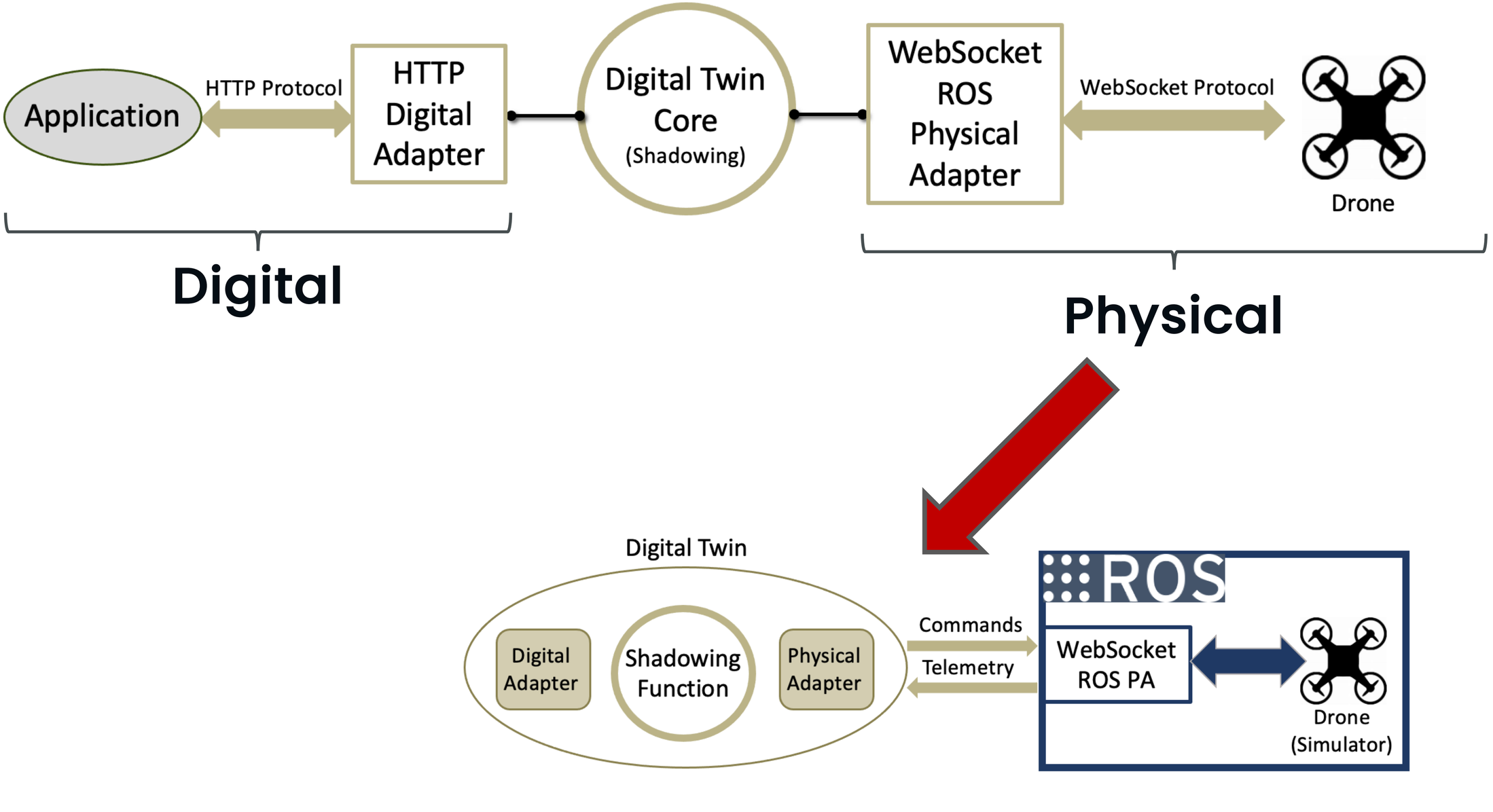
Fonte: FlyingBasket



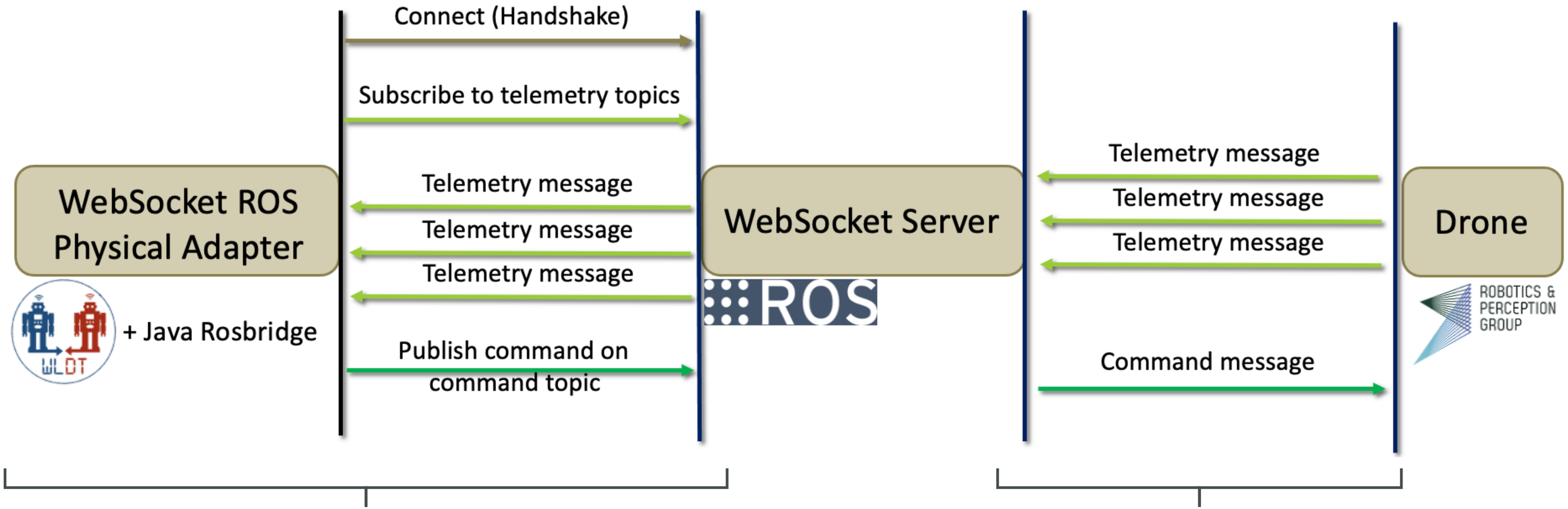
Digital Twins & Drones



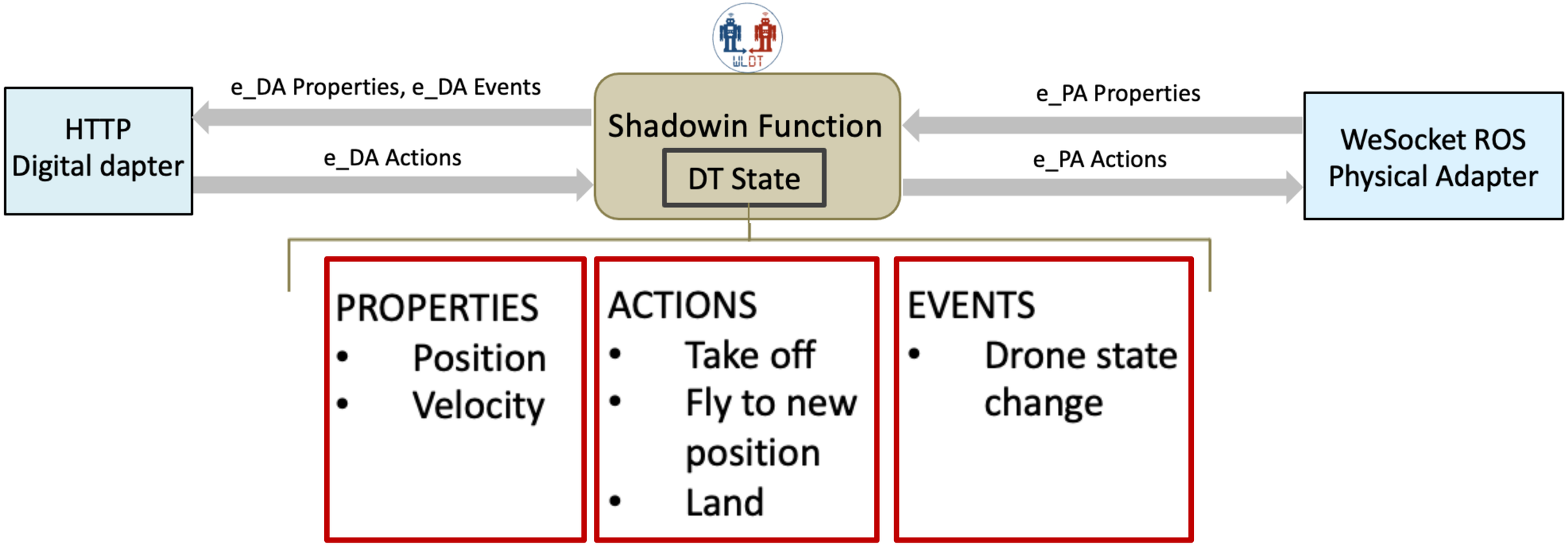
Digital Twins & Drones



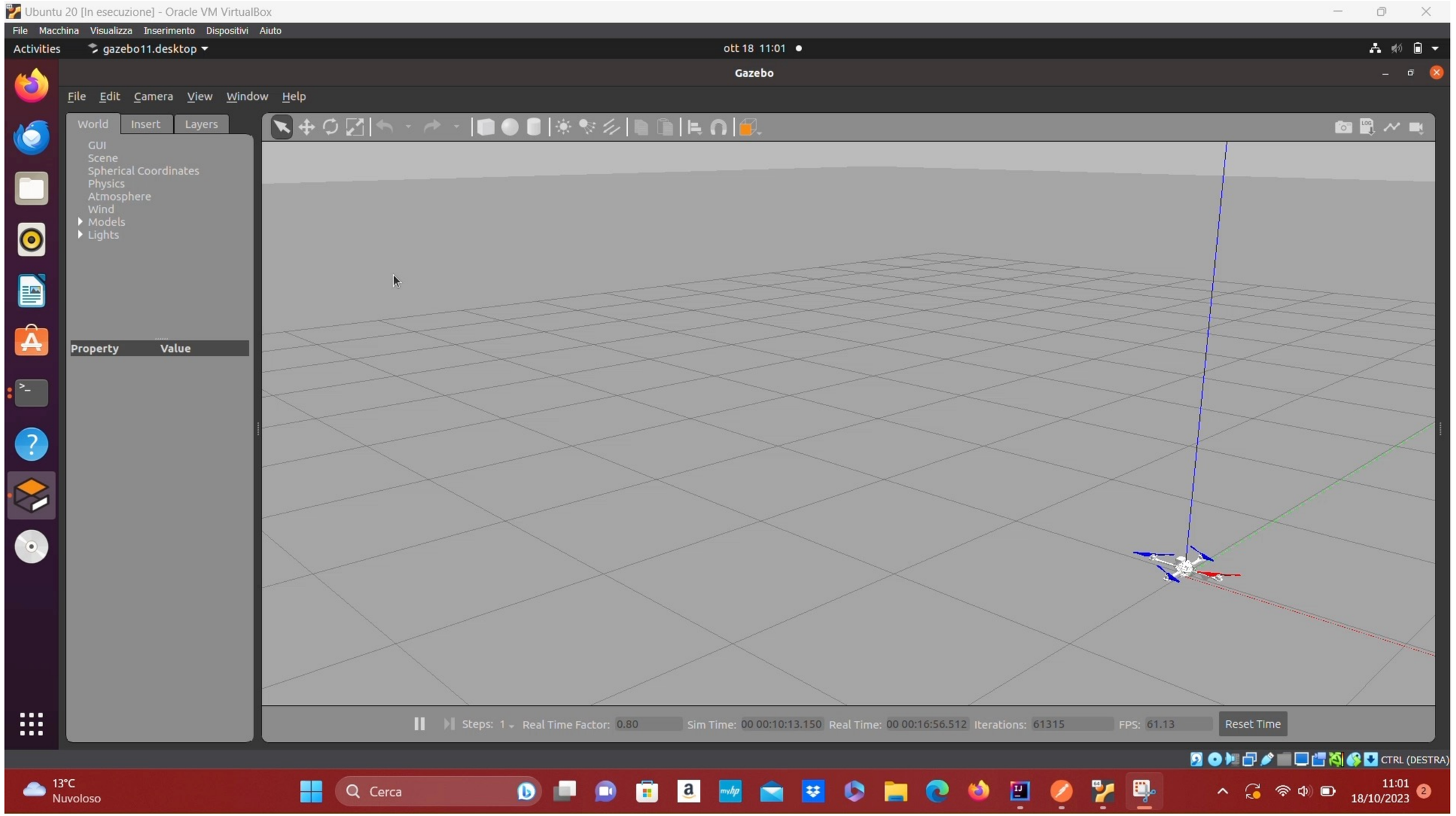
Digital Twins Drones – Physical Adapter



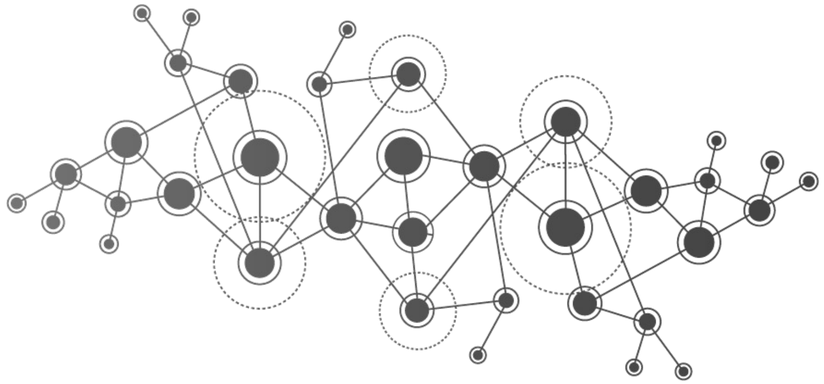
Digital Twins & Drones



Digital Twins & Drones



UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA



Intelligent Internet of Things

IoT Digital Twins
HandsOn Session

Prof. Marco Picone

A.A 2023/2024