Meta-MES

A Factory Abstraction Architecture for I4.0

ESSM Workshop – 16 June 2022

Franco Fummi
Alessia Bozzini
Marco Bizzantino
Sebastiano Gaiardelli
Michele Lora
Marco Panato
Stefano Spellini
Carlo Tadiello

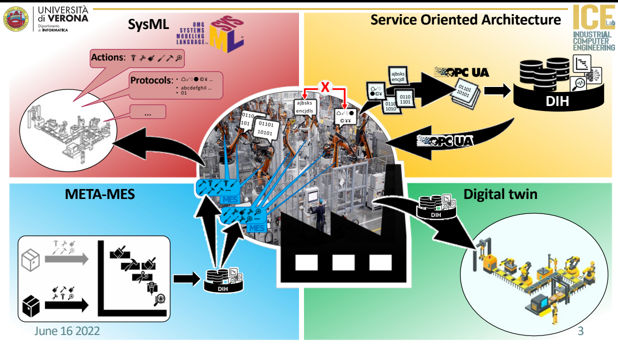
1

Abstract

- Smart factories can be seen as complex cyber-physical production systems (CPPSs)
  - their design, implementation, management and evaluation needs some abstraction strategies to focus on the relevant aspects instead of on the low-level details
- This tutorial proposes an abstraction methodology, and related tools:
  - It starts from the way to build a complete model of the CPPS, based on SysML
  - there is the description of protocols (like OPC-UA) to see the CPPS as a service-oriented architecture,
  - where IIoT data are collected by an ad-hoc architecture,
  - the model allows the automatic configuration of this data collection architecture with the generation also of the digital-twin of the production line,
  - finally, the integration of different Manufacturing Execution Systems (MESs) produces an integrated view through the so-called Meta-MES
- The theoretical part of the tutorial is completed:
  - by the visit of an actual production line

June 16 2022 2

2



June 16 2022 3

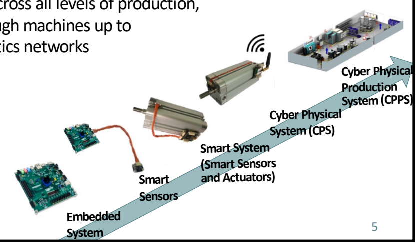
3

CPPS

4

What is a CPPS?

- **Cyber-physical production systems** (CPPS) consist of autonomous and co-operative elements and subsystems that are connected based on the context within and across all levels of production, from processes through machines up to production and logistics networks

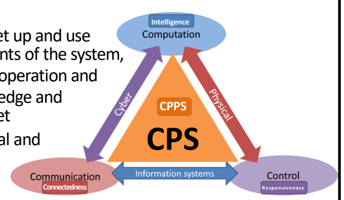


June 16 2022 5

5

Main CPPS characteristics

- **Intelligence** (smartness): the elements are able to acquire information from their surroundings and act autonomously and in a goal-directed manner
- **Connectedness**: the ability to set up and use connections to the other elements of the system, including human beings, for co-operation and collaboration, and to the knowledge and services available on the Internet
- **Responsiveness** towards internal and external changes



https://arxiv.org/ftp/arxiv/papers/1811/1811.03122.pdf

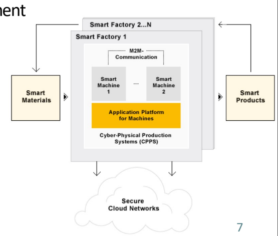
June 16 2022 6

6

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA — ICE INDUSTRIAL COMPUTER ENGINEERING

CPPS and smart factory

- Smart factories can be seen as complex cyber-physical production systems (CPPSs)
- Their design, implementation, management and evaluation needs some abstraction strategies to focus on the relevant aspects instead of on the low-level details

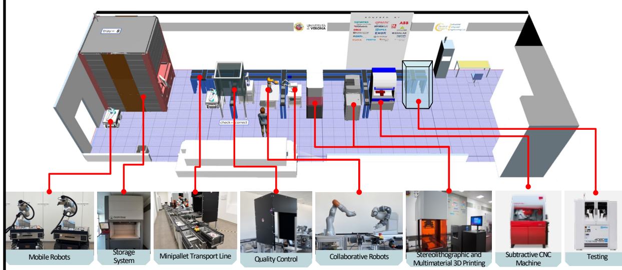


June 16 2022 7

7

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA — ICE INDUSTRIAL COMPUTER ENGINEERING

ICE Laboratory



June 16 2022 8

8

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA — ICE INDUSTRIAL COMPUTER ENGINEERING

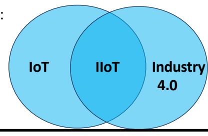
IIoT

9

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA — ICE INDUSTRIAL COMPUTER ENGINEERING

Sensor: IIot vs Iot

- **Sensor**: a device, module, machine, or subsystem
  - Detect events or changes in its environment
  - Send the information to other electronics
- **IoT** (Internet of Things):
  - A system of interrelated computing devices, mechanical and digital machines, objects that are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction
  - Embedded systems such as sensors and actuators
- **IIoT** (Industrial Internet of Things):



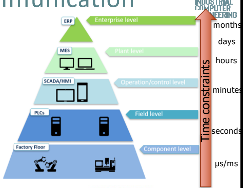
June 16 2022 10

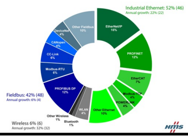
10

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA — ICE INDUSTRIAL COMPUTER ENGINEERING

Industry 4.0 and the communication

- Standard automation pyramid
  - from actuators and sensors to enterprise systems
  - result of the Third industrial revolution
- At the lowest level, different vendors provide different fieldbus protocols
  - Fielbuses like Profibus DP, Modbus-RTU
  - Industrial Ethernet networks like PROFINET, EtherCAT, Modbus TCP, EtherNet/IP
- **Challenge**: communication technology for all the levels of the pyramid.





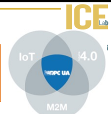
June 16 2022 11

11

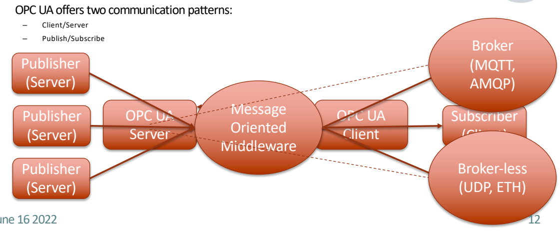
UNIVERSITÀ di VERONA Dipartimento di INFORMATICA — ICE Lab

Solution: OPC Unified Architecture

Designed for secure, reliable and interoperable M2M communication
Service oriented architecture (SOA) follows the request/response paradigm.



- OPC UA offers two communication patterns:
  - Client/Server
  - Publish/Subscribe

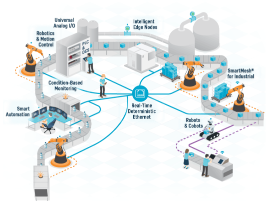


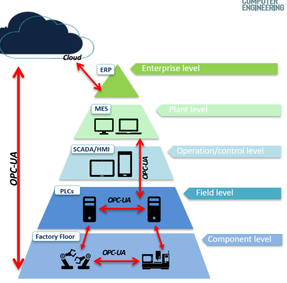
June 16 2022 12

12

Client/Server VS PubSub

- Different aspects are considered when using the model / Analysis of PubSub applicability in
  - Cloud scenarios
  - Real-time requirements / Real-time scenarios
  - Information fan-out





June 16 2022 13

13

IoT vs Industrial IoT - I

1. *Communications*:
   - IoT devices use *standard IT protocols* like *Wi-Fi*, *Ethernet*, …
   - IIoT devices must support *industrial protocols* like *Profinet*, *Ethercat*, …
2. *Safety and Security*:
   - IoT devices *access the Internet* directly, so they must be *secured*
   - IIoT devices live in the *OT network*, so they must take care of *safety rather than security*
3. *Reliability*:
   - it is *not so critical* for IoT devices
   - IIoT devices must guarantee *high reliability* to *keep safe the production plant*

June 16 2022 14

14

IoT vs Industrial IoT - II

4. *Architecture*:
   - IoT devices communicate with a *public cloud* to access user information
   - IIoT devices uses a *private* or hybrid *cloud* which contains *company data*
5. *End devices:*
   - IoT devices *sense data* (like steps, blood pressure, …) for a consumer-type ecosystem
   - IIoT devices provide *precise measures* to both *cloud and PLCs*
6. *Costs*:
   - IoT devices must be *as cheap as possible*
   - IIoT devices need to be *certified* so their cost can be slightly higher

June 16 2022 15

15

AN EXEMPLIFICATION CASE: ICE LABORATORY

16

(I)IoT Solution - Layers

An IoT solution is composed of many layers:

- Security
- Management

4. analytics platform
3. core platform
2. communication protocols / device gateway
1. device layer

June 16 2022 17

17

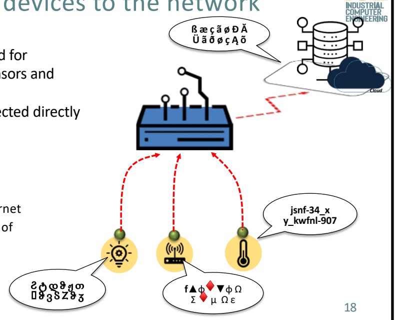
Connect devices to the network

PROBLEM

- Absence of a unique standard for communication between sensors and central processing
- The sensors cannot be connected directly to the IP/internet network

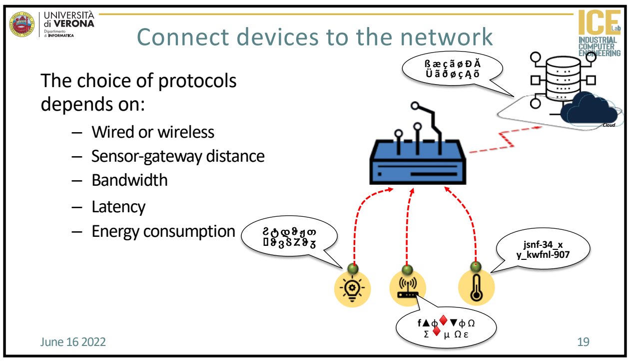
SOLUTION: Gateway

- Support different protocols
- Connect devices to the internet
- Provides an additional level of security
- Pre-process data
- Reduce transmission size

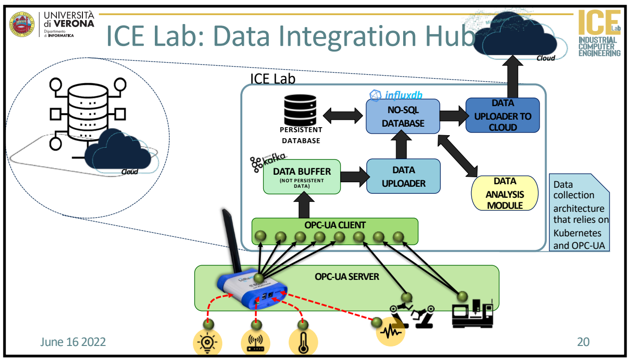


June 16 2022 18

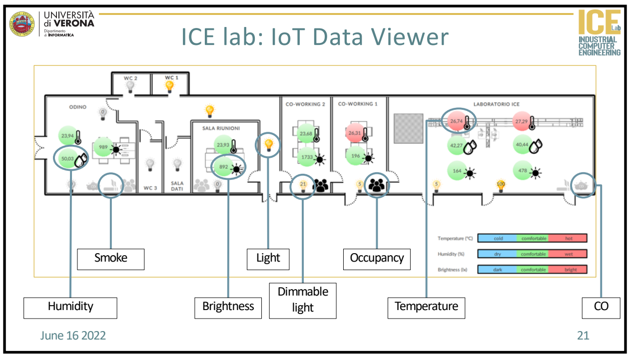
18



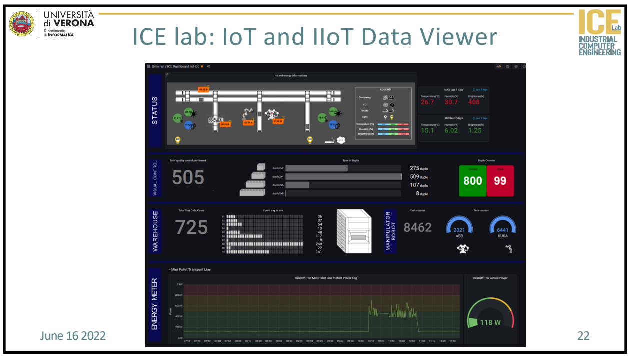
19



20

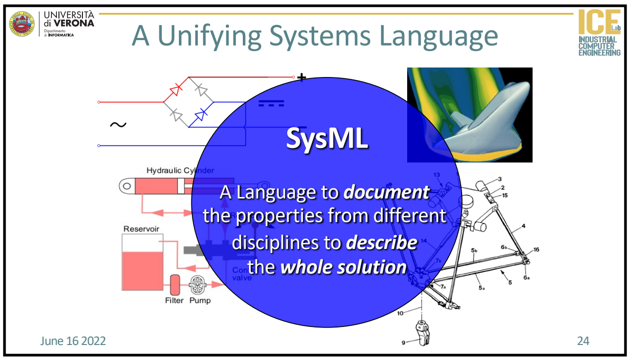


21



22

23

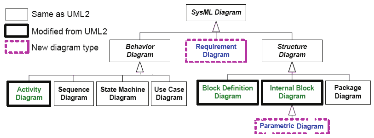


24

UNIVERSITÀ di VERONA
ICE INDUSTRIAL COMPUTER ENGINEERING

Systems Modeling Language (SysML): origin

- Extension (dialect) of **UML**
- Supported system's viewpoints:
  - Structure
  - Behavior
  - Parameters
  - Requirements
- Elementary modeling entity: **diagrams**

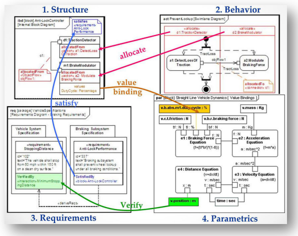


25

UNIVERSITÀ di VERONA
ICE INDUSTRIAL COMPUTER ENGINEERING

Systems Modeling Language (SysML)

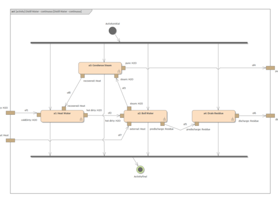
- Elementary modeling entity: **diagrams**
- Supports the **specification**, **analysis**, **design**, **verification** and **validation** of a broad range of systems and systems-of-systems



26

UNIVERSITÀ di VERONA
ICE INDUSTRIAL COMPUTER ENGINEERING

Activity Diagram (ACT)



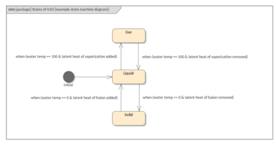
- Activity diagram shows **system dynamic behavior** using a combined Control Flow and Object (data) Flow model
  - **Control Flow** = flow of functional behaviors
  - **Object Flow** = data flow of object inputs/outputs into/from an Activity or Action
- Composed of **Activity Nodes**
  - **Action = atomic Activity, which is a primitive executable behaviour**
- **ACT Purpose**
  - Specify **dynamic system behaviors** that Satisfy system **Functional Requirements**
  - ACT diagrams are simulatable

27

UNIVERSITÀ di VERONA
ICE INDUSTRIAL COMPUTER ENGINEERING

State Machine Diagram (SMD)

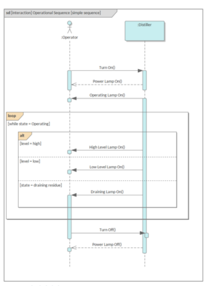
- State Machine diagram shows the **sequences of States** that an object or an interaction go through during its lifetime
  - In response to Events (Triggers)
  - May result in side-effects (Actions)
- Composed of **States**
  - A condition in the lifecycle of an object, during which it is
    - Satisfying a condition
    - Performing an activity
    - Waiting for some event
- SMD Purpose
  - Specify dynamic system behaviors for time-critical, mission-critical, safety-critical, or financially-critical objects
  - SMD are simulatable



28

UNIVERSITÀ di VERONA
ICE INDUSTRIAL COMPUTER ENGINEERING

Sequence Diagram (SD)



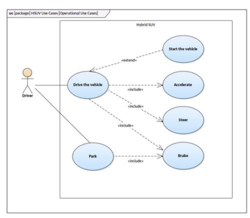
- A Sequence diagram is a dynamic behavioral diagram, showing:
  - Interactions (collaborations) among distributed objects or services using
    - **sequences of messages** exchanged, along with corresponding (optional) **events**
  - Supports **recursive nesting**
- A message represents the **communication** between objects
  - Messages may be **synchronous** (notation: open arrowhead) or **asynchronous** (notation: black-triangle arrowhead)
- Purpose:
  - Specify dynamic system behaviors as **message-passing collaborations** among prototypical Blocks (Parts)
  - SDs are simulatable

29

UNIVERSITÀ di VERONA
ICE INDUSTRIAL COMPUTER ENGINEERING

Use Case Diagram (UC)

- A Use Case diagram shows
  - **Communications** among system transactions (**Use Cases**) and external users (**Actors**)
  - Actors may represent persons, organizations, software systems, or hardware systems
  - Defines the **system scope**
- A Use Case represents a **system transaction** with an external system user (Actor)
  - *Sometimes considered as high-level functional requirements*
- Purpose:
  - To provide a **high-level view** of the subject system
  - Convey the top-level system requirements in non-technical terms for all stakeholders

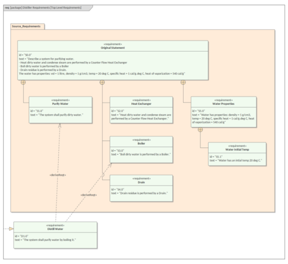


30

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA

ICE INDUSTRIAL COMPUTER ENGINEERING

Requirement Diagram (REQ)



- Requirement diagram shows the relationships among
  - Requirement constructs
  - Model elements that satisfy them
  - Test Cases that verify them
- Composed of **Requirements**
  - **Capability or condition** that a system must ("shall") **satisfy**
  - Can express **functional** (a function that a system must perform) and **non-functional** (quality criteria) requirements
- REQ Purpose
  - Specify both Functional and Non-Functional Requirements within the model
  - Can be traced to other model elements that **Satisfy** them and Test Cases that **Verify** them

June 16 2022 31

31

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA

ICE INDUSTRIAL COMPUTER ENGINEERING

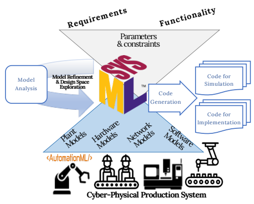
MODELING CPPS WITH SYSML

32

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA

ICE INDUSTRIAL COMPUTER ENGINEERING

Methodology Overview



- We propose a methodology based on the **Platform-based Design (PBD)** paradigm:
  - Top-down **modeling** of requirements and functionalities
  - Bottom-up **reuse** of available components and models
- Core to the methodology a language able to capture both cyber and physical aspects of CPPSs
  - The System Modeling Language (SysML)
    - A **unified representation** and modeling language for the entire system design process
- **Bottom-up**:
  - **Reuse** of AutomationML (AML) specifications to generate SysML models
- **Top-Down**:
  - Model **novel functionalities**
  - **Refine** them onto the model of the **already existing** architecture

June 16 2022 33

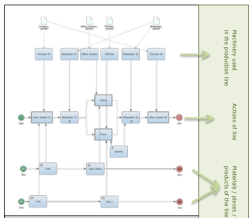
33

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA

ICE INDUSTRIAL COMPUTER ENGINEERING

Taxonomy DIN – Business Process Model and Notation

- The **DIN 8580** standard categorizes the existing industrial processes in five macro sections.
- Each machine is represented by the actions it performs
- **Business Process Model** and **Notation** (BPMN)
  - **Graphical representation**
  - Allows to specify business processes
- We use a **custom** BPMN as a front-end
- BPMN is **consistent** with the ISA-95 standard



Example in BPMN of a small production line

June 16 2022 34

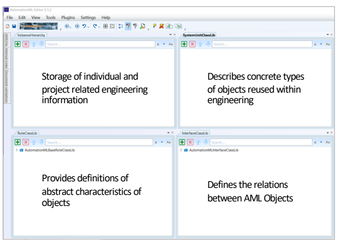
34

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA

ICE INDUSTRIAL COMPUTER ENGINEERING

Automation Markup Language

- **Automation Markup Language** (AML) is an open standard
- Neutral data format based on XML for:
  - Storage
  - Exchange of plant engineering information
- Goal: **interconnect** the heterogeneous modern **engineering tools** in their different disciplines
- AML is divided into 4 parts:
  - Instance Hierarchy
  - System Unit Class Lib
  - Role Class Lib
  - Instance Class Lib



June 16 2022 35

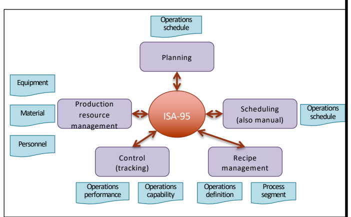
35

UNIVERSITÀ di VERONA Dipartimento di INFORMATICA

ICE INDUSTRIAL COMPUTER ENGINEERING

ISA-95

- ISA-95 is an international **standard**
- Used to develop an **automated interface** between enterprise and control systems
- Applied in:
  - all industries
  - batch, continuous and repetitive processes
- We create an AML model **consistent** with the ISA-95 standard
- AML is divided **according to the standard** subdivisions of ISA-95



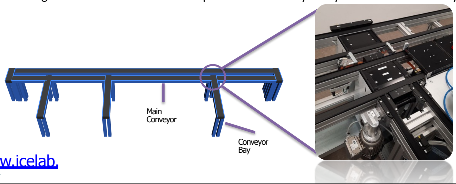
Subdivision of standard ISA-95

June 16 2022 36

36

Case-study: UNIVR ICE Laboratory

- To exemplify the system modeling methodology, we adopt the **transportation system** of the ICE Laboratory as a case-study
  - The **production line** available at University of Verona research facility
- We concentrate on a **subset of components**
  - The switching mechanism that routes mini-pallet to the conveyor bay from the main conveyor



https://www.icelab.di.univr.it

June 16 2022 37

37

Exploiting Models for CPPSs Design

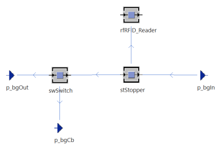
- Models produced using the proposed methodology may be used for:
  - ***Verification and Validation*** of SysML models based on **formal methods**
  - ***System Analysis and Optimization*** through design-space exploration
    - Intrinsic of any PBD flow
    - **Optimization problems** and formal models can be built on top of SysML models
  - ***Code Generation:***
    - **Hardware-software integration**
    - **Control software implementation**
      - E.g., PLC software consistent with the IEC 61131-3 standard
    - **Simulation**

June 16 2022 38

38

Generation of Simulation Scenarios

- **Generate the code** for simulation
  - From SysML diagrams
- **Code import** in a **plant simulation** tool
  - E.g., *Tecnomatix Plant Simulation*
- A two-phased task:
  - ***Structural Information Extraction***: extraction of **plant topology** from SysML structural diagrams
  - ***Behavioral information extraction***: translation of behavioral diagrams to C code implementations



June 16 2022 39

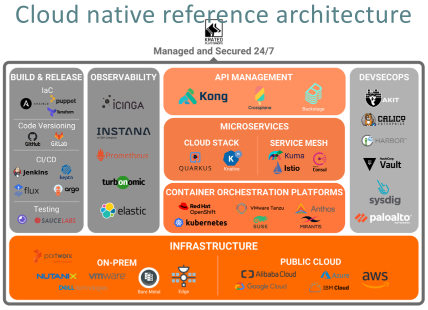
39

Containerization and Kubernetes

KIRATECH

40

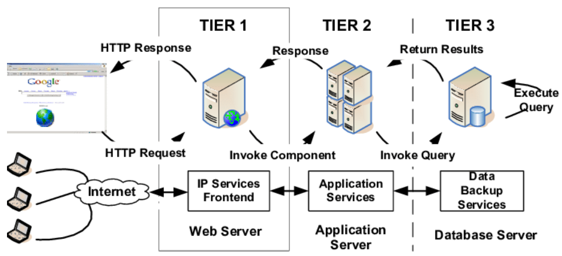
Cloud native reference architecture



June 16 2022 41

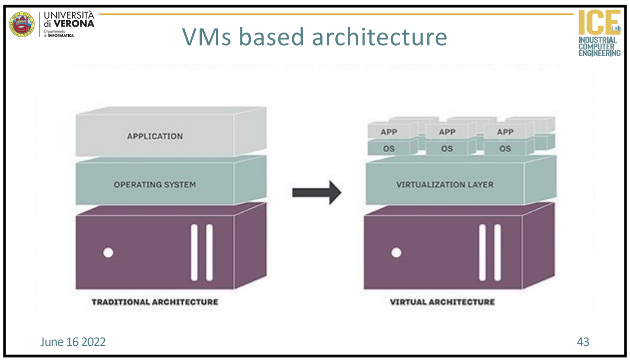
41

Traditional architecture

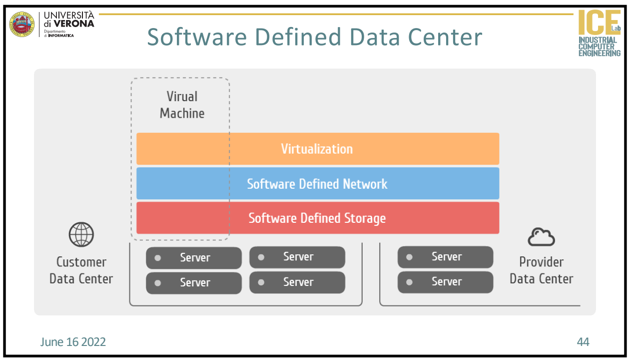


June 16 2022 42

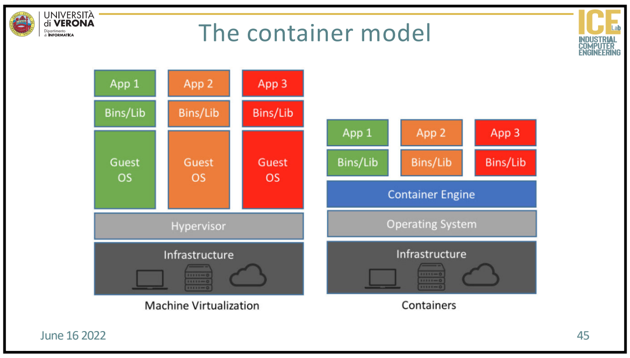
42



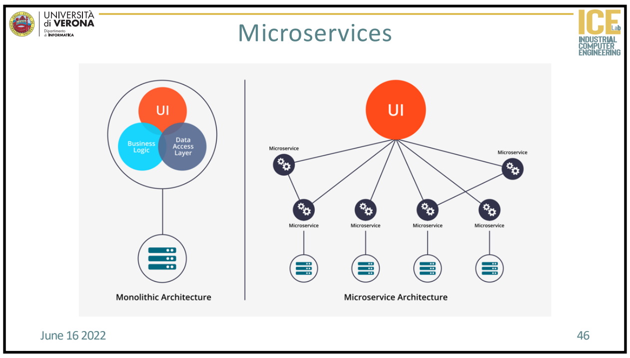
43



44



45



46

What is Kubernetes

- Project that was spun out of Google as an open source container orchestration platform.
- Built from the lessons learned in the experiences of developing and running Google's Borg and Omega.
- Designed from the ground-up as a **loosely coupled** collection of components centered around deploying, maintaining and scaling workloads.
- Known as the **linux kernel of distributed systems**.
- **Abstracts away the underlying hardware** of the nodes and provides a uniform interface for workloads to be both deployed and consume the shared pool of resources.
- Works as an engine for resolving state by converging actual and the **desired state** of the system.

June 16 2022 47

47

Decouples Infrastructure, Scaling and Self Healing

All services within Kubernetes are natively Load Balanced.

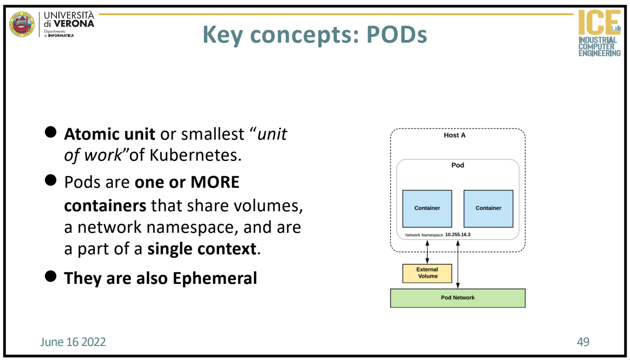
Can scale up and down dynamically.

Used both to enable self-healing and seamless upgrading or rollback of applications.

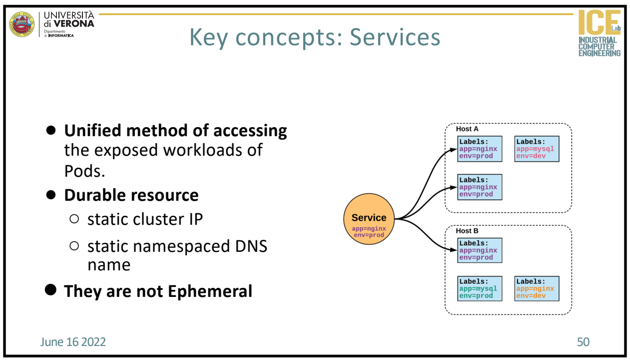
Kubernetes will **ALWAYS** try and steer the cluster to its desired state.

June 16 2022 48

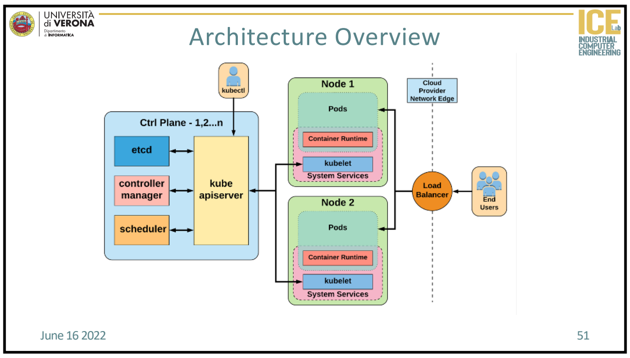
48



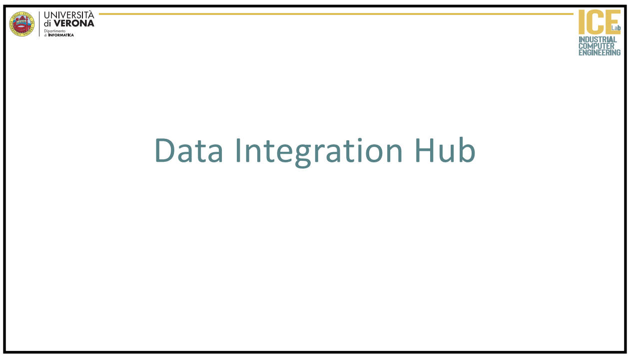
49



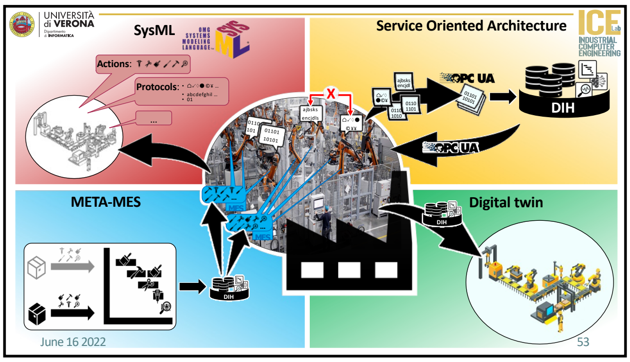
50



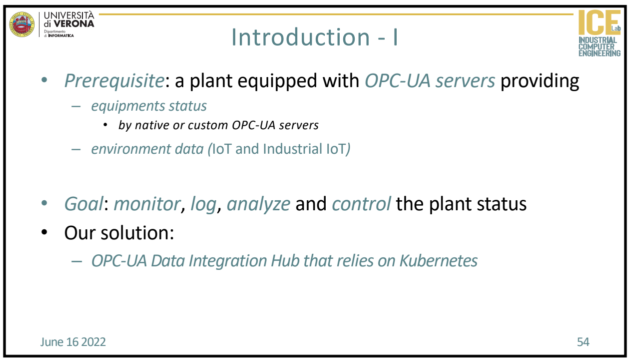
51



52



53



54

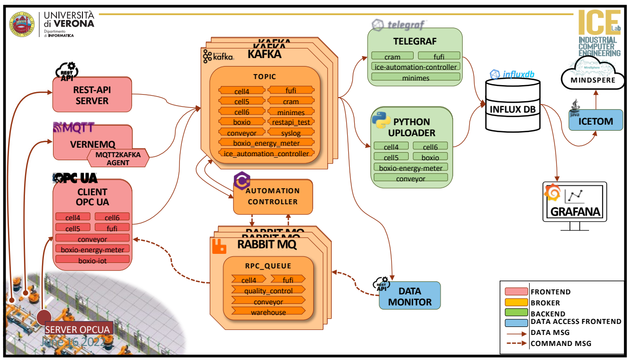
UNIVERSITÀ di VERONA

Introduction - II

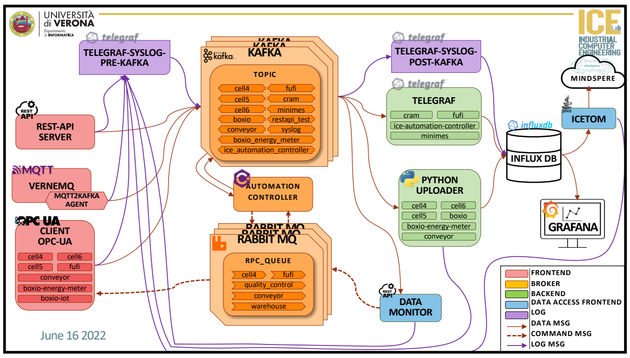
- Architecture goals:
  - OPC-UA servers *monitoring*
  - multi-protocol support for *data ingestion*
  - data *logging*
  - *data analytics* and filtering
  - data *upload to different cloud* providers (hybrid cloud approach)
  - provide a *unique interface* to access data
  - recipe *execution* and *scheduling* on supported equipments

June 16 2022 55

55



56



57

UNIVERSITÀ di VERONA

Frontend – OPC-UA Client

- Frontend Node
- Reconfigurable:
  - *OPC-UA server uri*
  - variable list to subscribe, for each:
    - *sampling interval*
    - *datatype* and *unit*
    - custom tags
  - *topic name* to publish data to
- *At least* one instance per machine

- RPC Methods
  - *read(variablePath)*
    - requests an OPC-UA variable read operation
  - *write(variablePath, value)*
    - writes the provided value to the OPC-UA variable
  - *call_method(methodName, args)*
    - calls one OPC-UA method with the provided arguments

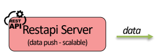
June 16 2022 58

58

UNIVERSITÀ di VERONA

Frontend – RestAPI Server

- Simple node that provides data upload through HTTP PUT request
  - /api/v1/push/<topic>
    with json payload



- *Scalable: ~7000req/s per core*
- All data is sent to the provided kafka topic

June 16 2022 59

59

UNIVERSITÀ di VERONA

Frontend – MQTT Broker

- Vernemq MQTT Broker
  - *native integration with kubernetes*
  - *multiple instances can be spawned*
    - *actually 1 is alive*
- MQTT2Kafka Agent (python)
  - transfers messages to Kafka
  - topic translation mechanism
    - MQTT: *ice/data/xxxx*
    - Kafka: *ice_data_xxxx*

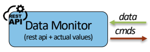
- No performance data available yet

June 16 2022 60

60

Data Access Frontend – Data Monitor - I

- Subscribed to all Kafka topics
  - *reads each message and stores the last value of each variable*
- Allows to query the latest values for each variable through HTTP GET

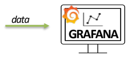


- Endpoints
  - */data -> list measurements*
  - */data/measurement -> list fields*
  - */data/measurement/field -> get value(s)*

61

Data Access Frontend – Grafana

- Grafana is an analysis and monitoring solution for every database
- We created different dashboards to show stored data from InfluxDB

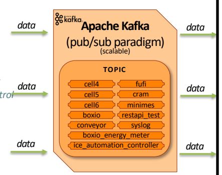


62

Broker – Kafka - I

- A *streaming platform* that supports mutiple producers and consumers
  - *topic concept*
  - *consumer group concept*
- Possibility to store temporarly all topic messages (1 week)
- Actually deployed with *3 instances*
- Used to move data from ICE lab to consumers (dbs, data monitor, …)

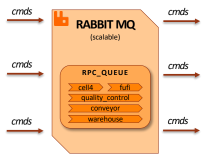
- Data partitioned in *topics*
  - *ice_data_cell4*
  - *ice_data_cell5*
  - *ice_data_cell6*
  - *ice_data_conveyor*
  - *ice_data_fufi*
  - *ice_data_boxio*
  - *ice_data_boxio_energy_meter*
  - *ice_data_ice_automation_controller*
  - *ice_data_syslog*



63

Broker - RabbitMQ

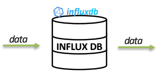
- Message Oriented Middleware adopted to handle the *platform commands*
  - *deployed with three replicas*
- Messages are moved through *queues*
- One queue per RPC service (e.g. OPC-UA clients)
  - conveyor_rpc_queue
  - cell4_rpc_queue
  - cell5_rpc_queue
  - warehouse_rpc_queue
  - fufi_rpc_queue



64

Backend – InfluxDb Agent

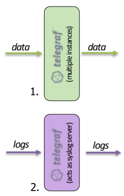
- Python application
  - subscribes to provided Kafka topics
    (possibility to specify group-id)
  - rewrites incoming data to DB Abstraction Layer
    (possibility to send chunk of data to reduce write frequency)
- Multiple instances to cover all kafka topics
- Two writing modes
  - *synchronous*: sends immediatly each value
  - *asynchronous*: acts as buffer sending chunks



65

Frontend – Backend - Telegraf

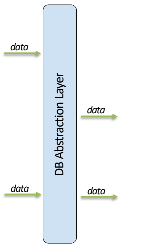
- *Go* application born to collect, aggregate, process and write metrics
- It can read data from *Kafka topics* and write it to the *DB Layer* *(image 1)*
- Can accept data formatted accoding to *syslog protocol* and write it to a *Kafka topic* *(image 2)*
- …



66

Backend – DB Abstraction Layer

- Application able to *partition data* to multiple databases
  - configurable
  - supports *NoSQL (InfluxDB)* and *SQL (Postgres)* dbs
  - exposes API to write data and perform queries
- (this node has not been yet implemented!)

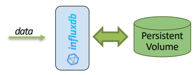


June 16 2022 67

67

Backend – InfluxDB2

- NoSQL Database optimized to ingest *time series*
- Supports multiple buckets to store data
- Very powerful query language: *flux*
- Integrated web interface to perform queries

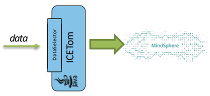


June 16 2022 68

68

Backend - ICETom

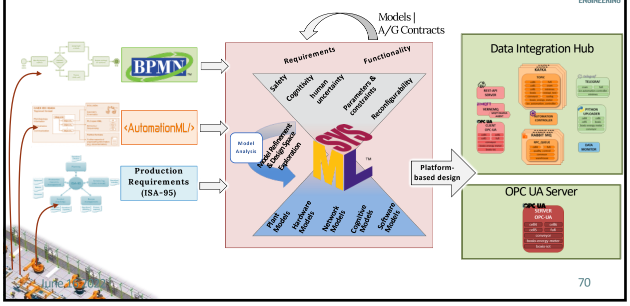
- Java application that is bundled with Mindsphere APIs
  - the *dataselector* part select chunks of data from InfluxDB
  - selected data is *aggregated* as configured
  - data is *pushed to Mindsphere* using its API



June 16 2022 69

69

Data Integration Hub generation

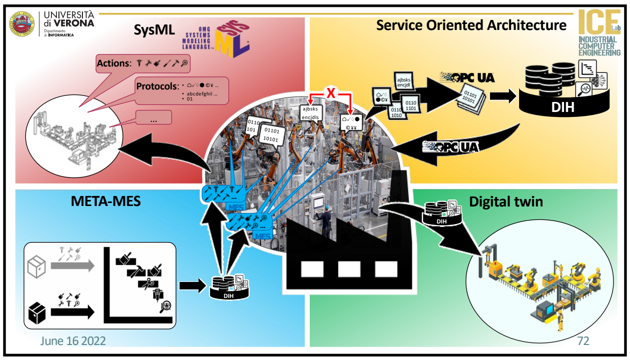


June 16 2022 70

70

Service-oriented architecture

71

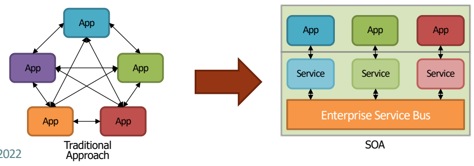


June 16 2022 72

72

What is SOA?

- **Service-Oriented Architecture (SOA)** defines a way to make software components reusable and interoperable through shared services
  - **Each service** implements a simple task such as retrieving information or performing an action
    - Exposed by software components
  - Services are accessible through an **(Enterprise) Service Bus (ESB)**
    - Loose coupling
    - Location transparency



June 16 2022 73

73

SOA benefits

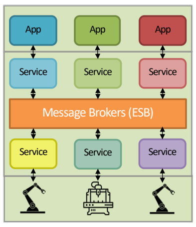
Reusability	Business logic is divided into services shared among multiple applications
Easily maintained	Since all services are independent, they can be easily modified and updated without affecting other services.
Promotes interoperability	The use of a ESB allows to easily transmit data between applications and services regardless of the languages they're built on.
High availability	SOA facilities are available to anyone upon request
Composability	Collections of services can be coordinated and assembled from composite services to build more complex services
Autonomy	Services have control over the logic they encapsulate
Scalable	Services can be replicated on multiple instances, increasing scalability

June 16 2022 74

74

Service-oriented Manufacturing

- **Service-oriented manufacturing** applies the concept of SOA to smart manufacturing:
  - **Manufacturing resources** connected with SOA principles
    - E.g. Through message brokers (ESB)
  - Each **machine exposes** its interface as services
    - Increases production flexibility and interoperability
    - Standard M2M communication
  - Ease the application of **modern production paradigms**
    - Mass customization
    - Automatic re-configuration
    - Flexible manufacturing
  - **Challenging application** in manufacturing
    - Previous software doesn't support modern paradigms
    - Requires huge investment
    - Difficult to integrate new technologies

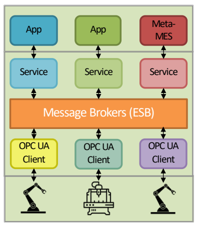


June 16 2022 75

75

Service-oriented Middleware (SOM)

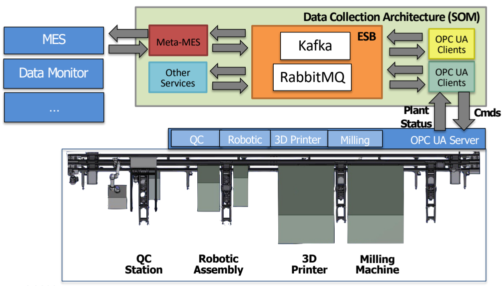
- **Service-oriented middleware (SOM)** to ease the application of modern production paradigms
  - Adapt the previous software extending its capabilities
    - Easy integration of new technologies
    - Hide the complexity necessary to support new technologies
    - Avoid changing the entire information infrastructure
  - Service orchestration performed by another component (E.g. Meta-MES)
    - Work orders as a sequence of services
    - Services composability enables correct sequence based on behavior models
  - OPC UA Client offer machine capabilities as services
    - Share machine capabilities among all the applications inside the SOM



June 16 2022 76

76

SOM – ICE

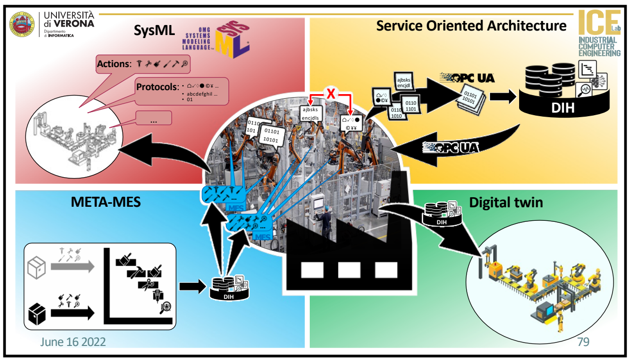


June 16 2022 77

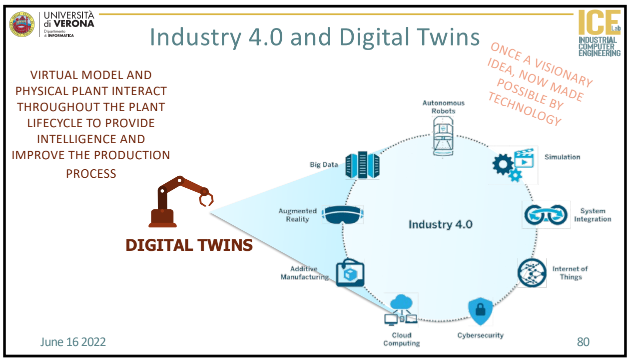
77

Digital Twin

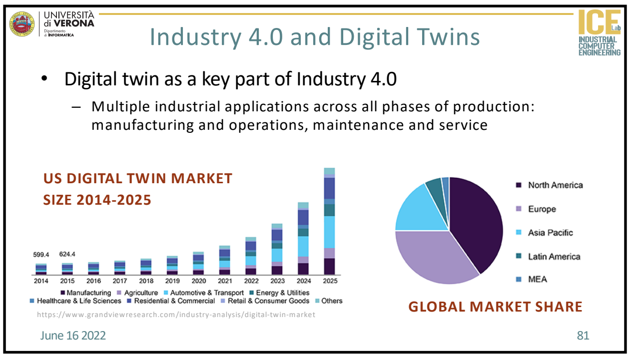
78



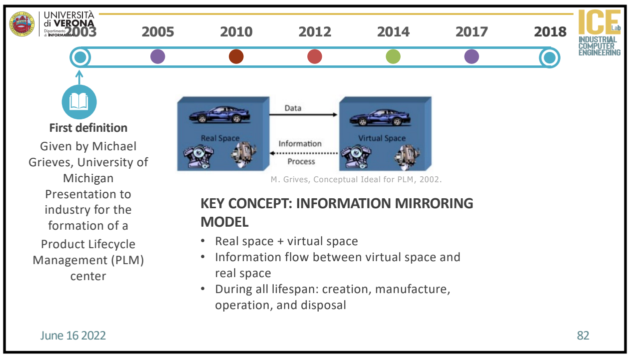
79



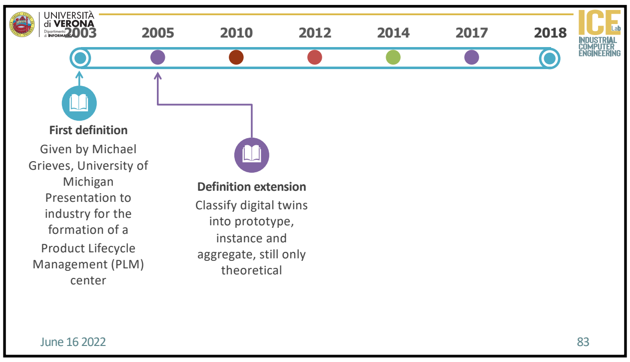
80



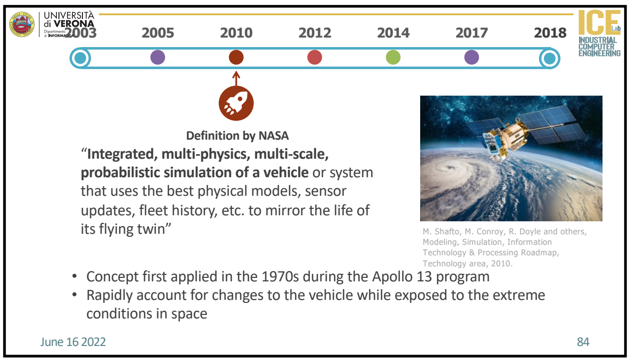
81



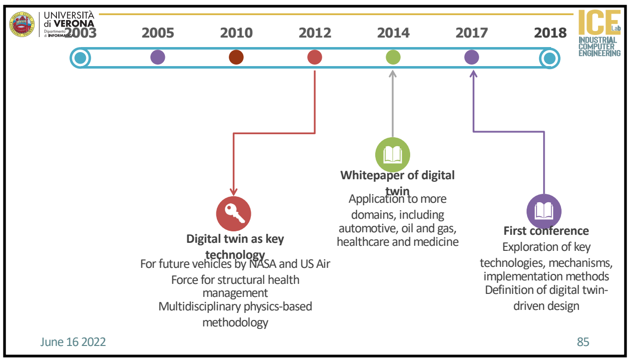
82



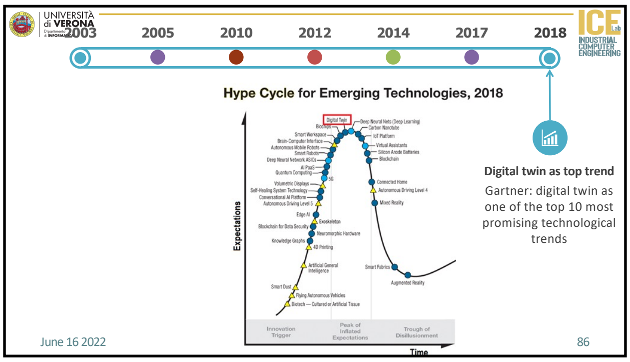
83



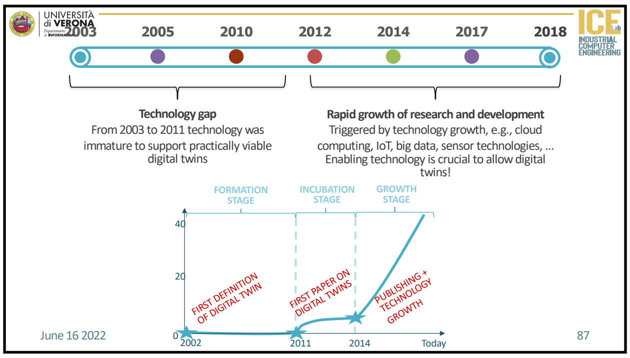
84



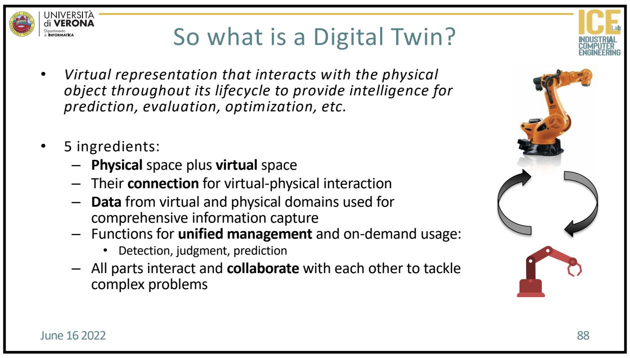
85



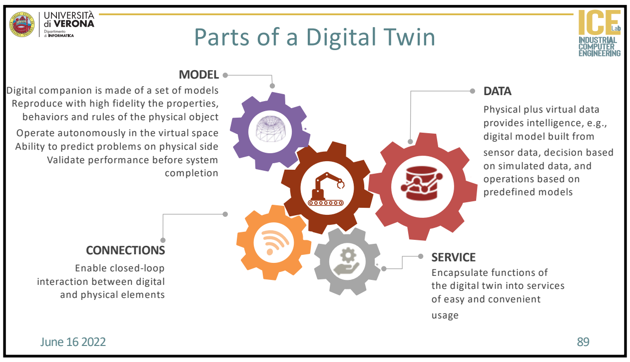
86



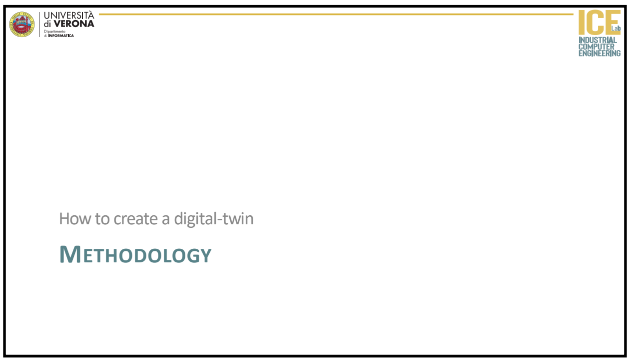
87



88

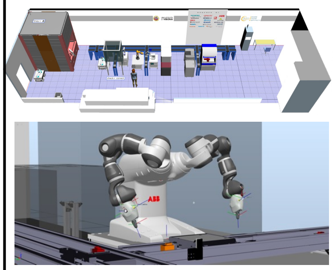


89



90

Digital twin



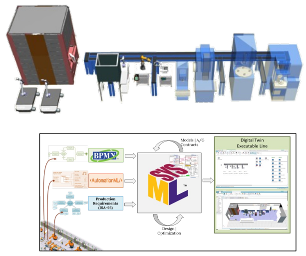
- A **digital twin** is a digital replica of a living or non-living physical entity
- A digital replica can represent physical assets, processes, people, places, systems and devices
- Can be used for various purposes

- **Types of Digital Twin:**
  1. Autonomous
  2. Connected (digital shadow)
  3. Hybrid

June 16 2022 91

91

1 Autonomous

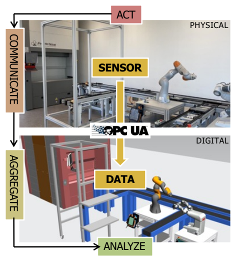


- Independence from the real line
- The line can be:
  - Already existing
  - A new model
- ADVANTAGES:
  - Errors or bottlenecks detection before production validation
  - Number reduction of physical prototypes with initial virtual validation
  - Cycle time optimization through simulation
- SETTINGS:
  - Speed
  - Time
  - Power

June 16 2022 92

92

2 Connected (digital shadow)

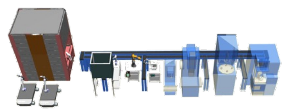


- Digital twin communicates with the real plant with sensors
- The line must exist
- ADVANTAGES:
  - Production line monitoring even when far from the physical line
  - Failures or bottlenecks monitoring
  - Production process optimization thanks to the production data collection
  - Creation of statistics of intermediate and finished products produced by the line
- SETTINGS:
  - The digital twin utilizes the OPC-UA protocol for the communication with machines

June 16 2022 93

93

3 Hybrid

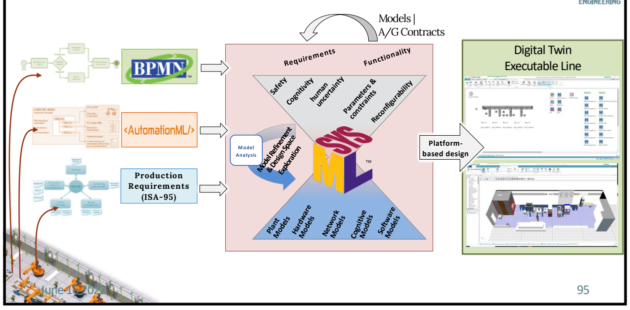


- Digital twin where some machines are real others are hypothetical
- The line exists in the middle
- ADVANTAGES:
  - Statistics of a hypothetical production
  - Optimization of production cycles
- SETTINGS:
  - The digital twin communicates with the existing machines with the OPC-UA protocol
  - Speed of hypothetical machines
  - Time of hypothetical machines
  - Power of hypothetical machines

June 16 2022 94

94

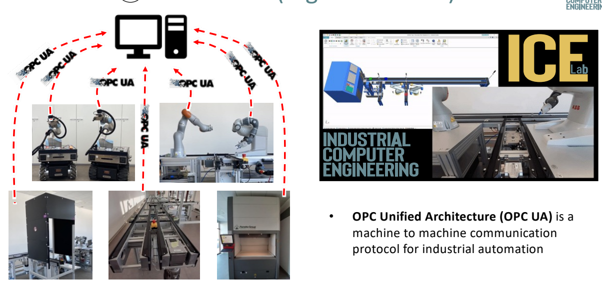
1 Autonomous



June 16 2022 95

95

2 Connected (digital shadow)



- **OPC Unified Architecture (OPC UA)** is a machine to machine communication protocol for industrial automation

June 16 2022 96

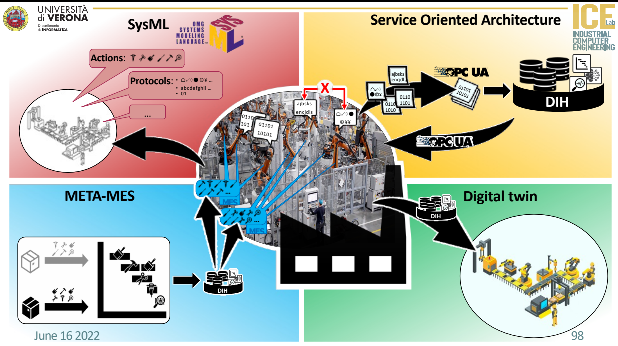
96

UNIVERSITÀ di VERONA

MES

97

UNIVERSITÀ di VERONA



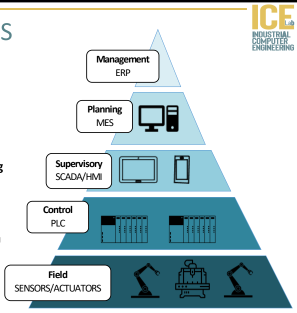
June 16 2022 98

98

UNIVERSITÀ di VERONA

MES

- **Functional layer between enterprise resource planning (ERP) and process control systems**
  - ERP focuses on creating and managing plant schedules including production
  - MES, focuses on managing and monitoring manufacturing operations and reporting on production line activities in real-time
  - ERP and MES create an integrated ecosystem, offering a complete view of the entire manufacturing.
- **Essential in today's competitive and rapidly changing manufacturing environment**
  - Track and document the transformation of raw materials into finished goods
  - Manage, monitor, synchronize the execution of real-time, physical processes involved in manufacturing operations
  - Coordinate the flow of work orders with production scheduling and enterprise-level systems line ERP or product lifecycle management PLM systems
  - Enable the control of multiple elements, such as inventory, personnel, machines, etc..



June 16 2022 99

99

UNIVERSITÀ di VERONA

MES - Functionalities

- **Dispatching Production Units:**
  - Dispatch work instruction from the ERP
    - Scheduling
    - Provide suggestions for unforeseen events
  - Ensure resource availability
    - Real-time state of the resource usage
  - Meet production requirements
    - Production deadline
    - Throughput
  - Maximize the production
    - Increase machine uptime
    - Avoid bottleneck

June 16 2022 100

100

UNIVERSITÀ di VERONA

MES – Functionalities (1)

- **Product Tracking and Genealogy:**
  - Track all the items in the production process
    - Source
    - Materials
    - Equipment
    - Personnel
  - Real-time production visibility
    - Progress tracking
    - Inventory planning
    - Estimate labor costs
  - Warehouse management
    - Intermediate materials
    - Product label
    - Product data management

June 16 2022 101

101

UNIVERSITÀ di VERONA

MES – Functionalities (2)

- **Process Management:**
  - Managing the production process from order release to finished goods
    - Work Steps
    - Work Instruction
  - Reduce human errors
- **Resource Allocation and Status :**
  - Manage the allocation and status of resources
    - Equipment
    - Tools
    - Materials
    - Labor

June 16 2022 102

102

UNIVERSITÀ di VERONA

MES – Functionalities (3)

- **Data Collection and Storage:**
  - Facilitate the acquisition of data
    - Equipment
    - Personnel management
    - Quality control systems
  - Collects data from different task areas
    - Consolidation of data
    - Data correlation
    - Data visualization
- **Quality Management:**
  - Quality monitoring
    - Preventive actions
    - Verification
    - Nonconformance workflows

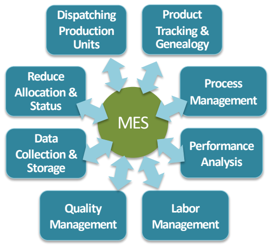
June 16 2022 103

103

UNIVERSITÀ di VERONA

MES – Functionalities (4)

- **Labor Management:**
  - Manage the employees
    - Work time logging
    - Personnel qualifications
    - Personnel certifications
- **Performance Analysis:**
  - Define and track key performance indicators (KPIs)
    - Advanced analytics
    - Data visualization
    - Performance monitoring
    - Reporting



June 16 2022 104

104

UNIVERSITÀ di VERONA

MES - Limits

- **MES are rigid systems with narrow, clearly defined features and system architectures**
  - Hard to customize
    - Time-consuming
    - Expensive licenses
  - Manufacturing workflow needs to be adapted to fit the MES
    - Easier to change the manufacturing operation rather the MES
- **MES have a blind spot for human data**
  - Still poorly human interaction
    - Human errors
    - Production processes are manually

June 16 2022 105

105

UNIVERSITÀ di VERONA

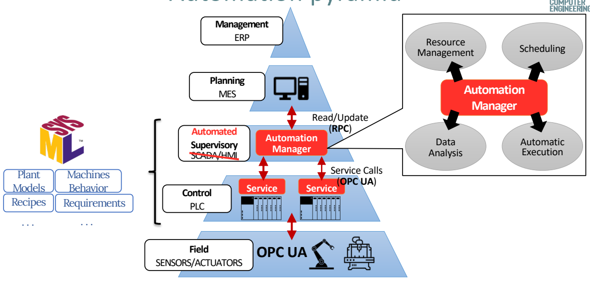
Dynamic control of a production line

META-MES

106

UNIVERSITÀ di VERONA

Automation pyramid



June 16 2022 107

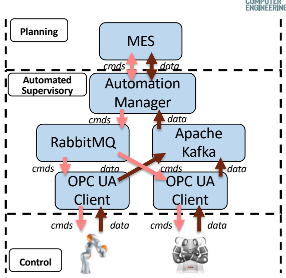
107

UNIVERSITÀ di VERONA

Cloud-native architecture

The current market is characterized by rapid changes in consumer demands. To respond to such issue, we need manufacturing:

- Easily reconfigurable
- Supports the integration of new technologies transparently
- Based on independent microservices



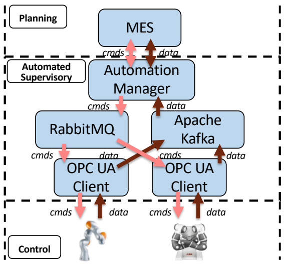
June 16 2022 108

108

Cloud-native architecture

This architecture has many advantages:

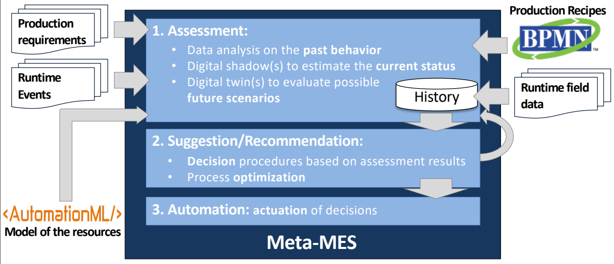
- Seamless integration with existing cloud architecture
- Automated reconfiguration
- Easy integration of new technologies
- Autonomous execution and supervision of production orders
- Runtime dynamic scheduling
- Support for advanced data analysis



June 16 2022 109

109

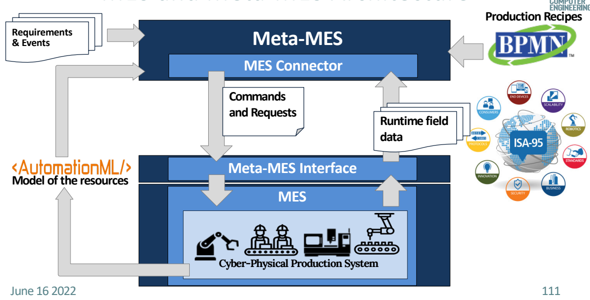
Meta-MES Overview



June 16 2022 110

110

MES and Meta-MES Architecture



June 16 2022 111

111

Recipes

Current MES

- Represent only recipes as a sequence of tasks with dependencies with each other
- Doesn't specify the implementation on the machines

Automation Manager

- Represent only recipes as a sequence of tasks with dependencies with each other
- Specify the implementation of each task
- Support runtime rearrangement of the implementation
- More detailed view of what the system is doing
- The recipes are model in SysML

June 16 2022 112

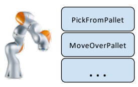
112

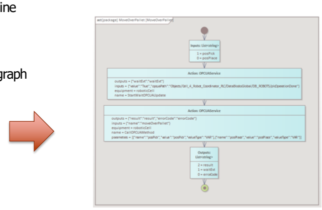
Recipes - Machine Function

Each machine behavior can be described as a sequence of basic actions and basic services exposed

- Avoid modeling the same sequence of actions in different Tasks multiple times
- Increase the flexibility of the production line
- They have input and output parameters
- Each function has a private scope

Inside SysML, are represented with a direct graph





June 16 2022 113

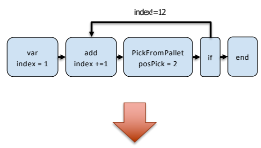
113

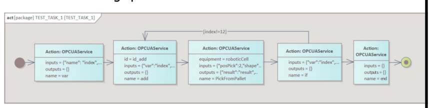
Recipes - Tasks

Tasks are represented as a direct graph

- Nodes represent actions that are executed when a visit takes place
- An action can be a Machine Function, OPC UA action (read/write, method, subscription), or a control action (var, if, add, end)
- Each action has input and output parameters

Inside SysML, tasks are represented with a direct graph





June 16 2022 114

114

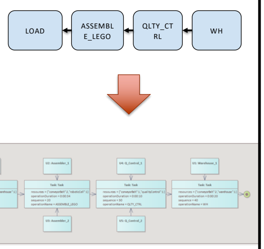
Recipes

Recipes are represented as a direct acyclic graph

- Nodes represent tasks that can be executed on a working cell
- Edges between nodes describe the relation of type "start after"

Inside SysML, Recipes are represented as a Resource-task network (RTN) Graph

- Model also a set of equipment where a certain task can be executed
- Can also represent the intermediate state of the production process and parameters for each machine (energy consumption, precision, etc..)



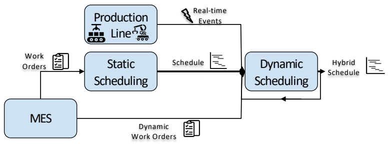
June 16 2022 115

115

Hybrid Scheduling

Scheduling with two phases:

- Static scheduling based on an optimization algorithm that produces an optimal solution used as a basis
- Dynamic scheduling adapts the produced schedule to unexpected events every seconds



June 16 2022 116

116

Dynamic Scheduling

Takes as input the optimal static schedule and unexpected events(new work orders, delays, machine breakdown, etc.). Schedule tasks with three different heuristics:

- Dynamic Assignment
- Production Extension
- Schedule adaptation

June 16 2022 117

117

Dynamic Assignment

Assigns the runtime task in the free space between operations

	10:56	10:57	10:58	10:59	11:00	11:01	11:02	11:03	11:04	11:05
Robotic Cell	OP1	OP1	OP5	OP5	OP6	OP6				
Quality Control 1	OP4	OP4	OP2	OP2			OP7	OP7		
Quality Control 2	OP10	OP10	OP10	OP10	OP4	OP4				
Warehouse					OP3	OP3			OP8	OP8

	10:56	10:57	10:58	10:59	11:00	11:01	11:02	11:03	11:04	11:05
Robotic Cell	OP1	OP1	OP5	OP5	OP6	OP6				
Quality Control 1	OP4	OP4	OP2	OP2	OP9	OP9	OP7	OP7	OP9	
Quality Control 2	OP10	OP10	OP10	OP10	OP4	OP4				
Warehouse					OP3	OP3			OP8	OP8

June 16 2022 118

118

Schedule Adaptation

Swap the order of two tasks if this reduce the total execution time

time

	10:56	10:57	10:58	10:59	11:00	11:01	11:02	11:03	11:04	11:05
Robotic Cell 1	OP1	OP5	OP6	OP6	OP6					
Robotic Cell 2	OP1			OP6						
Quality Control	OP4	OP2	OP2	OP7		OP7	OP7	OP7		
Warehouse			OP3	OP3	OP8		OP8	OP8	OP8	

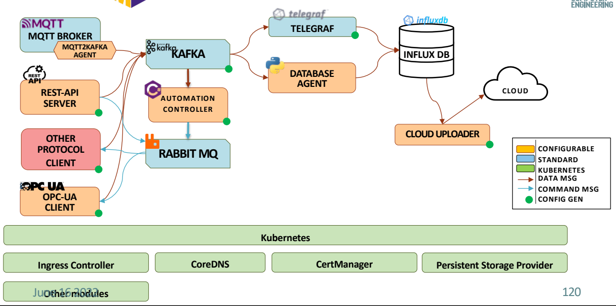
time

	10:56	10:57	10:58	10:59	11:00	11:01	11:02	11:03	11:04	11:05
Robotic Cell 1	OP1	OP5		OP6	OP6					
Robotic Cell 2	OP1			OP6						
Quality Control	OP4	OP2	OP2		OP7	OP7	OP7			
Warehouse			OP3	OP3		OP8	OP8	OP8		

June 16 2022 119

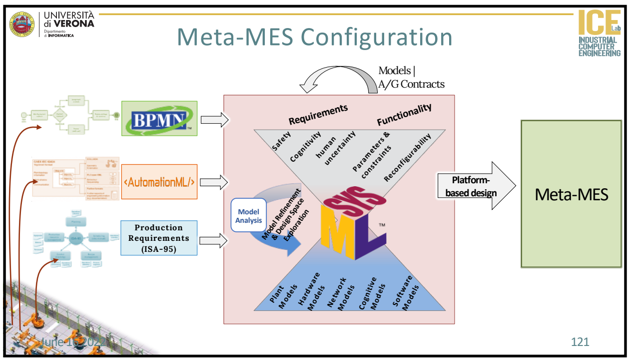
119

Meta-MES Architecture

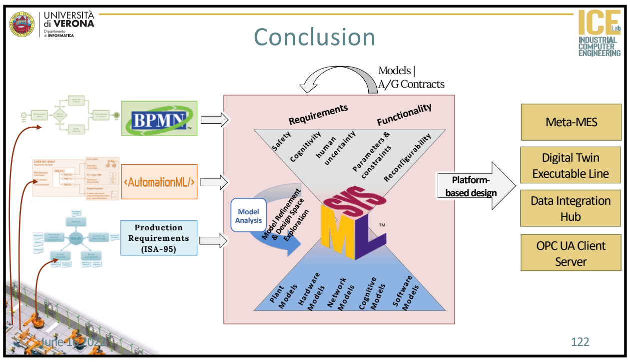


June 16 2022 120

120


UNIVERSITÀ di VERONA
Meta-MES Configuration
ICE
INDUSTRIAL COMPUTER ENGINEERING
Models | A/G Contracts
BPMN
<AutomationML/>
Production Requirements (ISA-95)
Requirements
Functionality
Safety
Cognitivity
human uncertainty
Parameters & constraints
Reconfigurability
Model Analysis
Model Refinement & Design Space Exploration
SYSML
Plant Models
Hardware Models
Network Models
Cognitive Models
Software Models
Platform-based design
Meta-MES
121

121


UNIVERSITÀ di VERONA
Conclusion
ICE
INDUSTRIAL COMPUTER ENGINEERING
Models | A/G Contracts
BPMN
<AutomationML/>
Production Requirements (ISA-95)
Requirements
Functionality
Safety
Cognitivity
human uncertainty
Parameters & constraints
Reconfigurability
Model Analysis
Model Refinement & Design Space Exploration
SYSML
Plant Models
Hardware Models
Network Models
Cognitive Models
Software Models
Platform-based design
Meta-MES
Digital Twin Executable Line
Data Integration Hub
OPC UA Client Server
122

122