SAPIENZA
UNIVERSITÀ DI ROMA

Smart Manufacturing

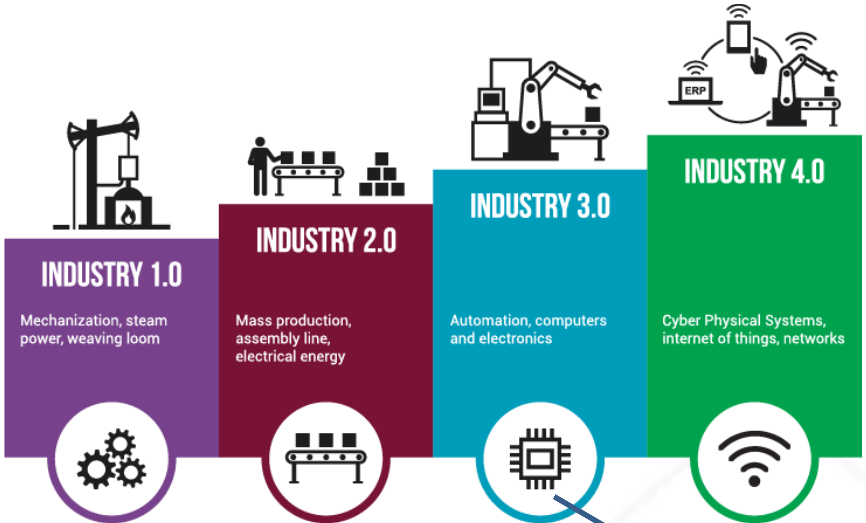
PhD Course on Smart Environments:
Technologies, state of the art and
research challenges

Francesco Leotta
leotta@diag.uniroma1.it

Thanks (in alphabetical order) to ingg. Daniele Bracciani, Marco Calamo, Lucrezia Geusa, Ugo Okeadu and Lucia Rodinò for the work on the die cutter case study

Smart Manufacturing (Industry 4.0)

- Evolution of the traditional world of industrial automation
  - continuously evolving technologies for networking, storage and computing
- Functional to
  - increase productivity and quality
  - ease workers' lives
  - define new business opportunities



- Digital factory is a key concept
  - a multi-layered integration of the **information related to activities** along the factory and related resources

Introduction of Fieldbus (e.g., EtherCAT)

Emerging Trends

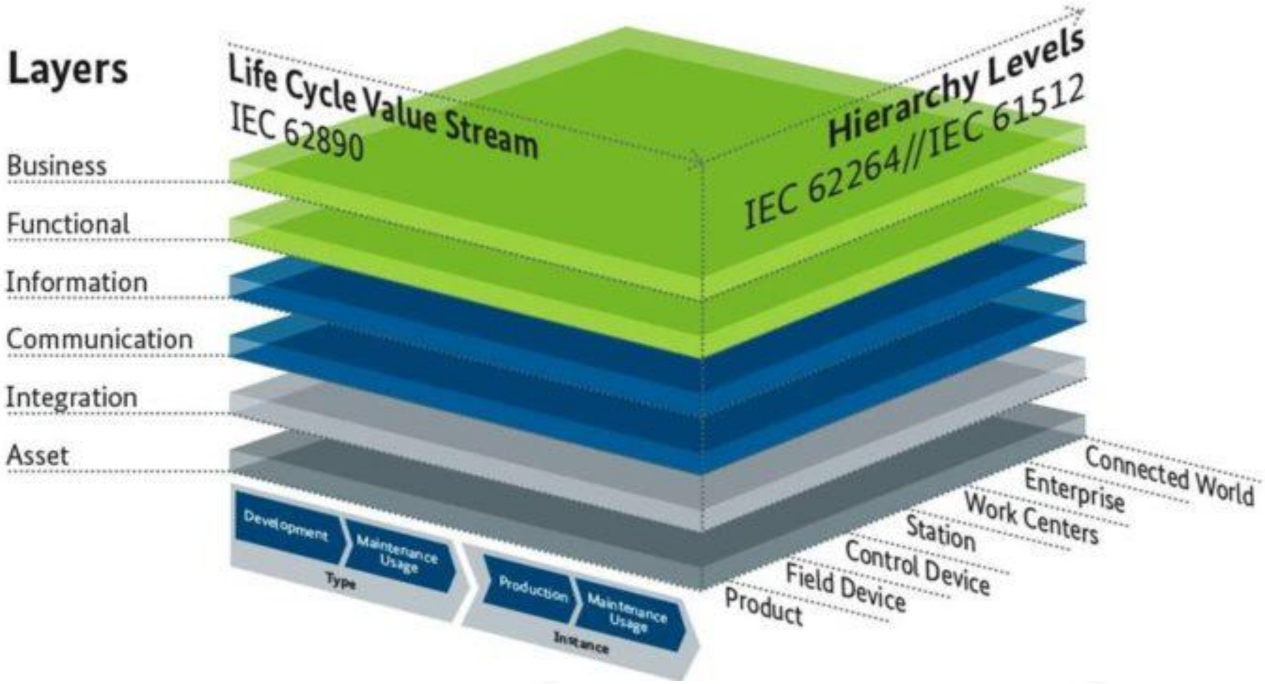
- Factories and machines increasingly complex
  - Human-centricity and robot collaboration are crucial
- Mass customization
  - Customer requires customized services and products
- **Business Processes** are not considered isolated
  - Multiple stakeholders involved
- **Smart Supply Chains**
  - **Agility** as responsiveness and adaptivity
    - **Responsiveness**: quickly react to disruptions along the supply chain
    - **Adaptivity**: ability to adapt to changes in the market while remaining competitive → **business opportunity**
  - **Dynamic situations** must be managed during the product lifecycle
    - Natural disasters, climate change, sociopolitical shifts, economic crises, wars

Digital Twins

- Different categories of **physical actors** in digital factories processes
  - humans (i.e., final users or participants in the production process)
  - industrial machines
- These physical entities must have a faithful representation in the digital world, usually referred to as **digital twins (DTs)**
  - **provide software interface (APIs) to receive instructions and to query data**
  - **Connectivity, autonomy, homogeneity, customizability, traceability**
  - **Fundamental for MRO – Manteinance, Repair, Operation**
  - **Synchronizing the physical world with the digital world**

Introducing RAMI

- RAMI - Reference Architectural Model for Industry 4.0
  - German initiative adopted at EU level
  - a 3-dimensional model
  - life-cycle/value-stream vs. organizational hierarchy vs logical layers



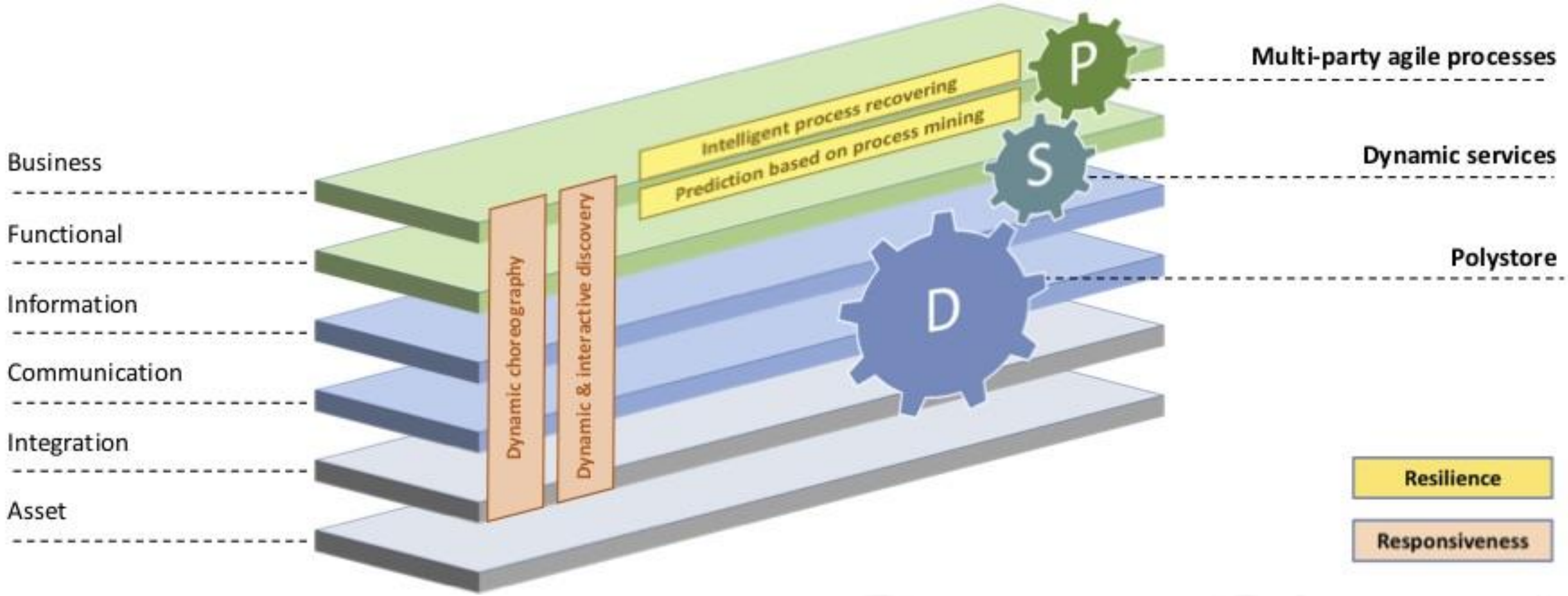
Zezulka, Marcon, Vesely, Sajdl. Industry 4.0 – An Introduction in the phenomenon. IFAC-PapersOnLine 2016.

Logical Layers in RAMI

- **Assets layer**: physical components such as robots, but also non-physical objects such as software and ideas.
- **Integration layer:** information for assets in a form that can be digitally processed
  - Corresponds to the DT. See later.
- **Communication layer**: standardization of communication using uniform data format and predefined protocols. E.g., MQTT.
- **An information layer**: is processing and integrating available data into useful information.
  - Corresponds to context extraction in ambient intelligence
- **Functional layer:** includes formal descriptions of functions. Also ERP functions belong to this layer.
- **Business layer:** includes mapping of the business model and links between different business processes.

A Possible Vision of RAMI

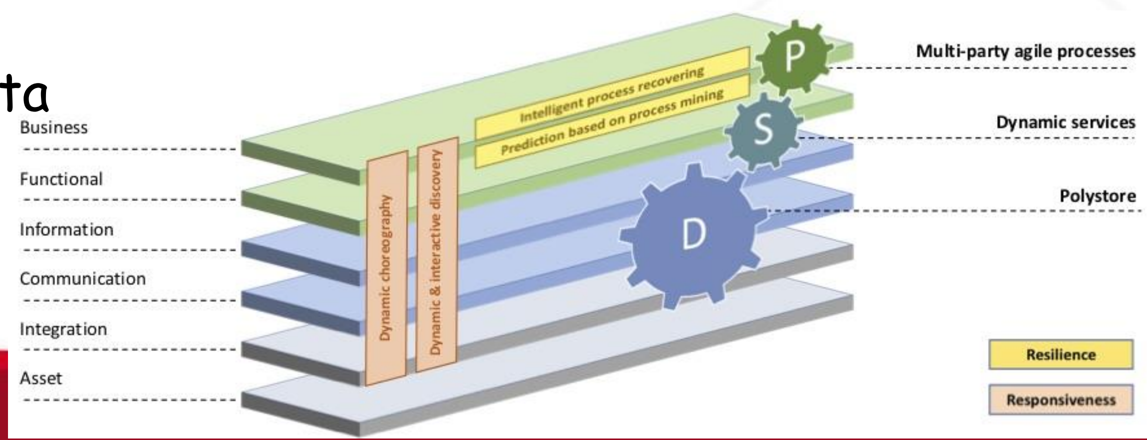
- RAMI defines logical functions of the different layers
  - How RAMI is implemented is up to the designer



Bicocchi, Cabri, Leotta, Mandreoli, Mecella, Sapio. An architectural approach for digital factories. 27th Italian Symposium on Advanced Database Systems, SEBD 2019.

Implementing RAMI – Data Space

- Physical entities must have a faithful representation in the digital world
  - usually referred to as **digital twins**
  - both for machines and humans (operators and supervisors)
- Machines are heterogeneous
  - different vendors, different standards and vocabulary
  - data are managed by different systems as well
- Polystore: a collection of heterogeneous data sources
  - Huge data volumes coming from IoT sensors, databases and datawarehouses
  - Streaming vs Static data
  - Declarative mappings between data stores
  - External data



The Data Space of a Digital Factory

- **Classical, native digital, information systems**
  - E.g., orders, inventory, ERPs
- Machines and information systems are complex and heterogeneous
  - different vendors, standards and vocabularies
  - data sometimes not available in the form of databases
    - query expressivity limited by exposed web services
- Interaction with DTs is not only synchronous:
  - huge data volumes coming from sensors (**streaming**)
  - alarms and warnings produced according to **manual or (machine) learnt rules**
- **DTs provide not only current data but also prediction about the future**
- **Other digital factories belonging to the supply chain expose their services to be queried and receive instructions**

Real World Example

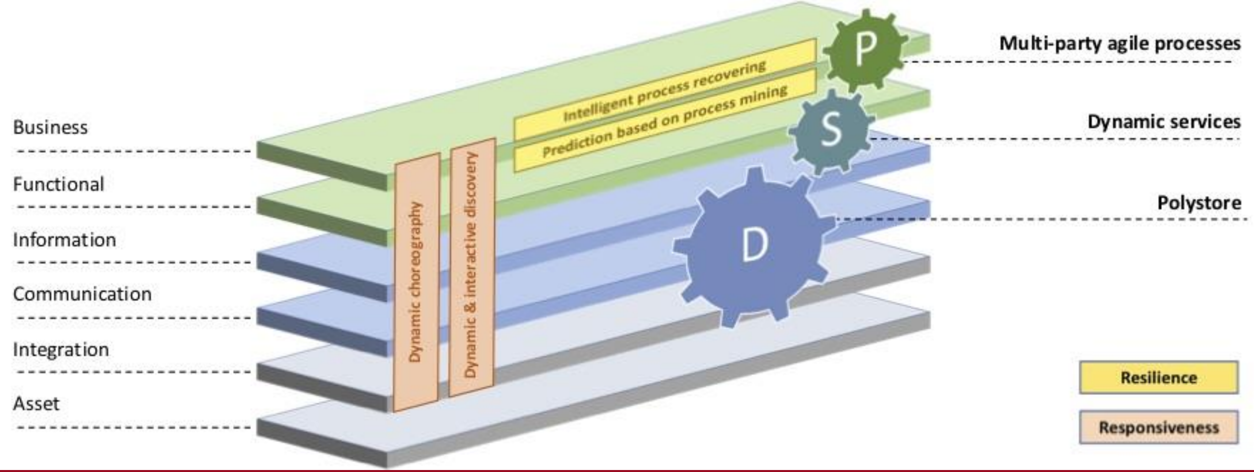
- A cardboard boxes real manufacturing scenario involving three companies
  - the cardboard manufacturer
  - the die cutter manufacturer
  - a set of delivery services
- We can consider three twins
  - the twin corresponding to the die cutter manufacturer
  - the twins corresponding to the delivery services
  - a twin corresponding to a single die cutter installed at the cardboard manufacturer factory

Real World Example (cont.)

- The polystore will contain
  - the data from the twins
  - the historical production from the die cutter manufacturer
  - the shipping history from the delivery services

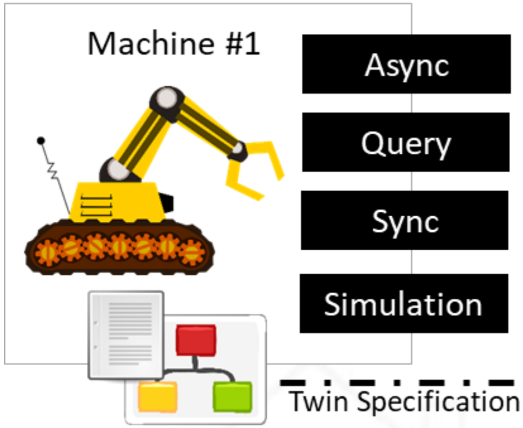
Implementing RAMI - Services

- Services can be provided at different levels of granularity
  - fine grained services (e.g., access to data)
  - coarse grained service (e.g., predictive monitoring, operations)
- Dynamic services
  - Get information and perform operations on the manufacturing parts of the system
  - Managed in a dynamic ways: control, discover and compose them
  - Rich semantic description to obtain specific goals



Services of a Digital Twin

- A DT exposes a set of service interfaces:
  - **Sync**: executing operations on physical twin
  - **Query**: allowing to query DT data
  - **Async**: producing asynchronous data describing DT activity and status
  - **Simulation**: performing predictive monitoring
- For each DT interface a formal specification is available



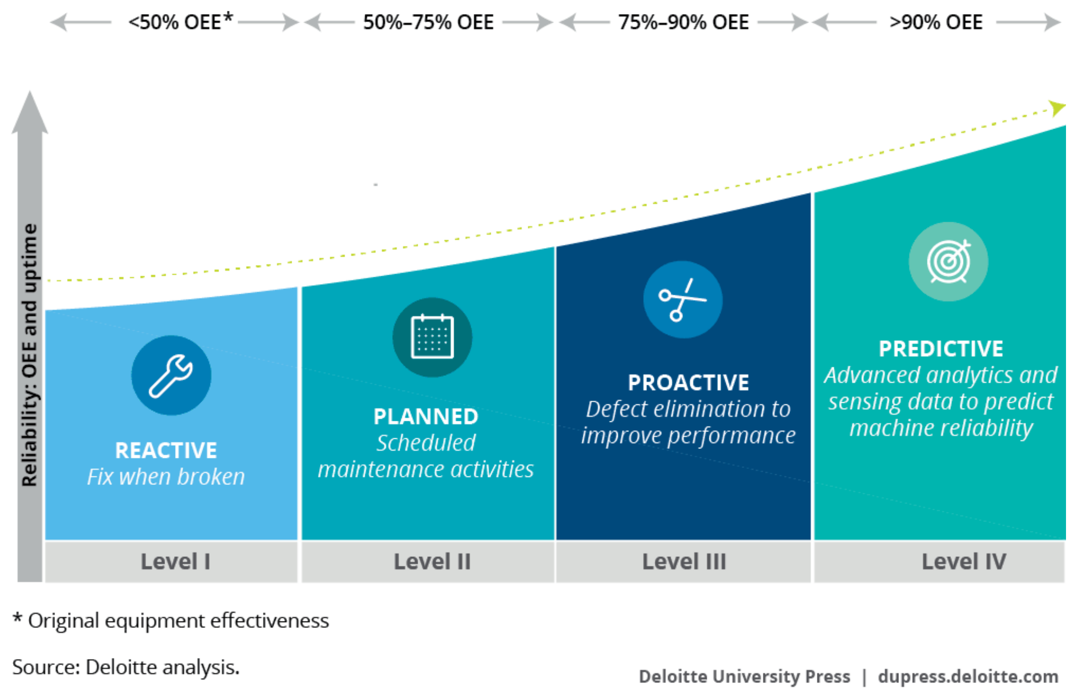
Real World Example (cont.)

- The DT of the die cutter contains at least two states: mounted and unmounted
  - At any time, the twin provides information about
    - the number of rotations performed
    - the residual life expectancy
  - It may expose a simulation API simulating what happens if the die cutter undergoes a given setting of the rotation speed throughout a 24h period
- The DT of the die cutter manufacturer may provide information about the time needed to produce a new die cutter
- The DT of a delivery service may provide information about the expected shipping time

Predictive Monitoring and Simulation

- Simulation used to predict how the physical twin of a DT can be expected to perform in the real world
  - crucial in a hypothetical DT composition task
- Concept of **Experimentable Digital Twins (EDTs)**
  - Virtual testbeds including several networked EDTs
  - Hybrid approaches with both DTs and EDTs

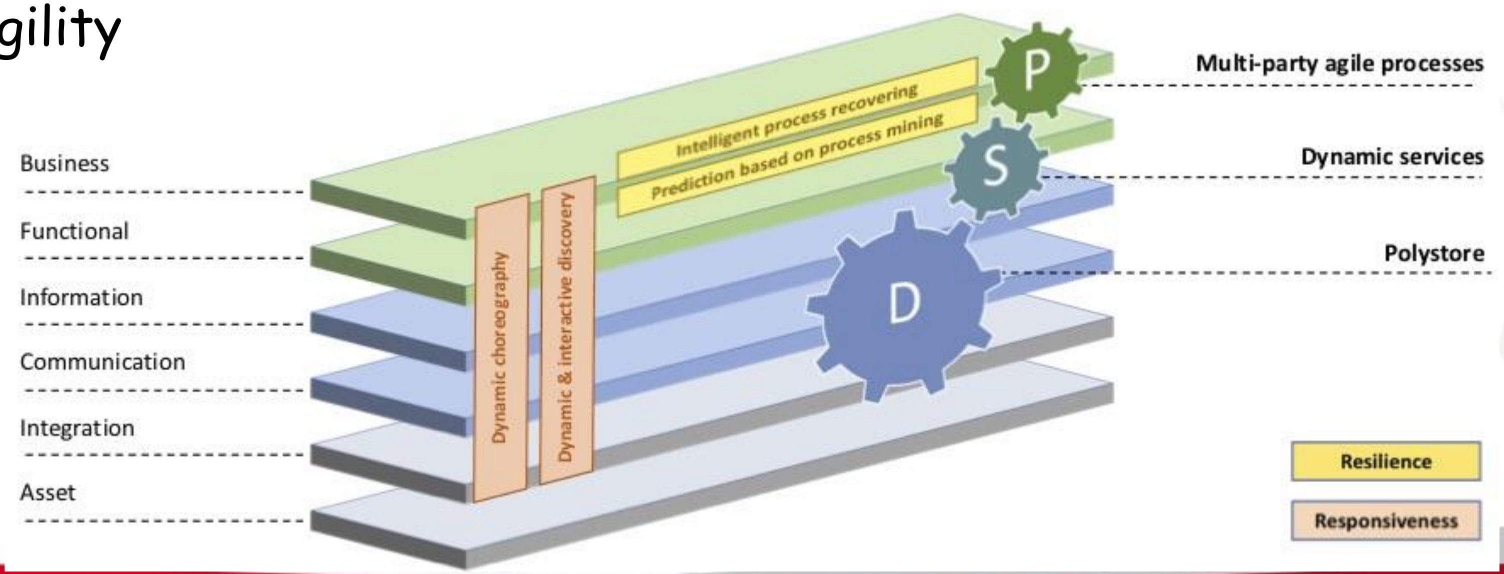
Predictive Maintenance



- Picture source https://www2.deloitte.com/us/en/insights/focus/industry-4-0/using-predictive-technologies-for-asset-maintenance.html
- Overall (Original) Equipment - measures the technical performance and capacity utilization of a manufacturing asset
  - allows to judge the effectiveness with which an asset is being used

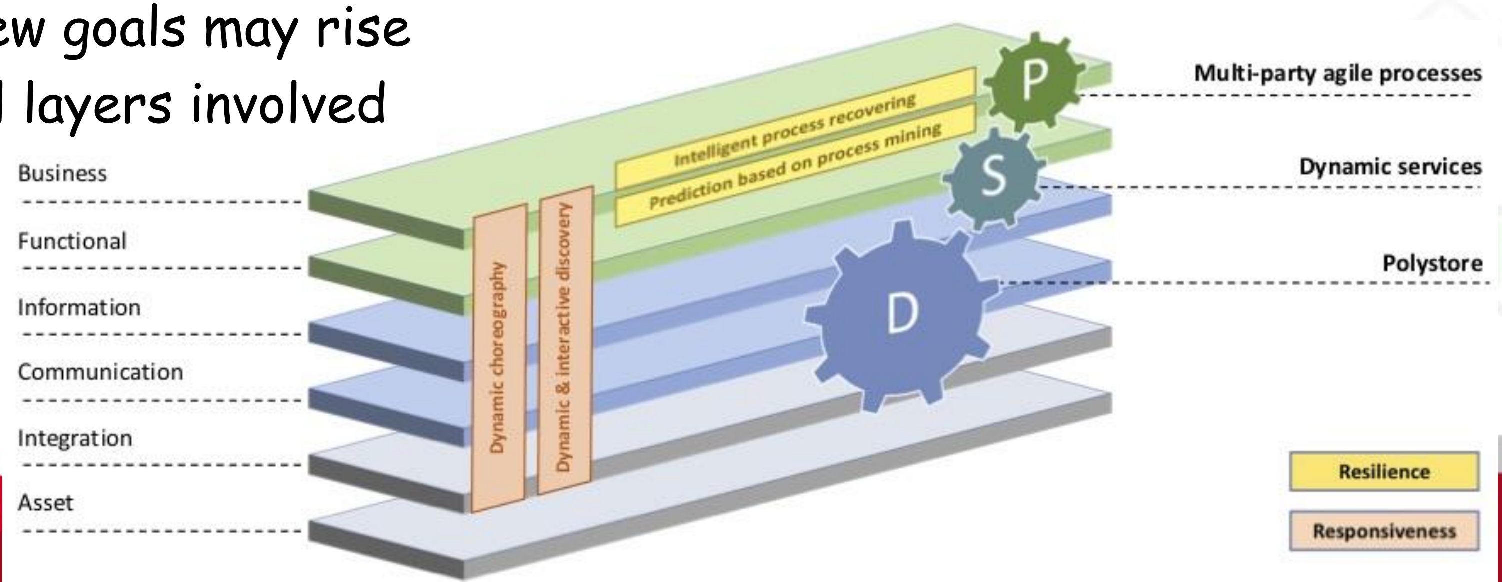
Implementing RAMI – Business Processes

- Multi-party agile business processes
  - Capture not only orchestrated (single-organization) processes but also choreographed (spanning different organization) processes
  - Shift from activity-centric processes to artifact-centric ones
    - Data and goods as first-class citizens
  - Focus on goals more than on the way to obtain it → obtaining agility



Obtaining Supply Chain Agility

- Responsiveness
  - Intelligent process recovering → even to partially fulfilled goals
  - Prediction based on process mining
  - Obtained on the top levels of RAMI
- Adaptivity
  - New possible partners provide new ways to obtain a goal being more efficient and competitive
  - New goals may rise
  - All layers involved



Real World Example (cont.)

- The production goal may be to avoid interruptions in the production process
  - for economic reasons, manufacturers prefer not to store in the warehouse replacement production machines and tools
- The supervisor may instruct the mediator to order a new die cutter
  - by predicting when the current one is going to break
  - taking into account production time on the die cutter manufacturing site
  - taking into account shipping time.
- The mediator will discover the different services and automatically understand how to combine them

Modeling Digital Twins

- Semantic approaches for representation of data streams
  - Reuse data analytics pipelines
  - Digital factory data space much more complex and heterogeneous
- Employing the popular AutomationML to model physical and logical components of machinery
- Using the Manufacturing Service Description Language (MSDL)
  - a descriptive ontology for representing capabilities of manufacturing services
- Models of data in different lifecycle phases
- **Model can describe the physical twin at any level of granularity**

Frameworks for Digital Twins

ditto

Bosch IoT Things

- All-around platform to manage DTs
- DTs are described with JSON
- Several services (DT search, selective notifications, fine-grained access control)
- Wide range of supported protocols
- Open source

- Cloud platform resulting from the integration Eclipse Ditto with other open-source projects
- User-friendly environment

Frameworks for Digital Twins

- Focus on modelling physical environments with spatial intelligence graphs
- Oriented to task automation
- Ideal in case there is a large number of sensors that must be organized hierarchically

- JSON document that stores a set of desired and reported states of a device
- When desired and reported states differ, the device is notified

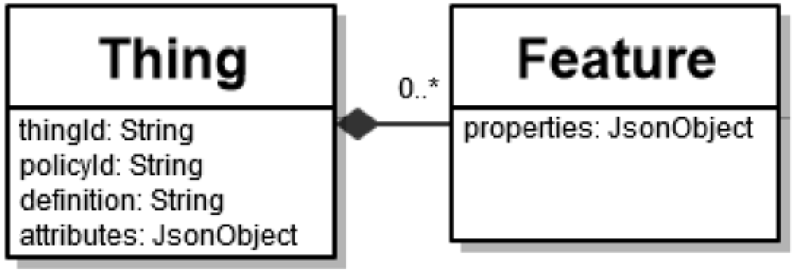
Frameworks for Digital Twins

- Other players:
  - General Electric Predix
    - Creation of DTs for analytics
  - IBM Watson IoT introduces Rational Rhapsody, Rational Engineering Lifecycle Manager and Rational Lifecycle Integration Adapters to deal with DTs
- No automatic DT composition, simulation and data integration
- **Interoperability rather than operation semantics**
- ***As an example we will show a short Ditto DT example***

Eclipse Ditto

- An open-source platform implementing the software pattern of DTs
- Main features:
  - API abstracting from the hardware
  - Routing requests between hardware and customer apps
  - Access control policies management
  - Persisting last reported state of hardware as cache
  - Change notification to interested parties

Representing Things in Ditto

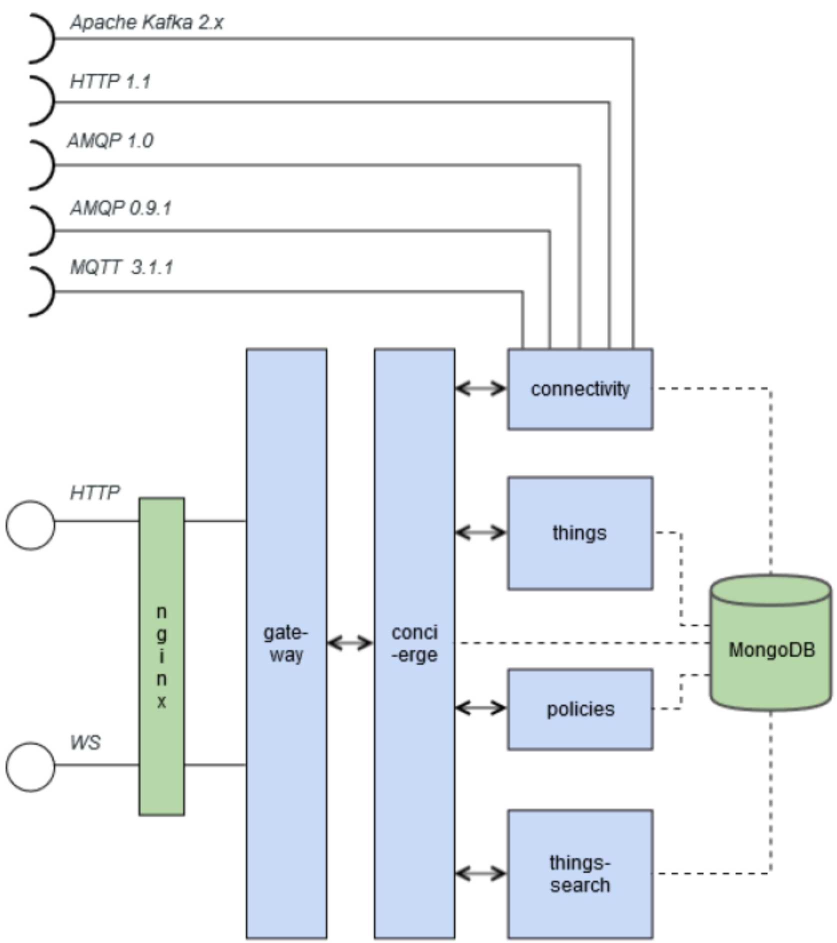



```
{
        "thingId": "my.namespace:myFridge",
        "policyId": "my.namespace:myFridgePolicy",
        "definition": "digitaltwin:DigitaltwinExample:1.0.0",
        "attributes": {
                "location": "Kitchen"
        },
        "features": {
                "temperature": {
                        "properties": {
                                "cur_temp": 5
                        }
                }
        }
}
```

Static properties

State of the thing

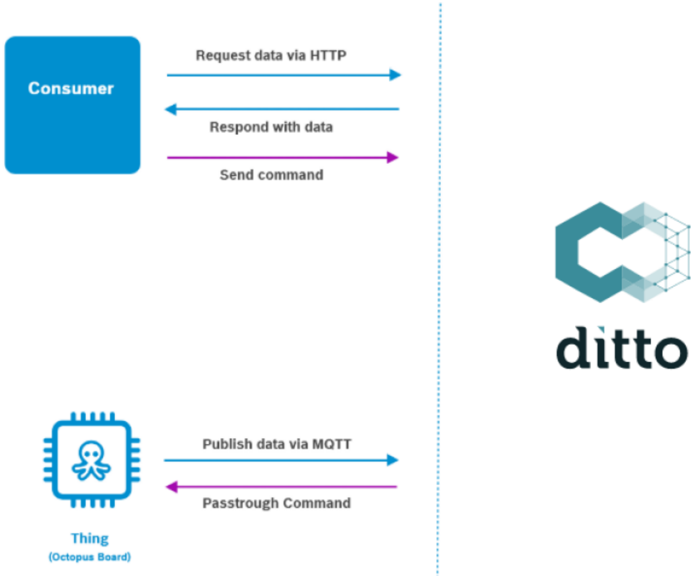
Architecture of Ditto



- **Policies**: fine-grained access control to the Things
- **Things**: persistence of Things
- **Things-Search**: tracking changes to Things, Features, Policies and updating an optimized search index and executing queries on this search index
- **Concierge**: orchestrates and authorizes the backing persistence services
- **Gateway**: provides HTTP and WebSocket API
- **Connectivity**: sends Ditto Protocol messages to external message brokers and receives messages from them

A Simple Ditto Example

- Example taken from https://github.com/eclipse/ditto-examples
- A board equipped with a temperature sensor and a LED
  - the temperature and the status of the LED status are periodically sent via MQTT
- A client able
  - to query the board via the HTTP REST API
  - to set the value of the LED remotely via HTTP REST API



A Simple Ditto Example

```
curl -X PUT 'http://localhost:8080/api/2/things/my.test:octopus'
  -u 'ditto:ditto'
  -H 'Content-Type: application/json' -d '{
    "policyId": "my.test:policy",
    "attributes": {
      "name": "octopus",
      "type": "octopus board"
    },
    "features": {
      "temp_sensor": {
        "properties": {
          "value": 0
        }
      },
      "LED": {
        "properties": {
          "value": 0
        }
      }
    }
  }'
```

Board Representation

A Simple Ditto Example

```
curl -X POST
  'http://localhost:8080/devops/piggyback/things/connectivity?timeout=1000'
  -u 'devops:foobar'
  -H 'Content-Type: application/json' -d '{
    "targetActorSelection": "/system/sharding/connection",
      "headers": {
        "aggregate": false
      },
      "piggybackCommand": {
        "type": "connectivity.commands:createConnection",
        "connection": {
          "id": "mqtt-example-connection-123",
          "connectionType": "mqtt",
          "connectionStatus": "open",
          "failoverEnabled": true,
          "uri": "tcp://test.mosquitto.org:1883",
          "sources": [{
            "addresses": ["ditto-tutorial/#"],
            "authorizationContext": ["nginx:ditto"],
            "qos": 0,
            "filters": []
          }],
          "targets": [{
            "address": "ditto-tutorial/{{ thing:id }}",
            "topics": [
              "_/_/things/twin/events",
              "_/_/things/live/messages"
            ],
            "authorizationContext": ["nginx:ditto"],
            "qos": 0
          }]
        }
      }'
```

Create internal connection operation

MQTT Server

Where to read data generated from things (#)

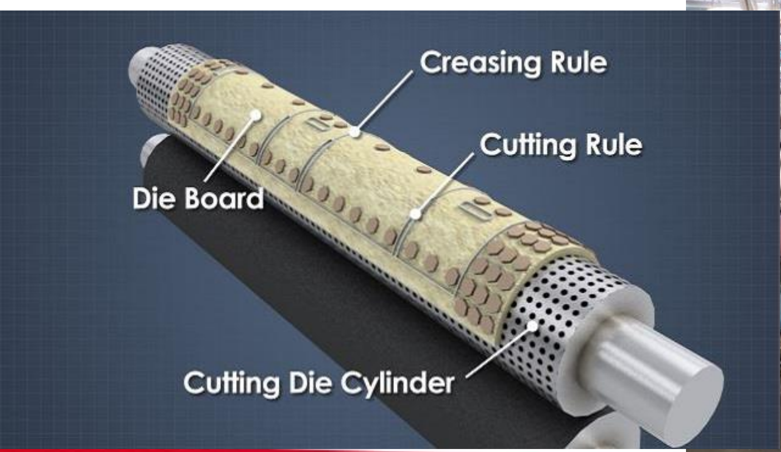
Where to publish data directed to the thing

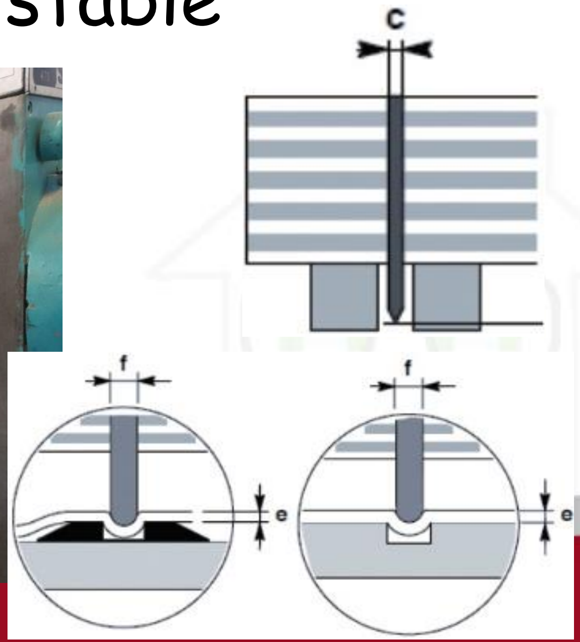
A Simple Ditto Example

- A thing publishes on MQTT
- Ditto read from MQTT
  - create an handler for the message if you want to store the state of the thing
- Clients read data from things from the Ditto API
- Send a command:
  - curl -X POST http://localhost:8080/api/2/things/my.test%3Aoctopus/features/LED%2Fproperties%2Fvalue/inbox/messages/LED%20switch?timeout=0
    -H "accept: */*" -H "Content-Type: application/json«
    -d "1" -u 'ditto:ditto'
- Receive thing data
  - curl -X GET http://localhost:8080/api/2/things/my.test%3Aoctopus
    -H "accept: application/json" -u 'ditto:ditto'
- Other details on https://github.com/eclipse/ditto-examples/tree/master/mqtt-bidirectional

A Predictive Maintenance Case Study

- A die cutter boards manufactures for cardboard cutting
  - Rotary or plane
  - Provided with blades for cutting and rubber pieces to keep the cardboard stable

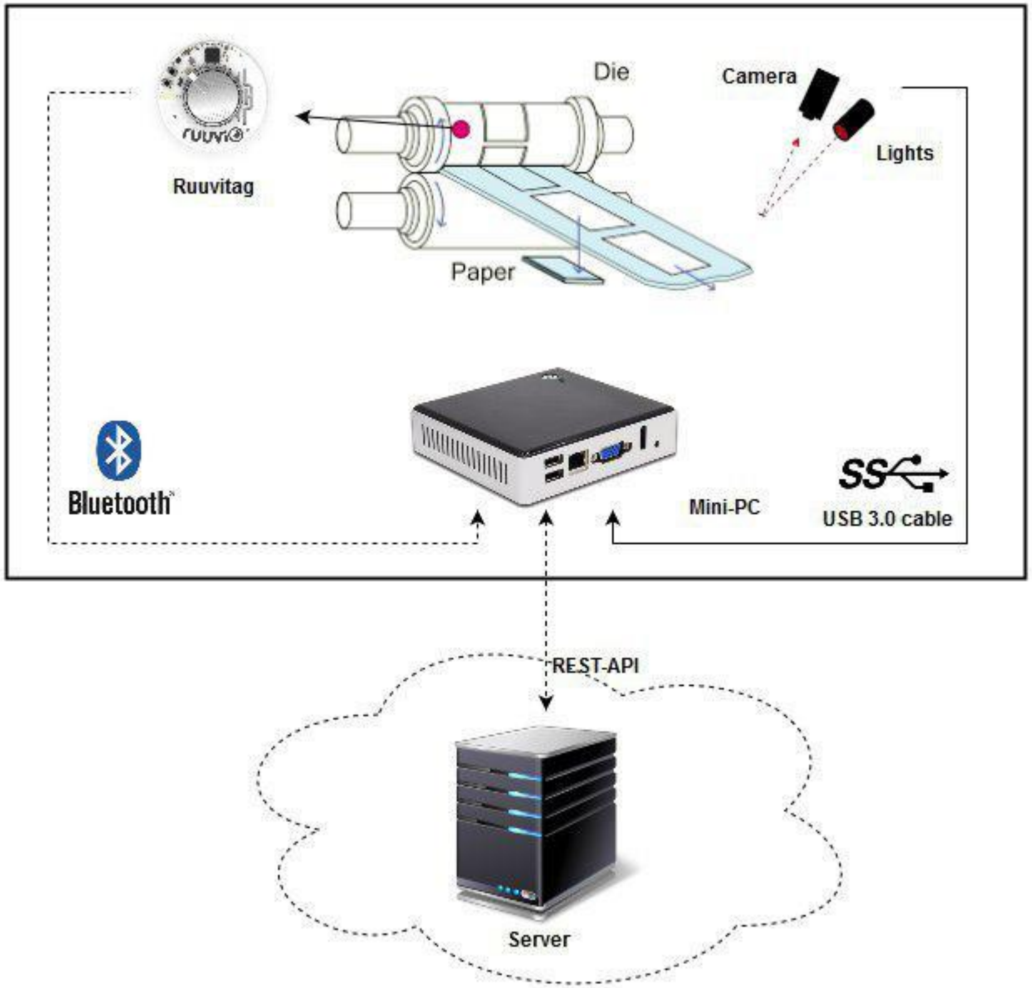




A Predictive Maintenance Case Study

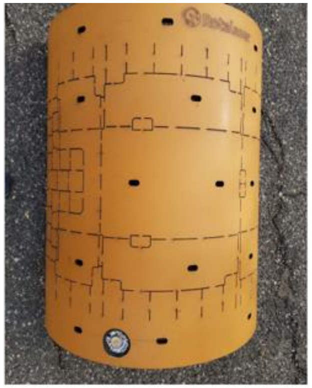
- Goals of the manufacturer:
  - Monitoring the consumption of dies
  - Understanding if the consumption is accelerated by some other factor
  - Reduce the issues coming from lack of rotary dies warehouse
    - A common problem since 2008 crisis
    - Production stop for customers ☹ → Need to reduce shipping times
    - Sudden requests from customer may disrupt die cutter manufacturer schedule → Need predict sudden requests from customers
- Avoid to intervene on the cutting machine (different manufacturer) to avoid responsibility

Proposed Architecture

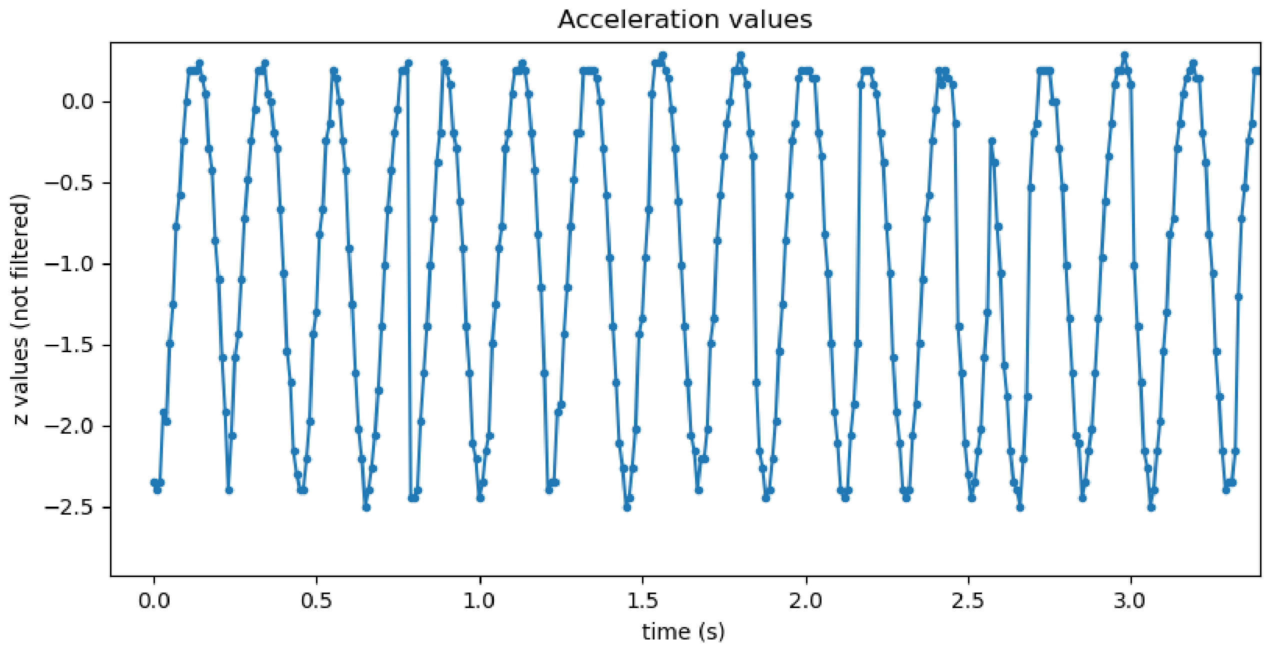


Rotation Monitoring Solution (1)

- Sensor RuuviTag installed on the die board
  - Only 9 mm depth → Possibility to instull directly on the die board
  - Unique device MAC Address → Unique ID associated to each die board
  - Accelerometer sensor
  - Environmental sensor (temperature, humidity)
  - Bluetooth Low Energy (BLE) for broadcasting information
  - Open-source, both software and hardware
- A custom firmware has been implemented to extend battery life, while increasing the data rate
  - Estimated 1 year life

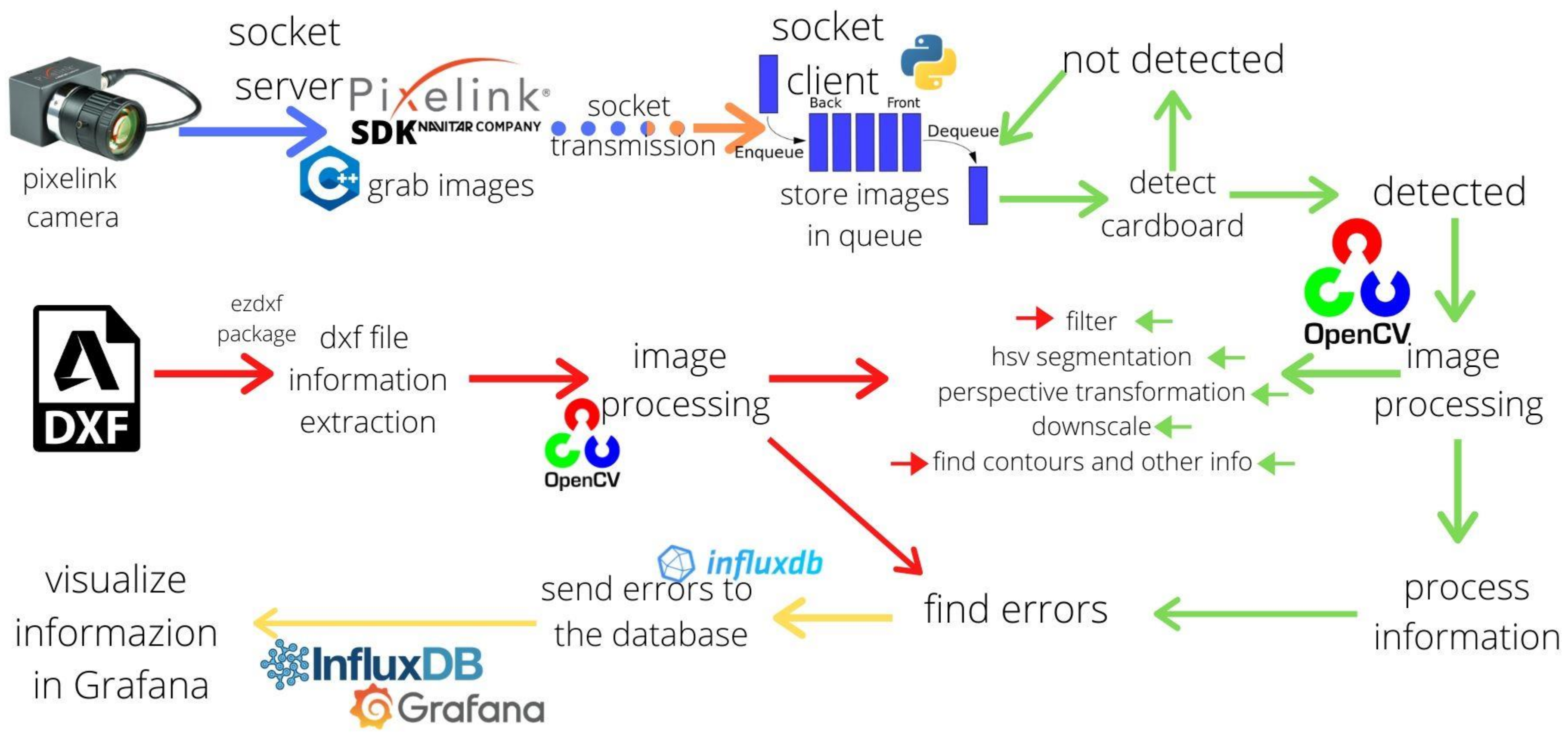


Rotation Monitoring Solution (2)

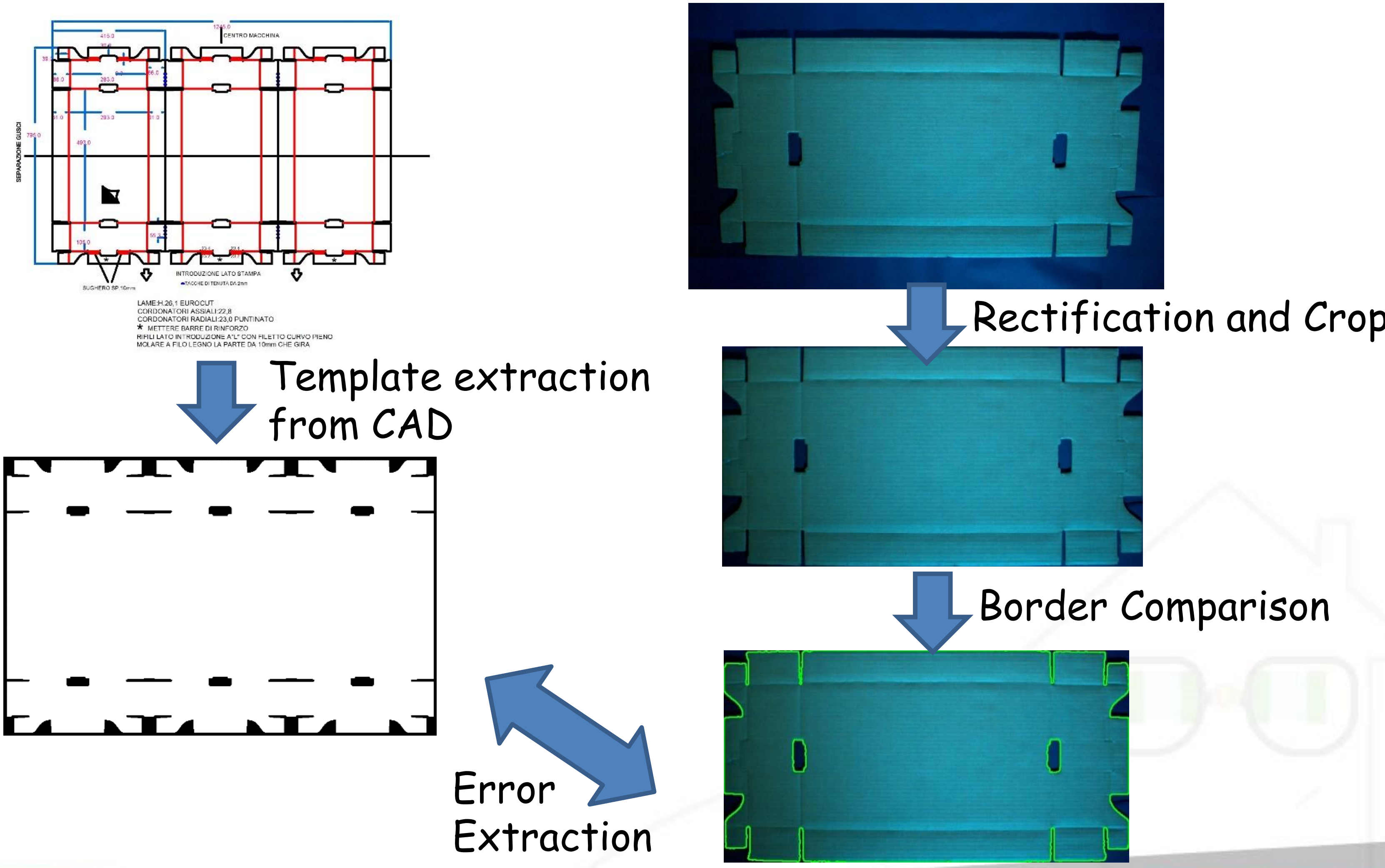


- A rotation in computed by using:
  - Acceleration on the Z axis
  - Couple of Zmin and Zmax peaks
  - A global average weighted on max(Zmax) and min(Zmin)
  - width = |(Zmax - Zmin)|

Cardboard Defect Detection (1)



Cardboard Defect Detection (2)



Computing Remaining Useful Life (RUL)

- Estimating how long a device can still function with acceptable performance
- Case study: ~20 thousand measurements collected by 24 sensors on 100 turbofan engines
  - https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/
  - Training set: data until fan end of life
  - Test set: data until a specific age + ground truth
- Neural network employed to estimate RUL

Narendhar Gugulothu, Malhotra, Vig, Agarwal, Shroff. (2017). Predicting remaining useful life using time series embeddings based on recurrent neural networks. In 2nd ACM SIGKDD Workshop on ML for PHM

Our Experiment

Task 1: regression problem

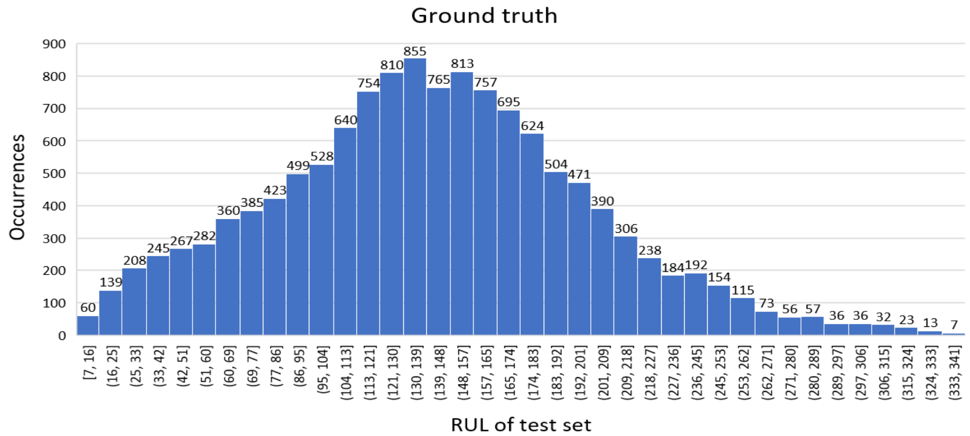
- "How many cycles an engine will last before it fails?"
- Result: Remaining Useful Life (RUL) prediction.

Task 2: binary classification

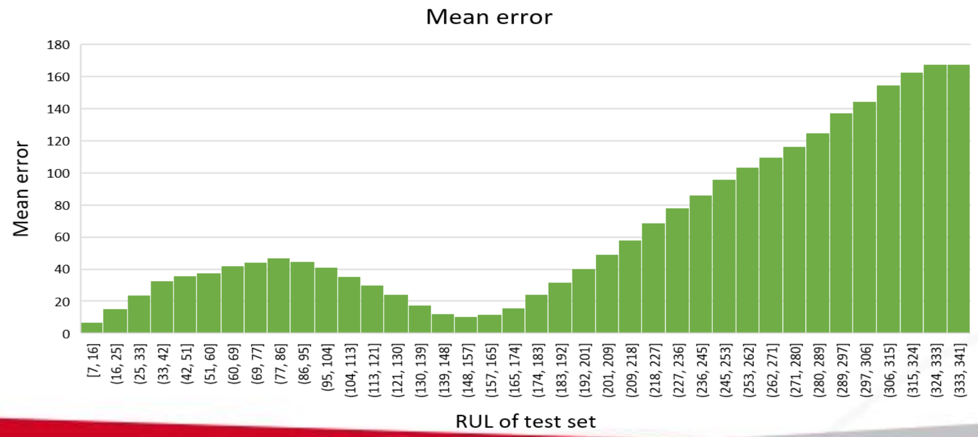
- "Is an engine going to fail within Lower Control Limit cycles?"
- Result: Negative (0) or Positive (1) class prediction.

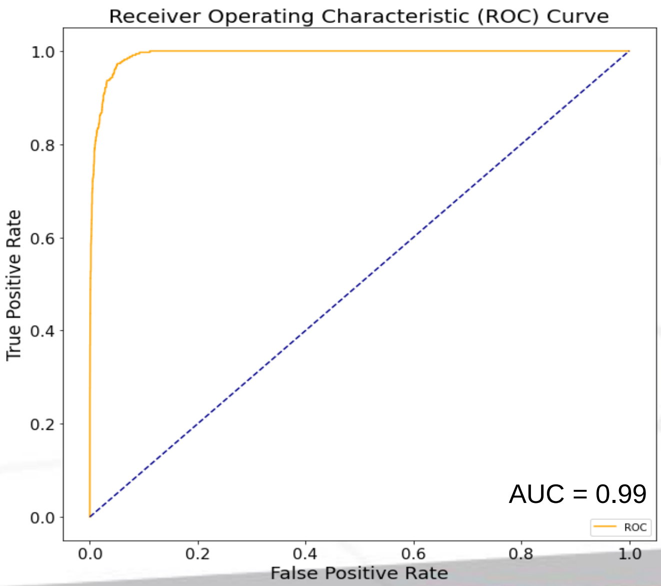
$W = 2$ time steps, $X = 24$ features

Configuration			TASK 1: regression		TASK 2: binary classification		
Set	**Shape**	**Labels**	**MAE**	**RMSE**	**Accuracy**	**Precision**	**Recall**
Training	(20531, 2, 24)	(20531, 1)	29.6075	41.53	0.9404	0.7312	0.9577
Test	(12996, 2, 24)	(12996, 1)	34.8322	45.71	0.9793	0.5645	0.8433



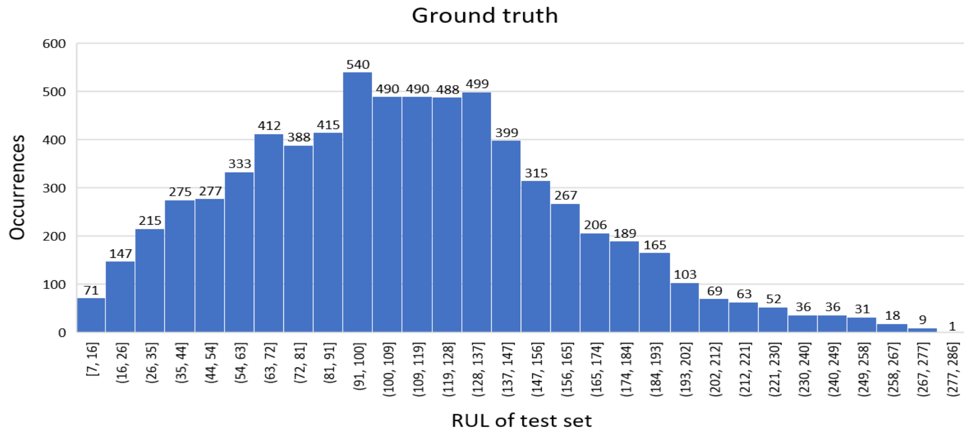
		Predicted values 0	Predicted values 1
True values	0	**TN** = 12448	**FP** = 216
	1	**FN** = 52	**TP** = 280

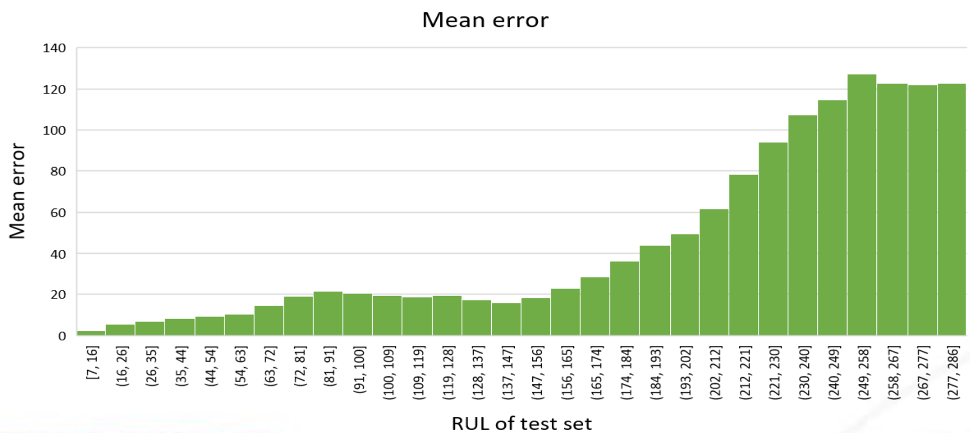




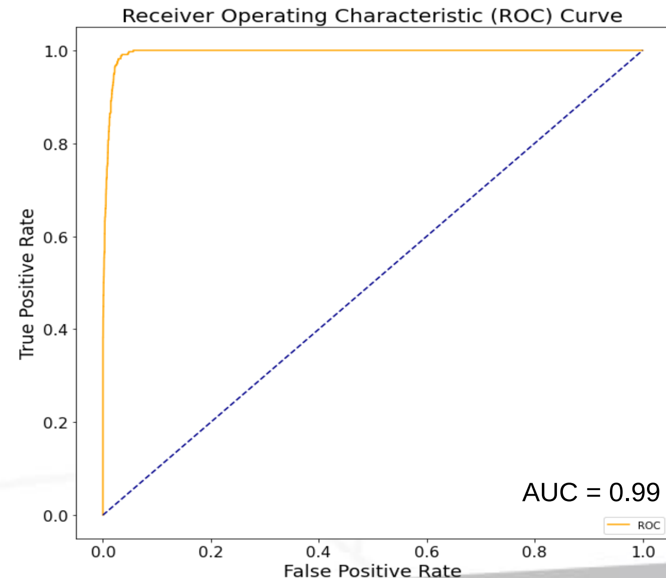
W = 64 time steps, X = 14 features

Configuration			TASK 1: regression		TASK 2: binary classification		
Set	**Shape**	**Labels**	**MAE**	**RMSE**	**Accuracy**	**Precision**	**Recall**
Training	(14331, 64, 14)	(14331, 1)	13.1626	21.29	0.9621	0.8660	0.9758
Test	(6999, 64, 14)	(6999, 1)	21.8087	31.17	0.9805	0.7512	0.8825





		Predicted values	
		0	1
True values	0	TN = 6570	FP = 97
	1	FN = 39	TP = 293

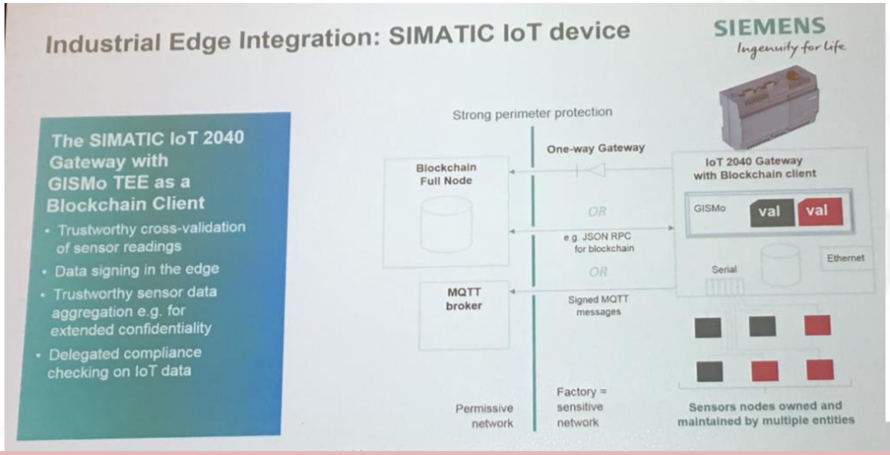


Digital Object Memory

- Smart manufacturing is intended to follow the productions along the entire product lifecycle
- A digital object memory is a digital space where all information about the product are stored
- Each product must have a unique ID → Electronic Product Code (EPC)
- Traceability
  - RFID
  - QR Code

Blockchain

- IoT devices generate data that need to be shared for demonstrating some properties with the guarantees of no modifications
  - In a distributed and multi-organization context, in which there is no trusted third party available
- Supply chain management
- A way to implement digital object memory
- Examples:
  - Modum.io
  - Siemens SIMATIC IoT 2040 Gateway



Saberi, Kouhizadeh, Sarkis, Shen. Blockchain technology and its relationships to sustainable supply chain management. International Journal of Production Research 2019

Final Project Ideas

- Use one of the DT frameworks to simulate the execution of an industrial process
- Implement a blockchain to support a multi-part industrial process
- Analyze a free industrial dataset for prediction and forecasting
- ....