IoTwins
BIG DATA OPTIMIZE INDUSTRY AND SERVICES

Grant Agreement N°857191

Distributed Digital Twins for industrial SMEs: a big-data platform

DELIVERABLE 3.1 – REQUIREMENTS FROM LARGE-SCALE INDUSTRIAL PRODUCTION AND FACILITY DIGITAL TWINS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement № 857191

Document Identification

Project	IoTwins
Project Full Title	Distributed Digital Twins for industrial SMEs: a big-data platform
Project Number	857191
Starting Date	September 1st, 2019
Duration	3 years
H2020 Programme	H2020-EU.2.1.1. - INDUSTRIAL LEADERSHIP - Leadership in enabling and industrial technologies - Information and Communication Technologies (ICT)
Topic	ICT-11-2018-2019 - HPC and Big Data enabled Large-scale Test-beds and Applications
Call for proposal	H2020-ICT-2018-3
Type of Action	IA-Innovation Action
Website	iotwins.eu
Work Package	WP3 - AI services for distributed digital twins
WP Leader	UNIBO
Responsible Partner	BRI
Contributing Partner(s)	UNIBO
Author(s)	Francesco Millo (BRI), Andrea Torcelli (BRI), Gaetano Ciaravella (BRI)
Contributor(s)	Michela Milano (UNIBO), Andrea Borghesi (UNIBO)
Reviewer(s)	Claudio Arlandini (CINECA), Florian Kintzler (SAGOE)
File Name	D 3.1 – REQUIREMENTS FROM LARGE-SCALE INDUSTRIAL PRODUCTION AND FACILITY DIGITAL TWINS
Contractual delivery date	M13 – 30 September 2020
Actual delivery date	M13 – 29 September 2020
Version	1
Status	Final
Type	R: Document, report
Dissemination level	PU: Public
Contact details of the coordinator	Francesco Millo, francesco.millo@bonfiglioli.com

Executive summary

This deliverable reports the results of the activity performed in the IoTwins project Work Package N.3 in the period M1-M6, focusing on task 3.1 and 3.2 -- whose outcome provides the core of this deliverable.

WP3 aims at the design and development of intelligent services, leveraging Artificial Intelligence (AI) techniques, to enable the creation of digital and hybrid twin models. Tasks 3.1 and 3.2 consist in the collection of requirements concerning AI-inspired services from the manufacturing and large scale facilities test-beds. At the beginning of the activity, a set of questions for the pilots' leaders were identified and sent to the partners. These questionnaires were then followed by interviews conducted by WP3 leaders and test-beds' leaders. The answers to the questionnaires and to the interviews were analyzed and used to create this deliverable.

The activity faced no blocking issue and was completed in due time.

Table of Contents

Executive summary 3
1 Introduction 5
2 Questionnaires for the test-beds 6
2.1 General questions 7
2.1.1 Machine Learning Models 7
2.1.2 Optimization Models 8
2.1.3 Simulation/Physics Models 9
2.1.4 Visualization 9
2.2 Test-bed specific questions 9
2.2.1 Test-bed 01: BRI-KKWS – Wind Turbine Predictive Maintenance 9
2.2.2 Test-bed 02: FILL-TTT – Machine Tool Spindle Predictive Behaviour 10
2.2.3 Test-bed 03: ETXE-BSC – Predictive Maintenance for a Crankshaft Manufacturing System .. 10
2.2.4 Test-bed 04: GCL – Predictive Maintenance and Production Optimization for Closure Manufacturing 10
2.2.5 Test-bed 05: NOUCAMPNOU – Sport Facility Management and Maintenance 10
2.2.6 Test-bed 06: EXAMON (CINECA) – Holistic Supercomputer Facility Management 10
2.2.7 Test-bed 07: SAG-SAGOE – Smart Grid Facility Management for Power Quality Monitoring 10
3 Analysis of the answers to the questionnaires 11
3.1 Questionnaire Responses & Interviews 11
3.1.1 TB01 (Wind Turbine Predictive Maintenance) 11
3.1.2 TB02 (Machine tool spindle predictive behaviour) 12
3.1.3 TB03 (Predictive maintenance for a crankshaft manufacturing system) 14
3.1.4 TB04 (Predictive maintenance and production optimization for closure manufacturing) 15
3.1.5 TB05 (Sport facility management and maintenance) & TB11 (Replicability towards smaller scale sport facilities) 16
3.1.6 TB06 (Holistic supercomputer facility management) & TB09 (Examon replication to INFN/BSC data centres) 17
3.1.7 TB07 (Smart Grid facility management for power quality monitoring) 17
3.1.8 TB08 (Patterns for smart manufacturing for SMEs) 20
3.1.9 TB10 (Standardization/homogenization of manufacturing performance) 21
3.1.10 TB12 (Innovative business models for IoTwins PaaS in manufacturing) 22
3.2 Discussion 23
3.2.1 Suggested Workflows 24
4 Conclusions 29

1 Introduction

This deliverable reports the results of the requirements collection activity. The requirements were collected from both large-scale test-beds for manufacturing (Task 3.1) and for facility management (Task 3.2). The goal of the activity was to analyze the type of available data needed to build the Machine Learning (ML) and physics-based models, depending on the pilot objectives. Another goal of this activity was to extract functional and non-functional requirements from the pilots and properly define the distributed twin features needed by each test-bed.

In order to collect these requirements a questionnaire had been prepared and given to each test-bed leader to be answered. After the questionnaires were answered an interview with each test-bed was conducted, with a twofold objective: I) clarifying points not entirely covered by the questionnaires and II) providing the occasion to treat in more detail the specific issues of the test-bed. These interviews were conducted in conjunction with leaders of WP2, who were collecting the technical requirements for the IoTwins platform (Task 2.1).

The rest of this document is devoted to the description of the activity. Section 2 describes the questionnaires given to each test-bed. Section 3 reports the responses of the test-bed to both questionnaires and interviews, and analyzes the answers.

2 Questionnaires for the test-beds

In the context of IoTwins project there are a series of test-beds (also referred to as "pilots") that will be used as practical case studies for the development of the digital twin platform. The test-beds belong to two main macro-areas: 1) *manufacturing* pilots (WP4) and 2) *facility management* pilots. It is also present a third, transversal category: *replicability* test-beds which aims at replicating in other systems the methodologies developed for the previous pilots (the replicability testbeds are centered around manufacturing and facility management as well). The complete list of the test-beds (TB) is the following:

- **TB N.1** Wind turbine predictive maintenance (*manufacturing*) – creation of a digital twin of a wind farm by aggregating simulation and ML models of single turbines;
- **TB N.2** Machine tool spindle predictive behaviour (manufacturing) -- development of multiple target-oriented digital twins of machine tools for the production of automotive components/parts;
- **TB N.3** Predictive maintenance for a crankshaft manufacturing system (manufacturing) -- digitalization and creation of a digital twin for a semi-autonomous system that produces crankshafts for the automotive industry;
- **TB N.4** Predictive maintenance and production optimization for closure manufacturing (*manufacturing*) -- establish an overall production management optimization, with specific goal in predictive maintenance;
- **TB N.5** Sport facility management and maintenance (*facility management*) -- management of facilities involving the flow of large crowds, both during normal operations and during maintenance and upgrade phases involving construction;
- **TB N.6** Holistic supercomputer facility management (*facility management*) -- development of data-driven prescriptive maintenance and optimization of large computing facilities, using the Examon framework;
- **TB N.7** Smart grid facility management (*facility management*) -- smart grid computation as close as to the data sources as possible in order to create a digital twin for prescriptive analytics;
- **TB N.8** Patterns for smart manufacturing for SMEs (*replicability*) -- definition of a general and replicable methodology for Small and Medium Enterprises (SMEs) based on the convergence of data analytics, AI, Internet of Things (IoT), and physics-based simulations;
- **TB N.9** Examon replication to INFN/BSC datacenters (*replicability*) -- replicate the Examon IoTwins-enabled management for large datacenters, definition of a methodology for the monitoring infrastructure reuse and deployment in new and different contexts;
- **TB N.10** Standardization/homogenization of manufacturing performance (*replicability*) -- extensions of the models investigated in TB N.4 to a wider series of machinery;
- **TB N.11** Replicability towards smaller scale sport facilities (*replicability*) -- demonstrate the replicability and scalability of the solutions developed in TB N.5;
- **TB N.12** Innovative business models for IoTwins PaaS in manufacturing (*replicability*) -- validation of innovative business models that brings resources available in the cloud accessible to applications running on manufacturing machines related to the machine monitoring business.

The questionnaires for the test-beds were divided into two parts: 1) test-bed specific questions and 2) general questions for all test-beds. The questionnaires did not distinguish between manufacturing and facility management.

2.1 General questions

The questions for all benchmarks were divided in four macro-areas:

1) questions related to ML models (e.g. anomaly detection/prediction) – **Q_ML.x**;
2) optimization models – **Q_OPT.x**;
3) simulation and physics models - **Q_SIMU.x**;
4) questions concerning visualization tools/mechanisms/techniques – **Q_VISU.x**.

2.1.1 Machine Learning Models

Q_ML.1: What is the format of the data collected from the sensors? Please provide details

- Is a common, standard format used? E.g. csv file, pandas data frame[1], spreadsheets, DB tables, etc.
- What is the data structure? How different components are handled (e.g. multiple files or tables)?

Q_ML.2: Is the data reliable? Namely, are there guarantees that the data has been gathered from healthy, well-configured, trustworthy sensors?

- If this is not the case, are the periods corresponding to unreliable data known? E.g. a wrongly calibrated sensor

Q_ML.3: Please provide as much information as possible regarding the system components

- Data sheet
- Measurement of sensors sensitivity
- Normal operative ranges
- Known issues
- Expected life time of components, mean time between failures, etc.

Q_ML.4: Regarding ML models for anomaly detection - Please, identify the classes of faulty behaviour to be automatically identified/detected/predicted

- Can a set of discrete different behaviours be identified? Or is it only possible to distinguish between "macro-behaviours", e.g. the system in normal state and the system in non-normal state
- What are the temporal dynamics of the faults? Does the problem happen suddenly or after a prolonged period -- for example, the first case could be a machine entering abruptly in an unwanted state due to misconfiguration, while the second case could correspond to a component that breaks after a slow degradation caused by usage w
- ear

Q_ML.5: Regarding ML models for anomaly detection - Are the data at our disposal labeled or not? Do we have exact information about the state of the system/component at any time?

- Identify one among the following 4 cases:
  1. the data can be clearly divided in normal samples (where the system/component was behaving in its expected manner) and anomalous ones (one or multiple classes for different anomalies)
  2. part of the data is labeled as normal samples, part is labeled with the corresponding anomalous state, another part is not labeled

[1] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html

3. labels are available only for the system in normal state (e.g. there are guarantees that at least a subset of the available data correspond to normal behaviour of the system)
4. no labels available at all

Q_ML.6: Please clearly highlight sensitive/confidential data

- Can confidential data be sent from the edge to the cloud for training of large ML models or global optimization strategies? Should the communication be encrypted?
- Can sensitive data be rendered anonymous on-edge to decrease the risk of spreading confidential data?

Q_ML.7: Please provide information about the feature of the on-edge and near sensors components

- Is there any computing power that can be used for training ML models? If not for training, can a trained ML model be deployed and used for real-time inference using live data?
- Details about computing units, storage capacity, memory (RAM), installed OS, installed SW, configuration info, etc.
- Are there additional limitations to the deployment of ML models on edge components? For instance, restricted access, impossibility to install new software (e.g. libraries for ML), OS-specific limitations, etc.

2.1.2 Optimization Models

Q_OPT.1: Please provide an accurate description and definition of the optimization problem

- Can the problem be modeled analytically (formulated through a set of equations/constraints)?
- What are the involved factors (variables)? Integer (discrete) or continuous variables? How many variables?
- What is the objective of the optimization problem? Cost minimization? Energy savings? Higher performance (e.g. quality of the final product)? Identify performance metrics and solution quality indicators
- If the goal is a combination of factors, can a weight be assigned to each factor?

Q_OPT.2: Please estimate real-time constraints on the time to find a solution for the optimization problem

- Is there an upper bound on the solution time? Hard-bound (the solution *has* to be provided within a certain limit) or soft-bound (the solution *should* be provided within a time limit, in order not to incur in penalties or additional costs)? At least provide the order of magnitude (a solution is needed within a few seconds, a few minutes, a few hours, etc.)
- Can the problem be decomposed on a “global”, general problem (that can be solved offline and thus is not subjected to stringent time constraints) and “local” problem to be solved on the edge (e.g. the solution for a specific instance of the problem, a refinement of the general solution, etc.)

Q_OPT.3: Are there baseline values that can be used for the comparison with optimized strategies?

- Current best practice, historical data

Q_OPT.4: What is the computational power of the computing resources available on the edge (if any)?

2.1.3 Simulation/Physics Models

Q_SIMU.1: What kind of simulation is needed?

- System simulation? Does it exist today? If so, which software is used today?
- FEM (Finite Elements Method] simulation? Does it exist today? If so, which software is used today?

Q_SIMU.2: What are the simulations to perform on the edge? What are the simulations to perform in the cloud?

Q_SIMU.3: What model data are stored? How to connect to the model data repository?

Q_SIMU.4: Data-driven anomaly detection: What kind of faults should be detected?

Q_SIMU.5: Is the simulation involved in control loops (real-time control)?

- Define the real-time constraints

Q_SIMU.6: For the construction of reduced order models, we need results from high fidelity simulations. Who will perform these simulations?

2.1.4 Visualization

Q_VISU.1: Is a data visualization tool needed? For which usage and which public?

Q_VISU.2: Is a data introspection tool needed?

Q_VISU.3: 3D FEM data visualization

- From an edge?
- From the cloud?
- Format of the data?
- Type of data: mesh, contours, curves?

Q_VISU.4: How will the visualization tools be provided?

Q_VISU.5: Dashboard creation

- Which tools [are / are planned to be] used to create dashboards?
- Should the platform provide component(s) dedicated to dashboard creation?
- Will it be possible to automate the “Dashboard tools” in order to create Tailored Menus/Options?

Q_VISU.6: Define the level of integration of the visualization tools

2.2 Test-bed specific questions

Each test-bed has its own set of specific questions, identified with **QP[test-bed_id][question_id]**. The test-beds for replicability (TB08-TB12) did not require specific questions for the requirements collection.

2.2.1 Test-bed 01: BRI-KKWS – Wind Turbine Predictive Maintenance

QP1.1: What is the current storage location of the data? On edge nodes? On the cloud or other centralized infrastructure?

QP1.2: What is the meaning of the field "Turbine data" in the table at page 12 of the project agreement document? Why is it available in such larger amounts w.r.t. the remaining fields? Does it refer to the whole wind farm? If this is the case, what do the other sensors refer to?

QP1.3: About the test-bed objectives, please define in more details the parts of the control system to be optimized and the desired outcome.

QP1.4: How can the optimization of the control system be measured?

2.2.2 Test-bed 02: FILL-TTT – Machine Tool Spindle Predictive Behaviour

QP2.1: About test-bed objectives: they present a problem with multiple objectives -- is it possible to quantify the relative importance of each aspect?

QP2.2: How can the optimization of the control system be measured?

2.2.3 Test-bed 03: ETXE-BSC – Predictive Maintenance for a Crankshaft Manufacturing System

QP3.1: About test-bed objectives, can the optimization strategies be defined in detail? E.g. reduce energy consumption, minimize components breakages, etc.

2.2.4 Test-bed 04: GCL – Predictive Maintenance and Production Optimization for Closure Manufacturing

QP4.1: Where is stored the current data? Cloud infrastructure? The proposed 300TB of collected data per year will requires significant amount of storage: is the data forecast to be stored long-term?

2.2.5 Test-bed 05: NOUCAMPNOU – Sport Facility Management and Maintenance

QP5.1: Is there any sensitive (or confidential) data that cannot leave the edge twins, for instance to be analyzed in the BSC data center? E.g. people's faces collected via cameras. If so, is there a plan to deal with this sensitive data?

2.2.6 Test-bed 06: EXAMON (CINECA) – Holistic Supercomputer Facility Management

No test-bed specific questions.

2.2.7 Test-bed 07: SAG-SAGOE – Smart Grid Facility Management for Power Quality Monitoring

QP7.1: Regarding the test-bed objectives, please define with more detail the "openness" to be demonstrated.

3 Analysis of the answers to the questionnaires

We begin this section by reporting the responses of each test-bed. To provide a compact overview of the requirement of each pilot, we will integrate the questionnaire answers with the details that emerged during the interviews. Consequently, Section 3.1 contains both the outcome of the questionnaires and the interviews, summing up all the pilots' requirements in terms of ML, optimization, physics/simulation models and visualization. Section 3.2 discusses common themes and provides an overview of the requirements.

3.1 Questionnaire Responses & Interviews

As the questionnaires contained many "general purpose" questions, not all of them applied to every pilot. Hence, the test-bed leaders did not answer all questions, but only to those that were applicable to their pilot. This was expected and it is not a cause of concern. Some of the test-beds did not directly answer the questions listed in Section 2, but they rather provided their responses in a more discursive form; conversely some pilots provided explicit answers to all the questions applicable to their test-bed. In the former case we preserve the structure of the answer given by the pilots, while in the second case we list the responses to all answered questions.

3.1.1 TB01 (Wind Turbine Predictive Maintenance)

Q: What kind of WP3-related tasks are needed for this TB?

Fault modeling & prediction?

- Fault modeling analysis will be performed at several levels. Training will help building a database. Pattern recognition will be performed on the edge using the information coming from the previous actions. Task 3.4 will provide tools to support this part.

Predictive Maintenance

- These results of the Fault modeling will drive us to predictive maintenance prescriptions. One of the main expectations will be to determine the remaining life duration of the systems & sub-systems.

Power generation optimization

- The power generation will be monitored and comparisons will be performed in respect of the nominal values. Warnings may therefore be sent. Moreover, action for improving the performances may be taken into account.

Q: What will be the goal of the ML models? On which level they should focus?

- Model Reduction Techniques (MOR) will be built using results of high-fidelity simulations. This reduced order model which reduces the computational cost without losing the accuracy will be used in control loop of the considered system at the edge level. To take into account the gaps between the model and the sensors data collected which we referred to it as uncertainty, ML techniques based on regression trees, decision trees, etc., will be applied.

Q: What kinds of simulation are expected at both edge and cloud level? For the whole turbine farm? For the single turbines? Single subsystems/components?

- Simulations will be performed for components/sub-systems for both system modeling and 3D finite element simulation. This will be deployed at the cloud level to train the models and the results will be used to control the system. The typical sequence of actions will be:
    - Simulation to build a model and determine the parameter to control the system. This will be done on historical data mainly using the cloud,
    - Transfer of the result of the model parameters to the edge to control the system,
    - During the exploitation, using the data collected we will continuously adjust the model and the parameters.

Q: Do you envision providing visualization techniques? For instance, "cloud-level visualization", which could be relevant in terms of required network bandwidth

- Several kinds of visualization will be used at the cloud level:
    - Dashboard that will show the data collected, showing the historical data raw, transformed or aggregated.
    - Data introspection (using for example an application called Inspector). This will require a large amount of data (historical). At the same time, this operation may require large network bandwidth; to be investigated together with WP2.

3.1.2 TB02 (Machine tool spindle predictive behaviour)

QP2.1 – The pilot leaders identify three cases:

1. The most important aspect would be to predict the behaviour of the machine spindle in the context of the machine. Are there spots or positions where the position of the axis in relation to the basis of the machine is leading to problems?
2. Another important aspect is to predict the wear and tear of the TOP 3 maintenance parts. E.g "in 2 weeks from now Part XYZ would break in 80% of the cases".
3. Cycle time improvement would be nice to have but with a lower priority.

QP2.2 – The optimization can be measured with the following actions:

1. Compare digital twin to real machine (force Sensor, vibration sensor);
2. Check if the wear and tear prediction forecast arrives;
3. Check if the optimized cycle time on the real machine is faster.

QP2.3 – The time scale of the control problem can be divided into three cases:

1. Adaptive short-term control or pre-calculated matrix - if a pre-calculated matrix is sufficient then maybe there are only set points to check;
2. Monitoring the data stream and analyze it from time to time (example: part is finished) - if it detects some anomalies – raise warning (or prediction);
3. Part-to-Part analysis.

QP2.4 – Not known at the moment.

QP2.5 - It is possible to load a docker container into the edge device through the available platform. In the docker container you can have all kind of analysis (python program, c# program, etc.).

QML.1 – InfluxDB[2] is used for the measurements; the data is structured in various tables (measurements) for different cases (always*: Timestamp;ValueA;ValueB;etc.).*

QML.2 – The data cannot be guaranteed to be always reliable; each sensor or measured value is related to a function of the machine. Depending on the function performed (e.g. part processing), only the corresponding sensor may be evaluated.

QML.4 – At the current state it is not clear which kind of anomalies can be detected. More research is necessary. However, there are likely different types of unknown granularity detectable. It is unclear whether one model can be well-suited to find all of them. Our data is non-stationary, hence there are definitely temporal dynamics which need to be considered. Data exhibits short as well as long term distribution changes due to e.g. temperature effects or tool wear.

QML.5 – For some anomalies there will be labels. E.g. tool wore down and broke. Other anomalies can only be assigned a pseudo-label. We expect to detect long term anomalies via unsupervised learning. The exact state can be assumed to be known ("digital twin"), but it is unclear which data-points are useful for the task at hand.

QML.6 – The communication from edge to cloud for training the models should be encrypted; sensitive data can be rendered anonymous.

QML.7 – There is no computing power at the edge for training ML models. Models should be trained in the cloud with support of GPUs; then deployed on edge for using live data – docker could be provided on the edge.

QML.8 – The data collection is fully autonomous; no input from customers.

QOPT.1 – Possible targets of optimization actions: I) no chattering occurring due to tool wear; II) dynamic limits not used fully due to safety factors in process. The problem is too complex to be model analytically, at the moment (State-of-the-Art, SoA, not detailed enough for multi spindle machine tools).

QOPT.2 – There is temporal hard-bound for active control feedback; this is a challenging to be developed if limits should be reached at any time (max performance at its edge required). A soft-bound can be considered for processing alarm.

QSIMU.1 – Simulations needed: Matlab/Simulink[3] for drivetrain and control behavior coupled with FEM /NX Nastran[4] - can be replaced by Ansys or other) for static and dynamic models. Matlab simulation models of simplified multi degree of freedom oscillator, calibrated by dynamic measurements.

QSIMU.2 – Simplified calibrated models to be code generated to C, to be run on edge in real-time. More complex simulation models run in the cloud, with parameters collected of the process to predict manufacturing quality, e.g. surface roughness.

QSIMU.3 – Model data stored as: .MDL Simulink files, .DAT FEM-data, text file on nodes and displacement, velocities, acceleration matrix, .txt / .csv measurements.

[2] see https://www.influxdata.com/
[3] https://www.mathworks.com/products/matlab.html
[4] https://www.plm.automation.siemens.com/global/en/products/simcenter/simcenter-nastran.html

QSIMU.5 – Simulations control loops from SoA: Siemens PLC cycle-time 200ms on status / boundary conditions, and NC cycle-time 6ms.

QSIMU.6 – The high-fidelity simulations will be performed by system engineers, reduced order models, state-space-models are derived from NX Siemens[5], SoA it is capable to cover till 4000 Hz Eigenfrequencies.

QVISU.1 – Different levels/targets of visualization: I) service technician, II) R&D engineer and researchers; public for demonstration, confidential for series machines.

QVISU.2 – Data introspection tools: Matlab , Flexpro[6], Catman easy[7].

QVISU.3 – 3DFEM visualization: Eigenmodes are interesting to be displayed at machine display or any other e.g. AR/VR for better analysis in the case of quality aspects; data format: for the static case .op2 txt file for editor, for the dynamic case .op2 txt file for editor animated; data types: .dat shell, hex, tetrahedral meshes, bushing, beams, Nastran finite element analysis.

QVISU.4 – The data visualization tools will be provided by internal department software engineering.

QVISU.5 – The current dashboard used for visualization is Graphana; it is not strictly necessary that the IoTwins platform will provide component for the dashboard creation.

QVISU.6 – Integration level of visualization tools: current Laptop / PC; hololens under investigation.

3.1.3 TB03 (Predictive maintenance for a crankshaft manufacturing system)

QML.1 – Data from the sensors is collected as multiple .csv files

QML.2 – The data come from highly trusted sources but no guarantees can be made at this stage; for some sensors there are measurers regarding the measurement error and related confidence interval, but not for all the collected metrics.

QML.3 – Failures are extremely rare; more interesting than detecting a failure is the detection of mechanical degradation.

QML.4 – Labels are not available (only extremely unbalanced ones – one anomaly every few years); this is an unsupervised problem.

QML.5 – Confidentiality does not affect TB03.

QML.6 – There are no guarantee of installing the needed SW modules on the edge and IoT levels; it is possible to install software tools on the lab used for experiments and prototypes; it is very hard to install additional software on the machines, as it should be certified in order to guarantee that it wouldn't interfere with the machine itself.

QVISU.1 – Not needed.

QVISU.2 - Not needed.

QVISU.3 - Not needed.

[5] https://www.plm.automation.siemens.com/global/en/products/nx/
[6] https://www.weisang.com/en/flexpro/at-a-glance/#
[7] https://www.hbm.com/en/2290/catman-data-acquisition-software/

QVISU.4 - Not needed; visualization tools already in place.

3.1.4 TB04 (Predictive maintenance and production optimization for closure manufacturing)

QP4.1 - The amount of 300 TB is related to 2 plants with all the machines connected (at least 200). For the test-bed 4 we will collect data just from one machine, therefore we forecast to store for this purpose 2-3 TB/year. We don't forecast to store them long-term but to keep just their aggregation.

QML.1 – Data collected from sensors will be saved into a database. Probably will be used a no-SQL DBMS like InfluxDB[8]. The expected data format will have a datetime + tag name + tag value structure; it will be saved into a unique file.

QML.2 - Data arrives from the production process, so they are reliable.

QML.3 – The collected metrics have been provided as an additional file; since it contains thousands of lines (the monitored sensors) it wasn't included in the deliverable.

QML.4 - We can just identify normal state and fault sate behaviour; problems happen after a prolonged period.

QML.5 – Data is labeled and there is information about components state anytime; the system will provide a series of process value and the information related to the status of the monitored component; the data can be clearly divided into normal samples (where the system/component was behaving in its expected manner) and anomalous ones (one or multiple classes for different anomalies).

QML.6 – Data is not confidential; the communication between edge and cloud should preferably be encrypted but it is not strictly necessary.

QML.7 - Data sets should contain information about the process status directly from the machine, no user interaction.

QOPT.1 – The objective is the predictive maintenance of one component of the machine.

QOPT.2 – After training of the predictive models, the inference (detection/prediction) can be performed on the edge.

QOPT.4 – On the edge there are virtual resources; computational power can be added if requested.

QVISU.1 – There is the need to visualize the most significant variables in the model, both for operators and maintainers.

QVISU.2 – Not needed.

QVISU.3 - Not needed.

QVISU.4 – Via a web interface.

QVISU.5 - Not needed.

[8] see https://www.influxdata.com/

QVISU.6 - The Data will be visualized inside a platform that manages the production cycle.

3.1.5 TB05 (Sport facility management and maintenance) & TB11 (Replicability towards smaller scale sport facilities)

The two test-beds are grouped as they both deal with the management and maintenance of sport facilities (at different scale).

QP5.1 – The plan is to analyze the sensitive data, as the people faces or profile, in the field, so, in the cameras, that might support GPU processing, so that raw sensitive data is never sent nor stored.

QP5.2 – The flow modeling should be done in batch, once the building plans/expected people profile/ etc. is set. Thousands of simulations are done and sent to the edge layer, which should be able to choose, in real time, which model to execute, depending on the IoT nodes real time output.

QML.1 - Structured data. The specific format (CSV, frame, etc) is still unknown. It could be a fixed-schema json as well.

QML.2 - Sensors are healthy but data extracted (such as distance between someone and the WIFI hotspot) has a high error variance and might be confirmed by several sensors (like triangulation methods) for getting a better trustfulness.

QML.3 – Unknown.

QML.4 - Supervised and not supervised methods. An anomaly could be high people concentrations, but anomaly detection is not the aim of this TBs.

QML.7 - The camera with GPU support should be provided by the project.

QML.8 – No.

QOPT.1 – Mix of continuous and integer variables; based on the people profile, the number of people, etc. the optimization model should choose the best agent-based model input to send to the Cloud layer.

QOPT.2 - Pandora[9], the Agent based modeling to perform simulations is not real time. The real time data might affect the model to choose, but the models are already created in batch.

QOPT.3 - Random for the Agent based modeling.

QSIMU.1 - Agent based modeling (ABM).

QSIMU.2 - On the cloud: Pandora (BSC software).

QSIMU.3 - Fog layer and cloud layer.

QSIMU.6 – BSC.

QVISU.1 – Visualization tool not really needed; maybe just an internal dashboard, still unknown.

QVISU.2 – No introspection.

[9] https://www.bsc.es/research-and-development/software-and-apps/software-list/pandora-hpc-agent-based-modelling-framework

3.1.6 TB06 (Holistic supercomputer facility management) & TB09 (Examon replication to INFN/BSC data centres)

TB06 and TB09 are grouped together as the former deals with the development of a monitoring infrastructure to enable the creation of a digital twin of a supercomputer -- Examon infrastructure, and the latter replicates Examon on different data centres.

QML.1 – Data is accessible via tag-keys queries, no-SQL data base table (CASSANDRA[10]) and its CQL language); client software is available to retrieve snapshots and time series.

QML.2 - The available data is reliable; some periods are missing in 2019 (due to the infrastructure calibration phase) and monitoring started on different nodes (partitions) at different time (the deployment has been incremental).

QML.3 - Information about the MARCONI system can be found at the following link[11] data collected are presented in the following table:

Plugin	Partition	Nodes (#)	Metrics (#)	Ts (s)	Db rate (metrics/s)
Confluent	Marconi SKL	3216	414735	60	6912.25
Confluent	Marconi KNL	3600	183526	20	9176.30
Nagios	Marconi KNL+Service	3717	101802	900	113.11
Nagios	Marconi SKL+Service	3318	103994	900	115.55
Ganglia	Service	29	9412	60	156.87
Ganglia	Marconi SKL	3215	105203	60	1753.38
Ganglia	Marconi KNL	3601	118833	60	1980.55
IPMI	Galileo	532	9576	20	478.80
Total		**7348**	**1047081**		**20686.81**

QML.4 - The data is partially labeled.

QML.6 - Names of the users that executed jobs on the HPC system and the names of their application should be obfuscated.

QML.7 - Edge nodes could be used for inference, not training.

QVISU.1 – A data visualization tool is needed and already in place, Graphana.

3.1.7 TB07 (Smart Grid facility management for power quality monitoring)

QP7.1 – The "openness" to be demonstrated is the increased independence of the containerized applications from the underlying (hardware) system, which enables the system to execute the edge applications unchanged in a simulated environment to test and predict their behaviour in the real environment. In addition, closed systems become open for customer applications while maintaining safety and secrecy requirements of the core control and monitoring functionality.

[10] http://cassandra.apache.org/
[11] https://wiki.u-gov.it/confluence/display/SCAIUS/UG3.1:+MARCONI+UserGuide

QML.1 – Sensor data is currently stored in CSV files. Harmonization and transfer to InfluxDB is ongoing. Data published by devices using IEC61850 protocol and transferred to cloud storage using OPC [12] Unified Architecture (UA) PubSub[13].

QML.2 - The trustworthiness of the sensor data was initially guaranteed through manual engineering and tests for each data point. But there is no active supervision or validation of incoming data streams. It will be part of the event detection and root cause analysis tasks (see tasks 3.3 and 5.3.3) to either detect the failure of sensors directly or consider the failure of a sensor as the root cause of another event. Especially with respect to topological data, there can be a discrepancy between the time of actual change and time of recording the change. Example: Operator closes or opens a switch at time t1 and records the change in the database at t2, whereby t2 might be even weeks or month after t1.

QML.3 – Information regarding the used edge devices can be found in the following web pages:

- SGW 1050: https://www.downloads.siemens.com/download-center/download?DLA03_2196
- A8000: https://support.industry.siemens.com/cs/attachments/109757713/HB_CP-802x_CP-8000_ENG.pdf
- X300: https://support.industry.siemens.com/cs/document/109477846/simatic-ipc227e?dti=0&lc=en-RS

Data like expected lifetime and mean time between failures are highly confidential.

QML.4 - The failure classes are still to be defined (this is part of tasks 5.3.1 and 5.3.2 / 3.3.) However, it is possible to distinguish between “normal operation” and “non-normal operation”. On the micro level, the periods, duration and abruptness changes from fault class to fault class. In case a breaker disconnects a part of the grid, the incident is sudden and might have consequences in large area. In case of voltage instabilities (fluctuations) or highly unbalanced phases the change might happen very slowly, and the consequences only manifest themselves after a prolonged period of time.

QML.5 - No labels available yet; the data will be analysed and labelled manually by a domain expert.

QML.6 - Communication should always be encrypted. Sensitive data might be transferred to the cloud for training, when the corresponding users agreed that their data can be used for this purpose. Sensitive data should preferably be rendered anonymous on-edge to decrease the risk of spreading confidential data.

QML.7 - The edge devices SGW 1050 and A8000 are enabled to run custom applications in parallel to their core functionality, which is configured and partly implemented for the concrete use case. The app framework on these devices handles the assignment of computation resources so that the correct functionality of the core functions is guaranteed. The computational power that can be used for the custom applications thus highly depends on the functionality that is implemented in the core. It is planned to develop a deployment chain in order to be able to test the computational power of these devices together with the core functionality that is implemented for the use case. On both the SGW 1050 and the A8000 a Linux based operating system is used. The application framework allows the deployment of containerized applications, which enables the user to use many programming languages and to include a wide range of libraries. The application framework ensures security of the core functionality of the device.

[12] https://opcfoundation.org/
[13] https://www.exorint.com/en/blog/the-introduction-of-opc-ua-pubsub-publish-subscribe-and-its-importance-to-manufacturers

QML.8 – No customer feedback/info is integrated into the data.

QOPT.1 – The objective of the implementation is to enable smart grid analytics on the edge and in the cloud. This enables the operator to optimize the performance of the system. However, there is no optimization problem solved by the system itself.

QSIMU.1 – Simulation needed: Electrical load flow simulation, weather simulation, Power usage simulation (typical household, etc.). The PSS® power system simulation and modeling software Siemens Sincal provides advanced algorithms with multiple modules i.e. Power Flow and Fault Analysis, Contingency Analysis/Probabilistic Reliability, Network Optimization, Transient Stability, EMT and Eigenvalues, Harmonics and Flicker evaluation, Protection simulation and coordination. For demonstration and tests with restricted functionality Pandapower[14] was used in the past. For demonstration purposes Siemens Bifrost can be used, while for co-simulation the Mosaik[15] framework might be used.

QSIMU.2 - Currently only the usage of trained models is envisioned for the edge level. Training or simulation of local processes is currently not planned. All data is planned to be transferred to the cloud and to be used for training and adaptation of algorithms on the loud level.

QSIMU.3 - To be investigated (see tasks 3.3 and 5.3.3).

QSIMU.4 – The kind of events to be detected are to be investigated during the project (see tasks 3.3, 5.3.2 and 5.3.3). Not only faults and errors are considered, but also events like “car charging pole changes below expected charging capacity”, which might be an unexpected behaviour, but not a fault. Possible faults include “power outage”, “power line is not conductive”, “voltage levels out of range”, “phases out of balance”, “load levels too high”, “unusual load fluctuations”, “network equipment not available”, “broken equipment like sensors or actuators”, “discrepancy between world status and world model” (topological data might be inconsistent, see above, or forecasts used for control algorithms might be incorrect).

QSIMU.5 - No simulation is directly involved in control loops. No real-time constraints.

QSIMU.6 - Simulations will be highly specific for the smart grid environment and are therefore envisioned to run in the Mindsphere[16] backend.

QVISU.1 – No User Interface (UI) needed from the project. UIs for displaying and monitoring time series data and to monitor the state of the system including a map-view to locate the part of the system in which an anomaly is detected or in which the root cause of an anomaly is located are typical components for the test-bed.

QVISU.2 - When AI algorithms are used on the edge level, there should be a way to evaluate the quality of its application. This might be a tool that could be beneficial for all test-beds.

QVISU.3 – Not needed.

QVISU.4 - Not needed.

[14] https://pandapower.readthedocs.io/en/v2.2.2/
[15] https://mosaik.offis.de/
[16] https://siemens.mindsphere.io/en

QVISU.5 - No UI components will be developed. In case they are needed pre-existing solutions of open source solutions (Grafana[17] etc.) are re-used.

3.1.8 TB08 (Patterns for smart manufacturing for SMEs)

QML.1 – Data collected and stored as .csv files. Data format: for the raw values measuring force, power, and analog signals; each line corresponds to an acquisition sample, each column for a separate channel. For raw values of accelerations, each line corresponds to an acquisition sample and each column is for a separate channel.

QML.2 - Sensors are calibrated, and compatible with an industrial use; data is reliable.

QML.4 - In machining, the problems can occur suddenly and also after a slow degradation due to the tool wear for example.

QML.5 - Some data will be labeled, but raw sensors signals not, of course. Continuous mark with threshold may be associated after analysis of experimental and numerical data (level of vibration, waviness defects, etc.). There is exact information about the state of the system/component at any time. Acquisitions are synchronized with the machining process (one series of acquisitions for each part produced). The IoT layer and edge layer computers are synchronized with the production machine-tool.

QML.6 – It must be taken into account the fact that the industrial end-users could not agree to send the data coming from their machine-tools making production of parts. Obfuscation of confidential data is a strategy that could be applied and should be considered (i.e. avoid to explicitly giving name & reference of the part, name of machine, name of the industrial company …).

QML.8 – Feedback from users is available in the form of structured data files for the geometrical measurements (for instance, coming from a coordinate-measuring machine).

QOPT.1 – There are three possible optimization targets:

1. Method preparation: decide the machine parts, optimize the process parameters that the machine user is defining (w.r.t. cost);
2. Configuration phase, decide the settings on the machining;
3. Production phase (parameters are fixed), parts are being produced, no changes allowed in production parameters, the optimization problem is to decide when to change the tool (to prevent the tool to become worn), optimize the substitution pattern (when to change a component).

Cutting of metal is not easy to solve, and not an exact science. There are some models and formulas, but not every process is analytically solvable. We will have to deal with a mix of equations, models, simulations and AI-ML models. The cost function will be linked mainly to geometrical criterion (acceptable shape, waviness, roughness) and possibly vibration amplitudes.

QOPT.2 – Real-time constraints depend on the corresponding operating phase. In the preparation of the work before machining, the answer of a computing problem should be provided in less than 1/2 hour (if full accuracy, and in 10 seconds if using light simplified models). In the production phase, the analysis of the data (local problem solved at edge level) coming from sensors should be done in less than 30 seconds.

QSIMU.2 - Simulation SW on the cloud (not installed on each user PCs); same thing for the ML models & data analysis (it happens on the cloud). Prediction can be done on the edge layer, but the heavy computations

[17] https://grafana.com/

should be done on the cloud. Production monitoring and data analysis can be performed on the edge layer. Reduced models simulations should be performed on the cloud.

QVISU.1 – Visualization tools can be useful for experts during the R&D phase of the project (for analyzing the data) before making models and ML solutions; for final users in order to visualize simply the key criteria used for monitoring the production process.

QVISU.2 – Introspection tools can be useful for experts during the R&D phase of the project (for analyzing the data) before making models and ML solutions.

QVISU.3 – 3DFEM visualization provided through ESI Player[18] (on the cloud).

QVISU.4 - Visualization tools located in the WEB based platform proposed to the final users.

QVISU.5 - Commercialized dashboard tools adapted to the test-bed specifications for the data visualizations or custom-made dashboard solutions if not successful (made with SCILAB[19]).

3.1.9 TB10 (Standardization/homogenization of manufacturing performance)

QML.1 - Multiple files: CSV file; ASCII files for logs (possibly already parsed to CSV).

QML.2 - Data are reliable.

QML.3 - Systems are those installed in the datacenter, mainly servers and the SW services running on it. Batch systems, parallel file systems, data transfer tools etc… Typically, each metric is collected every 5 minutes – it can be an average or a sum – it depends on the metric.

QML.4 - Normal versus non-normal state of the entire system can be identified; both sudden and gradual anomalies can happen.

QML.5 - In general, no labels available – but faulty situations can be identified through log error messages in the easiest cases – that however are the less common.

QML.6 - Names of the users that executed jobs on the HPC system and the names of their application should be obfuscated – this will be done at the IoT layer level.

QML.7 - Yes, we will use datacenter high-end server for the edge layers, so computing power is available from CPU only – GPUs will be available only on the cloud layer.

QOPT.1 - No optimization problem, predict faulty situations by correlating data from log and/or load/monitoring metrics.

QOPT.2 - Few minutes to identify/predict a faulty situation.

QSIMU.1 - No simulation foreseen for now.

QVISU.1 - For visualization the ELK dashboard[20] creator can be used; the public for the visualization dashboard is the data center owners and operators – not users.

[18] https://myesi.esi-group.com/downloads/software-downloads/esi-player-3.0
[19] https://www.scilab.org/
[20] https://www.elastic.co/guide/en/kibana/current/dashboard.html

QVISU.4 - We have Grafana and ELK installed on the EDGE and possibly on the CLOUD layers.

QVISU.5 - Grafana and ELK dashboard (KIBANA[21]).

3.1.10 TB12 (Innovative business models for IoTwins PaaS in manufacturing)

QML.1 - Data collected from PLC and Sensors are stored into a proprietary binary format = *.dpf files of max 5MB each containing process data recorded during cutting operations. A subsequent processing of these data is stored in standard SQL database (MS-SQL[22] or Posgresq[23]l).

QML.2 - The data from the machines are reliable.

QML.3 - The target system is composed of several Marposs-made Genior Modular[24] (GEM), consisting of hardware CPU-01, primary measurement transducer VM-01 with vibration sensors VA-1 and VA-3D, along with software systems GEM VISU and C-Thru. These modules are installed on several hundred machines and ancillary computer systems for collecting, storing and analyzing the data produced from the GEM systems. More specifically, the physical quantities under monitoring include the torque value of several axes, vibration velocity and acceleration, together with axis position, velocity, acceleration, jerk. The derived indicators include areas, ripples, deviations, duration and classical statistical derivatives. The GEM system controls the machining operation interacting with the CN to monitor the process and reduce failures and faulty operations.

QML.6 - No confidential data must be sent to the cloud: historical data used to train the ML models must be obfuscated. The current edge computational resources may not be sufficient to perform this anonymization process, so we may need to do it once data are stored on the cloud.

QML.7 - The training of the ML models can only happen on the cloud; only inference can be performed on the edge. The controlling system is based on a real-time embedded OS (QNX 6.4.0[25]) and cannot be extended with additional software.

QML.8 - Data contains only control measurements, no users' feedback.

QOPT.1 - The main focus of the test-bed is the detection/prediction of anomalies. Performance metrics can be, in this situation, replicability of the solution, accuracy of algorithm generation for different edges and platform, speed up the training of monitoring systems in their capability of recognizing and anticipating patterns of process degradation.

QOPT.2 - The time to provide solutions must be within few minutes: the digital twin of the process generated using AI techniques is used to test the efficiency of new/different controlling algorithms. Different algorithms, or different sets of configuration parameters, can be tested to find the best solution before installing it on the controlling edge. Again, the problem must be solved offline on the Cloud and only after a solution is found it is deployed to the edge.

QOPT.3 - As baseline, historical data is available.

[21] https://www.elastic.co/kibana
[22] https://www.microsoft.com/en-us/sql-server/sql-server-2019
[23] https://www.postgresql.org/
[24] https://www.marposs.com/eng/product/tool-and-process-monitoring-system-2
[25] http://www.qnx.com/download/feature.html?programid=18776

QOPT.4 - On the edge, the available hardware is the following. CPU: PPC MPC5121e[26], single core, little endian; RAM 512BM; *1 GB* Flash memory.

QSIMU.2 - Starting from Historical data a "digital twin" of the controlled process must be created. This digital twin is then used to test different controlling algorithms simulating their ability to recognize and anticipate patterns of process degradation, testing them on a simulation environment before deploying the best one to the real plant. The simulation is performed on the Cloud only that must be used by the final customer on a "pay-per-use" basis.

QSIMU.5 - No control loops.

QVISU.4 - Web interface

3.2 Discussion

After having reported all the answers provided by the pilot leaders, it could be useful to recap the common themes and shared issues.

First, one can observe that the majority of test-beds share the goal of having methods/tools to detect and forecast faults and/or anomalous behaviours of target systems and components; this is true for both manufacturing and facility pilots. For this purpose, ML models could be studied and implemented. Whilst anomaly prediction is a common desire, the definition of anomalies differs.

TB06, TB07 consider as anomalies a large class of events, encompassing components malfunctioning, components and/or sub-systems not properly configured, emergent behaviour diverging from the expected, optimal ones, etc. – probably since these are facility test-beds and their target system are composed by a collection of heterogeneous components. Other test-beds are mainly interested in predicting the faulty behaviour of particular components, critical in their production process (TB03, TB04, TB12). In some cases, anomalies are so rare – one every few years – that the pilots are more interested in detecting/predicting the degradation process that could lead to a fault, rather than the fault itself (TB03).

The data collected by the pilots also have a very diverse nature in terms of the available labels; the presence labels (e.g. in the form "at time T component X is in a normal/abnormal state") or their absence should guide the selection of different predictive models. For example, fully supervised models[27] can be used only when the labels are always available (TB04); most of the pilots have only partial labels (TB06, TB07, TB08). For some test-beds, we can distinguish between two classes of behaviour, normal and non-normal, and it is also possible to identify data sets composed only by normal data (TB03, TB06, TB07); this is a useful piece of information as it allows the usage of semi-supervised strategies[28]. In other test-beds, there is a combination of labelled data but also degradation detection models are required for the digital twins (TB01, TB02). In these cases, a preliminary analysis with unsupervised models should be likely conducted, to identify classes of behaviours and to better characterize the distinction between normal situation and anomalous one.

Another important requirement raised by some test-beds is the possibility to *inspect* the cause of the prediction made the ML model, for example in the case of root cause analysis (TB07) or identification of the

[26] https://www.nxp.com/docs/en/reference-manual/MPC5121ERM.pdf

[27] *A machine learning approach to online fault classification in HPC systems*, Netti A., Kiziltan Z, et al., Future Generation Computer Systems, 2019

[28] *A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems*, Borghesi A., Bartolini A., et al., Engineering Applications of Artificial Intelligence, 2019

culprit among a sub-system with many (thousands) of components (TB06). For these situations, pure sub-symbolic ML approaches such as neural networks could be insufficient, as it is not straightforward to "explain" their decision. For this scope, a variety of other ML models can be employed (decision trees, random forests) can be used. Furthermore, using a neural network is also possible, albeit without using straight *off-the-shelf* approaches but rather devising models capable of providing explanations[29].

A common theme among all test-bed is the (expected) limited computing power available on the edge. This has repercussions on every macro-area dealt within the WP3, predictive models, optimization models, and physics simulations, as it implies that the heavy lifting should be performed on the cloud and only simple operations can be done on the edge. For example, regarding the prediction models, the training of the models will have to be performed on the cloud, and then the models should be sent back and loaded on the edge, where they can be used to perform inference on the live data. All benchmarks provide a minimum of on-edge computational power, and thus edge inference should be doable for all pilots. Concerning the optimization models, the bulk of the computation has to be performed on the cloud, in an off-line fashion (exploiting also looser time requirements), while only simpler techniques can be used on the edge, where in general real-time requirements are more stringent. In this situation, the decomposition of the problem in off-line and on-line optimization is the most promising strategy.

The visualization tools are not mandatory for all test-beds but in general they would be considered a nice addition to have. In some cases, the test-beds do not require specific visualization solutions as there are already tools in place (for instance, in the case of TB03 and TB06). For other test-beds (TB01, TB02, TB04, TB07), there are already some visualization tools that are expected to be expanded with added functionality, such as introspection capability or web pages interfaces; in this case, the visualization would take place on the cloud and not directly on the edge. For this reason, the available bandwidth is a critical aspect to be considered for the development of the IoTwins platform, as video or image streaming is very costly in terms of network bandwidth.

3.2.1 Suggested Workflows

In general, we can observe that ML and physics/simulation models are the fundamental components required to create a digital twin. ML models can be used for *descriptive* and *predictive maintenance* (identify and predict faults) and for the forecast of specific target values, for instance to build power/thermal/electrical models. Simulation models serve the purpose of *emulating* the behaviour of the target system; they can be very complex if deployed on the cloud but tend to require simplifying assumptions when adopted on the edge, for example by using ML models to approximate the behaviour of sub-systems which are very hard to simulate (let alone be analytically computed). Afterwards, simulation and ML models can be combined to help finding the solution of optimization problems, a definition that encompasses multiple areas addressed by test-beds, ranging from *prescriptive* maintenance (e.g. schedule a service crew to substitute components and machine parts before they reach their breaking point) to improved power generation (TB01) or load management (TB06). However, providing precise guidelines applicable to every case is not simple, as each test-bed has its own particular optimization problem in mind. Visualization is a common requirement that is already partially addressed by many test-beds and whose critical issue is the network bandwidth needed in case of data transfer from cloud to edge (or vice-versa) in case of real-time visualization.

[29] *A Survey of Methods for Explaining Black Box Models*, Guidotti R., Monreale A, et al., in ACM computing surveys (CSUR), 2018

Based on the collected requirements (described in the previous sections), we can define a set of general workflows for the various activities to be performed to reach the desired goals. We identify 3 main workflows: 1) ML models, 2) simulation models, 3) optimization models. The following workflows will be the basis for the identification of the AI-based services to be provided to the pilots, which will be developed in T3.3, T3.4 and T3.5 (whose outcome will be reported by deliverables D3.2 and D3.3). The AI-based services will be general and applicable to the diverse test-beds; the services will then be used by the test-bed partners to implement their own, pilot-specific workflows. The IoTwins project also forecast an additional phase where the pilots' partners provide their feedback on the first version of AI-based services, with an additional round of collection of requirements and suggestions.

3.2.1.1 Workflow for ML Models

1. Identify the desired learning task
    a. Determine the type of the task: is it a regression or a classification task?
    b. Determine the type of the outcome: is the output used to detect a state change or to predict a state change? (Detection VS prediction)
    c. Identify the temporal dynamics involved in the phenomena to be learned/approximated.
    d. Is root cause identification required? Do the model's predictions need to be explainable?
2. Analyse the available data
    a. Time-series format? Uni-variate or multi-variate?
    b. Find the type of learning technique supported by the data:
        i. There is a label associated to each data point → supervised learning
        ii. Some data points are labeled, while others are not → semi-supervised learning
        iii. No labels are available → unsupervised learning
    c. How much data do we have?
3. Select a ML model by matching the identified characteristics of the task and the available data to the properties of the available algorithms:
    a. Unsupervised learning
        i. Clustering techniques
        ii. Auto-encoder neural networks
    b. Semi-supervised techniques
        i. Novel-detection methods
        ii. Zero-shot learning methods
        iii. Auto-encoder neural networks
        iv. Nonlinear regression
        v. Code2vect
    c. Time-series: use ad-hoc models
        i. ARIMA (statistics technique, close to state-of-the art for uni-variate series)
        ii. Recurrent Neural Networks
        iii. LSTMs/GRUs neural network
        iv. Topological Data Analysis
        v. Dynamic model Decomposition
    d. Predictions need to be explained
        i. Decision trees, random forest and other ensemble methods
        ii. Use DL models such as neural networks with caution → predictions explanation is not native for these model (but can be obtained with ad-hoc strategies)
        iii. Regressions
    e. Very few data → active learning models

   f. Decide the loss function
      i. For instance: with regression problems Mean Average Error (MAE) or Root Mean Squared Error (RMSE), with classification problems binary or categorical cross-entropy
4. Prepare the data
   a. Create a training set, a validation set, and a test set
   b. Normalize the data
   c. Can the data be moved around between edge to cloud? If so does it need encryption? Encrypt if required (also obfuscate sensitive information)
5. Train and validate the ML models
   a. Select target hardware for the training – possibly on the cloud due to the computational requirements
   b. Train the model using the training data – the validation data is used during the training phase to avoid overfitting
   c. Evaluate the model performance on the test set; is the model accurate?
      i. Yes
      ii. No
         1. Analyze the unsatisfying results to understand the cause of the failure: overfitting (→ try dropout)? Model not powerful enough (→ change the model architecture)? Data not sufficient to properly train a model (→ collect more data?)
         2. Apply corrective changes by going back to bullet 5
   d. Explain the decision of the ML model
      i. Easy for decision forest and similar (look at features importance)
      ii. Less straightforward for DL models (look at activation function patterns, significance analysis, etc.)
6. Deploy the ML models
   a. Can the model be used for inference directly on the edge?
      i. Yes: deploy it there directly attached to the data source
      ii. No: use the cloud to make predictions and send back the results (if needed on the edge)
   b. Periodically monitor the model's performance
      i. The monitored component's behaviour can change → periodical re-training of the models might be needed
      ii. If the edge hardware is power enough, online training strategies can be adopted

3.2.1.2 Workflow for Optimization Models

1. Define the optimization task
2. Identify the type of problem and the objective function
   a. Integer variables
   b. Continuous variable
   c. Single objective VS multiple objective
   d. Formalize the constraints that need to be respected
      i. Are the bounds hard (they need to be respected at all cost) or soft?
   e. Identify uncertainty sources
3. Identify components/functions that could/should be approximated
   a. Exploit ML or simulation models and integrate them in the optimization model

   i. Do parts of the optimization problem require estimate/forecast? → use ML models and embed the ML model within the optimization problem (e.g. Empirical Model Learning)
  b. Integrate simulation results in the optimization model
4. Select the optimization model type
  a. Heuristic algorithms?
  b. Linear programming (constraints and objective needs to be expressed in linear form, variables are continuous)
  c. Constraint programming
  d. Integer programming (integer variables)
  e. Mixed integer linear programming
  f. Constraint programming
  g. Stochastic programming (handle with uncertainty and possible different scenarios)
  h. Composition of the aforementioned techniques
5. Implement the optimization model
  a. Start from a simplified experimental setting (do not start with on-production systems), e.g. if the optimization problems involved many components/parts, begin with a sub-system
  b. Are there real time constraints?
    i. Any-time solution required?
    ii. On the edge VS on the cloud
    iii. Development of optimization algorithms that can be decomposed in two phases, offline and online[30]
    iv. Problem decomposition (e.g. Benders technique[31])
    v. Hierarchical decomposition of the problem [32]
    vi. Employ problem relaxations and/or surrogate models and/or approximations [33]
    vii. Suitable solution space search techniques [34]
    viii. Heuristic algorithms[35]
  c. Can the problem be decomposed within an off-line and on-line scheme?
    i. Off-line solution can be searched for on the cloud
    ii. On-line optimization could start from a pre-computed solution and improve it (or adapt it) for the local edge-level context
  d. Write the optimization model in the chosen computing paradigm

[30] A. De Filippo, M. Lombardi, and M. Milano. Methods for off-line/on-line optimization under uncertainty. In *IJCAI*, 2018; A. De Filippo, M. Lombardi, and M. Milano. Off-line and on-line optimization under uncertainty: A case study on energy management. In *CPAIOR*, 2018.

[31] Maheo A, Kilby P, Van Hentenryck P. Benders decomposition for the design of a hub and shuttle public transit system. Transportation Science. 2019 Feb;53(1):77-88; Hooker JN. Logic-based Benders decomposition for large-scale optimization. InLarge Scale Optimization in Supply Chains and Smart Manufacturing 2019 (pp. 1-26). Springer, Cham.

[32] Chaieb M, Jemai J, Mellouli K. A hierarchical decomposition framework for modeling combinatorial optimization problems. Procedia Computer Science. 2015 Jan 1;60:478-87.

[33] Gokbayrak K, Cassandras CG. Generalized surrogate problem methodology for online stochastic discrete optimization. Journal of optimization theory and applications. 2002 Jul 1;114(1):97-132; Taghavi M, Huang K. A Lagrangian relaxation approach for stochastic network capacity expansion with budget constraints. Annals of Operations Research. 2020 Jan 1;284(2):605-21.

[34] Zivan R. Anytime local search for distributed constraint optimization. InAAAI 2008 Jul 13 (pp. 393-398); Luna R, Şucan IA, Moll M, Kavraki LE. Anytime solution optimization for sampling-based motion planning. In2013 IEEE international conference on robotics and automation 2013 May 6 (pp. 5068-5074). IEEE.

[35] Borghesi A, Bartolini A, Lombardi M, Milano M, Benini L. Scheduling-based power capping in high performance computing systems. Sustainable Computing: Informatics and Systems. 2018 Sep 1;19:1-3.

e. Evaluate the quality of the solution
    i. Are there standard references to be used as comparison?
    ii. Is the solution provided on time?
6. Scale up the model to larger systems
    a. Assess the scalability of the solution

3.2.1.3 Workflow for Simulation Models

1. Define the simulation task
2. Define the simulation model
    a. Simulation type (FEM, System, computational fluid dynamics, ..)
    b. input, outputs and concerned physics
    c. Identify the application phase (Design, validation, or production)
    d. Solutions or MOR solution usage?
        i. Is MOR (Model Order Reduction) needed?
        ii. Prediction
        iii. Optimization
        iv. Inverse analysis
        v. Uncertainty propagation
        vi. Control
    e. Couplings (multi-domain / multi-physics)
3. Define and collect data
    a. FEM
        i. Geometry and material,
        ii. Process parameters
    b. System simulation
        i. System components, connectivity
        ii. System parameters
        iii. Define Fault components
    c. MOR
        i. Define the parameters and quantities of interest
4. Define Simulation Process and Build the simulation model
    a. Loads,
    b. Boundary conditions
    c. Identify the validation data
    d. Define simulation experiment(s)
        i. If MOR required construct DOE
5. Run the calculations
    a. On Edge (system) or Cloud?
    b. Validate results against validation data
6. Build the Reduced Order Model (if necessary)
    a. Identify training set
    b. Build the ignorance model to reality to simulation
    c. Integrate ROM in system simulation model
7. Deploy ROM / System Simulation model
    a. On Edge? On Cloud?

4 Conclusions

This document describes the requirements regarding AI services, physics simulation, and visualization collected from the test-beds during a two-phase strategy:

- Questionnaires sent to the test bed leaders;
- An in-depth follow-up interview to cover remaining details.

The involved partners complied and provided their responses in a timely manner. The collected requirements have also been analyzed, identifying common needs, test-bed specific exigencies and potential critical aspects.