Array 14 (2022) 100176

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

Industrial digitalization in the industry 4.0 era: Classification, reuse and authoring of digital models on Digital Twin platforms

Valentina Zambrano [a,*], Johannes Mueller-Roemer [b], Michael Sandberg [c], Prasad Talasila [c], Davide Zanin [d], Peter Gorm Larsen [c], Elke Loeschner [e], Wolfgang Thronicke [e], Dario Pietraroia [f], Giuseppe Landolfi [d], Alessandro Fontana [d], Manuel Laspalas [a], Jibinraj Antony [g], Valerie Poser [g], Tamas Kiss [h], Simon Bergweiler [g], Sebastian Pena Serna [i], Salvador Izquierdo [a], Ismael Viejo [a], Asier Juan [a], Francisco Serrano [a], André Stork [b]

[a] Instituto Tecnológico de Aragón – ITAINNOVA, C/ María de Luna 7-8, 50018 Zaragoza, Spain
[b] Fraunhofer Institute for Computer Graphics Research IGD, Fraunhoferstraße 5, 64283 Darmstadt, Germany
[c] Aarhus University, Nordre Ringgade 1, 8000 Aarhus, Denmark
[d] Scuola Universitaria Professionale della Svizzera Italiana – SUPSI, Via Pobiette 11, 6928 Manno, Switzerland
[e] Atos Information Technology GmbH, Otto-Hahn-Ring 6, 81739 Munich, Germany
[f] Technology Transfer System – TTS, Via Francesco d'Ovidio 3, 20131 Milano, Italy
[g] Deutsches Forschungszentrum Für Künstliche Intelligenz – DFKI GmbH, Trippstadter Str. 122, 67663 Kaiserslautern, Germany
[h] University of Westminster, Centre for Parallel Computing, New Cavendish Street, W1W 6UW London, United Kingdom
[i] Clesgo GmbH, Seyfferstraße 34, 70197 Stuttgart, Germany

ARTICLE INFO

Keywords:
Digital Twin
Cyber–Physical System
Industry 4.0
Reusable models

ABSTRACT

Digital Twins (DTs) are real-time digital models that allow for self-diagnosis, self-optimization and self-configuration without the need for human input or intervention. While DTs are a central aspect of the ongoing fourth industrial revolution (I4.0), this leap forward may be reserved for the established, large-cap companies since the adoption of digital technologies among Small and Medium-size Enterprises (SMEs) is still modest. The aim of the H2020 European Project "DIGITbrain" is to support a modular construction of DTs by reusing their fundamental building blocks, i.e., the *Models* that describe the behavior of the DT, their associated *Algorithms* and the *Data* required for the evaluation. By offering these building blocks as a service via a DT Platform (a Digital Twin Environment), the technical barriers among SMEs to adopt these technologies are lowered. This paper describes how digital models can be classified, reused and authored on such DT Platforms. Through experimental analyses of three industrial cases, the paper exemplifies how DTs are employed in relation to product assembly of agricultural robots, polymer injection molding, as well as laser-cutting and sheet-metal forming of aluminum.

1. Introduction

A successful manufacturing company must be agile, innovative, and highly efficient. Not only do companies today face fierce competition from globalization, but product requirements constantly increase due to new legislation, regulations, and customer expectations. Therefore, manufacturing companies already exploit digitalization techniques to be successful in the competitive market. This is the basis of the ongoing fourth industrial revolution (i.e., I4.0), which is already bringing a paradigm shift to manufacturing engineering.

I4.0 builds on the foundation of *Internet of Things* (IoT) technology, where sensors and software are embedded in devices (i.e., cyber–physical systems) to exploit different aspects of computerization. As such, a central aspect of I4.0 is the deployment of the so-called Digital Twins (DTs) [1] that ultimately act as real-time digital models and allow for self-diagnosis, self-optimization and self-configuration without the need for human input or intervention. There is, however, data that indicate that the leap forward offered by I4.0 might be reserved for the established, large-cap companies. For example, the Digital Transformation Scoreboard in EU signals that the adoption of digital technologies among SMEs and mid-caps [2] is below 10%, which

* Corresponding author.
E-mail addresses: vzambrano@itainnova.es (V. Zambrano), prasad.talasila@ece.au.dk (P. Talasila).

https://doi.org/10.1016/j.array.2022.100176
Received 25 April 2022; Accepted 25 April 2022
Available online 10 May 2022

poses a significant barrier if I4.0 technologies are to be adopted across the entire manufacturing sector.

DTs are becoming increasingly prominent in the manufacturing industry. DTs are capable of optimizing processes and products prior to their execution thanks to their ability of high-fidelity forecasting. In manufacturing industries, DTs can be virtual representations of different systems, e.g., machinery, production lines, products, or any other operation or service related to the manufacturing process. Therefore, DTs can be introduced at any time in the manufacturing process, hence their usefulness is not limited to pre-production planning and design, where prediction and optimization of the product or production lines apply, but they can also be used for maintenance, market analysis, etc. [3].

One way to allow SMEs and mid-caps access to I4.0 technologies is to offer the services via a platform, which is sometimes referred to as a *Digital Twin Environment* [4]. Not only does this lower the investment and infrastructure requirements of companies, but it also gives non-technical staff access to key technologies with only limited training. A Digital Twin Environment can utilize collected data to let models and algorithms remotely steer and optimize products and processes according to the operating conditions. As such, manufacturing companies can now outsource both expertise and parts of the supply chain [5] to service providers, which is referred to as Manufacturing as a Service (MaaS) [6–10].

To provide a MaaS platform, several building blocks are needed, many of which have already been explored, among others, in previously funded European H2020 projects. A non-exhaustive list of these EU projects includes: digital services marketplaces for manufacturing companies (CloudiFacturing [11], MANUSQUARE [12]); deployments of smart applications in open-source IoT platforms (FIWARE [12]); cloud orchestration engines (COLA [13,14]); as well as basic research into cyber–physical systems (INTO-CPS [15]) and simulation and forecasting technologies (MAYA [16]). The natural step from present state of the art is to develop a complete set of solutions that further extends the concept of DTs and enable manufacturing companies to tap into the full potential of I4.0. This is the goal of the European H2020 project DIGITbrain and this paper presents the project's novel approach how models for Digital Twins can be made reusable based on the DIGITbrain technology [17].

In order to provide a digital integrated platform, both software and hardware components need to be considered. However, one of the core technologies of the DIGITbrain project is the device-agnostic software-based verification mechanism by the implementation of a lightweight cryptographic library. This paper strictly focuses on Models, as a separate and reusable asset in DIGITbrain.

This paper is organized as follows: Section 2 provides background for the paper by reviewing related work; Section 3 identifies and reviews a classification of models that may be used on MaaS platforms for evaluation, including the behavior of models, organization, and interaction of coupled models, embedded models, and stateful and stateless models; Section 4 presents the proposed model characterization and metadata structure within DIGITbrain; Section 5 explains how models can be authored; Section 6 lists some of the most relevant results obtained during the first year of DIGITbrain Project leading to enhanced model reusability; and finally, Section 7 concludes the findings of the paper.

2. Background and related work

The use of DTs dates more than 50 years back to the Apollo 13 program, where NASA had designed physical simulators to mirror the conditions that astronauts would experience in the spacecraft during spaceflight [18]. While no formal, generic definition followed its initial introduction, the consensus now characterizes a DT as a digital model with near real-time, bidirectional communication to the physical system [19]. Accordingly, models with one-way communication are defined as a *digital shadow*, and models without any communication link are no more than digital models of the physical system. A DT can reflect the entire life cycle of a product [20,21], however, targeted domain-specific applications such as monitoring and prediction of crack growth are commonly seen as well [22].

In order to implement and use DTs in an I4.0 manufactory scenario, protocols, access profiles and communications must be set up between hardware and software nodes of the DT architecture [23–25]. Nevertheless, as described in Section 1, the Project DIGITbrain allows a clear separation between different assets, mainly Data, Algorithm and Microservices, and Models. In this scope, only Models are analyzed as an independent and reusable building blocks for DTs in I4.0.

The data to be utilized by the DT (i.e., the cyber–physical system) can be collected from various sensors and other devices, such as cameras, Radio Frequency Identification (RFID) tags, gauges (e.g., strain, deformation, temperature), etc. [26]. Naturally, to allow the system to perform any meaningful autonomous decision-making, data must be transmitted in real time or near real time. Recent reviews of data processing in a DT setting can be found in [27,28].

The role of models in a DT setting is to state, estimate and forecast. Models can be purely semantic in the sense that they are solely based on previously collected data. These models are also called database models and can be considered as black-box or gray-box approaches depending on the need for manual configuration. Machine Learning (ML) is the basis for semantic models, and several examples in literature can be found where data has been used to train Neural Networks (NN) [28,29].

Another approach to model system behavior is using physics-based models, also called first-principle models. Here, models are established based on geometrical information, material parameters, process conditions, and the underlying physics is characterized with partial differential equations. In a DT setting, for instance, the governing equations can take the basis in simple considerations of the conservation of energy in thermal system [30] or high-fidelity computational fluid dynamics models [31]. When solving the partial differential equations that govern the physics-based models, discretization methods such as Finite Element Method (FEM) or Finite Volume Method (FVM) must be employed. Therefore, the physical product or process is resolved into a large number of discrete variables, and this system can require significant computational effort to solve. As already discussed, the runtime of simulation models must be close to real-time in a DT setting, so physics-based models are often supplemented with semantic (i.e., ML) [32] or Reduced Order Modeling (ROM) techniques [33,34] to bring down computation time, although the latter technique can be also used to build completely data-driven DTs, as detailed in Section 3.

Based on some prior initiatives in Germany, a joint effort named *Plattform Industrie 4.0* [35] was launched in 2012 to promote the digital transformation of manufacturing, by bringing together companies, their workforces, trade unions, related associations, science, and politics. A comprehensive literature review of I4.0 and related technologies shows that the label *Industry 4.0* was later taken up internationally by many other approaches in the area of *smart manufacturing* [36]. The I4.0 approach was hence at first driven by the German initiative with broad support and high effort in its conceptual foundation, standardization [37] and practical application. The foundation of this approach was mainly based on the results of the working group on Reference Architectures and Standards and Norms, i.e., the *Reference Architectural Model Industrie 4.0* [38]. Over the years, the platform extended internationally by cooperation or alignment with similar initiatives in other countries in Europe, the US, Japan, and China. An important cornerstone was the alignment with the *Industrial Internet Reference Architecture* (IIRA) of the Industrial Internet Consortium (IIC) [39]. As the key concept of I4.0 platform, the Asset Administration Shell (AAS) has been introduced. In alignment with the IIC, the AAS is understood as a realization of DT in terms of I4.0 [40].

While all the previously mentioned research efforts enable the creation of DTs, their construction and creation remain a long and

time-consuming process. Each DT is typically constructed individually, without reusing previous results and existing building blocks. The work described in this paper is the first step to overcome this obstacle by categorizing and describing models with a rich set of metadata to enable their reusability when building DTs.

3. Types of models

The DIGITbrain Project includes and supports several kinds of models. A model in DIGITbrain is an asset that contains the knowledge related to a specific industrial product instance (i.e., a concrete manufacturing machine or production line), which can hence describe and forecast the behavior of such an instance when specific operating conditions are given (n.b., the process of forecasting a system's behavior according to specific operating conditions is also known as model evaluation). In this section, we detail the different model types.

3.1. Co-simulation models

Models related to Cyber–Physical Systems (CPSs) are often difficult to develop, due to the large variety of sub-systems (e.g., networks, control algorithms, mechanical components, electrical circuits, sensors) and components with different formalisms. Since sub-systems and components can be reused in different scenarios and applications, it becomes very helpful to be able to model them separately, instead of creating a single monolithic model that includes all of them. For this purpose, co-simulation is a very useful approach for coupling together the different parts of a whole system.

The sub-systems and components are created by model developers and exported as co-simulation units [41]. In order to perform co-simulation of a complete system, these co-simulation units need a standard way of interacting with each other. Since manufacturing systems are typically produced by a combination of heterogeneous components, these components are usually supplied by several legal entities; hence protection of the intellectual property for the underlying model is needed.

The Functional Mockup Interface (FMI) [42] is a widely used solution for the above described problem where the orchestration of combining differing simulation units is made by independent orchestration engines [43]. FMI is a cross-platform and open-source standard to exchange models and perform co-simulation. Components that conform to the FMI standard are called Functional Mockup Units (FMUs). An FMU is distributed as an archive with the file extension *.fmu*. An FMU contains the following:

- an XML-file with metadata, definitions of all the variables inside the FMU, and desired outputs
- implementation in source and/or binary form, specifically:
  - binaries: this directory contains the executable files of the FMU and can also contain shared library executable code for different OS (Operating System) platforms
  - resources (optional): the contents of this directory can be used by the FMU during execution time
  - sources (optional): the source code of the FMU, compiled to produce the shared libraries placed in the binaries directory
- additional Data, such as documentation

It is important to remark that the model itself inside the FMU may be distributed as binary and hence can be exchanged as *black boxes*, which is also beneficial to protect the intellectual property of the source code.

3.2. Machine learning models

The use of ML in industrial manufacturing offers immense potential for the optimization of procedures and processes. As a result of their effectiveness, they are used in various tasks like computer vision, speech and pattern recognition, natural language processing etc. [44]. However, the creation of the required models requires expert knowledge, since modern production plants are very individual and thus also very complex. The goal of the methods is to capture interrelationships that cannot be comprehensively achieved by classical modeling with the aim of capturing the optimization potential, such as product quality, resource utilization and to be able to initiate corresponding improvements in the sense of predictive maintenance, containment of unplanned machine failures or an improved quality assurance process. To develop models in these application domains, significant amounts of data are needed in the first stage of preparation. These data need to be collected and prepared with the help of experts, matching the applied ML technical requirements and the production-automation technologies. They serve as *training data* for the model.

ML models are usually trained in a High Performance Computing Cluster (HPC) and subsequently deployed in less compute-intensive environments, such as highly available computers in the cloud or edge devices. The *trained model* can be used for prediction, feature extraction, decision support, and pattern recognition once adaptation to the training data is complete. Such a trained model consists of a collection of numbers *(called weights)*, rules, and data structures specific to ML algorithms, required for decision-making and prediction. For example, in an ML deep learning application on an image classification use case, the NN algorithm, together with the backpropagation [45] and gradient descent algorithms [46], generates an ML model and optimizes the training data, which consists of a collection of correctly classified images.

3.3. Reduced Order Models

Together with ML, ROM is another popular technique among AI methods for DTs. It shows several similarities with ML, as detailed below.

ROM models are created through a numerical strategy that aims to transform a complex, time- and resources-consuming simulation or laboratory test-based model into a significantly simpler system. ROM models can be obtained through both intrusive [47–50] or non-intrusive methods [51–54]. Intrusive methods allow to directly modify the system's equations and are especially useful when the system's equations are known, but too complex to be solved for a general case, in a reasonable amount of time and with limited resources. On the other hand, non-intrusive methods can be employed either when the system's equations are known or completely unknown. These methods are hence purely data-based and can be compared to ML data-trained model.

An example for generating non-intrusive ROM models is provided by CAELIA^{TM} Twinkle authoring tool [55], i.e., a library for creating Tensor-Rank-Decomposition (TRD)-based ROMs, according to Eq. (1). Twinkle allows for an extensive manifold exploration, and it can work on dense, sparse or even unstructured data; for further details please refer to Zambrano et al. [56]. According to Eq. (1), the TRD method is a powerful method for simplifying and understanding a system. In fact, through the TRD approach it is possible to describe the system's behavior (i.e., output) by means of independent functions, one for each system variable separately. Using the TRD method, a problem with N variables can therefore be expressed as the product of N one-dimensional functions (i.e., one for each system variable).

$$F(v_1, \ldots, v_N) = \sum_{m=1}^{M} \alpha_m \prod_{n=1}^{N} f_{m,n}(v_n) \tag{1}$$

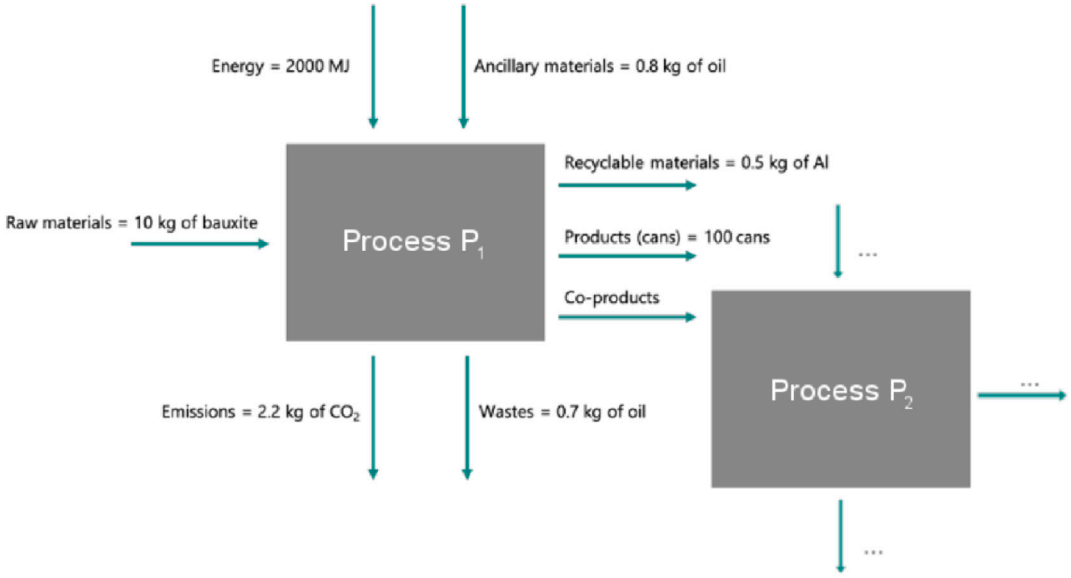

Fig. 1. Example of a SoS characterization from an LCA perspective.

where M is the ROM's approximation order, so that the first addend corresponds to a first approximation of the system, while following terms are corrections to it, being $\alpha_m, m = 1, \ldots, M$ weighting coefficients with generally decreasing values.

CAELIA^{TM} ROMs, for instance, can be easily obtained from input data files such as *.txt* or *.csv* that typically include data points where both inputs and output values are provided, similarly to supervised ML algorithms [57]. These ROM models can be easily embedded and managed by virtually any kind of environment, e.g., desktop applications or web interfaces, where users can provide keyboard inputs and navigate to obtain real-time results. The computed ROM model file can be straightforwardly embedded in different platforms as a *.txt* file that includes all the necessary information and parameters to compute the system's response in real-time.

3.4. System of Systems models

System of Systems (SoS) models can be based on both structural and behavioral models. One example of a SoS model, from a structural point of view, is the Life Cycle Assessment (LCA) tool [58] model applied in a manufacturing domain. LCA is a well-acknowledged methodology for analyzing the environmental impacts of manufacturing processes along their entire life cycle. LCA assessment leverages on the use of well-founded background data that enable users to personalize their operation information so to calculate the environmental indicators that represent impacts related to a specific process. This process is supported by the LCA Process Templates (PT) tool which creates process characterization by formalizing the Life Cycle Inventory (LCI) description, where for each specific process inputs and outputs are identified and quantified. LCI considers resources (i.e., inputs) coming from the ecosphere (e.g., raw material, water) or from another technosphere (e.g., ancillary material such as lubricating oil) and energy of various types. LCI assesses emissions (i.e., outputs), such as waste, products and co-products. LCI data represent the variables belonging to the LCA model and are retrieved directly from the production line or collected manually through IoT devices. Fig. 1 shows an example of LCA process characterization. The figure shows how LCI output of a certain process can become the input for another process in a SoS model.

From a LCA point of view, an output of a specific process (i.e., a system) can be formalized as an input to another process. Since a process is considered the fundamental unit for the LCA evaluation, a complex system might be represented by a production machine (as supplier of several processes) or a production line (obtained as the composition of production machines or single processes). This approach can be repeated by composing more processes on various levels of complexity.

In addition to LCI data, LCA methodology is also based on Life Cycle Impact Assessment (LCIA) information on the impacts related to the inventory, i.e., the computed environmental indicators. LCIA data, for the analysis of a specific system, can be calculated from the LCI data via characterization factors, i.e., factors meant to translate the inventory into environmental impacts. On the basis of the approach explained before, the LCA assessment of a SoS can be obtained as a composition of LCA assessments of each system.

Another type of SoS models supported by DIGITbrain is integrated 3D and Discrete Event Simulation (DES) models. DES models are behavioral model which describe the evolution of a system during a given time, in which the simulated time is advanced at discrete intervals only when a change in the system occurs, updating the state of the system. In DIGITbrain, DES is supported through the DDDSimulator simulation engine, a DES tool with integrated 3D virtual environment. Simulation models are composed by reusable elements called prototypes, each one representing a component of the simulated system (machine, transporters, logic module); prototype instances communicate among each other with ports that are used to exchange data between different modules. The models realized can be used to simulate the system for a period of time and obtain statistics that give information on the performance of the system in the simulated conditions. A graphical Integrated Development Environment (IDE) called DDD Model Editor supports the visual editing of both prototypes and models.

3.5. Stateful vs. stateless models

Some models always generate the same results for a specific set of in-slot values when instantiated (i.e., stateless models), while other models depend on their previous state and must therefore remain persistently instantiated (i.e., stateful models). On the one hand, in stateless models, outputs only depend on the model itself and current inputs, such as static equilibrium simulations, non-recurrent NNs, and LCAs. On the other hand, for stateful models, the outputs also depend on their internal state and thereby on a potentially unbounded number of previous inputs. Examples include transient or dynamic simulations and recurrent NNs.

Distinguishing between stateful and stateless models is particularly important for the correct evaluation of models. On the one hand, while stateless models can be reconstructed at any time, stateful models and their evaluating algorithms must remain instantiated on a given compute node. Alternatively, the algorithm must support serializing its state and restoring it on a different machine. On the other hand, keeping stateless models instantiated is not a requirement, although it might be beneficial to avoid instantiation overhead. In this context, it is important to consider the cost of on-demand instantiation of a model with a given algorithm, as well as (de)serialization costs for stateful models. If the cost of on-demand instantiation is less than the benefit derived from continuous model evaluation, then on-demand instantiation might be a better choice.

4. Reusable models for DIGITbrain

4.1. Reuse in the DIGITbrain ecosystem

The key to efficiently create DTs lies in the reuse of existing artifacts. The reuse of software artifacts is a well-established concept in software engineering [59]. The DIGITbrain ecosystem enables reuse by decomposing DTs into their constituting parts, i.e., data, models, and algorithms. Metadata describing these assets and their physical locations in external repositories are stored in the Platform, enabling users to create DTs by composing the parts into a Data-Model-Algorithm (DMA) tuple.

Focusing on models, the core problem is not the availability, but the identification, occasional adaptation, and composition of models with suitable data and algorithms to create DTs (i.e., DMA Tuples). In DIGITbrain a model classification has been developed to allow a model developer creating a meta-description, which ensures discoverability and composability of models with algorithms by DT experts after publication.

The major challenge tackled by DIGITbrain is to present the characterization in such a way that the model user can conveniently find the right model for efficient reuse. The core service responsible for enabling reuse is the DIGITbrain Asset Metadata Registry that stores a rich set of metadata about all DIGITbrain assets, including models. The actual models are only referenced from this Registry and stored in external model repositories (the project set up a sample model repository for demonstration purposes). The access service relies on the metadata descriptions to provide filtering, search functionality and to assure that the selected model is fed into the chosen algorithm at execution time.

Within this context, the main classification characteristics proposed in DIGITbrain are the various aspects of the model, for example modeling language, model inputs and outputs, and the structure and construction of the model. Moreover, further characteristics are taken into account, related to the evaluation of the model by the selected algorithm, such as its storage requirements during execution, and the information about the model's fidelity in its range of validity.

For this purpose, generalized metadata description tables were created (see Tables 1 to 3), to be filled with the selected characteristics of each model. The tables, represented as key–value pairs, are described in the following sections.

4.2. Model metadata description

The process of publishing a model on the DIGITbrain Platform requires providing all information needed for the model to be evaluated with a compatible algorithm. Once the model information has been collected, it can be published on the DIGITbrain Platform, using the dedicated DIGITbrain publishing interface.

As mentioned earlier, in order to facilitate data collection from different types of models, metadata specifications for models have been defined as a common structure to be used within the DIGITbrain Platform and are stored as key–value pairs in a relational database. Model metadata were designed not simply based on the analysis of various models, but also taking the characteristics of algorithms evaluating these models, and the requirements for the deployment and execution of the resulting DMA Tuples (DTs) into consideration. Model metadata are divided into three main categories: definition, parametrization, and publication. Tables 1 to 3 show a selection of model metadata related to each category, while Tables 4 to 6 are optional metadata tables very specifically referring to co-simulation models. A full specification will be released on the DIGITbrain Website (https://digitbrain.eu) in the near future.

Table 1
Selection of model metadata related to model definition.

Key	Type	Description
`ID`	`UUID`	SemVer ID of the model.
`name`	`string`	Name of the model.
`version`	`SemVer2`	Model version.
`license`	`string`	Licensing model chosen from a fixed set of known licenses.
`provider`	`enum`	Provider name: Institution or Person.
`provider_contacts`	`obj`	*(optional)* Dictionary with keys being phone, email, address.
`AuthTool`	`obj`	Authoring tool used to create the model.
`type`	`enum`	Model type, e.g., ML, LCA, 3D FEM, CFD, system simulation, discrete event simulation, or co-simulation; any algorithm that supports the given type can be used to evaluate this model.
`fidelity`	`number`	*(optional)* Error of the model's prediction.
`model_URI`	`URI`	Where the model(s) file(s) is/are stored.
`state_depend`	`bool`	Defines whether a model is stateful or stateless, c.f. Section 3.

Model definition fields (Table 1) include a human-readable description of models, model properties (i.e., I/O, hardware requirements etc.), and references to the externally stored models. This section contains an ID value, which is generated automatically by the DIGITbrain Asset Publishing Interface. The other fields are provided by the asset owner through the publishing interface. These include generic information as the model's name, version, license, and provider fields. The *AuthTool* key provides information about the authoring tool used for creating the model, while further fields describe its type (e.g., ML, LCA, etc.), the model's fidelity (i.e., reliability of prediction or errors), a URI for storage location and a *state_depend* key that differentiates between stateful and stateless models (see Section 3).

The parametrization section takes into account I/O and other parameters that might be needed for its evaluation. In Table 2 the generic features to be considered for model parametrization are listed, such as the *in_slots* (i.e., inputs and parameters) and *outputs* fields. These are composed of a unique model's key, a name, the number of dimensions, model's units (i.e., human-readable name, SI – International System of Units – exponents in a $[\mathrm{m}, \mathrm{s}, \mathrm{mol}, \mathrm{A}, \mathrm{K}, \mathrm{cd}, \mathrm{kg}]$ format, a scale offset if needed and a scale's order of magnitude), a default value if available, valid range, and a description. Moreover, the model parametrization metadata foresees optional fields presented in Tables 4 to 6 and related to co-simulation, in which dependencies, OS and hardware requirements are covered.

Publication metadata (Table 3) is currently a simple human-readable description of the model. However, this can potentially be replaced with a more complex ontological approach in the future that enables the automated interpretation and selection of models (a functionality that is not currently supported by the DIGITbrain Platform and done by the human operators/users).

Using the metadata specified above, model reusability becomes a reality. Once *Models* are published separately from other DIGITbrain *Assets* (e.g., *Algorithms* and *Data*), dynamic composition of these assets

Table 2
Selection of model metadata related to model parametrization.

Key	Type	Description
`in_slots`	`array[object]`	Input values and/or parameters for the model. The objects in this array contain input's or parameter's: unique key, name, number of dimensions, units (i.e., a human-readable name, SI – International System of Units – exponents in a [m, s, mol, A, K, cd, kg] format, a scale offset if needed and a scale's order of magnitude), default value if available, ranges (i.e., minima and maxima) and a description
`outputs`	`array[object]`	Model-specific outputs. Structured analogously to *in_slots*.
`cosim_solver_info`	`object` *(optional)*	Co-simulation models bundle binary solvers. Therefore, execution information such as operating system, CPU and GPU architecture, etc. are required in that case (see Tables 4 to 6).

Table 3
Selection of model metadata related to model publication.

Key	Type	Description
`description`	`CommonMark`	Human-readable description of the model, e.g., version, scope (simulation, control, etc.). Provided as CommonMark Markdown for rendering as a web page.

Table 4
Cosim_solver_info Dependant FMUs metadata *(optional)*.

Key	Type	Description
`dependencies`	`array[URI]`	Dependant FMUs for co-simulation.

Table 5
Cosim_solver_info OS requirements metadata *(optional)*.

Key	Type	Description
`osArch`	`enum`	OS architecture type (e.g., x86_64).
`osType`	`enum`	OS type (e.g., Windows, Linux).
`osDistribution`	`enum`	OS distribution (e.g., Ubuntu, Fedora).
`osVersion`	`SemVer2`	Version of the OS.

into executable DMA Tuples becomes possible at run-time. Obviously, the compatibility of the combined assets still needs to be checked and assured, either by the human composer (as in the current version of the DIGITbrain Platform) or via an ontology-based automated approach (which may be considered for the future). However, from the published metadata the DIGITbrain Platform is capable of generating the executable "artifact" automatically by combining the model with the algorithm that evaluates it and the data that is required as input for the calculation, without further human intervention.

Please note that while the above-mentioned composition and automatic code generation are important and interesting topics, these are out of the scope of this paper as we only concentrate on models and their reusability.

5. Authoring of DIGITbrain models

There are several authoring tools for generating, importing, managing, evaluating, and exporting various models. In this section the authoring tools related to DIGITbrain models detailed in Section 3 are described. Please note that such authoring tools are outside the DIGITbrain Platform. With such decision DIGITbrain does not limit the choice of authoring tools that can be used to generate models. Any authoring tool can be applied, and the generated models can be registered in a uniform way with the Platform, as described in Section 4. As a consequence, the authoring tools detailed in this section are examples only. Other authoring tools can also be used freely in relation to the DIGITbrain approach.

5.1. Authoring tools for co-simulation models

Many modeling and simulation tools are used to author functional mockup unit (FMUs) – co-simulation units – conforming to the functional mockup interface (FMI) standard; over 100 of these authoring tools can be found on the FMI website [60]. Using FMI, FMUs can be authored by the manufacturers of industrial products and shared among the users.

The Integrated Tool Chain for Model-based Design of Cyber–Physical Systems (INTO-CPS toolchain), a Project led by Aarhus University, is an example co-simulation toolchain [15,61]. The INTO-CPS toolchain consists of a framework (i.e., a family of tools) built around FMI-compatible co-simulation for a collaborative development of Cyber–Physical Systems (CPS). The core of the INTO-CPS toolchain is the co-simulation orchestration engine called Maestro [43]. This essentially enables any authoring tool that can export their models to an FMU to be usable. The toolchain integrates with FMU authoring tools such as Modelio, and Overture [62,63]. Modelio is an open-source modeling environment capable of generating FMUs from SysML profiles [64]. Overture is a tool used to specify and execute formal models described in Vienna Development Method (VDM) [65]. Overture is capable of exporting VDM models into FMUs. Other FMU authoring tools do exist; OpenModelica is one such example as an interactive development environment for system representation in Modelica language [66,67]. The three tools mentioned above model Cyber Physical Systems (CPS) in different ways and export these into FMUs. The users can then apply the INTO-CPS toolchain to import these FMUs and combine multiple FMUs into a co-simulation model of an industrial product.

5.2. Authoring tools for machine learning models

As ML models are generated through the data driven training process on ML algorithms, necessary algorithm packages have to be developed in the particular ML framework. These pre-configured algorithms are created with the help of external authoring tools. Such authoring tools create specific microservices performing certain data handling tasks and build up an ML algorithm by combining one or more individual microservices. In ML applications DIGITbrain makes use of reference architectures, combination of widely-used machine learning tools and libraries that can be deployed by *one click* in the cloud, to enable this automated creation of containerized, ready-to-use software stacks, for non-experienced users. Many steps of the authoring process could be automated, by providing domain specific templates and building on tools that are already available, for example Jupyter Labs [68].

Within the reference architectures, the domain experts (data scientists) can build on and use well-known ML frameworks. The DIGITbrain Platform provides a fast way to reuse models and train them individually with customized datasets. This fast and easy reproducibility of implementations provides a very fast way to implement AI in production. This model processing takes place in a predefined working environment supported by ML frameworks such as TensorFlow [69], PyTorch [70], etc. After training, the generated models can be stored in ML framework-specific formats and executed on less powerful hardware. To support model portability, the open source community has proposed the Open Neural Network Exchange (ONNX) standard [71]. With this approach, we propose a standard to exchange and transfer trained ML models to the different devices for applications. As another option, we also rely on the *SavedModel* format [72] favored by Google.

Table 6
Cosim_solver_info Hardware requirements metadata *(optional)*.

Key	Type	Description
`recommendedNumberOfGPUCores`	`number`	Recommended number of GPU cores.
`minimumNumberOfGPUCores`	`number`	Minimum required number of GPU cores.
`recommendedGPURAM`	`number`	Recommended GPU memory.
`minimumGPURAM`	`number`	Minimum required GPU memory.
`recommendedRAM`	`number`	Recommended Memory.
`minimumRAM`	`number`	Minimum required memory.
`recommendedCPUs`	`number`	Recommended number of CPU cores.
`minimumCPUs`	`number`	Minimum required number of CPU cores.
`requiredDiskSpace`	`number`	Required amount of disk space in GB.

5.3. Authoring tools for Reduced Order Models

As described in Section 3, ROM is an umbrella of methods that can be handful in different situations where a reduction in terms of system variables and their relations should be taken into consideration. The most common scenarios for ROM application are the followings:

- a large amount of data is to be analyzed,
- the physics behind the system under investigation remains uncertain,
- the system's equations are well known, but the solution is not trivial and/or its computation is highly time- and resource consuming.

To support such scenarios, the CAELIA™ tool was developed at ITAINNOVA for ROM generation and management, using TRD technique. CAELIA™ can be used for different manufacturing processes, such as injection molding, rubber extrusion, hot stamping, and laser welding. CAELIA™ consists of a set of libraries, where models are generated using Twinkle library [56]. Since TRD-based ROM models are based on data (please, refer to Section 3) that can be obtained through experiments, simulations or by mathematically solving equations (if known), a careful Design of Experiment (DoE) must be performed beforehand. Hence, this authoring tool also includes a library for automatic DoE with an enhanced space coverage. Additionally, CAELIA™ includes different optimization algorithms and several visualization interfaces. All these libraries and tools can be combined and used for enhancing ROM computation and management.

CAELIA™'s ROMs are a valuable tool, not only capable of forecasting unexplored behaviors of the system, but also useful for optimization, i.e., for finding the system's operating conditions that correspond to a desired output.

The CAELIA™ toolset has been already tested against a large variety of cases, such as fluid- and thermodynamics, friction modeling, etc. In Section 6 an injection application CAELIA™ will be detailed.

5.4. Authoring tools for System of Systems models

The Sustainability Assessment Application (SAA) is an evolution of an already existing application developed by SUPSI, i.e., Scuola Universitaria Professionale della Svizzera Italiana, in previous EU projects (MANUTELLIGENCE [73] and MANUSQUARE [12]). SAA is an authoring tool which allows users to characterize the processes provided by a production machine or a production line according to a LCA point of view. Fig. 2 shows an example of formalization of a set of processes. The tool enables the users to specify the SoS composition where the output flow (LCI output) of a certain process can be formalized as input flow (LCI input) for another process. The model underlying the SAA has been extensively described in Section 3, where the concept of PT emerges as an element related to a specific Functional Unit (i.e., the quantification of the system's function analyzed by LCA), that is meant to quantify the function of the process under investigation (examples are: 1 h of milling process execution or 1 kg of removed steel for milling, 1 kg of injected plastic for injection molding, etc.; please refer to Fig. 2). Starting from the background data, retrieved from a LCIA database such as Ecoinvent [74], the percentage contribution of inputs and outputs to the selected environmental indicators are evaluated in order to identify process parameters critical from the LCA point of view. For instance, concerning the Climate Change indicator of the injection molding operation, it has been estimated that inputs such as *Electricity* and *Natural Gas (heat)* represent over 80% of the indicator value's input. Through this sort of Pareto analysis, performed by LCA experts and partially automatized by the SSA, whenever a new operation type is introduced into the ecosystem, the LCI data affecting most of the process in terms of environmental impacts is determined. The identified crucial parameters are hence considered as *free* ones that, starting from the default value proposed by the SSA, can be customized by the supplier, together with more specific indicators values, in order to better represent its manufacturing operation. The Pareto analysis, here presented only for the Climate Change impact category, has to be performed on the complete set of the selected indicators. By clicking on a specific process, it is possible to visualize the LCI data characterizing the whole, where the most relevant ones from the impact point of view are highlighted in green, as shown in Fig. 3. Only the highlighted LCI can be customized by the user considering a different input value (n.b., the user usually knows the amount of the electricity that is actually consumed by a given machine).

Another example for SoS authoring tools is the DDDSimulator which a DES tool with an integrated 3D virtual environment. Simulation models are composed by reusable elements called prototypes each one representing a component of the simulated system (machine, transporters, logic module). Prototype instances communicate among each other with ports that are used to exchange data between different modules. The developed models can be used to simulate the behavior of the entire system by obtaining statistics that give information on the performance of the system within the simulated conditions. A graphical IDE called DDD Model Editor supports the visual editing of both prototypes and models. The modules can be combined in different configurations to realize DTs of plants to evaluate and compare the performances of alternative design. Due to this modular approach, with each module encapsulating the behavior of a complex system, the modules can be reused to compose SoS models.

6. Experimental results

In this section the results obtained in some of the experiments performed during the first year of the DIGITbrain Project are summarized. The examples described in this section show how DIGITbrain architecture helps developing DTs. Moreover, the experimental results obtained during the first year of the project, some of them being summarized below, show that the DIGITbrain's framework can be straightforwardly applied in I4.0 to accelerate industrial digitalization.

6.1. Co-simulation in agricultural robots

AgroIntelli manufactures and sells semi-autonomous agricultural vehicles to farmers, called Robotti. Robottis are delivered to farmers and used in diverse agricultural environments. The company expects a rapid growth in the sales of Robotti and would like to know the best

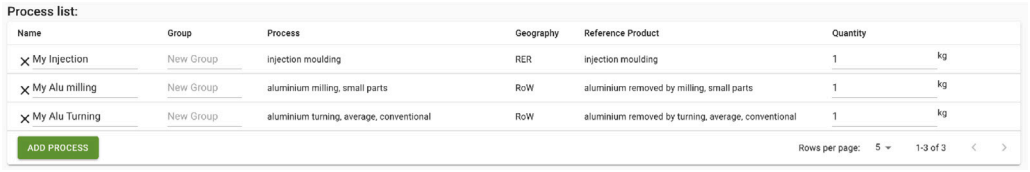

Fig. 2. List of the processes, for example belonging to a production line.

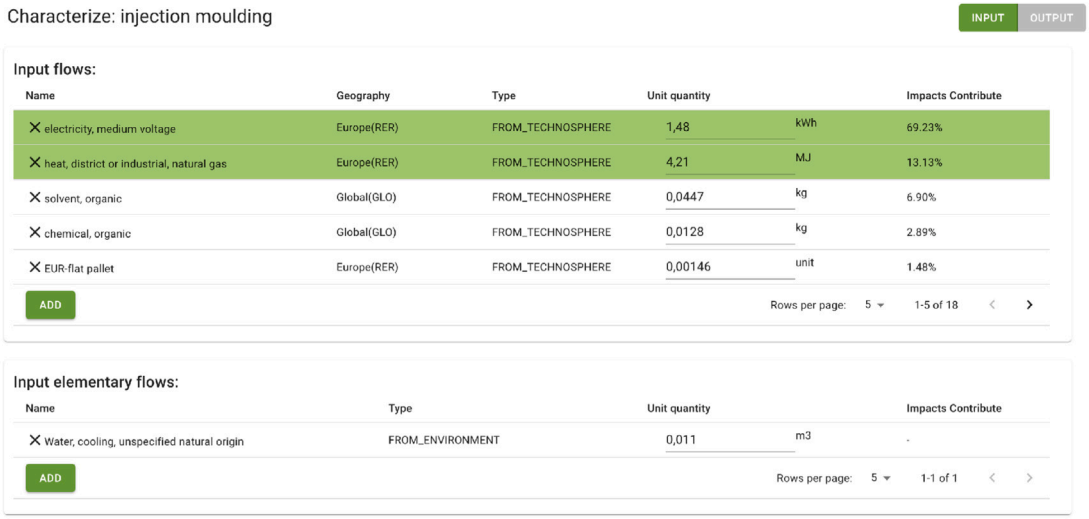

Fig. 3. Example of process characterization (LCI input and output) by the SSA. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

way of scaling up the production based on the expected sales. Right now, AgroIntelli assembles each robot on a made-to-order basis, at a single workstation, with predominantly manual work being required to assemble and interconnect all subcomponents and devices to each order. AgroIntelli would like to design and validate an automated manufacturing line for Robotti that can take the manufacturing blueprint and produce the robot.

Additionally, certain parts of Robotti, for example the air filter, the hydraulic oil, or the oil filter, need to be replaced during scheduled maintenance at different service intervals. When required, farmers also place orders for Robotti parts. Once the part orders from farmers are received, AgroIntelli needs to integrate the production of these parts into the manufacturing schedule. The updated production schedule will have to produce an expected delivery date for all the ordered parts.

A DT of AgroIntelli's manufacturing line helps the company performing an analysis of alternative scenarios for future factory construction. The projected manufacturing capacity, based on estimated sales and the manufacturing requirements of various parts based on sample Robotti usage scenarios, are key inputs of the factory DT. On the other hand, a feasible production schedule is the key output of the DT. The creation of the DT for the manufacturing line is done using the FMI [60] co-simulation standard. The key elements/workstations of the manufacturing line are modeled as FMUs. The DT for the manufacturing line is created by putting the FMUs together as one system and then co-simulated using Maestro [43]. The co-simulation *Algorithm* uses a configuration file for connecting the FMUs and performing a single co-simulation.

The factory DT is created from reusable FMUs which together form the *Model* of the DT. The Maestro co-simulation *Algorithm* becomes the *Algorithm* evaluating the factory DT *Model*. The evaluation of the factory *Model* by the Maestro *Algorithm* also requires *Data*. This *Data* come from the planned production schedules and the on-demand part orders from farmers.

In the context of agile manufacturing practices, it is important to keep the manufacturing line flexible and reroute the parts based on the current demand. Therefore, FMUs need to be connected differently for co-simulating alternative manufacturing scenarios. One such scenario could be the creation of multiple models of Robotti. Another scenario could be the installation of parallel workstations to scale up the production of complicated machine parts. Each of these alternative scenarios can be explored by using one factory *Model* (implemented as multiple FMUs) and different co-simulation configurations. Therefore, the Maestro co-simulation *Algorithm* is reusable for all co-simulation models. Additionally, different manufacturers can bring in their co-simulation models and perform co-simulation using Maestro co-simulation *Algorithm*. The factory *Model* can also be reused by AgroIntelli to perform different design scenarios for the planned manufacturing line. In other words, the same *Model* can be used to perform Design Space Exploration (DSE) for designing an optimum manufacturing line [15]. Since each FMU represents a factory workstation, there is a potential for reusing single FMUs to create new co-simulation scenarios of other factory configurations/factories. As a result, a DT created for AgroIntelli is reusable within the broader manufacturing industry too. This has been made possible by the clean separation of *Data*, *Algorithm*, and *Model* of the factory DT using the DIGITbrain approach.

6.2. Reduced order models for injection molding

A physically based DT for a thermoplastic injection molding consists of the digitalization of a thermoplastic material injection molding process for the quick initial set-up of process parameters and the optimization during production. The DT is based on the offline exploitation of state-of-the-art numerical simulation tools to model the relevant fluid-dynamics, heat transfer and thermo-mechanical physical mechanisms, as well as subsequent encapsulation of relevant Key Performance Indicators (KPIs) into ROMs that are made available for the user to get a real time virtual try-out and estimated production quality monitoring.

Within the DIGITbrain experiment performed for Inymon, a Spanish injection molding company, current state-of-the-art simulation models

representing the physics of the material transformation process, including transient flow fluid-dynamics, heat transfer by conduction and convention, and solid thermo-mechanics were used. Specific material relations, such as non-Newtonian rheology and equations of state, reproduce the material's behavior. These models are expressed as a set of differential equations corresponding to the conservative laws (i.e., mass, moment, energy) in fluid and solid domains. These physical equations are solved using Volume of Fluid (VoF)/FEM by the Moldex3D commercial program [75]. Furthermore, an offline execution of a simulation DoE is carried out to allow the virtual exploration of the *Model* to generate a database of simulation results, covering suitable variation ranges of the processing parameters (i.e., melt temperature, filling time/ram speed, switch over point, packing pressure and time, coolant temperature and flow rate, cooling time, etc.).

As mentioned previously, online process modeling for quality control requires real-time simulation capabilities, which cannot be achieved by executing the computationally intensive physically based simulation models (as these require hours to run) within a real cycle time (i.e., <1 min). Creating ROMs has proved to be a valid approach to provide real-time responses to Physically Based Simulations (PBS). A ROM retains only the relevant information required for quality evaluation (i.e., selected KPIs) and process control. Among the different techniques available for ROM generation, *a posteriori* or non-intrusive techniques have a higher potential, especially combined with real *Data*. CAELIA™'s ROM generation, based on the factorization of the information through TRD approach, results into a sequence of products of separable 1D functions that can be solved *on the fly* (see Eq. (1)), allowing having a transfer function directly related to input parameters, independently [56].

Within the scope of the DIGITbrain Project, the ROM generation solution, used to build the ROMs, has been deployed in the cloud-based platform as a job. Moreover, the CAELIA™ App including a Graphical User Interface (GUI) has also been deployed on a Virtual Machine (VM) instance on the Platform. The GUI allows the user to compose tuples of ROMs (as DIGITbrain *Models*), input *Data*, as well as evaluation and optimization *Algorithms* to try finding the parameters describing the best injection molding process.

ROMs are specific of each injection process (IP instance), as they are built on physically based simulation *Data* generated for a specific physical instance, which makes the reusability of ROMs anything but straightforward. Nevertheless, the algorithms used to build ROMs and the CAELIA™ App are generic, so that different users can employ them and upload their ROMs and *Data* from other injection molding lines. A possible way to reuse ROMs would be the development of an expert system that, by analyzing a database of ROMs, generated for different injection molding processes, could extract rules of dependencies for typical results obtained from defined process conditions (knowledge-based reasoning). As a further step, the aforementioned expert system could also, based on the characteristics of the part to be injected, be able to create a new ROM backbone by proposing 1D functions for the TRD factorization, that represent the dependency of a result on a specific process condition. This *Model* could be used as a low fidelity *Model* to analyze tendencies or as the basis for a new adjustment, based on PBS or measured *Data*.

6.3. System of systems models for laser-cutting and forming of aluminum

Creation of a simulation tool which allows the end user to create alternative layouts of production lines for forming aluminum sheets and compare the performances of the different solutions has high importance. The main limitations that reflect the current status of most EU manufacturing SMEs in the utilization of DTs of production lines, are the following:

- the inability to perform real-time *what-if* analysis based on the actual plant *Data*, that strongly impairs the ability to promptly tackle production needs and constraints,
- the inability to use DT in the bidding/negotiation phase, that hampers the possibility to demonstrate all plant configurations, leading to under-exploited machine customization potential,
- the inability to provide simulation services and analytics empowered by the knowledge developed all along the plant's life-cycle reduces plant productivity and efficiency, reducing customer satisfaction,
- the dichotomy existing between the real plant and its digital representation makes simulation and analysis tools of little use after the ramp-up phase, leading to under-exploited productivity during machine usage due to the inability to use context-synchronized and history-aware simulation.

The inability to connect to existing manufacturing plants in many cases comes from the presence of legacy control hardware, as this legacy hardware does not have the ability to be connected to the external world. Investing in updating the control hardware of a manufacturing line usually requires high investment. To overcome this, *CPSisers* are deployed at shop-floor level. Such *CPSisers* are software and hardware components used to turn the machines into CPS. The task of developing the CPSiser was assigned to NXT control, an automation consulting company partner of the DIGITbrain Project. To interact with the real plant, NXT control developed a CPSiser, which allows legacy hardware of different brands and vendors to be interfaced with the FIWARE [12] communication infrastructure, sending *Data* from sensor and Programmable Logic Controllers (PLCs) to the DT to synchronize it with the reality. Simulation prototypes for eight modules were created, which allowed the simulation of two alternative layouts with different configurations of the unloading system and selecting the better performer. The simulation *Model* was then uploaded on a VM in the cloud and from there it can be accessed and executed via an internet browser loading different production plans.

Within this context, the discrete simulation engine (DDDSimulator) [76] was used for SoS analysis. The DDDSimulator is a simulation tool for integrated 3D and DES, developed and distributed by Technology Transfer System (TTS). This simulator enables its users to design new layouts combining reusable modules, each representing an element of the plant such as machines, transporters, feeding systems and robots. The simulation models can be used to assess the performance of alternative layouts, after which the *Model* of the effective plant can be uploaded to the DIGITbrain Platform and used to simulate the performances of different production plans.

To simulate a production line, the end user uses the Domain-Driven Design (DDD) *Model* editor, a standalone Windows application that can be installed on a Windows PC. The user drags modules from the prototype catalog into the 3D environment. Each prototype simulates the behavior of a component of the plant and several instances of the same prototype can be added to one simulation *Model*. Once an instance is dragged into the 3D environment, it can be moved and rotated to adjust its position in space. Each specific instance also has parameters that can be configured, representing the peculiarities of the specific instance and enabling the configuration of both logical or physical characteristics of the module. For example, the speed of a linear axis in a robot or the length of a conveyor are parameters that can be configured for a module. Modules can exchange information between them using ports, and ports are connected in the visual interface to allow all the modules to work based on the entire logic of the system. This modular structure allows for the re-usability of modules, which already encapsulate the behavioral model of a single complex system, to compose SoS models of several complex production systems combining the reusable elements.

7. Conclusion

The European H2020 Project "DIGITbrain" aims at creating an integrated digital platform that enables Manufacturing as a Service (MaaS) which lowers the barriers for Small and Medium-size Enterprises (SMEs) to adopt key Industry 4.0 technologies. The demands for

increased production and the high standards of consumers' expectations in the Industry 4.0 (I4.0) era encourage the introduction of Digital Twins (DTs) in the manufacturing industry, especially to help SMEs forecasting and optimizing their systems (e.g., processes, machinery, production lines) under different operating conditions. Offering MaaS on an integrated digital platform is an efficient way to achieve that.

In the paper, it was highlighted how DIGITbrain's overall concept supports the reusability of *Models* by separating the fundamental building blocks of DTs, including the *Model* itself, its *Algorithms* and *Data*, with a rich set of metadata. Moreover, given the definition of a DT as a virtual model of a physical system, models represent a crucial asset for industrial digitalization and the introduction of DTs in manufacturing processes. We further detailed some model types that are currently supported by the DIGITbrain Platform, such as co-simulation, Artificial Intelligence (AI) and System of Systems (SoS) models. Moreover, we detailed three representative case studies, where practical applications of DTs were delivered to manufacturing SMEs and their end-users.

Further work is being conducted towards the generation of improved models for several applications. The models will be published on the project's integrated platform which is currently being upgraded to grant scalability, parallelization and efficiency.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was funded by the DIGITbrain – Digital twins bringing agility and innovation to manufacturing SMEs, by empowering a network of DIHs with an integrated digital platform that enables Manufacturing as a Service project, No. 952071, European Commission (EU H2020).

References

[1] Grieves M. Digital twin: manufacturing excellence through virtual factory replication. White paper 1, Florida Institute of Technology; 2014, p. 1–7.
[2] European Commission. Digital transformation scoreboard 2018. EU businesses go digital: Opportunities, outcomes and uptake. Luxembourg: Publications Office of the European Union; 2018, p. 10–2826, 10 (691861).
[3] Semeraro C, Lezoche M, Panetto H, Dassisti M. Digital twin paradigm: A systematic literature review. Comput Ind 2021;130:103469.
[4] Jones D, Snider C, Nassehi A, Yon J, Hicks B. Characterising the digital twin: A systematic literature review. CIRP J Manuf Sci Technol 2020;29:36–52.
[5] Fisher O, Watson N, Porcu L, Bacon D, Rigley M, Gomes RL. Cloud manufacturing as a sustainable process manufacturing route. J Manuf Syst 2018;47:53–68.
[6] Kusiak A. Service manufacturing: Basic concepts and technologies. J Manuf Syst 2019;52:198–204.
[7] Kusiak A. Service manufacturing = process-as-a-service + manufacturing operations-as-a-service. J Intell Manuf 2020;31(1):1–2.
[8] Rauschecker U, Meier M, Muckenhirn R, Yip ALK, Jagadeesan AP, Corney J. Cloud-based manufacturing-as-a-service environment for customized products. In: EChallenges E-2011 conference proceedings. IIMC International Information Management Corporation; 2011.
[9] Meier M, Seidelmann J, Mezgár I. ManuCloud: the next-generation manufacturing as a service environment. ERCIM News 2010;33–4.
[10] Hasan M, Starly B. Decentralized cloud manufacturing-as-a-service (CMaaS) platform architecture with configurable digital assets. J Manuf Syst 2020;56:157–74.
[11] Kiss T. A cloud/HPC platform and marketplace for manufacturing SMEs. In: 11th international workshop on science gateways, IWSG 2019. 2019.
[12] Landolfi G, Barni A, Izzo G, Fontana A, Bettoni A. A MaaS platform architecture supporting data sovereignty in sustainability assessment of manufacturing systems. Procedia Manuf 2019;38:548–55.
[13] Kiss T, Kacsuk P, Kovács J, Rakoczi B, Hajnal A, Farkas A, Gesmier G, Terstyanszky G. Micado—microservice-based cloud application-level dynamic orchestrator. Future Gener Comput Syst 2019;94:937–46.
[14] Cloud orchestration at the level of application iCal. About - COLA project. 2017, https://project-cola.eu/cola-project/.
[15] Larsen P, Fitzgerald J, Woodcock J, Fritzson P, Brauer J, Kleijn C, Lecomte T, Pfeil M, Green O, Basagiannis S, Sadovykh A. Integrated tool chain for model-based design of cyber-physical systems: The INTO-CPS project. In: CPS data workshop. Vienna, Austria: IEEE; 2016, p. 1–6. http://dx.doi.org/10.1109/CPSData.2016.7496424, This publication is part of the Horizon 2020 project: Integrated Tool chain for model-based design of CPSs (INTO-CPS), project/GA number 644047.; null ; Conference date: 11-04-2016 Through 11-04-2016. URL https://www.twt-gmbh.de/en/cpsdata.html.
[16] Negri E, Fumagalli L, Macchi M. A review of the roles of digital twin in CPS-based production systems. Procedia Manuf 2017;11:939–48.
[17] DIGITbrain EU Project, https://digitbrain.eu/.
[18] Boschert S, Rosen R. Digital twin-the simulation aspect. In: Mechatronic futures. Springer; 2016, p. 59–74.
[19] Kritzinger W, Karner M, Traar G, Henjes J, Sihn W. Digital Twin in manufacturing: A categorical literature review and classification. IFAC-PapersOnLine 2018;51(11):1016–22.
[20] Rosen R, Von Wichert G, Lo G, Bettenhausen KD. About the importance of autonomy and digital twins for the future of manufacturing. IFAC-PapersOnLine 2015;48(3):567–72.
[21] Gabor T, Belzner L, Kiermeier M, Beck MT, Neitz A. A simulation-based architecture for smart cyber-physical systems. In: 2016 IEEE international conference on autonomic computing (ICAC). IEEE; 2016, p. 374–9.
[22] Yang J, Zhang W, Liu Y. Subcycle fatigue crack growth mechanism investigation for aluminum alloys and steel. In: 54th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference. 2013, p. 1499.
[23] Stojanovic L, Usländer T, Volz F, Weißenbacher C, Müller J, Jacoby M, Bischoff T. Methodology and tools for digital twin management—The FA3ST approach. IoT 2021;2(4):717–40.
[24] González I, Calderón AJ, Figueiredo J, Sousa J. A literature survey on open platform communications (OPC) applied to advanced industrial environments. Electronics 2019;8(5):510.
[25] Lu Y, Liu C, Kevin I, Wang K, Huang H, Xu X. Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues. Robot Comput-Integr Manuf 2020;61:101837.
[26] Liu M, Fang S, Dong H, Xu C. Review of digital twin about concepts, technologies, and industrial applications. J Manuf Syst 2021;58:346–61.
[27] Bazilevs Y, Deng X, Korobenko A, Lanza di Scalea F, Todd M, Taylor S. Isogeometric fatigue damage prediction in large-scale composite structures driven by dynamic sensor data. J Appl Mech 2015;82(9):091008.
[28] He Y, Guo J, Zheng X. From surveillance to digital twin: Challenges and recent advances of signal processing for industrial internet of things. IEEE Signal Process Mag 2018;35(5):120–9.
[29] Ghosh AK, Ullah AS, Kubo A. Hidden Markov model-based digital twin construction for futuristic manufacturing systems. AI EDAM 2019;33(3):317–31.
[30] Feng H, Gomes C, Thule C, Lausdahl K, Sandberg M, Larsen PG. The incubator case study for digital twin engineering. 2021, arXiv preprint arXiv:2102.10390.
[31] Feng H, Gomes C, Sandberg M, Thule C, Lausdahl K, Larsen PG. Developing a physical and digital twin: An example process model. In: 3rd international workshop on multi-paradigm modeling for cyber-physical systems (MPM4CPS'21). IEEE; 2021.
[32] Brunton SL, Noack BR, Koumoutsakos P. Machine learning for fluid mechanics. Annu Rev Fluid Mech 2020;52:477–508.
[33] Magargle R, Johnson L, Mandloi P, Davoudabadi P, Kesarkar O, Krishnaswamy S, Batteh J, Pitchaikani A. A simulation-based digital twin for model-driven health monitoring and predictive maintenance of an automotive braking system. In: Proceedings of the 12th international modelica conference, Prague, Czech Republic, May 15-17, 2017. Linköping University Electronic Press; 2017, p. 35–46.
[34] Kapteyn MG, Knezevic DJ, Huynh D, Tran M, Willcox KE. Data-driven physics-based digital twins via a library of component-based reduced-order models. Internat J Numer Methods Engrg 2020.
[35] Plattform Industrie 4.0, http://www.plattform-i40.de.
[36] Oztemel E, Gursev S. Literature review of Industry 4.0 and related technologies. J Intell Manuf 2020;31(1):127–82.
[37] Standardization council Industrie 40. German standardization roadmap: Industrie 4.0. 2020.
[38] Beuth. DIN SPEC 91345: 2016-04 Reference architecture model industrie 4.0 (RAMI4. 0). 2016.
[39] Lin S-W, Murphy B, Clauer E, Loewen U, Neubert R, Bachmann G, Pai M, Hankel M. Architecture alignment and interoperability: An industrial internet consortium and plattform Industrie 4.0 joint whitepaper. 2017.
[40] Boss B, Malakuti S, Lin S, Usländer T, Clauer E, Hoffmeister M, Stojanovic L, Flubacher B. Digital twin and asset administration shell concepts and application in the industrial internet and Industrie 4.0. An industrial internet consortium and plattform Industrie 4.0 joint whitepaper. 2020.
[41] Gomes C, Thule C, Broman D, Larsen PG, Vangheluwe H. Co-simulation: a survey. ACM Comput Surv 2018;51(3):49:1–33.

[42] Blochwitz T, Otter M, Akesson J, Arnold M, Clauss C, Elmqvist H, Friedrich M, Junghanns A, Mauss J, Neumerkel D, Olsson H, Viel A. The functional mockup interface 2.0: The standard for tool independent exchange of simulation models. In: Proceedings of the 9th international modelica conference. Munich, Germany: The Modelica Association, Linköping University Electronic Press; 2012, p. 173–84.
[43] Thule C, Lausdahl K, Gomes C, Meisl G, Larsen PG. Maestro: The INTO-CPS co-simulation framework. Simul Model Pract Theory 2019;92:45–61. http://dx.doi.org/10.1016/j.simpat.2018.12.005, URL http://www.sciencedirect.com/science/article/pii/S1569190X1830193X.
[44] Popper J, Hermann J, Cui K, Bergweiler S, Weyer S, Ruskowski M, Wang M, Guang L, Hu Y, Kueh V, Wang D, Madigan M, Oppermann I, Ryu S-M, Kim L, Wei S, Li R, Du X, Shang Y, Thonet G. Artificial intelligence across industries. Tech. rep., IEC Whitepaper; 2018, p. 93.
[45] Rojas R. The backpropagation algorithm. In: Neural networks. Springer; 1996, p. 149–82.
[46] Bottou L. Stochastic gradient descent tricks. In: Neural networks: Tricks of the trade. Springer; 2012, p. 421–36.
[47] Rowley CW, Colonius T, Murray RM. Model reduction for compressible flows using POD and Galerkin projection. Physica D 2004;189(1–2):115–29.
[48] Rozza G, Veroy K. On the stability of the reduced basis method for Stokes equations in parametrized domains. Comput Methods Appl Mech Engrg 2007;196(7):1244–60.
[49] Baur U, Benner P, Feng L. Model order reduction for linear and nonlinear systems: a system-theoretic perspective. Arch Comput Methods Eng 2014;21(4):331–58.
[50] Chinesta F, Cueto E. PGD-based modeling of materials, structures and processes. Springer; 2014.
[51] El Halabi F, Gonzalez D, Chico-Roca A, Doblare M. Multiparametric response surface construction by means of proper generalized decomposition: An extension of the PARAFAC procedure. Comput Methods Appl Mech Engrg 2013;253:543–57.
[52] Cichocki A, Zdunek R, Phan AH, Amari S-i. Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation. John Wiley & Sons; 2009.
[53] Willcox K, Peraire J. Balanced model reduction via the proper orthogonal decomposition. AIAA J 2002;40(11):2323–30.
[54] Ibáñez R, Abisset-Chavanne E, Ammar A, González D, Cueto E, Huerta A, Duval JL, Chinesta F. A multidimensional data-driven sparse identification technique: The sparse proper generalized decomposition. Complexity 2018;2018:1–11.
[55] Izquierdo S, Rodr'iguez R, Zambrano V. TWINKLE: A digital-twin-building kernel for real-time computer-aided engineering. 2020, https://github.com/caeliaITAINNOVA/Twinkle.
[56] Zambrano V, Rodríguez-Barrachina R, Calvo S, Izquierdo S. TWINKLE: A digital-twin-building kernel for real-time computer-aided engineering. SoftwareX 2020;11:100419. http://dx.doi.org/10.1016/j.softx.2020.100419, URL https://www.sciencedirect.com/science/article/pii/S2352711019300664.
[57] Ayodele T. Types of machine learning algorithms. In: New advances in machine learning. InTech; 2010, p. 19–48. http://dx.doi.org/10.5772/9385, Ch. 3.
[58] International Organization for Standardization. Environmental management – Life Cycle Assessment – Principles and framework. URL https://www.iso.org/standard/37456.html.
[59] McClure C. Software reuse - a standards-based guide. Wiley-IEEE Computer Society Press; 2001.
[60] Functional mock-up interface. https://fmi-standard.org/tools/.
[61] Bagnato A, Brosse E, Quadri I, Sadovykh A. INTO-CPS: An integrated "tool chain" for comprehensive: model-based design of cyber-physical systems. In: ICSSEA 2015 proceedings. 2015, p. 31–5, This publication is part of the Horizon 2020 project: Integrated Tool chain for model-based design of CPSs (INTO-CPS), project/GA number 644047.; null ; Conference date: 27-05-2015 Through 29-05-2015.
[62] Modelio: Open source development environment. 2021, https://www.modelio.org/.
[63] Larsen PG, Battle N, Ferreira M, Fitzgerald J, Lausdahl K, Verhoef M. The overture initiative integrating tools for VDM. ACM SIGSOFT Softw Eng Notes 2010;35(1):1–6.
[64] Fitzgerald J, Gamble C, Larsen P, Pierce K, Woodcock J. Cyber-physical systems design: Formal foundations, methods and integrated tool chains. In: Proceedings of the 2015 IEEE/ACM 3rd FME workshop on formal methods in software engineering (FormaliSE). IEEE; 2015, p. 40–6. http://dx.doi.org/10.1109/FormaliSE.2015.14, This publication is part of the Horizon 2020 project: Integrated Tool chain for model-based design of CPSs (INTO-CPS), project/GA number 644047.; null ; Conference date: 18-05-2015 Through 18-05-2015.
[65] Larsen PG, Lausdahl K, Battle N, Fitzgerald J, Wolff S, Sahara S, Verhoef M, Tran-Jørgensen PW, Oda T, Chisholm P. VDM-10 language manual. 2010.
[66] Fritzson P, Pop A, Abdelhak K, Ashgar A, Bachmann B, Braun W, Bouskela D, Braun R, Buffoni L, Casella F, Castro R, Franke R, Fritzson D, Gebremedhin M, Heuermann A, Lie B, Mengist A, Mikelsons L, Moudgalya K, Ochel L, Palanisamy A, Ruge V, Schamai W, Sjölund M, Thiele B, Tinnerholm J, Östlund P. The OpenModelica integrated environment for modeling, simulation, and model-based development. Model Identif Control 2020;41(4):241–95. http://dx.doi.org/10.4173/mic.2020.4.1.
[67] Henriksson D, Elmqvist H. Cyber-physical systems modeling and simulation with modelica. In: Proceedings of the 8th international modelica conference. Linköping University Electronic Press; 2011, p. 502–9.
[68] Project jupyter, https://jupyter.org/.
[69] Tensorflow, https://www.tensorflow.org/.
[70] Pytorch, https://pytorch.org/.
[71] Bai J, Lu F, Zhang K, et al. ONNX: Open neural network exchange. 2019, https://github.com/onnx/onnx.
[72] Using the savedmodel format: Tensorflow core, https://www.tensorflow.org/guide/saved_model.
[73] Fontana A, Corti D, Alge M, Petrucciani M, Calvo L, Marangi L. Integrating a LCA tool with a design platform towards a sustainable-aware PSS design: application in a fablab environment. IFAC-PapersOnLine 2018;51(11):1125–30.
[74] Ecoinvent, https://ecoinvent.org/.
[75] Moldex3D, https://www.moldex3d.com/.
[76] Technology Transfer System's DDDSimulator, https://www.ttsnetwork.com/en/simulator/.

Valentina Zambrano M.Sc. in Physics (La Sapienza University of Rome, Italy). Ph.D. in Medical Physics (Medical University of Vienna, Austria). Worked at CERN, INFN (Italian Institute for Nuclear Physics), MedAustron (Austria). Taught at the University of Zaragoza (Spain). Currently teaches at the International University of La Rioja (Spain) and is Post-Doc at the Aragon Institute of Technology (Spain) on Reduced-Order-Modeling.

Johannes Mueller-Roemer joined Fraunhofer IGD's Interactive Engineering Technologies department in 2011 after receiving his M.Sc. in Information and Media Technology (with honors and best thesis award) from the BTU Cottbus. He received his doctorate in computer science (summa cum laude) from TU Darmstadt in 2019. His research interests include GPU-accelerated geometry processing, visualization, and simulation.

Michael Sandberg is an Assistant Professor at Aarhus University (Denmark), Department of Mechanical and Production Engineering. His research is focused on the manufacture of polymers and polymer composites, with emphasis on process simulations, flow- and thermo-mechanical modeling. Before academia, Michael worked as an engineer in the wind turbine blade industry.

Prasad Talasila is a researcher with the Cyber–Physical Systems research group in the Department of Electrical and Computer Engineering at Aarhus University. He has research experience in the areas of Cyber–Physical Systems, computer systems engineering including cloud computing, and data storage technologies.

Davide Zanin M.Sc. Eng. Davide Zanin works as a Researcher at the Institute of Systems and Technologies for Sustainable Production of the Department of Innovative Technologies within the Sustainable Production Systems Laboratory (SPS). He has been involved, as Software Engineer, in several European projects regarding Sustainability and Circular Economy. He is teaching assistant in Introduction of Programming course.

Peter Gorm Larsen is both head of section and professor in the Department of Electrical and Computer Engineering at Aarhus University, where he also leads the Cyber–Physical Systems research group as well as the Centre for Digitalisation, Big Data, and Data Analytics (DIGIT) and the AU Centre for Digital Twins. He is the author of more than 200 papers published in journals, books and conference proceedings, and several books and has an h-index at 35.

Elke Loeschner is a Senior Consultant with C-LAB, a joint research and development laboratory operated by Atos and the University of Paderborn, and is engaged primarily in innovative R&D projects. During more than 25 years, Elke Loeschner has gained competence and experience in all phases of the software lifecycle. Her recent topics of interest are Industry 4.0 and decarbonization.

Wolfgang Thronicke Dr., works as Principal Consultant and R&D Manager at the Atos Innovation Lab in Paderborn and is member of the Atos Scientific and Expert Communities. Since 2000 he is working on innovative technologies and trends both in research and customer projects with the current focus on digital twin technologies, AI, platforms, and especially GAIA-X related aspects.

Dario Pietraroia Engineer and MSc. in Science. Has worked as software designer and developer. Collaborates to European research projects. Researcher and lecturer at SUPSI (Scuola Universitaria Professionale della Svizzera Italiana) for the "Analysis and simulation of productive and logistics systems" course. Currently partner at TTS (Technology Transfer System) for the development of simulation models of industrial plants.

Giuseppe Landolfi works at the Institute of Systems and Technologies for Sustainable Production of the Department of Innovative Technologies within the Sustainable Production Systems Laboratory (SPS). He has been involved in several National and European projects as Software Engineer with a supervisory role of a group of developers. He is lecturer in several courses such as Introduction of Programming, Introduction of Object Programming and Industry 4.0.

Alessandro Fontana M. Sc. Eng. Alessandro Fontana is Lecturer & Researcher at SUPSI. Graduated in Material Engineering, he had a six-year industrial experience in Life Cycle Assessment of products and manufacturing processes within Legrand Group. Member of Sustainable Production Systems Lab of SUPSI, his main research domain is sustainability assessment in manufacturing. He teaches Industrial Sustainability at Master and Bachelor level.

Manuel Laspalas Ph.D. in Industrial Engineering. He has been leading ITAINNOVA's research lines on "Multiscale Analysis and Simulation of Unconventional Transformation Processes" (2013–2015) and "R+D+i in Polymers" (2017–2020). He currently performs functions of Project Manager in strategic projects for ITAINNOVA and leads the research line on "Sustainable Material Transformation Processes" (2021–2024).

Jibinraj Antony completed his M.Sc. in Mechatronics from University of Siegen, Germany, with focus on Deep Learning. Presently he is working as researcher at DFKI focusing on the digital transformation of SMEs. He is also part of the Mittelstand Digital Zentrum (MDZ), offering consultation for SMEs in their digitalization and AI implementation efforts.

Valerie Poser is a researcher at DFKI, focusing on software engineering in cloud environments. She received her B.Sc. and M.Sc. in Computer Science from Saarland University in the field of classical AI planning. Her current research activities address machine learning in cloud environments, with the goal to facilitate the application of state-of-the-art ML tools for industrial applications.

Tamas Kiss is a Professor of Distributed Computing at the School of Computer Science and Engineering and Director of the Centre for Parallel Computing at the University of Westminster. He is also Editor in Chief of the Journal of Grid Computing, published by Springer Nature. He attracted over £50 Million research funding and led several national and European research projects.

Simon Bergweiler received a diploma in the field of applied computer science at Trier University of Applied Sciences. Since then, he has been working on national and international projects at the German Research Center for Artificial Intelligence (DFKI). The focus of his current research activities as a senior engineer is on the Cognitive Factory and Human-Centered Subsidiary Industrial Production. In the context of this paper, Machine Learning models as well as the use of methods of AI were considered in terms of Manufacturing as a Service (MaaS).

Sebastian Pena Serna founded clesgo GmbH in 2016, a startup facilitating the democratization of ICT technology for the manufacturing sector. Sebastian is a mechanical engineer, finalizing his Ph.D. at the TU Darmstadt. He worked with ESI Group in the position of Domain Lead Geometry and he was the deputy head of the Competence Center "Interactive Engineering Technologies" at Fraunhofer IGD.

Salvador Izquierdo is M.Sc. Chemical Engineering (2003) and Ph.D. Fluid Mechanics (2008). Currently he is Coordinator of the Multiphysics & Multiscale Simulation Group at ITAINNOVA. He builds and manages industrial R&D projects providing innovative solutions for materials-, manufacturing- and energy-related problems based on scientific machine learning and high performance simulation. Additional interests: Artificial Intelligence; Digital Twins; Innovation Management; and Intrapreneurship.

Ismael Viejo received the M.S. degree in mechanical engineering and M.A.S. in computational mechanics from the University of Zaragoza, Spain, in 2006 and in 2010. In October 2008, he joined as permanent staff of ITAINNOVA. His research focus on advanced modeling of materials including multi-scale material, process simulation, multi-physic simulations, data analysis, optimization numerical tools and ad hoc applications.

Asier Juan B.Sc. in Mechanical Engineering and M.Sc. in Mechanics of Materials. He joined ITAINNOVA in 2017 as R&D Engineer in the Multiphysics and Multiscale simulation division. He has experience in Computational Fluid Dynamics simulations (CFD) in a variety of industrial products and processes and the generation of Reduced Order Models (ROM) for the user assessment, on live evaluations and optimization tasks.

Francisco Serrano is Mechanical Engineer (2016) and obtained his Ph.D. in Biomedical Engineering from the University of Zaragoza in 2021. In 2017 he joined ITAINNOVA in the Materials and Components department. He has carried out modeling and simulation tasks on composite material, fiber-reinforced plastics and plastic injections, as well as programming simulation flows, design of experiments and optimizations.

André Stork head of department Interactive Engineering Technologies at Fraunhofer IGD. Honorary professor at TU Darmstadt. Major research interests: geometry modeling, shape processing, 2D/3D interaction techniques, simulation, scientific visualization. Author of over 200 papers. Member of program committees, reviewer in international conferences, workshops, journals. Lectures: "Computer Graphics III", "Geometric Methods in CAD/CAE" at TU Darmstadt. Member of IEEE, Eurographics, ACM.