Model-Driven Development of Digital Twins for Supervision and Simulation of Sensor-and-Actuator Networks (Extended Abstract)

Gaël Pichot[1], Jérôme Rocheteau[1,2] and Christian Attiogbé[2]

[1]*Icam Ouest, 35 avenue du champ de manœuvres, 44470 Carquefou, France*
[2]*LS2N, 2 chemin de la Houssinière, BP 92208, 44322 Nantes Cedex 3, France*

Digital Twins are defined as virtual counterparts of physical objects, processes or systems that make it possible to supervise, control and simulate these objects, processes or systems. They are made of sets of software components interacting with physical devices. Unfortunately, there is currently no real standard neither on the very definition of Digital Twins nor on their method of development. This leads to empirical development of Digital Twin applications. In a previous work we had, firslty, proposed a meta-model able to represent Sensor-and-Actuator Networks and had, secondly, defined a model transformation that generates IoT applications for the supervision and the control of such networks. In this paper we extend this previous work: Firstly, the Sensor-and-Actuator-Networks meta-model is improved in order to better represent computations within such networks. This allows us to declare total or partial simulations of such networks. Secondly, we extend the model transformation in order to generate data-driven simulation programs that can interoperate with the IoT application for supervision-and-control of a given Sensor-and-Actuator Network. This can be seen as a digital twin of a Sensor-and-Actuator Network as it makes it possible to control the actuators by the means of computation units that provide their setpoints from the own measurements of the sensors of this network. Finally, we model and develop a digital twin of a connected plastic extruder.

Extension of the SAN meta-model The meta-model[1] of Sensor-and-Actuator Networks, or SAN for short, has been designed (1) to represent sensors, actuators and computation units of sensor-and-actuator networks and (2) to encompass concepts from other meta-models like SensorML, SOSA, SSN and SMM. The latter address the issue of modeling sensor networks. It merely allows us to declare elements of SAN models by specifying what kind of data is exchanged with such networks. It doesn't allow us to define these elements i.e. how data is processed. Moreover, a model-to-text transformation[2] makes it possible to obtain an IoT-based supervision-control-and-data-acquisition application, or SCADA for short, from SAN models by generating a set of HTTP requests that configure a previously developed IoT platform called

STAF 2022 Workshop - International Workshop on MDE for Smart IoT Systems
© 2022 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0)
CEUR Workshop Proceedings (CEUR-WS.org)

[1] The SAN meta-model is currently provided by the means of the following Xtext grammar: https://git.icam.fr/jerome.rocheteau/san/-/blob/main/fr.icam.san/src/fr/icam/san/SensorActuatorNetwork.xtext.

[2] This model transformation generates the following Java program that benefits from the EMIT Java HTTP client library: https://git.icam.fr/jerome.rocheteau/recyplast/-/blob/main/src/test/java/fr/icam/recyplast/Extruder.java.

CEUR-WS.org/Vol-3250/messpaper4.pdf

EMIT[3] once executed.

We extend this SAN meta-model in two ways. Firstly, we introduce another kind of *process triggers*, called *tasks*, that represent computations repeatedly launched at fixed-rate. This addition completes the SAN modelling of *process triggers*; *processes* represent computations and *triggers* represent when and how the latter are computed. More than anything, *tasks* make it possible to represent computations of simulations programs that continously listen to exchanged data into networks and that can send valuable data back at defined time spans. Secondly, we introduce another way to bind values to *process parameters* while declaring *triggers*. In fact, *processes* are (parametric) components with formal parameters and *triggers* are (parameterized) components with real parameters i.e. values. Values can be provided either as raw *values* (i.e. string, integers, floating point numbers, etc) as previously in SAN modelling, or as *measurements* i.e. references to given data channels. It then makes it possible to bind *process parameters* with values that can vary during the time but which type and source are ensured by the model compliance with the system.

Generation of simulation programs from SAN models We extend the model transformation from SAN models to *tasks* by the means of the Xtext code generation feature. It generates a Java abstract class per *task*[4]. The latter consists in a Java class that implements both interfaces Runnable and MQTTCallback. The Runnable interface is used to process the collected data and fire the output results. The MQTTCallback interface is used to collect the data. In fact, the model transformation associates a Java variable per *parameter* of the *trigger process*. This Java variable is initialized according to the corresponding *argument* of its associated *parameter*: it is initialized once with the specified value if this *argument* consists of a *value*; it is initialized each time a message arrived on the topic that corresponds to the *measurement* if this *argument* consists of a *measurement*. The way data is processed and results are computed is delegated to the inherited classes of the Java abstract class that correspond to the *task*. For instance, this makes it possible to implement several simulation algorithms from a single *task*.

This model transformation extension leads to a « 2-level application ». The first level is defined by the SCADA application whereas the second level is defined by the data-driven simulation programs. In fact, such simulation programs are based on and driven by the data that is exchanged and collected between sensors, actuators and computational units within the network.

Development of the digital twin of a plastic extruder We had starting updating the SAN meta-model by modelling the plastic extruder used for the Recyplast-Demo project at the Icam school factory[5]. An extruder is a machine that transforms input plastic balls into output plastic products of various shapes depending on the trailing equipment: the die. It consists (1) in feeding the machine with input material, (2) in pushing the material along the extruder by

[3] The EMIT platform consists of web-services that allows us to edit and manage MQTT clients and callbacks. Its repository is publicly available at https://github.com/jeromerocheteau/emit.

[4] The generated Java program for the *task* declared in the following extruder SAN model can be found at: https://git.icam.fr/jerome.rocheteau/recyplast/-/blob/main/src/test/java/fr/icam/recyplast/ExtruderSimulator.java.

[5] The SAN model of the extruder is available: https://git.icam.fr/jerome.rocheteau/recyplast/-/raw/main/src/main/resources/extruder.san.

the means of a motorized endless screw, (3) in heating this material along the extruder by the means of several heating barrels equipped with a heating resistance actuator and a temperature sensor such that the input material changes from solid state to liquid state and (4) in shaping this output material thanks by the means of the die. The latter has a pressure and a temperature sensor that monitor the output material. The screw motor speed and the barrels temperatures are servo measurements i.e. setpoints are automatically adjusted according to their respective measurements. A torque measurement of the screw motor is also provided by a dedicated sensor. In practice, the motor speed is surprisingly set to a constant value, even if it could be adjust at runtime, as well as the setpoints of the heating barrels. Moreover, the setpoints of the heating barrels are all set to the same value, even if it could be possible to provide different setpoints to each heating barrel. These setpoints of motor speed and barrel temperature are mainly specified by the plastic material providers. The SAN model transformation allows us to configure the EMIT platform such that the result corresponds to the SCADA application of the plastic extruder[6].

The Recyplast-Demo project aims at using recycled plastic in extrusion process. This leads to feeding the extruder with material which properties can vary during the time whereas properties of current plastic material used in extrusion doesn't. The underlying challenge then consists in ensuring a constant quality of the output plastic products despite the variable properties of the input materials. The investigated solution corresponds to slave the screw motor speed setpoint and the heating barrels temperature setpoints from the material pressure and temperature measurements of the die. In fact, the output products quality are correlated to the material pressure and temperature measurements of the die. This leads us to develop a program that monitors the different measurements at runtime – screw motor speed and torque, heating barrels temperature, material pressure and temperature – and that provides suggestions for the screw motor speed and heating barrels temperature setpoints. Adding such a program to the SCADA application directly corresponds to a Digital Twin, or DT for short, of the extruder. DTs differ from SCADA applications in the way that a DT is not just defined by a bidirectionnal exchange of data between the DT and its physical system: it also includes algorithms, like simulations such that it can be seen as the next level simulation based on collected data which is obviously the case in this context.

In this work[7], we prove the feasability of developing IoT-based full Digital Twins from models of Sensor-and-Actuator Networks i.e. enabling both the Supervision-Control-And-Data-Acquisition feature and the data-driven simulation one by the means of a IoT application. This has been achieved by a tooling based on the Xtext support for language engineering that relies on the Ecore meta-model and the EMF framework. Moreover, the way data-driven simulation programs are generated offers a valuable balance between the support provided by the code generation and the flexibility of finely customizing the simulation algorithms.

[6] A bridge from OPC-UA to MQTT has been developed in front of the extruder PLC in order to ensure communications between the extruder PLC and the EMIT platform.

[7] This work has received fundings under the Regional Council of Pays-de-la-Loire and the European Regional Development Fund (ERDF) programs for the Recyplast-Demo project under the grant agreement n°2021-15465.