Summary of Towards a Reference Architecture for Leveraging Model Repositories for Digital Twins

September 2024

The paper presents a reference architecture for building digital twins (DT) that leverages model repositories. The model repository is introduced as a bridging component between the design-time model and the DT runtime behaviour. The design-time model is a logical model that describes different structural and behavioural aspects of the DT. A model repository is a storage unit containing the model definitions of other components of the DT that operate the physical devices being used to mimic their behaviour. The authors argue that designing DTs using model-driven engineering (MDE) is a critical enabler in handling the complexity of the DT engineering process, as models are considered static entities, mainly used as blueprints in the design phase, which are neglected in the later life-cycle phases.

In their previous works, the authors developed and prototyped an implementation of a multi-domain framework that links hardware, software, runtime, and simulation data through interlocked execution loops to automatically identify the system's state. These prototypes allowed the enrichment of the logical models with runtime information but did not enhance the running system with structural and behavioural information from the logical modes. Therefore, the reference architecture proposed is presented to address such shortfalls.

The proposed architecture aims to integrate models stored in model repositories, utilize existing structural and behavioural information, leverage their capabilities, and explore the possibility of enhancing operational systems with a new reflection layer by incorporating design-time models as additional sources of knowledge for the DT. Their architecture offers the advantage that the system can have many components that do not necessarily need to be visualized, even though one element of the DT still confirms and stores the data about their functioning. If the DT operator has visibility that such components exist, then through the proposed architecture, the data collected through such components can be invoked and displayed if needed. This solution was possible through the use of a layered approach. The architecture contains three layers, namely, the information, the operation, and the communication layers. The information layer is the top layer that displays the visualization of the overall DT system. The operation layer acts as the data collection layer. Finally, the communication layer acts as middleware between both previously mentioned layers, offering the concept of a reflection engine that forwards processed design time model information to the running system. This last layer is the novelty contribution proposed by the authors.

The proposed architecture from the paper is successfully demonstrated in a practical application by executing the model repository and the reflection engine on a production process gripper that carries items from one physical production entity to another. Based on the proposed architecture, the gripper reflects on its potential behaviour to find the production space where a particular item must be placed next. The next state is predicted by identifying potential states from design-time models and refining them based on historical runtime model data. This successful demonstration reassured the feasibility and potential benefits of the proposed architecture in real-world applications.

In future works, the authors anticipate studying alternative architectures and variations of the presented architecture while considering different architectural drivers, such as the quality attributes of DTs. The long-term goal is to create an architectural style language for DTs that provides customization possibilities during design time and a high degree of automation in the development process.