Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

A DIGITAL TWIN APPROACH TO SUPPORT THE EVOLUTION OF CYBER-PHYSICAL SYSTEMS

Joost Mertens[1,2]

[1] Faculty of Applied Eng.: Electronics and ICT, University of Antwerp, Antwerp, BELGIUM
[2] Flanders Make at University of Antwerp, University of Antwerp, Antwerp, BELGIUM

ABSTRACT

A digital twin is a virtual representation of a real-world system of which data is continually collected. The data is fed back to the digital twin such that it may mirror system it reflects. In exchange, its users gain a variety of services. A key aspect in all digital twins is that of evolution, and this in two different ways. The first way deals with the mirroring of the real-world system when it evolves. The models in the digital twin should reflect that evolution. The second way deals with the evolution of the services in the twin itself. Like any other software system, the purpose and requirements of the digital twin evolve over time. In this thesis the focus is on a subset of issues encountered in these two types of digital twin evolution. It provides techniques that aid digital twin developers with the evolution of their digital twin.

1 INTRODUCTION

A digital twin is a virtual representation of a real-world system of which data is continually collected. Although various definitions with intricate differences exist, in practice, the essence of a digital twin is a digital model representing a real-world system, data gathered during the operation of that real-world system, and a way of adjusting/updating the model based on that data if deemed necessary (Wright and Davidson 2020). The term real-world system is kept generic on purpose, as the digital twin concept is found in all kinds of domains, including but not limited to manufacturing, healthcare, urban/city planning, maritime and shipping, aerospace and automotive fields (Botín-Sanabria et al. 2022; Semeraro et al. 2021).

The key novelty in digital twins compared to traditional modeling and simulation is that the models in the twin mirror the real-world instance, not only during design time, but also over the entire lifecycle of the real-world instance. As a consequence, as changes occur to the real-world system and/or its environment the digital twin is required to evolve. For example, when a new sensor is added to the real-world system, the model of that sensor should also added to the digital twin. When those components were developed using model-driven engineering, models of these updated components exist and integration in the digital twin is straightforward. However, not all cases of evolution of the real-world system are engineered changes, particularly those changes pertaining to the physical aspects. As the real-world system ages, it suffers wear and tear, parts break and get replaced, sometimes by suboptimal replacements. Such changes are usually not documented, yet they can have a considerable impact on the system's operation. For these cases, how to update the digital twin is less clear-cut.

Besides the evolution driven by the real-world system, there is also the less frequently considered evolution of the digital twin's requirements and/or purposes. As the twin itself is a software system, these change over the duration of its lifetime. A twin might gain a new service, or an existing service might be updated. In contrast to the ideal, all-encompassing, view of a digital twin, in practice they are complicated systems that only address the minimally required set of aspects of the real-world system that are necessary to provide its services. Hence, introducing a new purpose usually entails addressing a new aspect of the real-world system. Each aspect relates to a set of properties of interest of the system, that are potentially coupled to the properties of interest of another aspect. For example, a twin that previously provided movement optimization, but now is also required to provide energy optimization is going to evolve to contain new software components that address this energy optimization. Such components will undoubtedly be interwoven with the existing movement optimization, as movement requires energy. As a result, when adding, removing or updating the functionality of the twin, twin developers are required to think about the composition of the building blocks/components that constitute the digital twin.

This thesis tackles a subset of issues encountered in these two types of digital twin evolution. It provides reusable techniques that aid digital twin developers with handling the evolution of their digital twin. Coupling this back to the title, we note how the title gains two meanings. On the one hand, the digital twin reflects the real-world system's evolution. On the other hand, the digital twin, as extension of the "Cyber" aspect of the Cyber-Physical System, is evolved itself.

2 CONTRIBUTIONS

The main contributions of this thesis can be summarized as follows:

1. **Provide a notation and set of patterns supporting digital twin evolution and composition.**
   This notation and set of patterns allow digital twin developers and stakeholders to reason about the digital twin's evolution and composition at a higher level of abstraction. The notation makes influencing or competing components explicit, allowing developers to make trade-offs and compare alternative implementations of the digital twin composition with each other.
2. **Demonstrate the reuse of model validation methods for divergence detection at runtime.**
   We demonstrate how model validation methods that are traditionally used during model development can be reused at runtime to detect divergences between a real-world system and its digital twin.
3. **A demonstration of fault-localization in twin systems using state-of-the-art time-series classification methods combined with training data stemming from the digital twin.**
   We used state-of-the-art time series classification methods to perform fault localization for twinned systems. The digital twin was used to provide the large amount of training data necessary to train the fault localizer.
4. **A system variant detection scheme to be used in the continuous evolution process as implemented in a digital twin.**
   We developed a workflow that allows system operators to detect system variants during the operation of the system. These variants are used to parametrize digital twins, which in turn are used to support the continuous evolution of the system.
5. **A Domain-Specific Language (DSL) for model validation.**
   We created a domain-specific language to be used for the definition of model validation experiments. Such experiments compare measured results on a real system with a simulated model response and yield a metric of similarity. Bolstered with model-to-code and model-to-text transformations, the DSL makes defining such experiments more feasible for non-experts, e.g. digital twin developers.
6. **Lab-scale gantry crane with digital twin.**
   As a case study for our work, we developed a lab-scale gantry crane with an accompanying digital twin to demonstrate our techniques and findings.

REFERENCES

Botín-Sanabria, D. M., A.-S. Mihaita, R. E. Peimbert-García, M. A. Ramírez-Moreno, R. A. Ramírez-Mendoza and J. d. J. Lozoya-Santos. 2022. "Digital Twin Technology Challenges and Applications: A Comprehensive Review". *Remote Sensing* 14(6) https://doi.org/10.3390/rs14061335.

Semeraro, C., M. Lezoche, H. Panetto, and M. Dassisti. 2021. "Digital twin paradigm: A systematic literature review". *Computers in Industry* 130:103469 https://doi.org/https://doi.org/10.1016/j.compind.2021.103469.

Wright, L. and S. Davidson. 2020. "How to tell the difference between a model and a digital twin". *Advanced Modeling and Simulation in Engineering Sciences* 7(1):1–13.