



Experimentable Digital Twins in the Loop

Jorge Luis Jiménez Aparicio and Jürgen Roßmann

Abstract

Digital twins (DTs) have become prominent in the digitalization era due to the advantages they bring to fields of application like Industry 4.0. In this contribution, we briefly go over the role of experimentable DTs in a V-model-based development cycle of a system, which includes XiL methods. We also provide a proof of concept (in the form of two application examples) of their use in the model-in-the-loop development stage, where the controller model of the system is implemented in real-time capable hardware.

Keywords

Digital twins • Simulation • Model-in-the-loop

1 Introduction

Digital twins (DTs) offer several advantages in the Industry 4.0 era, where a DT follows the life cycle of its physical twin to provide better support and greater insight for tasks like analysis, optimization, maintenance, upgrading, and updating, among others. Here, we use the definition introduced in [16] for the so-called “Experimentable Digital Twins” (EDT) interchangeably with a DT. Virtual Testbeds (VTBs) [14] are advanced 3D simulation

J. L. Jiménez Aparicio (✉) · J. Roßmann

Institute for Man-Machine-Interaction, RWTH Aachen University, Ahornstrasse 55, 52074 Aachen, Germany

e-mail: jimenez@mmi.rwth-aachen.de

URL: <https://www.mmi.rwth-aachen.de>

J. Roßmann

e-mail: rossmann@mmi.rwth-aachen.de

© The Author(s) 2025

S. Ihlenfeldt et al. (eds.), *Annals of Scientific Society for Assembly, Handling and Industrial Robotics 2023*, https://doi.org/10.1007/978-3-031-74010-7_17

environments that represent all required parameters of the application to simulate a realistic behavior. These VTBs are enhanced with the use of DTs, as it allows an even more accurate description of the agents in the simulated environment.

Before having a functional physical twin, and its corresponding DT, all products or systems have to go through a development cycle. A popular model-based development cycle is the V-model, which was first introduced for software development in [4], was later adopted for mechatronic systems in the VDI 2206 guideline [21], and has continued to evolve to be applied in diverse sectors. One iteration of this model includes development steps where elements of the system are tested in a loop that includes parts of a greater system, or components of the planned target system itself, to have a faster and cheaper overall development process. These are commonly known as X-in-the-loop (XiL), where normally *X* stands for model, software, or hardware.

DTs and VTBs could be implemented in this development cycle to further increase the benefits inherent to it. This applies generally to many types of development cycles, in which an improved design, early detection of issues, increased efficiency, and better collaboration can be achieved [15, 19] when compared to development cycles that rely solely on physical prototypes.

In this contribution, we aim to analyze the feasibility of integrating DTs into the V-model-based development cycle of a product, which also includes XiL methods. To achieve this, we provide a brief overview of every XiL iteration with DTs and present two application examples as the proof of concept for the use of DTs with model-in-the-loop (MiL) development, along with the use of real-time capable hardware as part of the life data exchange schema.

2 State of the Art

There are several approaches of modeling DTs, most commonly including physics-based, data-driven, hybrid, and reduced-order modeling [2, 9, 12]. However, the majority of these are not based on the V-model, and only a few take into account XiL methods. For instance, the proposed big data-driven model in [17] follows a cyclical circle pattern.

The work in [10] focuses on creating a high fidelity twin of hardware/software interaction in a MiL setup. However, the framework is limited to only MiL from the XiL methods and it is not based on the V-model for development. Furthermore, the work does not involve real-time capable hardware, which is typically seen in hardware-in-the-loop (HiL) setups, such as the one in [7].

Another study, proposed by the authors in [18], introduces a twin-driven product design framework that employs DTs in the V-model. However, the stages of the V-Model in this framework are more oriented towards overall concepts such as market analysis, customer requirements, and sales among many others; rather than product development with the use of XiL methods.

In the publication [22], DTs are defined as a combination of models and data that creates a virtual copy of a system. The authors explore the use of DTs in engineering dynamic applications and provide an overview of integrating them into the V-model development cycle. They propose a “W-model” for product design, where the initial stage is done virtually with a virtual model, and the DT is only introduced in the later stage of development. From a virtual prototype, they start building a DT. In contrast, our perspective advocates the use of DTs from the outset of the development cycle. We believe that developing DTs from the beginning provides increased benefits such as improved collaboration, early issue detection, and more efficient designs.

The work presented in [20] introduces the Hybrid Testbed as a novel holistic approach to integrate DTs and VTBs into a XiL-based development cycle. The authors propose a modular and scalable framework that includes real data to improve the accuracy and effectiveness of the system. Although our work differs in the sense that we use the V-model as the basis for the development cycle and propose the use of DTs at a more granular level, we share the same goal of achieving a seamless transition from the design stage to the final product. Consequently, our work could serve as a proof of concept for the use of DTs and VTBs in a model-in-the-loop (MiL) setup. Overall, the concepts proposed in [20] align with our approach and provide valuable insights into the integration of DTs in the XiL-based development cycle.

3 Overview of the Model

The V-model is a well-known model-based development cycle used for developing products and systems. It consists of two parallel branches, one for design and one for testing. As shown in Fig. 1, the DT can be integrated into several stages of the V-model as a central component. This means that the DT should be included in these development stages of the model. There are mainly two types of development that can be applied in this context. The first type is for the development of a final product, which is considered as a whole once it is created. Here, the DT should be created before or, at the latest, together with the physical twin. The second type is for the development of a product that is part of a larger system, for example, designing the steering wheel of a vehicle. In this case, DTs of the larger system or its components can already be employed during the creation of the product’s DT. In both cases, VTBs with DTs can be used to enhance the development and testing process.

In the following, we give an overview of the various XiL methods defined in [1, 3, 5, 8, 11] and their alignment with our approach of using DTs and VTBS.

Model-in-the-loop (MiL): All components involved are executed in the form of simulation models in development computers. These components are usually not identical to the target system. If a new element of a larger system is being developed, using the DT to represent said system can already provide more accurate integration results of the model and speed up its development. If a new system is being developed, this would be the first step to create

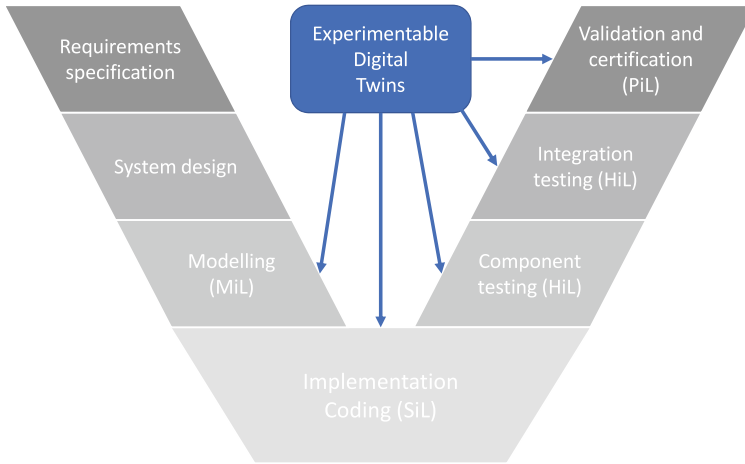


Fig. 1 V-model diagram of a product with DTs integration. DTs are integrated into all XiL methods for development, together with VTBs for testing

a DT of the target system. Our example applications shown in Sect. 5 are based on this part of the development cycle. *Software-in-the-loop (SiL)*: Software (e.g. a control algorithm) is designed and developed with the target hardware in mind. It is usually executed on a development computer and connected to the real process or parts of it. Here, the software can be implemented in a DT of the target hardware (e.g. an embedded system) for a more accurate behavior. This arrangement could be connected to a DT of the larger target system or a VTB representing the real process. *Hardware-in-the-loop (MiL)*: This improves on SiL by implementing the developed software on the target hardware and testing it with a simulation model of the real process running on a development computer. Here too, the larger system should be represented by its DT or the real process can be simulated in a VTB. *Product-in-the-loop (PiL)*: A more generalized analogy of the specific term used in the automotive industry Vehicle-in-the-loop (ViL). Here, the entire target system is tested and validated. The DT of the developed product is now part of the larger system's DT and they are tested as a whole. Testing can be done with the help of VTBs, which in turn can have more DTs. If the final product is not part of a larger system, the DT is finalized here and can keep being updated together with the physical twin during its life cycle.

4 Methodology for the MiL Implementation with Real-Time Capable Hardware

In an MiL setup, the DT is used to represent the behavior of a physical system or component in a virtual environment. This virtual environment typically includes other DTs, components, and systems with which the component or system being modeled interacts. By replacing

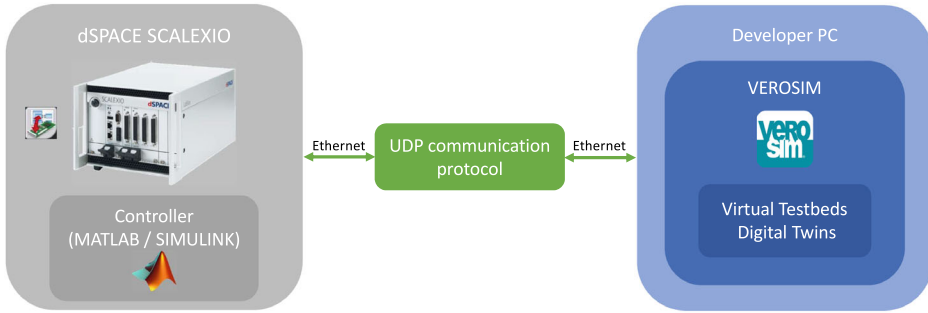


Fig. 2 Concept for the architecture implemented to enable MiL with DTs and VTBs and real-time capable hardware. Examples of the simulation software (VEROSIM) and hardware (Scalexio) are given. Communication between them is handled by an UDP protocol to support real-time communication

the physical system or component with its virtual representation, the DT can be integrated into the MiL setup, enabling simulation and testing of the system or component's behavior in a controlled environment. This methodology allows for a more efficient and accelerated development process, reducing the need for physical prototypes and field testing.

In this section, we describe the concept used to implement the application examples in Sect. 5. Our goal was to develop a modular and reusable architecture for MiL simulations in VEROSIM, our physically accurate 3D simulation software, by incorporating real-time capable hardware. As shown in Fig. 2, this architecture utilizes specific real-time hardware with its standard connection port, but it can be adapted to work with other real-time hardware. The DT in development is simulated within VEROSIM together with other DTs and the VTB.

To enable communication between the DT and the hardware, we have implemented an interface module in VEROSIM, which can be customized to handle different I/O data structures and port connections. We chose to utilize the UDP transport protocol due to its ability to provide the necessary speed for real-time applications, despite its lower reliability compared to the TCP protocol [13]. While there may be a risk of data loss during transfer, this can be mitigated by implementing a more robust controller for the system.

5 Application Examples

This section presents two application examples that demonstrate the use of DTs in MiL. The first one involves a turntable system where we control the angular velocity based on user input. The second involves the Gemini VIII spacecraft, where we control the rendezvous and docking with the Agena target.

Following the architecture shown in Fig. 2, we used the real-time capable hardware Scalexio from dSPACE to run the controller of the target system. The controller model is

created in Matlab/Simulink and loaded into the Scalexio. As stated before, we used the UDP connection protocol for the real-time communication between the Scalexio and VEROSIM. Finally, a DT of the target system was running in VEROSIM, whose controlling parameters were given by the Scalexio. Depending on the application, this DT is also deployed within a VTB for testing and validation.

Before setting up the connection with the Scalexio for the application examples, we conducted tests with a simple plant and controller in Matlab/Simulink, with VEROSIM serving as an echo server. The tests revealed no difference in performance compared to running the system within Simulink. However, when we performed the same test with the Scalexio as the controller, we measured a significant behavior change, likely due to information loss inherent to the UDP protocol and minor communication delays. This can be seen in Fig. 3. As a result, we had to make adjustments to the controller parameters when implementing the application examples to compensate for this. We achieved this after some iterations of the parameters, which we rapidly changed and tested within the explained setup thanks to the flexibility provided by the use of DTs instead of physical prototyping.

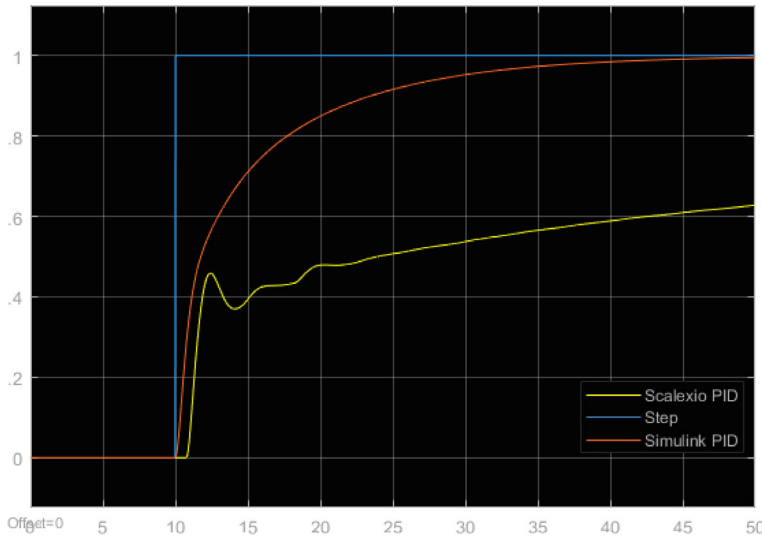
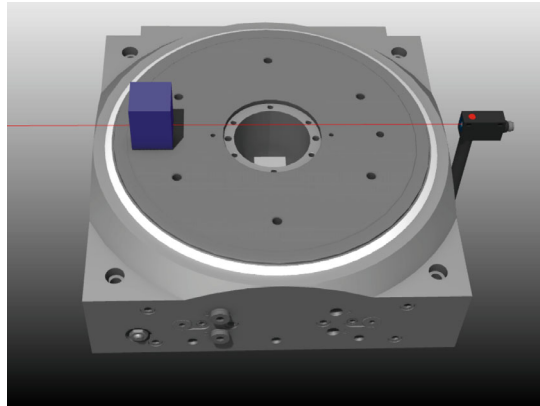


Fig. 3 Step response of a PID controller with a plant in Matlab/Simulink. Reference step response (blue), response with the Simulink controller (orange), and response with the Scalexio controller (yellow)

Fig. 4 Turntable in the 3D simulation software VEROSIM



5.1 Turntable

The turntable system consists of a turning plate mounted on a square mount (see Fig. 4). Our model includes a laser sensor that measures the presence of an object on the plate. This allows the calculation of an angular velocity error, which is the input of the controller. The Scalexio has a PID controller that will output the torque force to be applied by the turntable's motor to reduce the angular velocity error to zero. The target angular velocity is set by the user.

The turntable setup is shown in Fig. 5. To the left, the behavior of the target's angular speed (cyan), the object detection (orange), and the torque response (yellow) are shown. A small delay in the response of the torque can be observed, likely due to delays in the data transmission. Nevertheless, the PID can successfully control the angular velocity of the turntable. This setup still had room for improvement in performance but served as a first step to test the combination of the Scalexio and a MiL development stage.

5.2 NASA's Gemini Program

As part of NASA's Gemini program [6], the first rendezvous, docking, and extravehicular activity mission was conducted. This mission planned to launch, rendezvous, and dock the Gemini VIII spacecraft with the Gemini-Agena target vehicle (GATV). Figure 6 shows the DT of the Gemini VIII used for this test, together with the DT of the GATV, present in the VTB. Components for sensor and rigid body simulation as well as for orbital dynamics are used within the VTB to simulate a close-to-reality scenario for the DT under development.

The external guidance, navigation, and control (GNC) system was provided by one of our industry partners, Jena Optronik. It was implemented in Matlab/Simulink and simulated the space environment and Lidar measurements for a standard rendezvous and docking

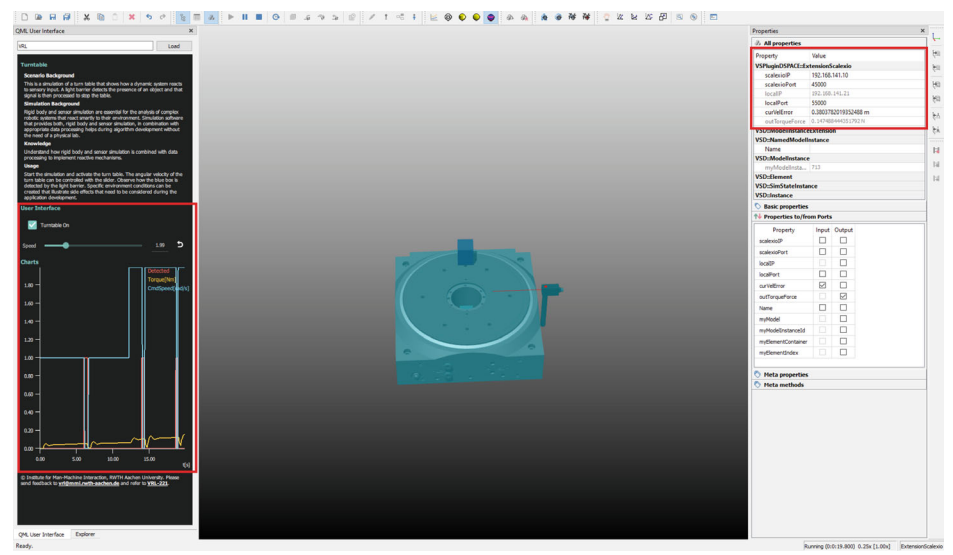


Fig. 5 Simulation of the Turntable system in VEROSIM. Left is the user interface where the target angular velocity can be set and some system variables are shown: target’s angular speed (cyan), object detection (orange), and torque response (yellow). Right are the properties of the interface module where the IP, port, and I/O values of the UDP connection are displayed

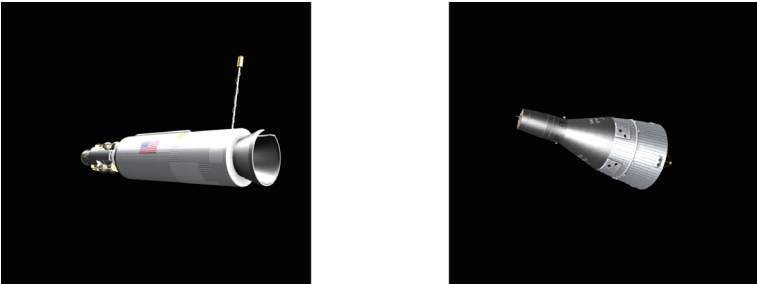


Fig. 6 DTs of the GATV (left) and GEMINI VIII (right)

mission. From this GNC, we were able to extract the necessary control system to implement in the Scalexio, which is depicted in Fig. 7. The control system is composed of the UPD communication blocks, a PD controller, a saturation block, a PWM generator, and low pass filters. The only differences between this version and the one loaded in the Scalexio, are the UDP communication blocks from Simulink, which are replaced by UDP blocks provided by dSPACE for compatibility with the Scalexio.

As the controller was initially designed for a chaser spacecraft docking to the International Space Station (ISS), it required adjustments in the hyperparameters, primarily scaling, to control the thrusters of the Gemini VIII properly. Further modifications were necessary to

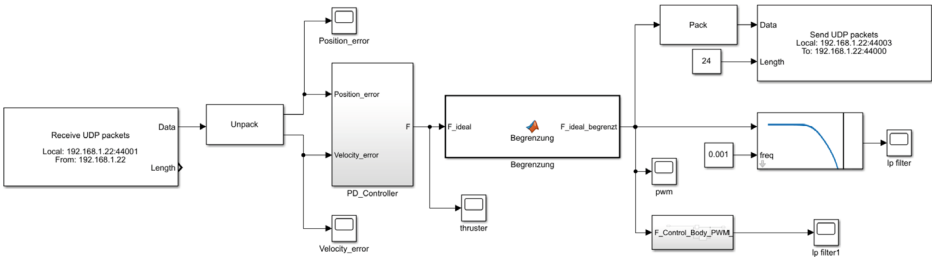


Fig. 7 Simulink model of the controller extracted from the GNC system provided by Jena Optronik. The controller was complemented with UDP communication blocks. The main component is contained in a black box named “Begrenzung”

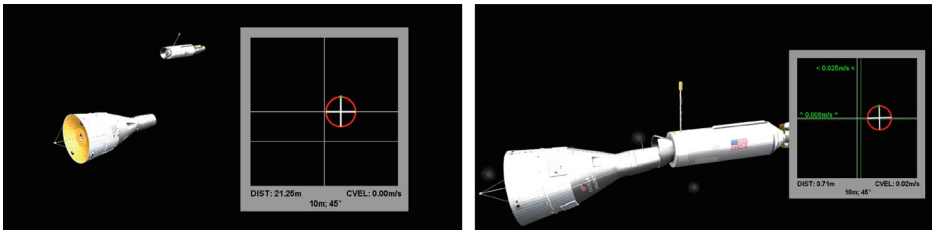


Fig. 8 Initial position of the GATV and GEMINI VIII’s DTs in the VTB in VEROSIM (left). Final position of the rendezvous and docking maneuver (right)

account for the time delays during communication. However, due to the constant communication of the DT with the control’s model, the modification process was done with increased efficiency.

In the left section of Fig. 8, the initial position of the Gemini VIII and the GATV is shown, with a distance of 21.25 m between them. The controller receives the current position and velocity error of the Gemini VIII with respect to the target and outputs the x,y,z thrust force vector. This is received by the Gemini VIII’s DT, which operates the thrusters accordingly. Although the controller was not thoroughly optimized, the spacecraft successfully reaches the target, as observed on the right section of Fig. 8, where the distance has closed to 0.71 m. This helps demonstrate the successful use of DTs in an MiL setup with the use of real-time capable hardware. Finally, this particular experiment highlights the benefits of using DTs in product development instead of physical prototyping. Running these experiments with physical prototypes would be extremely expensive and dangerous, if not impossible.

An additional approach of a mix between MiL and HiL was also conceptualized. Here, an embedded system specialized in space application (Jena Optronik’s DevBoard) handles the movement of the Gemini VIII and gives its pose information to VEROSIM, where the VTB with the DT of the GATV is implemented. The pose is given to the GNC inside the Scalexio and a Thrust vector is returned to VEROSIM and passed on to the embedded system. Although this configuration is left for future work, it is mentioned here as an additional benefit

to using DTs in MiL/HiL. This approach is useful for developers to test their GNC models running in the Scalexio together with the DevBoard in a HiL setup.

6 Conclusions

This contribution has shown a proof of concept for the integration of DTs at the MiL development stage of a system, together with the use of real-time capable hardware to run the controller of said system. This could be extended to other stages of the XiL development, where other parts of the system can be tested with the help of DTs and VTBs. The integration of DTs increases the efficiency, behavior accuracy, and predictability of the performance of the final product increasing the inherent benefits of development with the V-model. This is especially important in fields of application where running real-life development, testing, and validation can be very expensive and dangerous (e.g. in space).

A limitation to this approach is the necessity of DTs of other VTB's components (e.g. sensors) or a larger system (e.g. vehicles) to have closer-to-reality results. The compatibility of such DTs could be a problem if the development does not follow a scalable, modular approach like the one presented in [20].

This study is in its initial stages and only examines the use of DTs in the MiL stage, as demonstrated by the two examples presented. Further research is required to assess the potential benefits of integrating DTs and VTBs into other stages of the V-model and XiL development cycles. This future work should also determine the applicability of DTs in each stage and identify which stages stand to benefit the most from their use.

Acknowledgements This work is part of the project “ViTOS-3”, supported by the German Aerospace Center (DLR) with funds of the German Federal Ministry of Economics and Technology (BMWi), support code 50 RA 2120.

References

1. Basic, M.: On hardware-in-the-loop simulation. In: Proceedings of the 44th IEEE Conference on Decision and Control, pp. 3194–3198 (2005)
2. Bao, J., Guo, D., Li, J., Zhang, J.: The modelling and operations for the digital twin in the context of manufacturing. *Enterprise Inf. Syst.* **13**(4), 534–556 (2019)
3. Berg, G., Nitsch, V., Färber, B.: Vehicle in the loop. In: Handbook of Driver Assistance Systems, pp. 1–9. Springer International Publishing (2015)
4. Adolf, P.: Böhrh and Wolfgang Dröschel. Der Standard für die Softwareentwicklung mit Praxisleitfaden. Oldenbourg Wissenschaftsverlag, Das V-Modell (1993)
5. Demers, S., Gopalakrishnan, P., Kant, L.: A generic solution to software-in-the-loop. In: MIL-COM 2007 - IEEE Military Communications Conference, pp. 1–6 (2007)
6. Hacker, B.C., Grimwood, J.M.: On the Shoulders of Titans: A History of Project Gemini

7. Han, J., Hong, Q., Feng, Z., Syed, M.H., Burt, G.M., Booth, C.D.: Design and implementation of a real-time hardware-in-the-loop platform for prototyping and testing digital twins of distributed energy resources. *Energies* **15**(18) (2022)
8. Isermann, R., Schaffnit, J., Sinsel, S.: Hardware-in-the-loop simulation for the design and testing of engine-control systems. *Control. Eng. Pract.* **7**(5), 643–653 (1999)
9. Kapteyn, M.G., Knezevic, D.J., Huynh, D.B.P., Tran, M., Willcox, K.E.: Data-driven physics-based digital twins via a library of component-based reduced-order models. *Int. J. Numer. Meth. Eng.* **123**, 2986–3003 (2020)
10. Morris, B., Cheng, T., Koontz, C.: A model-in-the-loop technique for digital twin simulations (MIL-DT): a generalized approach for simulating embedded software, 2022-11
11. Plummer, A.R.: Model-in-the-loop testing. *Proc. Inst. Mech. Engin., Part I: J. Syst. Control Engin.* **220**(3), 183–199 (2006)
12. Qi, Q., Tao, F.: Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison. *IEEE Access* **6**, 3585–3593 (2018)
13. Rind, A.R., Shahzad, K., Abdul Qadir, M.: Evaluation and comparison of tcp and udp over wired-cum-wireless lan. In: 2006 IEEE International Multitopic Conference, pp. 337–342 (2006)
14. Rossmann, J., Schluse, M., Rast, M., Atorf, L.: E-robotics combining electronic media and simulation technology to develop (not only) robotics applications. In: *E-Systems for the 21st Century*. Apple Academic Press (2016)
15. Schleich, B., Anwer, N., Mathieu, L., Wartzack, S.: Shaping the digital twin for design and production engineering. *CIRP Ann. Manufact. Technol.* **66**, 141–144 (2017)
16. Schluse, M., Priggemeyer, M., Atorf, L., Rossmann, J.: Experimentable digital twins—streamlining simulation-based systems engineering for industry 4.0. *IEEE Trans. Indust. Inf.* **14**(4), 1722–1731 (2018)
17. Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., Sui, F.: Digital twin-driven product design, manufacturing and service with big data. *Int. J. Adv. Manufact. Technol.* **94**, 3563–3576 (2018)
18. Tao, F., Sui, F., Liu, A., Qi, Q., Zhang, M., Song, B., Guo, Z., Lu, S.C.Y., Nee, A.Y.: Digital twin-driven product design framework. *Int. J. Product. Res.* **57**(12), 3935–3953 (2019)
19. Tao, F., Zhang, H., Liu, A., Nee, A.Y.: Digital twin in industry: State-of-the-art. *IEEE Trans. Indust. Inf.* **15**(4), 2405–2415 (2019)
20. Thieling, J., Rossmann, J.: Scalable sensor models and simulation methods for seamless transitions within system development: From first digital prototype to final real system. *IEEE Syst. J.* **15**(3), 3273–3282 (2021)
21. VDI. 2206: Entwicklungsmethodik für mechatronische systeme. Verein Deutscher Ingenieure 2 (2004)
22. Wagg, D.J., Worden, K., Barthorpe, R.J., Gardner, P.: Digital twins: state-of-the-art and future directions for modeling and simulation in engineering dynamics applications. *ASCE-ASME J. Risk Uncert. Engin. Syst. Part B Mech. Engin.* **6**(3) (2020)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

