ISSN: 2766-9823 Volume 2, 2020

Discrete Event Simulations and Digital Twins

Dmitry Namiot
Lomonosov Moscow State University
Moscow, Russia
dnamiot@gmail.com

Oleg Pokusaev
Russian University of Transport (MIIT)
Moscow, Russia
o.pokusaev@rut.digital

Abstract— **This article is about development tools for digital twins. Digital twins are getting more and more attention. At the same time, one cannot fail to note the fact that most of the publications consider the use of digital twins. How these twins were built (designed) is out of the question. Meanwhile, it is obvious that it is the questions of the development (creation) of twins that are primary, especially if we consider this issue from the standpoint of the university. It is universities that should, first of all, teach exactly the creation (design) of twins. The paper deals with the relationship between systems for the development of discrete simulation models and digital twins.**

I. Introduction

Discrete-event simulation (DES) is one of the oldest and most widely used approaches to simulation. In the discrete-event approach, the system model (description of its functioning) is represented as a sequence of events in time. Each event (or events in multithreaded systems) occurs at a certain point in time and fixes some change in the state of the system [1].

The states of the system are determined by the values of the variables that are part of the model. The initialization (change of values) of variables occurs in accordance with logical rules that determine the reaction to an event (or events). Accordingly, such logical rules are also part of the system. Another part of any such system is a synchronizer - a mechanism (algorithm) that synchronizes state changes (realizing the "occurrence" of events).

The constructed models can be single-threaded, that is, they have only one event, or multi-threaded, which can have several current events.

The events themselves are instant and interval. By the nature of the occurrence of events, discrete-event models are divided into deterministic and stochastic, depending on how the events are generated [2].

The condition for the completion of the model can be:

- Occurrence of a given event
- Reaching a certain state
- Expiry of the specified time according to the clock of the simulation system
- Passing a given number of cycles by the clock of the simulation system

The result of modeling is the statistical characteristics of the system variables, accumulated (measured) during the operation of the model.

This is one of the oldest (long-used) approaches to modeling systems, which has accumulated extensive practice over the years of use and, most importantly, led to the emergence of popular tools (libraries, frameworks, problem-oriented programming languages).

A digital twin, according to the classical definition, is a digital copy of a living or artificial physical object [3]. The term digital twin refers to a digital copy of potential and real physical assets (physical twin), processes, people, places, systems and devices that can be used for a variety of purposes. Digital twins are designed to facilitate the control, understanding and optimization of the functions of all physical assets, ensuring the smooth transfer of data between the physical and virtual world [4].

Accordingly, the digital twin, by definition, includes a simulation model. Hence, it is natural to assume that the tools for developing simulation models can be used to create digital twins. Since most of these systems were built in the “pre-twin” era, they do not have everything that digital twins require. Therefore, the correct question is what should (possibly) be added to the toolkit for creating discrete-event models so that it can be used for digital twins.

Digital twin technology plays a critical role in Industry 4.0, providing a unified view of the real and virtual worlds.

The remainder of the article is structured as follows. Section II compares discrete event models and digital twins, in which a digital replica can be represented as a set of states. Section III discusses extensions for software systems for constructing discrete-event models.

II. Discrete event models and digital replica

Most of the publications devoted to digital twins (typical examples are [5], [6]) are devoted to exploitation. The topic of this article is development. How something was built, the exploitation of which is nicely described in most articles.

Virtual models in the digital twin must be exact copies of physical objects. This means, generally speaking, the exact reproduction of physical geometry, properties, behavior and rules of functioning [7]. For example, 3D geometric models can describe a physical object in terms of its shape, size, etc. Based on physical characteristics (properties), for example, speed and force, the physical model should reflect such phenomena as deformation, destruction, etc. It should be noted here that discrete-event models are usually not very suitable for describing just physical models in the above sense. This is not their disadvantage in terms of their use for designing digital twins, it is just a limitation in terms of applicability. For physical models, there are simply other development tools.

Behavior models, as the name suggests, describe behavior and response mechanisms to changes in the external environment. The description of behavior in the form of transitions between states is exactly what is the basis of discrete-event modeling.

Rules are closely related to patterns of behavior and model decision making, assessments, etc. Rules can be created based on historical data (if any and available) or set by experts. This component corresponds to the logic module in discrete event simulation.

The next element that is essential for digital twins is, of course, data. All digital twins potentially deal with heterogeneous and multidimensional data coming from different sources. Simulation systems most often accept simplified data models. For example, for a variety of discrete-event modeling models, it is typical to model the occurrence of events in accordance with some statistical distribution. For digital twins, simplifications are not possible (otherwise it will not be a digital twin). The model can receive this data directly from physical objects, while the data can be both static and dynamic, the data can be generated by the model itself, or come from various services that process both measurement data and external (in relation to the physical object) data ... Note also that data can also be generated as a result of data fusion.

Naturally, in this case, this data does not come by itself, and the digital twin must be characterized by support for multiple connections. This is necessary to receive data from a physical object, as well as from external services. These connections are shown in Figure 1.

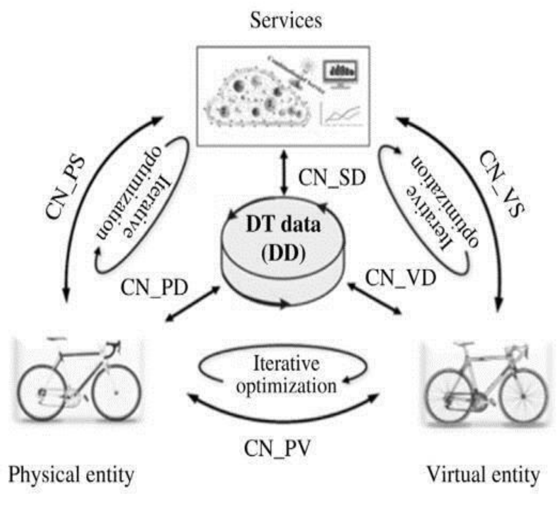

Fig. 1. Connections in a digital twin [8].

The main difference between the digital twin and the simulation model can be formulated as follows. Modeling is the analysis of “what will be if ...”, while the digital twin is “what is happening now and what will be if ...”. Key differences also include the following points.

The digital twin is a real-time simulation. Traditional simulations are performed in virtual environments, which can represent physical environments, but do not combine data in real time. The regular (continuous) transmission of information between the digital twin and the associated physical environment makes real-time simulation possible. This improves the accuracy of predictive analytical models, as well as the quality of management and monitoring.

More variables. Collecting all the data allows you to expand the number of possible variables available to the model. Naturally, such an analysis requires support in the descriptions of behavior models or in the rules of the model, but at least it becomes possible in principle. Accordingly, it becomes fundamentally possible to expand the use of simulation results. In other words, the digital twin should (can) get more results than just a simulation model.

The accumulation of large datasets in a digital twin increases its value over a simulation model. The digital twin becomes an integral part of the process of operating a physical object, which is used for predictive analytics, maintenance, etc. As a result, the digital twin affects the physical object, for example, provides data for its improvement, process optimization, etc. Examples include, for example, the digital twins of Tesla cars. Every Tesla car in production today has a digital twin that collects large data sets produced by each car. The collected data is used to optimize designs, predictive analytics, improve maintenance, etc. [9]. Another example is digital twins in BIM used in the operation of buildings and structures [6, 10]. Purely simulation models are usually used only at the design stage.

III. DISCRETE EVENT SIMULATION SOFTWARE TOOLS AND DIGITAL TWINS

In this section, we would like to focus on development tools for discrete-event models. Naturally, we proceed from the assumption that for our system the discrete-event approach will be suitable for representing a digital replica. In some cases, this will not be the case, and it will be necessary to use other tools. For example, in [11], discrete but dynamical systems are also described, etc.

Discrete Event Simulation (DES), as noted above, is a form of computer simulation of a system using a discrete sequence of events. The term "discrete" describes the fact that events develop forward in time at intervals determined in the model. The actual simulation is designed to track the evolution of the system over time. This is achieved using the so-called system (simulation) clock, which changes depending on the occurrence of events.

In the classic work [12], the schemes for using imitation clocks (model time) are divided into two types:

1) Determining the time of the next event (NETA - Next Event Time Advances). NETA is a mechanism that determines the timing of future events based on a list. In this typical approach, the model time is set to zero and then updated as events occur. Moreover, between successive events, under no circumstances should there be any changes in the system. All state changes are tied only to events.

2) Increasing the time using some fixed increment (FITA - Fixed Increment Time Advances). This alternative approach simply advances the system clock by some value. After changing the system time, events that have occurred during this time (time elapsed since the last increment) are determined. Internal variables (states) are updated according to the events. Then time advances again. As noted in the literature, in general, NETA is more widely used in modeling than FITA, due to its lower complexity [13].

The scopes of the discrete-event approach are quite wide. This is due to the simplicity and rather high efficiency of describing systems in the form of a set of states. Classic applications are everything that can be represented as a queuing system: service companies, transport systems, logistics and warehouse operations.

If we move from a discrete-event model to a digital twin, the digital replica of which is built on the basis of a discrete-event approach, then it is obvious that the model time, due to the requirement for data transmission in real time, must correspond to real time. Thus, for example, the zero count is simply the start time of the digital twin. The time of the next event is the real time of the occurrence of the event, the time increment is a new mark in which the occurred events are checked after the last check.

What and how should be changed in development tools can be represented by the example of the Simula language [14]. This language was one of the first to propose a practical implementation of a discrete-event modeling system based on coroutines [15]. The mechanisms proposed here are still the basis for discrete-event modeling tools.

Below is a classic example of a simple queuing system, taken directly from [16]. This program simulates the work of a certain fitting room (for simplicity, the only one), which visitors (Person) use in an exclusive manner. The visitor spends some time in the store (Hold (Normal (12, 4, u)) - a random value with a normal distribution), requests the use of the fitting room, waits in the queue (Wait.door), if the fitting room is occupied by another, then spends some then the time in the fitting room (Normal (3, 1, u) is a random value with a normal distribution), after which it leaves the system (fittingroom1.leave), promoting the next in line. The classical model of competition for a critical (single) resource.

```
Simulation Begin
   Class FittingRoom; Begin
      Ref (Head) door;
      Boolean inUse;
      Procedure request; Begin
         If inUse Then Begin
             Wait (door);
             door.First.Out;
         End;
         inUse:= True;
      End;
      Procedure leave; Begin
         inUse:= False;
         Activate door.First;
      End;
         door:- New Head;
      End;

      Procedure report (message); Text message; Begin
         OutFix (Time, 2, 0); OutText (": " & message); OutImage;
      End;

      Process Class Person (pname); Text pname; Begin
         While True Do Begin
            Hold (Normal (12, 4, u));
            report (pname & " is requesting the fitting room");
            fittingroom1.request;
            report (pname & " has entered the fitting room");
            Hold (Normal (3, 1, u));
            fittingroom1.leave;
            report (pname & " has left the fitting room");
         End;
      End;

      Integer u;
      Ref (FittingRoom) fittingRoom1;

      fittingRoom1:- New FittingRoom;
      Activate New Person ("Sam");
      Activate New Person ("Sally");
      Activate New Person ("Andy");
      Hold (100);
   End;
```

Note that the model (program) program creates all objects and activates all objects (Person) to place them in the event queue. The main program runs for 100 minutes in model time (Hold (100)).

Fragments that are to be replaced in the digital twin are highlighted in bold and underlined in the program.

First, there will be no runtime for the digital twin. Accordingly, the Hold (100) statement should simply be removed.

Next comes the creation of objects - New Person ("Sam"), etc. Obviously, the creation of objects should correspond to the actual appearance of users. Then it is not a static operator, but a process within the framework of the model that scans some sensors of a real system (opening a door, video analytics, issuing a number for a fitting room, etc.) and creates a new visitor based on information from these sensors - one or more.

And the most important thing is working with the Hold () model time. In this case, the model follows the NETA scheme and plays a random time when the next event occurs - how long the user is in the system before deciding to use the fitting room, and how long he will be in the chosen fitting room.

These statements (calls) should also be replaced by processes that poll the sensor data to determine the occurrence of the corresponding events.

It can be noted that the very scheme of a digital replica will change depending on what is actually possible to measure in specific conditions. For example, it is practically easy to organize the determination of the time of the fitting room occupation (presence sensor, badge for entry, which is returned upon exit, etc.). But it is not at all so easy to organize individual monitoring of being in the hall. It is possible, for example, that the arrival of a visitor will be recorded in the digital twin, and instead of the time spent in the hall, conversion will be measured - the number of visits to fitting rooms depending on the number of visitors. In other words, the digital replica (its structure) will depend on the data availability of the physical object. At the same time, accessibility should be understood as both the technical possibility of obtaining them and the economic feasibility of collecting them.

How can you provide access to measurement data for a discrete event model? The easiest way to achieve this is to introduce some bus (Kafka or similar products), where, on the one hand, the measurement data will be published, and on the other hand, there will be subscribers (the publish-subscribe model) who will read this data. Then, for example, instead of playing a random delay when simulating a fitting room occupation, there will be a subscription to a seizure signal and waiting for a release signal. The time difference between these two events (already real events) will be taken into account in statistics as the time for a particular visitor.

IV. Conslusion

This article briefly discusses the possibilities of using discrete-event modeling tools to design digital twins. A comparative analysis of classical discrete-event models and digital replicas in twins using the discrete-event approach is carried out. A set of extensions for systems for developing discrete-event models is presented, which are necessary for using such development tools when creating digital twins.

Acknowledgment

We would like to thank the reviewers for the early versions of this article for criticism and comments that have contributed to the improvement of this work.

References

[1] Varga, András. "Discrete event simulation system." Proc. of the European Simulation Multiconference (ESM'2001). 2001.

[2] Fishman, George S. Discrete-event simulation: modeling, programming, and analysis. Springer Science & Business Media, 2013.

[3] El Saddik, Abdulmotaleb. "Digital twins: The convergence of multimedia technologies." IEEE multimedia 25.2 (2018): 87-92.

[4] Khajavi, Siavash H., et al. "Digital twin: vision, benefits, boundaries, and creation for buildings." IEEE Access 7 (2019): 147406-147419.

[5] Kurganova, Nadezhda, et al. "Digital twins' introduction as one of the major directions of industrial digitalization." International Journal of Open Information Technologies 7.5 (2019): 105-115.

[6] Kupriyanovsky, Vasily, et al. "Digital twins based on the development of BIM technologies, related ontologies, 5G, IoT, and mixed reality for use in infrastructure projects and IFRABIM." International Journal of Open Information Technologies 8.3 (2020): 55-74.

[7] Qi, Qinglin, et al. "Enabling technologies and tools for digital twin." Journal of Manufacturing Systems (2019).

[8] Tao, Fei, et al. "Digital twin driven prognostics and health management for complex equipment." Cirp Annals 67.1 (2018): 169-172.

[9] Tharma, Rajeeth, Roland Winter, and Martin Eigner. "An approach for the implementation of the digital twin in the automotive wiring harness field." DS 92: Proceedings of the DESIGN 2018 15th International Design Conference. 2018.

[10] Kupriyanovsky, Vasily, et al. "BIM Technologies for Tunnels Used in Subways, Railways, Highways, and Hyperloop-IFC-Driven Real-Time Systems and Disruptive Innovation." International Journal of Open Information Technologies 8.9 (2020): 70-93.

[11] Ganguli, R., and S. Adhikari. "The digital twin of discrete dynamic systems: Initial approaches and future challenges." Applied Mathematical Modelling 77 (2020): 1110-1128.

[12] Law, Averill M., W. David Kelton, and W. David Kelton. Simulation modeling and analysis. Vol. 3. New York: McGraw-Hill, 2000.

[13] Tang, Jiangjun, George Leu, and Hussein A. Abbass. Simulation and Computational Red Teaming for Problem Solving. John Wiley & Sons, 2019.

[14] Nance, Richard E. "A history of discrete event simulation programming languages." History of programming languages---II. 1996. 369-427.

[15] Moura, Ana Lúcia De, and Roberto Ierusalimschy. "Revisiting coroutines." ACM Transactions on Programming Languages and Systems (TOPLAS) 31.2 (2009): 1-31.

[16] Simula https://en.wikipedia.org/wiki/Simula Retrieved: Jan, 2021.