Discrete-Event-Simulation Open-Source tools assessment for Digital Twin in manufacturing systems

Michela Lanzini*, Alberto Tamburini, Ivan Ferretti***, Simone Zanoni*****

** Department of Mechanical and Industrial Engineering, University of Brescia, Italy (michela.lanzini@unibs.it)*

*** Department of Technology, Management and Economics, Technical University of Denmark, Copenhagen, Denmark (albetam@dtu.dk)*

**** Department of Civil, Environmental, Architectural Engineering and Mathematics University of Brescia, Italy*

ivan.ferretti@unibs.it, simone.zanoni@unibs.it)

Abstract: This paper aims to present an analysis and assessment of the different options currently accessible for assisting the creation and execution of a Digital Twin (DT) that relies on a discrete event simulator. Discrete-event simulation (DES) is widely recognized for its industry standards such as Anylogic, Arena, Flexsim, Software Plant Simulation. However, especially in the context of Industry 4.0, it becomes crucial to have business systems that can be easily integrated and adapted with pre-existing systems and technologies, to guarantee a flexible and dynamic approach to handle innovations. As a result, the industry is shifting towards the use of open-source software for discrete-event modeling. Some examples of simulation libraries are SimPy, Ciw, and Salabim for Python, Simmer for R, and ConcurrentSim for Julia. The methodology for evaluating these tools, regardless of whether they are open or closed source, will involve conducting literature reviews and analyzing additional sources, such as GitHub and GitLab. This approach is necessary because descriptions of the utilization of open-source software may not always be readily available in academic papers. The purpose of this study is to identify metrics that can indicate the most relevant open-source tools in today's landscape and compare them to the most well-known and widely used industrial software. The study will critically evaluate the proposed metrics in order to highlight the key characteristics of a DES tool based on the needs of a company seeking a DT solution.

Keywords: Digital Twin, Framework evaluation, Discrete Event Simulation, Manufacturing, Open source tools

1. Introduction

Digital Twin (DT) has emerged as a promising approach to improve efficiency, productivity, and decision-making. In the context of Industry 4.0 and 5.0 technologies, the DT is seen as an extremely versatile tool. The concept of a real space and a virtual space, linked by two-way streams of data and information, was introduced almost twenty years ago by Grieves (2005) under the name "mirrored spaced models". It was later adopted by NASA, which provided an extensive definition as "an integrated multi-physics, multi-scale, probabilistic simulation of a vehicle or system that utilizes the most accurate physical models, sensor updates, fleet history, etc., to replicate the life of its flying counterpart." The NASA Modeling, Simulation, Information Technology & Processing Roadmap of 2010 and 2012 states that the system is highly realistic and takes into account multiple important and interconnected vehicle systems.

After the announcement of Industry 4.0 at the Hannover fair and the development of new technologies such as the Internet of Things, data science (Qi and Tao 2018), artificial intelligence (Salini et al., 2023), blockchain (Dolgui et al., 2020), and cyber-physical systems (Frazzon et al., 2018), the concept of a digital twin has started to be more extensively studied and explored. Boschert and Rosen (2016) define DT as the "next wave of simulation, characterized by continuous support throughout the entire lifecycle." The key elements of the definition of DT are those of the virtual representation of a physical counterpart, connected by a continuous and bidirectional flow of data and information.

The concept of DT has gained significance following the European Commission's programmatic announcement of Industry 5.0, where it is identified as one of the key technologies (European Commission, 2021). In addition to being part of European projects (Castañé et al., 2023), various uses of the digital twin in the manufacturing field can be found in the literature (Negri et al., 2017). These include production planning management, production control, scheduling, inventory management, design of production processes and layouts, and online monitoring (Agostino et al., 2020; Mosca et al., 2022; Pietrangeli et al., 2023). It has also been shown how data-driven tools and simulation can lead to optimization and better decision making, paving the way for the utilization of a proper DT (Lanzini et al., 2023, Fani et al., 2017 and 2023).

Despite the increasing interest in this solution, and despite the existence of numerous methods for structuring the virtual counterpart of the physical system (discrete event simulation, continuous time simulation, stochastic simulation, artificial intelligence, multiphysics modeling), it is extremely difficult to find literature that compares the various solutions or proposes adaptability analyses based on identifiable parameters. The lack of a clear tool evaluation guide is identified as one of the challenges in implementing digital twins in the industrial sector (Saporiti et al., 2023).

Specifically for discrete event simulation (DES) tools, there are several contributions in the literature that propose an evaluation framework for these tools. Particularly relevant is the work of Dias et al. (2016), which identifies and compares approximately twenty software for discrete event simulation. Fumagalli et al. (2019) present a systematic review of existing selection frameworks and propose a method based on the Analytic Hierarchy Process (AHP). Cafasso et al. propose an alternative method to AHP, which is integrated with benefits, opportunities, costs, and risks (BOCR), and further enhanced with a "best-worst method" (BWM). The main drawback of these methods is that they are designed for proprietary licensed software, the costs of which can be a deterrent for this solution, especially for small and medium-sized enterprises (Saporiti et al., 2023).

It is relevant to mention the presentation and discussion of frameworks for open source (OS) DES toolsby Dagkakis and Heavey (2016). Although a formal evaluation framework is not formally presented, the presentations of the tools and their usage are extremely useful. Peyman et al. (2021) propose methods for integrating Python with major DES software, while Lang et al. (2021) discuss whether OS software can be a viable alternative to proprietary software, comparing two commercial software and two OS tools.

Generally, in literature, it is not possible to find a comprehensive framework for selecting OS tools or a wide-ranging comparison between proprietary software and OS tools, primarily due to the abundance and fragmentation of open-source tools. This paper aims to present a selection of the most important open-source tools in the current landscape of DES tools. Using a well-known software evaluation framework from literature, it compares their general characteristics with some widely used commercial software, thus addressing the gap in literature regarding these topics.

The paper will be structured as follows: in section 2, the method used for conducting the research will be presented, while section 3 will contain the results and evaluations; the conclusions will follow.

2. Methodology

Following the guidelines of Fumagalli et al. (2019), a systematic review was conducted to assess whether there were existing contributions that provided insights into the use of both proprietary and open-source DES tools in relation to digital twin solutions. The search using the key [TITLE (("digital twin" OR "digital twins" OR "digital-twin" OR "digital-twins") AND ("criteria" OR "selection" OR "evaluation")).")) yielded 200 papers, none of which were relevant to the selection criteria DT tools. Due to the lack of formal guidelines, we have chosen to adopt the AHP-CORB dimension scheme presented by Cafasso et al. (2020)., with the following dimensions:

- Benefits: This section refers to the features of the tool being considered. The main subcategories include general characteristics (simulation speed, supported languages), visual aspects (icons, 3D objects or animations, virtual reality), coding aspects (coding, attributes and variables, routing rules, coding rules), and output (data export, animation export, statistical information on exported data).
- Costs: encompassing the expenses associated with the initial purchase, the implementation process, and the subsequent updates.
- Opportunities: This term refers to the availability of input data, statistical information on input data, ease of use through a graphical user interface, graphical model construction, real-time viewing capabilities, support and training provided through manuals, tutorials, online support, demo versions, and updates.
- Risks: refer to the reliability (compatibility, efficiency, integration with other systems) and the vendor, particularly the vendor's strength.

- The framework by Cafasso et al. (2020) was employed to provide qualitative insights into the main differences between proprietary and open-source tools. During the search on GitHub and GitLab, the most relevant open-source DES tools for various programming languages were identified. GitHub and GitLab are the two main hosting platforms for software projects that implement the "Git" version control system.

The metrics used, following the logic of certain dimensions identified in Dias et al. (2016), were the "stars" assigned by the online community, the number of "forks" (i.e., the number of alternative projects developed from already published projects), and the reasonably recent last update. The overall count of updates and releases was excluded due to their inconsistency in the considered time frames and their redundancy in case of problems arising from bug-containing updates. At the end of the analysis, the reference tools for comparison are:

- Anylogic (AnyLogic Company)
- Arena (Rockwell Automation)
- Flexsim (FlexSim Software Products)
- Software Plant Simulation (Siemens)
- Simpy (Matloff, 2008)
- Ciw (Palmer et al., 2019)
- Salabim (Van Der Ham, 2018)
- OMNeT++ (Varga, 2010)
- Simmer (Ucar et al., 2018)
- Sim#
- ConcurrentSim
- JaamSim (King and Harrison, 2013)

The first four are software with proprietary licenses that are already widely known in the literature (Dias et al., 2016), while for some of the open-source ones, it was not possible to find a reference for their presentation in the literature. The tools to be analyzed were identified, and a qualitative analysis of their characteristics was conducted, following the dimensions identified by Cafasso et al. (2020).

3. Evaluation

Based on the dimensions identified by Cafasso et al. (2020), it was possible to assess the differences in characteristics between commercial and open source tools, and potential areas for improvement were identified for the development of an evaluation framework. This framework would be able to evaluate both the specific use of a DES tool for DT and enable an unbiased comparison between commercial software and open source tools. The characteristics of OS tools are listed in table 1, while the main comparison evaluations are presented in table 2.

Regarding the composition of the identified OS tools, it is noteworthy that they tend to have lower numbers of stars and forks compared to other commonly used libraries (for example, the numpy library, for numerical analysis in Python, has 26,400 stars and 9,300 forks). It is hypothesized that the lower numbers are caused by a combination of factors: firstly, the most widely used tools, for which the GitHub and GitLab community is more active, are general-purpose tools that are traditionally introduced in basic programming language courses, while the discrete event simulation sector tends to be more niche, in academic and industrial applications; another reason may be that commercial software often offers academic or demo versions, leading those who approach DES techniques to use commercial software instead of OS ones.

To verify that all the tools were active projects and not discontinued, we analyzed both the frequency of version releases and commits, as well as the responsiveness of the community on GitHub/Lab. It was observed that update frequencies can exhibit non-linear trends over the years, with numerous frequent updates in short periods of time. This may be attributed, in some cases, to the limited number of individuals actively programming and maintaining these tools. (see Table 1)

Regarding the comparative evaluations, it is possible to outline the main differences between proprietary software and OS tools. In terms of the "Benefit" section, it is observed that licensed software exhibits a more pronounced standardization of use, along with more visual and intuitive interfaces for both model creation and output. On the other hand, OS tools tend to be lower-level solutions, but offer significant customization and integration possibilities with graphic and presentation tools. Regarding costs, open-source tools do not have any purchasing or acquisition costs.

Regarding costs, open-source tools do not have any purchasing or acquisition costs. However, this can make it difficult to identify who can implement the model if there are no internal resources available. There is a clear trade-off between costs and guarantees, and this decision point is particularly important to present to the company's decision-makers and evaluate in light of the implications of the specific project for which the tool selection is being carried out.

Regarding the opportunities, despite some differences between the analyzed software and tools, it is interesting to note the main macro-differences in the two groups. The ease of integration with other tools of the OS platform allows for a simplified possibility of connecting with existing tools, particularly when it comes to input data analysis, whereas commercial software often requires a custom connection. The learning curve also differs for the two categories: in the case of proprietary software, thanks to intuitive interfaces, it is possible to quickly become independent users, but to become expert users, it is still necessary to learn programming in order to allow for high levels of customization. On the other hand, operating systems have a steep initial learning curve in order to allow for high levels of customization. On the other hand, operating systems have an initially slow learning curve, especially for users who are not familiar with the development environment. However, the greatest difficulties are concentrated in the initial stages of learning.

Regarding support and training, OS tools offer a much wider but also more fragmented selection of tutorials, explanations, examples, and use cases compared to proprietary software, which tends to release official manuals and tutorials. Although documentation and tutorials are also available for OS tools, it is generally observed that the amount of information provided by the community for the community is significantly greater. This results in more widespread support, but also the inability to guarantee the quality of certain types of information. The size of the risks highlights the fact that proprietary software faces challenges in integrating with other existing systems and technologies compared to open-source tools.

However, proprietary software has the advantage of being managed by companies, making it less likely for solutions and software to be discontinued or become obsolete, while open-source tools are more vulnerable in this regard. (see Table 2)

Overall, it is possible to acknowledge that Cafasso's individualized dimension structure is valid for a qualitative evaluation of software proprietary versus operating system tools. However, it is evident that this structure provides dimension integration in order to discuss the adaptability of software or tools used in constructing the virtual counterpart of a DT.

Specifically, given that the bidirectionality of information flows between the physical counterpart and the virtual counterpart is a key aspect of the digital twin, it is necessary to further explore the potential for this exchange of information. Another dimension that should be further explored could refer to the possibility of synchronization, even online.

By incorporating these evaluations into the analysis, it becomes apparent that OS tools are increasingly becoming highly suitable and appropriate instruments for forming the foundation of a DES-based DT.

Conclusions

In this paper, an evaluation of the available tools for implementing a Digital Twin based on discrete event simulation (DES) has been conducted.

The research was conducted by identifying open-source tools to fill the research gap, and by using an existing comparison framework from literature to analyze the strengths and weaknesses of proprietary software and open-source tools. Proprietary software typically provides standardization, an intuitive interface, and more stable support from the provider, but it also entails some inflexibility in adapting to the context to be modeled, as well as difficulties in integrating with existing systems and technologies in the company. On the other hand, OS tools offer greater flexibility, customization options, and integration with other tools without initial costs, but they may require a steeper learning curve and involve risks related to maintenance and service continuity. Additionally, they are inherently more fragmented and dispersed in terms of support resources.

Thanks to their high customizability and integration, OS tools are emerging as valid options for Digital Twin solutions. However, it has been revealed that there are currently no adequate models of evaluation/selection frameworks to assess the suitability of a tool for the role of a virtual counterpart of a DT, particularly in terms of bidirectional data exchange and real-time synchronization.

Future work resulting from this contribution may involve developing an evaluation framework for DES tools that can easily encompass OS tools as well, and investigating the necessary dimensions to assess whether a tool is suitable to be the virtual counterpart of a DT.

Table 1: DES tool OS selected from GitHub and GitLab

Tool	Language	GitHub/Lab stars	Forks	Latest Version	Description?
Simpy	Python	91	41	November 2023	Process-based discrete-event simulation framework
Ciw	Python	144	42	April 2024	DES library for open queueing networks
Salabim	Python	253	62	May 2024	Library for object-oriented discrete event simulation
OMNeT++	C++	546	141	February 2023	Public-source, component-based, modular and open-architecture simulation environment with strong GUI support and an embeddable simulation kernel
Simmer	R	217	42	November 2023	Process-oriented and trajectory-based Discrete-Event Simulation (DES) package
Sim#	C#	119	31	June 2023	Process-based discrete-event simulation framework
ConcurrentSim	Julia	171	36	August 2023	Process-based DES framework
JaamSim	Java	148	65	April 2024	DES environment in which model logic can be coded directly in either a event- or process-oriented style

Table 2: Comparison between licensed software and open-source tools for DES

Category	Dimension	Licensed softwares	Open source tools
Benefits	General charateristics	Strong standardization	High customization
	Visual aspects	Advanced visualization tools (e.g. dashboards, 3D animations)	No built-in tools, but easy to integrate with other libraries
	Coding aspects	Usually drang-and-drop, limited coding	The entire model is described through a program
	Output	Data visualization improved through intuitive dashboards and animations	No built-in dashboards and visualization, but can be integrated with other libraries
Costs	Purchasing cost	Licence cost	No licence cost
	Implementation cost	Can be externalized to the DES company, another consultant company, or internal resources can be used	Can be externalized to a consultant company or internal resources can be used
	Cost of updates	Update cost and adaptation cost	Adaptation cost
Opportunities	Input	No built-in tools to analyze/prepare the input	Easy to integrate with input analysis library
	Ease of use	Initial fast learning curve	Initial slow learning curve
	Support and training	Company support and tutorials	Documentation and community
Risks	Reliability	Reliable, but hard to integrate with other systems	Easy to integrate with heterogeneous systems and technologies
	Vendor	Companies can discontinue projects or features	Continuous updates are not guaranteed

References

Agostino, Ícaro Romolo Sousa, Eike Broda, Enzo M. Frazzon, and Michael Freitag. 2020. "Using a Digital Twin for Production Planning and Control in Industry 4.0." In *Scheduling in Industry 4.0 and Cloud Manufacturing*, edited by Boris Sokolov, Dmitry Ivanov, and Alexandre Dolgui, 289:39–60. International Series in Operations Research & Management Science. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-43177-8_3.

Boschert, Stefan, and Roland Rosen. 2016. "Digital twin—the simulation aspect." Mechatronic futures: Challenges and solutions for mechatronic systems and their designers: 59-74.

Cafasso, Davide, Cosimo Calabrese, Giorgia Casella, Eleonora Bottani, and Teresa Murino. 2020. "Framework for Selecting Manufacturing Simulation Software in Industry 4.0 Environment." *Sustainability* 12 (15): 5909. https://doi.org/10.3390/su12155909.

Castañé, G., Dolgui, A., Kousi, N., Meyers, B., Thevenin, S., Vyhmeister, E., & Östberg, P. O. 2023. The ASSISTANT project: AI for high level decisions in manufacturing. International Journal of Production Research, 61(7), 2288–2306. https://doi.org/10.1080/00207543.2022.2069525

Dagkakis, G, and C Heavey. 2016. "A Review of Open Source Discrete Event Simulation Software for Operations Research." *Journal of Simulation* 10 (3): 193–206. https://doi.org/10.1057/jos.2015.9.

Dias, Luis M. S., Antonio A. C. Vieira, Guilherme A. B. Pereira, and Jose A. Oliveira. 2016. "Discrete Simulation Software Ranking — A Top List of the Worldwide Most Popular and Used Tools." In *2016 Winter Simulation Conference (WSC)*, 1060–71. Washington, DC, USA: IEEE. https://doi.org/10.1109/WSC.2016.7822165.

Dolgui, Alexandre, Dmitry Ivanov, Semyon Potryasaev, Boris Sokolov, Marina Ivanova, and Frank Werner. 2020. "Blockchain-Oriented Dynamic Modelling of Smart Contract Design and Execution in the Supply Chain." *International Journal of Production Research* 58 (7): 2184–99. https://doi.org/10.1080/00207543.2019.1627439.

European Commission, Directorate-General for Research and Innovation, Breque, M., de Nul, L., & Petridis, A. 2021. Industry 5.0 – Towards a sustainable, human-centric and resilient European industry. Publications Office of the European Union.

Fani, Virginia, Sara Antomarioni, Romeo Bandinelli, and Maurizio Bevilacqua. 2023. "Data-Driven Decision Support Tool for Production Planning: A Framework Combining Association Rules and Simulation." Computers in Industry 144 (January):103800. https://doi.org/10.1016/j.compind.2022.103800.

Fani, Virginia, Romeo Bandinelli, and Rinaldo Rinaldi. 2017. "A Simulation Optimization Tool For The Metal Accessory Suppliers In The Fashion Industry: A Case Study." In ECMS 2017, 240–46. ECMS. https://doi.org/10.7148/2017-0240.

Frazzon, Enzo M., Mirko Kück, and Michael Freitag. 2018. "Data-Driven Production Control for Complex and Dynamic Manufacturing Systems." *CIRP Annals* 67 (1): 515–18. https://doi.org/10.1016/j.cirp.2018.04.033.

Fumagalli, Luca, Adalberto Polenghi, Elisa Negri, and Irene Roda. 2019. "Framework for Simulation Software Selection." *Journal of Simulation* 13 (4): 286–303. https://doi.org/10.1080/17477778.2019.1598782.

Grieves, M. W. (2005). Product lifecycle management: the new paradigm for enterprises. International Journal of Product Development, 2(1–2). https://doi.org/10.1504/ijpd.2005.006669

King, D. H., and Harvey S. Harrison. 2013. "Open-source simulation software "JaamSim"." *2013 Winter Simulations Conference (WSC)*. IEEE.

Lang, Sebastian, Tobias Reggelin, Marcel Müller, and Abdulrahman Nahhas. 2021. "Open-Source Discrete-Event Simulation Software for Applications in Production and Logistics: An Alternative to Commercial Tools?" *Procedia Computer Science* 180: 978–87. https://doi.org/10.1016/j.procs.2021.01.349.

Lanzini, Michela, Nicola Adami, Stefano Benini, Ivan Ferretti, Gianbattista Schieppati, Calogero Spoto, and Simone Zanoni. 2023. "Implementation and Integration of a Digital Twin for Production Planning in Manufacturing." In *Proceedings of the 35th European Modeling & Simulation*

Symposium, EMSS. CAL-TEK srl. https://doi.org/10.46354/i3m.2023.emss.032.

Matloff, Norm. 2008. Introduction to discrete-event simulation and the simpy language. *Davis, CA. Dept of Computer Science. University of California at Davis. Retrieved on August*, *2*(2009), 1-33.

Mosca, Roberto, Marco Mosca, Roberto Revetria, Fabio Currò, and Federico Briatore. (2022). "Smart Inventory 4.0. Case Study Application to an Italian SME Operating in the Cosmetics Sector: Mediterranea Cosmetics Belonging to Fratelli Carli Spa, Imperia (ITALY)."

Negri, Elisa, Luca Fumagalli, and Marco Macchi. 2017. "A Review of the Roles of Digital Twin in CPS-Based Production Systems." *Procedia Manufacturing* 11: 939–48. https://doi.org/10.1016/j.promfg.2017.07.198.

Palmer, Geraint I., Vincent A. Knight, Paul R. Harper, and Asyl L. Hawa. 2019. "Ciw: An Open-Source Discrete Event Simulation Library." *Journal of Simulation* 13 (1): 68–82. https://doi.org/10.1080/17477778.2018.1473909.

Peyman, Mohammad, Pedro Copado, Javier Panadero, Angel A. Juan, and Mohammad Dehghanimohammadabadi. 2021. "A Tutorial on How to Connect Python with Different Simulation Software to Develop Rich Simheuristics." In *2021 Winter Simulation Conference (WSC)*, 1–12. Phoenix, AZ, USA: IEEE. https://doi.org/10.1109/WSC52266.2021.9715511.

Pietrangeli, I., Mazzuto, G., Ciarapica, F. E., & Bevilacqua, M. (2023). Artificial Neural Networks approach for Digital Twin modelling of an ejector. Proceedings of the 35th European Modeling & Simulation Symposium, EMSS. https://doi.org/10.46354/i3m.2023.emss.007

Qi, Qinglin, and Fei Tao. 2018. "Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison." *IEEE Access* 6: 3585–93. https://doi.org/10.1109/ACCESS.2018.2793265.

Salini, S., and B. Persis Urbana Ivy. 2023. "Digital Twin and Artificial Intelligence in Industries." In *Digital Twin for Smart Manufacturing*, 35–58. Elsevier. https://doi.org/10.1016/B978-0-323-99205-3.00014-6.

Saporiti, Nicolò, Violetta Giada Cannas, Rossella Pozzi, and Tommaso Rossi. 2023. "Challenges and Countermeasures for Digital Twin Implementation in Manufacturing Plants: A Delphi Study." *International Journal of Production Economics* 261 (July): 108888. https://doi.org/10.1016/j.ijpe.2023.108888.

Shafto, M., Conroy, M., Doyle, R., Glaessgen, E., Kemp, C., LeMoigne, J., & Wang, L. (2010). DRAFT Modelling, simulation, information technology & processing roadmap - technology area 11. National Aeronautics and Space Administration, November.

Shafto, M., Conroy, M., Doyle, R., Glaessgen, E., Kemp, C., LeMoigne, J., Wang, L., & A. (2012). Modeling, Simulation, Information Technology & Processing Roadmap-NASA. National Aeronautics and Space Administration, 1–38.

Ucar, Iñaki, Bart Smeets, and Arturo Azcorra. 2018. "Simmer: discrete-event simulation for R." *arXiv preprint arXiv:1705.09746*.

Varga, Andras. 2010. OMNeT++. In *Modeling and tools for network simulation* (pp. 35-59). Berlin, Heidelberg: Springer Berlin Heidelberg.

Van Der Ham, Ruud. 2018. "Salabim: Discrete Event Simulation and Animation in Python." *Journal of Open Source Software* 3 (27): 767. https://doi.org/10.21105/joss.00767.