GESTÃO & PRODUÇÃO

ISSN 0104-530X (Print)
ISSN 1806-9649 (Online)

ORIGINAL ARTICLE

# Systematic review of the literature on Digital Twin: a discussion of contributions and a framework proposal

## *Revisão sistemática da literatura sobre* Digital Twin*: discussão sobre contribuições e proposta de um framework*

Richard Junior Maraschalchi da Cruz[1], Luiz Antonio Tonin[1]

[1]Universidade Federal de São Carlos – UFSCar, Centro de Ciências Exatas e de Tecnologia, Departamento de Engenharia de Produção, São Carlos, SP, Brasil. E-mail: richardmaraschalchi@gmail.com; tonin@dep.ufscar.br

**How to cite:** Cruz, R. J. M., & Tonin, L. A. (2022). Systematic review of the literature on Digital Twin: a discussion of contributions and a framework proposal. *Gestão & Produção*, 29, e9621. https://doi.org/10.1590/1806-9649-2022v29e9621

**Abstract:** This study aimed to investigate the definitions, requirements and applications of Digital Twin (DT) models, and for that, a systematic literature review was carried out using Science Direct, Scopus and Web of Science as the bases for the articles. The method used was based on bibliometric analysis of the 332 articles extracted from the bases and a content analysis, which sought to identify contributions, common topics and gaps among the 17 articles that were filtered and identified as relevant to the second objective of this work, the framework proposal. The analyzed sample indicated that authors diverge on the definitions and use different processes to build a DT, models that seek to represent elements of the real environment, which could be products and/or processes/services, with the objective of being a tool for performance monitoring, simulation of scenarios and source of information for decision making. The use of software that often does not have trial licenses or versions for students was also observed in the sample, based on this point and on the analyzed content, an attempt was made to build a conceptual framework for the construction of a DT model in a game engine, using software that does not require licenses.

**Keywords:** Digital Twin; Framework; Game Engines.

**Resumo:** O presente estudo teve como objetivo investigar as definições, requisitos e aplicações dos modelos de *Digital Twin*, para tanto, se realizou uma revisão sistemática da literatura, utilizando como bases para extração de artigos a ScienceDirect, a Scopus e a Web of Science. A abordagem utilizada se baseou nas análises bibliométricas de 332 artigos extraídos das bases e da análise de conteúdo, que buscou identificar contribuições, pontos em comum, e lacunas entre os 17 artigos que foram filtrados e identificados como pertinentes com o segundo objetivo desse trabalho, a proposta de framework. A amostra analisada indicou que os autores divergem quanto as definições e se utilizam de processos distintos para a construção de um *Digital Twin*, modelos esses que buscam representar elementos do meio real, podendo ser de produtos e/ou processos/serviços, com o objetivo de ser uma ferramenta para o acompanhamento de desempenho, simulação de cenários e fonte de informações para a tomada de decisão. Também foi observado na amostra o uso de softwares que muitas vezes não tem licenças para avaliação ou versões para estudantes, a partir desse ponto e dos conteúdos analisados, buscou-se

Received Oct. 8, 2021 – Accepted Apr. 4, 2022
Financial support: None.

elaborar um framework conceitual para a construção de um modelo de *Digital Twin* em uma *game engine*, utilizando softwares que não necessitam de licenças para o seu uso.

**Palavras-chave:** *Digital Twin*; Framework; *Game Engines*.

## 1 Introduction

The term Digital Twin (DT) and the first studies of these types of models are credited to Dr. Michael Grieves, having his first mention during a lecture at the University of Michigan on Product Lifecycle Management (PLM), which is the tracking of the stages of development and market introduction, growth, maturity and decline of a product (Grieves, 2014). Although the first models focused on the study of products, Tao et al. (2018) add that there is also the application of DT models for the representation of processes and for the use of services.

The DT, or digital twin, is a type of simulation model that seeks to represent the real/physical environment through a computational/digital model (Quinalha, 2018). Also, regarding these models, considering the existence of other definitions, it is possible to say that they can be divided into three parts: the physical/real environment, the digital medium and the data connection between these two media (Tao et al., 2018), which enables the exchange of data in real time, the storage of historical data and the simulation of the performance of this process, product or service (Lee et al., 2015).

These characteristics allow the models to fulfill the main functions of being a prototype, a means for testing and studying scenarios or a tool for monitoring performance and a source for data extraction (Quinalha, 2018). Rosen et al. (2015) highlight that the combination of states and data from the real environment with the simulation can provide a good prediction of performance and assistance for planning and decision making.

As well as simulation models in general, DT models are one of the many technologies of the so-called Industry 4.0, which began in Germany, but has already spread to several other countries and which aims to develop smart factories with a high degree of autonomy and flexibility (Dalenogare, 2018).

There are several other technologies linked to Industry 4.0, commonly referred to as the fourth industrial revolution or information revolution (Tropia et al., 2017), with the Internet of Things (IoT), Big Data and Artificial Intelligence being some of them, which can be related to some degree with DT models. It is important to highlight the growing discussions about Industry 4.0, which consequently help in the diffusion of correlated theories, such as the Digital Twin models.

Just as there is an increase in the number of publications about Industry 4.0, there is also an increase in the number of publications that address DT models, but it is possible to notice a lack of practicality to have access to these models, as well as to build them. In addition to the existence of several definitions for them, many possible final objectives and functional requirements, there is also not a wide range of free software that offer trial versions or even student versions to disseminate this knowledge to the academic environment or small organizations.

With that came the interest of exploring an alternative for the construction of a DT model using free or open source software, and one of the platforms that could support all the functions and requirements seen are the Game Engines, software created for the construction of video games, capable of providing good visualizations, physical simulations on objects, integrating artificial intelligence models and that are highly

customizable, being, for example, used in markets such as architecture, assisting in the visualization of off-plan properties (Fritsch & Kada, 2004).

In order to elaborate this framework proposal with the main steps necessary for the construction of a Digital Twin model in a game engine, it was necessary to investigate the definitions, requirements and applications of the Digital Twin models, and for that, a systematic review of the literature, along with a summary of the main topics discussed that will be detailed throughout the text.

Given the context described, highlighting the differences between the authors, observed in the literature review, and the possibility of seeking ways to spread these models more comprehensively and easily in the academic and business environment, three questions were elaborated for which this article intends to provide theoretical contributions: (a) What are the main concepts and definitions about the Digital Twin models? (b) In the analyzed literature, how does this knowledge converge or not to a precise definition on the topic? (c) How would it be possible to create a Digital Twin model through free software and thus help to spread its use in small companies and educational institutions?

The last question is answered by proposing a framework that demonstrates the steps for the elaboration of a Digital Twin model using a Game Engine, for the other questions a series of contributions resulting from the systematic literature review are presented.

## 2 Research method

The research method was based on a Systematic Literature Review (SLR) seeking to identify the main themes on Digital Twin, along with the identification of the main authors, relevant publications and literature history. For the SLR, three databases were selected for the extraction of articles, these three were ScienceDirect, Scopus and Web of Science, chosen because they index searches in journals and allow the extraction of data for the StArt software, a tool to manage bibliometric reviews.

StArt was developed to support the realization of SLR, covering steps such as the creation of a review protocol, the definition of procedures for acceptance or rejection of selected studies, in addition to providing means for the creation of reports that will support to characterize the state of the art of the topic in question (Hernandes et al., 2012). An example of the use of StArt software can be seen in the article *Systematic bibliographic review on curricular "environmentalization" in higher education: an analysis focusing on legal education*, written by Eliana et al. (2019). In addition to StArt, the Microsoft Excel software was also used for performing other data processing and for generating visualizations and reports.

Returning to the research method, it is important to highlight that a clipping was made regarding studies and works that addressed Process Digital Twin models, since there are also product and service models, a second motivation for the clipping was the objective of understanding the construction of these models, in addition to the desire to establish a framework for the construction of one through a game engine, therefore, this cut influenced the choice of sets of keywords for searches in the databases.

The sets of keywords used were ("*digital twin*" AND "framework" AND "*production line*") and ("*digital twin*" AND "*game engine*"), along with the application of a filter to search only articles and the restriction by the search for texts in English and

Portuguese. As a result, articles published between 2016 and 2020 were found, along with articles that were already scheduled to be published in 2021.

After the initial search in the databases, carried out in October 2020, a sample of 367 articles was collected, 177 extracted from Science Direct, 183 from Scopus and 7 from Web of Science, as searches were carried out in three different databases, some articles were indexed in more than one base at the same time, thus appearing cases of duplicate articles in the sample. To solve this point, StArt provides the resource for removing duplicate documents, and with this, 35 duplicate articles were excluded, leaving 332 articles for analysis.

For the descriptive analysis, the 332 articles in the sample were considered, but for the content analysis, a filtering was performed seeking articles that had in their content points referring to the Digital Twin of processes, as well as the use of frameworks for the construction or tools that support the construction of a model in a game engine, which are the secondary objectives of this work.

For the selection of articles for content analysis, a filtering step was first carried out through reading the title, keywords and abstract, discarding articles that deviated from the theme and separating for a second analysis those that had some relation with the points described or articles that generate any doubt regarding their exclusion.

Finishing this first stage, a total of 39 articles were read in their entirety, of which 17 actually addressed the subjects sought and will have their contents analyzed in the discussion session. A flowchart with the synthesis of the research method can be seen in Figure 1.

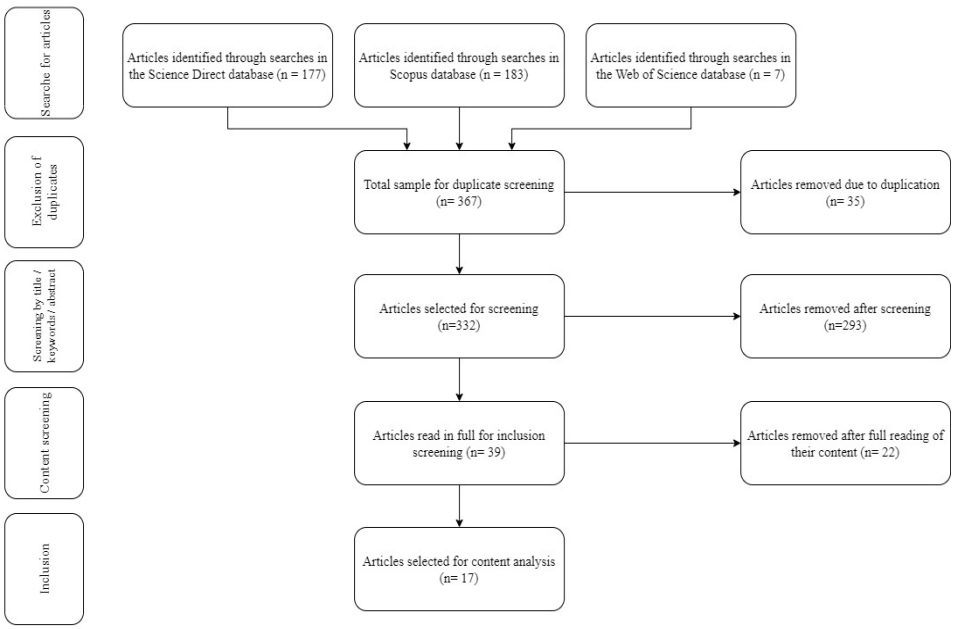


**Figure 1.** Flowchart with representations of the steps carried out throughout the literature review.

As previously highlighted, the research only addressed articles that involved DT models of processes, because of this objective, there were no results from articles prior to the year 2016, since the first works with DT models addressed only product models. To complement the theoretical basis of this study and enrich the analysis, an

exploratory search was carried out for the most important articles published before 2016 and that addressed the topic of DT, whatever were the types of models.

With the end of the research, it was noticed that some of the most cited articles in the literature, encompassing all that discuss the Digital Twin theme are *About The Importance of Autonomy and Digital Twins for the Future of Manufacturing,* by Rosen, Wichert, Lo and Bettenhausen, along with *Digital Twin in manufacturing: A categorical literature review and classification*, by Kritzinger, Karner, Traar, Henjes and Sihn, and *Shaping the digital twin for design and production engineering*, by Schleich, Anwer, Mathieu and Wartzack, having two of these three articles and their contents explored and addressed in this study.

## 3 Results and discussions

The following section seeks to present the descriptive analysis of the sample and the content analysis, together with a discussion of the literature.

### 3.1 Descriptive analysis

As described in the study method, a clipping was carried out focusing only on articles, which were written in English and Portuguese. Observing the results, as illustrated in Figure 2, it is possible to see that the theme has been showing a growth in terms of the number of publications, a phenomenon that is also related to the growing diffusion of the theme of Industry 4.0, not only in the academic environment, but also in the business world.

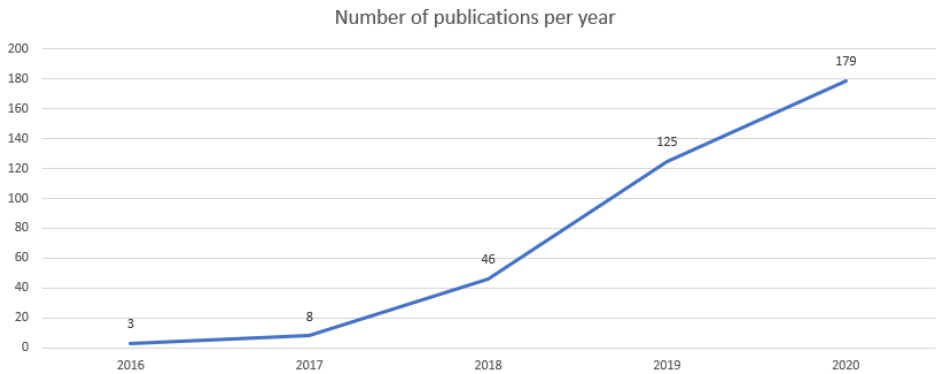


**Figure 2.** Annual evolution of the number of publications between the years 2016 and 2020.

In the sample, the year 2021 was omitted, although articles with publications scheduled for 2021 already appeared in the sample, due to the fact that the collection took place at the end of 2020, which contributed to the fact that this number of articles published in the year 2021 was not so expressive.

Moving on to an analysis of the main journals publishing these articles, it is observed that at least 7 journals have more than 10 registered publications, accounting for about 80% of the total records of the 332 publications under analysis.

The journals with the highest number of articles in the sample are Procedia CIRP (28 articles, 16.47% of the sample), together with the Journal of Manufacturing Systems

(28 articles, 16.47% of the sample), followed by Procedia Manufacturing (27 articles, 15.88% of the sample), IFAC-PapersOnLine (17 articles, 10.00% of the sample), Robotics and Computer-Integrated Manufacturing (14 articles, 8.24% of the sample), International Journal of Advanced Manufacturing Technology (12 articles, 7.06% of the sample) and IEEE Access (11 articles, 6.47% of the sample), while the other journals share the remaining 20% of publications. Figure 3 shows the ranking of the top 12 journals in this sample.

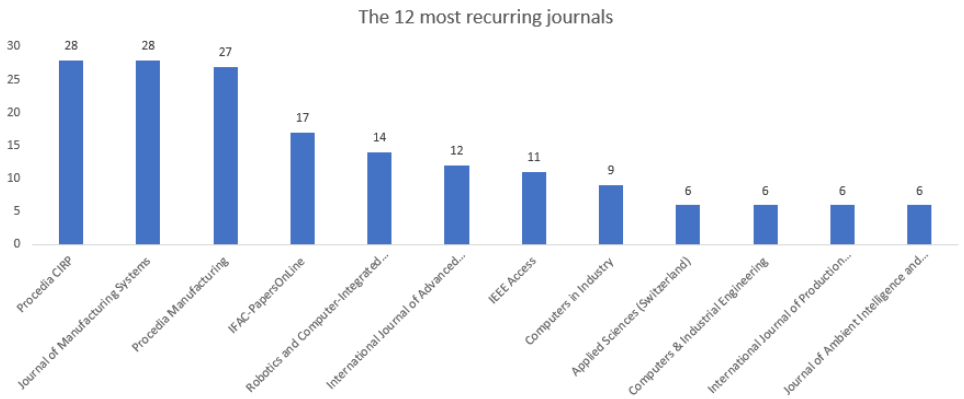


**Figure 3.** Distribution of publications among the 12 main journals in the sample.

Regarding the authors, there is no great emphasis on a specific author, there is a very uniform distribution, where the most recurrent author in the sample was Fei Tao (10 entries), followed by Jiewu Leng (7 entries) and Qiang Liu (7 entries).

Finally, looking only at the clipping of articles that loaded their keywords from the StArt software import, as seen in Figure 4, the 5 most cited were Digital Twin (43 entries, 22.40% of the total) followed by Industry 4.0 (39 entries, 20.31% of total), Smart Manufacturing (15 entries, 7.81% of total), Machine Learning (11 entries, 5.73% of total) and Cyber-Physical System (9 entries, 4.69% of total).

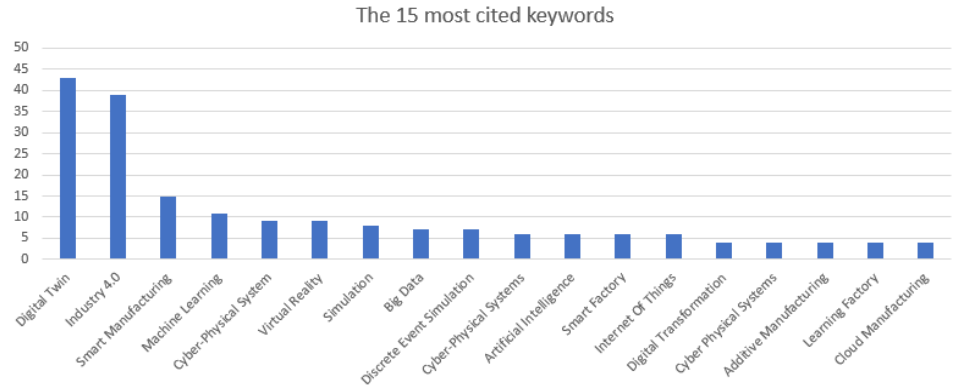


**Figure 4.** Distribution of the 15 most cited keywords.

## 3.2 Content analysis

After the proper reading and compilation of data/information, it was possible to observe commonalities in the Digital Twin studies, as well as points that diverge from one work to the other, mainly, the discussions focus on the architecture of model construction, communication between real and digital environment, data and information management and the functionalities/uses of the models, points that will be discussed individually in the next topics.

### 3.2.1 Construction architecture of Digital Twin models

As already pointed out, a commonly used architecture is the three-part division described in Tao et al. (2018), where it is possible to observe the physical/real environment, which is the object of study (a product, a service or an existing process), the digital environment, which would be the modeling that represents the characteristics of the real environment and the data connection between them, responsible for taking information from the real environment to be replicated in the digital environment, as well as to feed back the settings and data collected from the digital medium to the real medium, but this is not the only way to structure and build a Digital Twin model, as several other authors defend other architectures.

Qi et al. (2019) use a five-dimensional model in their work to characterize the construction of a DT, with another peculiarity of this work being the use of the word “dimension”, differing from the use of the word ” environment ” seen in Tao et al. (2018).

In this architecture proposed by Qi et al. (2019), the first dimension corresponds to physical entities, that is, the real environment that is intended to be represented, and the second is the virtual model that replicates the real environment. Moving on to the third dimension, there are utilities, representing the tools and technologies used in the model such as simulation and diagnosis of scenarios, while the fourth dimension is the data and information themselves and the fifth dimension is the connection that involves all four previous dimensions. The five-dimensional model subdivides the three environment cited in Tao et al. (2018), but both share many similarities.

Zhang et al. (2017b) demonstrate a four-dimensional architecture for building a model, the first dimension being the real environment from which data and information are taken; the second dimension is the three-dimensional model, whose function is to visually simulate the process and also generate data for analysis; the third dimension is the digital environment, responsible for analyzing and transforming data and information from the first two dimensions; and finally we have the fourth dimension, which would be an intermediate means that would carry out the communication between the other three, also having the function of storing the data.

Several other authors will use variations of these same architectures, each one with the objective of having more control in a given field of the model or of fragmenting another process aiming at greater customization. The choice for one of the architectures will depend on the construction objectives of the model and its own characteristics.

### 3.2.2 Communication and data capture in Digital Twin models

A gap identified with the reading of the articles was the consensus on how to capture and communicate data between the real environment and the model that represents it,

despite the existence of several variations on how to build a model, the form of this model communicating between its most diverse dimensions is not so simple.

In their work, Negri et al. (2020) even comment on the paradigm proposed by Kritzinger et al. (2018), something that is also explained in Watanabe (2020). The paradigm states that there are three types of digital replications and what differentiates the three is the way in which the communication between the real and the digital environment takes place.

The first type is the so-called Digital Model, being characterized by the need for manual communication between the real and the digital, there is no automatic way of communication between the parties.

The second type described is the Digital Shadow, where the communication from the physical to the digital medium is done in an automated way, but the way back, from the digital to the physical medium, needs to be done manually. Any insight or new information generated in the simulation is not automatically passed on to the real, it takes the manual intervention of an actor to make the change.

Finally, the last type described is the Digital Twin, which is characterized by an automated flow from the real to the digital, and vice versa, in this case, the digital model can communicate with the real and modify parameters and states without the need of human interference.

The aforementioned paradigm falls precisely on these last two types of communication, many of the models said to be Digital Twin, according to Negri et al. (2020), correspond to the Digital Shadow type model, and enabling both directions to be automated is a real challenge when building a DT model.

In addition to the paradigm, it is necessary to think about the complexity of ensuring real-time communication of data, something fundamental for various assumptions and objectives of the models, the work of Rolle et al. (2020) proposes the use of some software and means of communication to ensure a low latency between the generation of the data and its feeding into the model, in the study the capture and transmission of data is done through the Simulink platform and the communication standards used are the User Datagram Protocol (UDP) and Transmission Control Protocol/Internet Protocol (TCP/IP), in tests carried out, the low delay in real/digital communication was proven.

### 3.2.3 Data and information management

Another topic raised and widely discussed refers to the management of data and information using Digital Twin models, Qi et al. (2019) establishes dimensions in the model solely for the purpose of managing and storing data, other authors such as Bao et al. (2018) perform this function together with the dimension of communication between the real and digital environment.

According to Weyer et al. (2016), the model built must be able to test scenarios and measure the impacts of changes, whereas Moyne et al. (2017) also highlight the need for the model to represent the given physical element, being a faithful representation that is updated in real time, mirroring any changes and new states.

To make both possible, it is first necessary to think of ways to capture this data and information from the real environment, Bambura et al. (2020) talk about the use of sensors to capture the states of the physical environment, citing the use of motion, sound and light sensors, the use of historical data such as those from the company's Enterprise Resource Planning (ERP) is still cited. Bao et al. (2018) also mentions the

use of Programmable Logic Controllers (PLCs) and the extraction of data from them, as well as the capture of data with Radio Frequency Identification (RFIDs) to feed databases that will be read by the models.

In addition to capturing data, it is necessary to think about the issue of storage, both intermediate and final, for that, it is possible to observe the use of physical storage in local databases and the use of cloud storage, but with the recent scale in the amount of data generated, Zhang et al. (2017a) already cite the need to effectively incorporate Big Data in the construction of these models.

Moving on to the model itself, it is important to emphasize that this data must be managed, processed and used in order to enable performance monitoring and the generation of indicators for decision making. In this sense, Negri et al. (2020) describe the use of a model with two instances, the first being configured to monitor the performance and states of the physical environment and a second responsible for testing scenarios, performing changes independently, simulating results and in case of gains, changing the physical environment through feedback and real parameters changes.

It is possible to see that there are already paths and good practices for capturing, managing and storing data and information, but as seen in the texts, the escalation in the volume of data generated and the need to incorporate the use of Big Data in the models can be improvement opportunities for the models.

### 3.2.4 Features and uses

Finally, a topic that also generates some debate focuses on the functionalities and types of uses for a Digital Twin model, Lee et al. (2015) describe that a model must synthesize data from the physical environment in real time, while also storing information for analysis of historical series, in addition to simulating the performance of what is being represented.

Tao et al. (2018) complement this list of requirements with the need for the model to interact between its dimensions and adapt over time, where a change in one dimension would result in changes in the behavior of the others, something along the lines of Kritzinger's paradigm. Tao et al. (2018) also state that the model must present an evolutionary characteristic, improving with time and the accumulation of data, in addition to following the evolution of the real environment.

In their works Negri et al. (2020) and Jiang et al. (2020) describe a very similar functionality, which, in their view, is important for DT models, the creation of distinct instances with the objective of monitoring the performance of the environment, together with the simulation of scenarios.

Analyzing the objectives of such a model, Weyer et al. (2016) mention the possibility of testing scenarios with the variation of parameters and states, as well as the use when measuring the impacts of changes, whereas Moyne et al. (2017) add that the models can be used to generate predictions about the performance of the physical environment, as well as to identify critical points to the functioning of the same.

## 4 Conceptual framework for building a Digital Twin

After analyzing the sample obtained, it is possible to notice that there is more than one way to build a Digital Twin model, as well as there are several resources, tools and

software that can support this construction, software such as ARENA, Powersim and WITNESS, are cited in Guizzi et al. (2019). Another example is FlexSim, cited in Zhang et al. (2017b), a proprietary software for building simulation models, which has been updated over time and claims to have the ability to support the construction of Digital Twin models.

As many of these software requires licenses for use, and some do not offer easy access to demo or student versions, an opportunity was identified to seek viable alternatives, the use of free software, to build these types of models, thus enabling the dissemination of these studies and tools in the academic environment and their use for the representation of simpler environments.

Identifying this opportunity, searches were made to understand the tools that would be necessary for this construction, thus making a compilation of software that does not need a license for its use or that are open source and that would support this final objective, in addition, a conceptual framework was developed for the construction of a Digital Twin model, although general, the framework already indicates the important steps that can be followed in the construction of the model.

Some assumptions adopted for this framework were that it will initially correspond to a process Digital Twin, built with three dimensions, the physical environment being any process, and the digital environment being its representation on a platform that supports the requirements and the communication/data management between them.

## 4.1 Understanding the needs for building the model

Throughout the bibliographic review, it was possible to notice several requirements, assumptions and functions that the DT models should fulfill, contemplating all these points and at the same time using open/free tools that would help in the greater dissemination of the models is not an easy task.

Unlike the paid options, which often offer an entire environment for creating a model from scratch to its final version, without the need to integrate with a second software, here it was necessary to look for several software that could collaborate with the construction process, and that, at the same time, they had compatibility with each other to enable the workflow.

The first need was for a software that would allow the construction of objects and entities involved in the process that will be replicated in the model, because as seen in some studies, the visual part, especially when it comes to product and process models, is something important, with this, it would be necessary to select a modeling, texturing and three-dimensional animation software, which would make it possible to replicate the characteristics of real entities to digital models.

After obtaining these digital models, it would also be necessary to find a software that could operationalize the Digital Twin, a software that should simulate behaviors and physics, provide good visualizations and have the ability to read, generate, manage and extract data, for this function the possibility of using game engines was observed, software created for the construction of video games and that include most of these premises.

In addition to the two software already listed, it would be necessary to find a way to establish communication between the real environment and the game engine, to control its behavior, store/manage the data and make the model functional, for this we came to the conclusion that the programming language interpreters could be used, which are code editors that allow the creation of an infinity of functions and subroutines ranging

from reading files on a machine, to sending commands to the game engine to execute shares.

As seen, it was concluded that, considering the need to use free/free software, at least three software would be indispensable to support the creation of a DT model, while using proprietary software to build DT, this same model could be created without the need to learn or have to integrate a second software into the workflow.

This need to use more than one software brings with it some negative points, such as a longer learning curve, since it would be necessary to learn three software instead of just one, another negative factor is the risk of incompatibility between one software and another, which can make the desired objective unfeasible and, finally, a greater workload, since the integration of the software would be one more task, where the use of just one would eliminate it.

Choosing free software, on the other hand, also brings benefits, one that can be easily mentioned is the enhancement of the dissemination of knowledge and products, as it eliminates the need to spend money to have access to both, something very relevant for theoretical studies and applications in small businesses, which justifies the objective of building this framework.

Finally, after recalling the necessary requirements, listing the types of software needed for this construction, reporting the trade-off between using a paid option or opting for free options, it is now necessary to list possible candidate software and which would be chosen to help in building the model.

## 4.2 The choice of software

To make the choice of software, aiming at the construction of the conceptual framework, an exploratory research was carried out, which means that the software listed or chosen here are not the only ones available, not even the best, they are only the ones that were identified and that had more affinities with the intended use at the moment.

### 4.2.1 *Game Engines*

The first set of software analyzed was that of the game engines, since the choice between one of the available ones can impact on compatibilities regarding the modeling software and especially regarding the programming code interpreter, since each game engine works with a specific programming language.

Game engines are used in the creation of video games, therefore there are software for creating two-dimensional environments (2D) and software for creating three-dimensional environments (3D), as the objective of this work is to represent the physical environment, a focus was given in engines that support the creation of 3D environments.

Another very important point is that there are several game engines in use on the market, but some are not released to the public, on the contrary, we have open source engines, which are freely available for use, as is the case with Unity, Unreal Engine, Godot and the Cry Engine, which will be analyzed here in the search for the one that best suits the use in the construction of the model.

Chart 1 compiles the exploratory comparison carried out with the four engines, for the choice between them, we took into account, mainly, the factors that facilitated the

construction of the model or factors that reduced the learning curve for a more inexperienced user, after analyzing each one, it was noticed that Unity had a more user-friendly interface, much simpler than Unreal and Cry Engine, another very positive factor of Unity is the huge user base that moves its forums, where can answer several questions, receive help from other more experienced users and even find tutorials and ready-made content very easily, and finally, another positive factor for Unity compared to the other three available, was the large number of courses available on the internet, from courses for beginners to more advanced courses, some even available for free.

As Unity met all the requirements expected by the game engine and presented several extra factors, mainly those related to the reduction of the learning curve, it was chosen to be the engine used in the framework.

**Chart 1.** Synthesis of the exploratory comparison between the analyzed game engines.

| **Software: Game Engine** | | | | | |
|---|---|---|---|---|---|
| **Criteria** | **Group** | **CryEngine** | **Godot** | **Unity** | **Unreal Engine** |
| | | **Meeting the criterion** | **Meeting the criterion** | **Meeting the criterion** | **Meeting the criterion** |
| Ease of Use | General features | Quite complete software, but with a high degree of complexity in terms of its interface and use | Very simple and intuitive interface and use | Very simple and intuitive interface and use | Relatively complex interface and usage |
| Availability of resources on the internet | | Average amount of content (assets and ready-made materials) available on the internet | Average amount of content (assets and ready-made materials) available on the internet | Large amount of content (assets and ready-made materials) available on the internet | Large amount of content (assets and ready-made materials) available on the internet |
| Available documentation | | Available on the website | Available on the website | Available on the website | Available on the website |
| User community | | Forum with up-to-date discussions, but with an average number of active members | Forum with up-to-date discussions, but with an average number of active members | Forum with up-to-date discussions, and a large number of active members | Forum with up-to-date discussions, and a large number of active members |
| Importing Data (.xls;.csv;.txt) | Importing | Compatible | Compatible | Compatible | Compatible |
| Importing 3D files | | Compatible | Compatible | Compatible | Compatible |
| Importing images | | Compatible | Compatible | Compatible | Compatible |
| Importing sounds | | Compatible | Compatible | Compatible | Compatible |
| Project exporting | Exporting | Limited range of exporting and generating executables | Considerable range of exporting | Considerable range of exporting | Extensive list of possible exports |

**Chart 1.** Continued…

| Software: Game Engine | | | | | |
|---|---|---|---|---|---|
| Compatibility with Mobile | | Still in testing phase | Compatible | Compatible | Compatible |
| Creation of Cameras | Functionalities | Compatible | Compatible | Compatible | Compatible |
| Creation of Animations | | Compatible | Compatible | Compatible | Compatible |
| Trigger of Events | | Compatible | Compatible | Compatible | Compatible |
| Physics Elements | | Compatible | Compatible | Compatible | Compatible |
| Use of code language | Programming | Use of C++ language | Use of its own language, GDScript, but there is support for the use of the C# language | Use of C++ language | Use of C# language |

### 4.2.2 3D modeling software

To perform the three-dimensional modeling of objects, there are a series of software with different functionalities and focuses, from the most specific to engineering such as AutoCAD, SolidWorks and Inventor, to those with a greater focus on artistic modeling such as 3DS Max and Maya.

As with game engines, there are paid software, such as those mentioned above, as well as free and/or open-source software, which do not lack anything in terms of the expected features. As the focus of this choice involves the identification of software that allows the modeling of elements that replicate those seen in the physical environment, it was decided to choose a more artistic software, and for the analysis Blender, Clara.io and Wings 3D were selected.

Chart 2 presents the exploratory comparison of the three software, and as with the game engine, factors that reduced the learning curve for the user had a greater weight in the choice of one of the software.

As much as the three software performed very similar roles and functions, Blender ended up standing out more in relation to its competitors, for presenting a larger list of resources and for having a more friendly and stable interface (Clara.io, for being a online software, had several problems during use), but what weighed the most in the decision to use Blender, as it was with Unity, was the large number of active users in the forums, answering questions, providing materials and tutorials and above all the vast list of available courses that would help a lot even the most inexperienced with modeling software.

**Chart 2.** Synthesis of the exploratory comparison between the analyzed modeling software.

| Software: 3D Modeling | | | | |
|---|---|---|---|---|
| **Criteria** | **Group** | **Blender** | **Wings3D** | **Clara.io** |
| | | **Meeting the criterion** | **Meeting the criterion** | **Meeting the criterion** |
| Ease of Use | General Features | Very simple and intuitive interface and use | Very simple and intuitive interface and use | Very simple and intuitive interface and use |
| Availability of resources on the internet | | Large amount of content (assets and ready-made materials) available on the internet | Large amount of content (assets and ready-made materials) available on the internet | Large amount of content (assets and ready-made materials) available on the internet |
| Available documentation | | Available on the website | Available on the website | Available on the website |
| User community | | Forum with up-to-date discussions and a large number of active members | Forum with up-to-date discussions, but with an average number of active members | Forum with up-to-date discussions, but with an average-low number of active members |
| Importing of drawings for modeling | Importing | Compatible | Compatible | Compatible |
| Importing 3D files | | Compatible | Compatible | Compatible |
| Importing textures | | Compatible | Compatible | Compatible |
| Exporting Files | Exporting | Extensive list of possible exports | Extensive list of possible exports | Extensive list of possible exports |
| Modeling of 3D shapes | Functionalities | Compatible | Compatible | Compatible |
| Creation of Textures | | Compatible | Compatible | Compatible |
| Creation of Animations | | Compatible | Not Compatible | Compatible |

### 4.2.3 Programming code interpreter

As Unity was chosen to be the game engine used, therefore it was necessary to use an interpreter that accepts scripts in C# (“C Sharp”). Among the available options, there are Visual Studio and MonoDevelop, in terms of interface and functionality, both are very similar, but Visual Studio has greater support, since it belongs to Microsoft, a large and recognized company, potentially offering greater reliability.

It was also noted that Visual Studio presents a more complete documentation on the internet, something very important for beginners in a new programming language, because there it is possible to understand the construction of scripts, including the exposition of examples, it was also noticed the existence of various training and tutorials on the use of Visual Studio with Unity, which contributed to its choice as the programming code interpreter.

## 4.3 Proposal of a conceptual framework

Given the choice of software and resuming the concepts seen throughout the literature review, a conceptual framework was elaborated for the implementation of a Digital Twin in a game engine, for the same, the Blender software was used to build the three-dimensional entities of the model, Visual Studio for creating the scripts that command the model and manage the data and Unity as a building platform.

For the simplicity aimed at, a general step-by-step will be built for a process Digital Twin model. The same step can be changed depending on the needs of the model to be built, as well as, with some modifications, it can be used in the construction of product models or service monitoring.

A summary of the conceptual framework can be seen in the flowchart in Figure 5, where it is possible to have a macro view of all the steps and the execution order of each step. It is worth mentioning that this figure was built to be a summary of the framework that will be explained below, not being based on one or another work in the literature, but on the proposal made through the study.

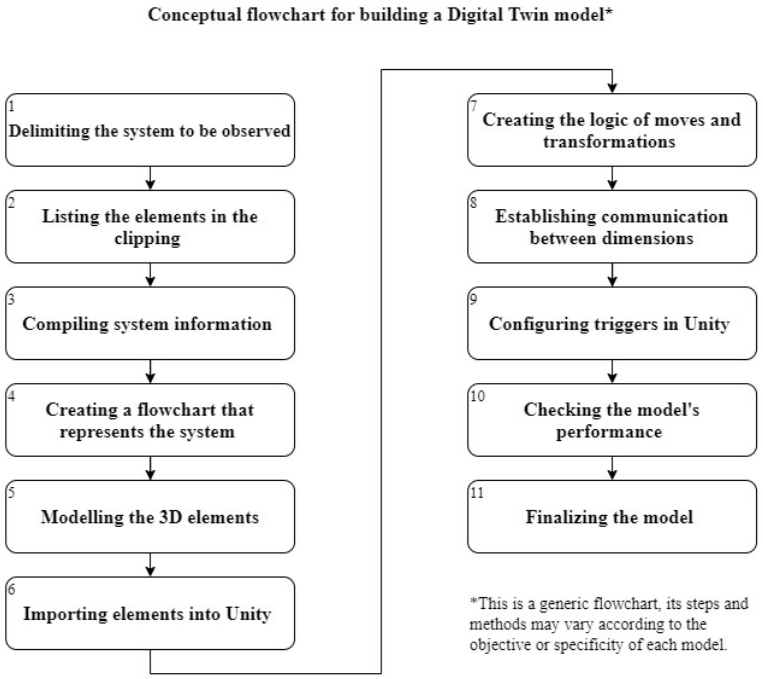


**Figure 5.** Step-by-step representation for the construction of a Digital Twin model built according to the premises seen throughout the research.

**Step 1: Delimiting the system to be observed –** To start building the model, it is first necessary to understand the context and define the system clippings that will be represented. It is necessary to pay attention to the inputs and outputs in this system, to understand the transformations that occur in the environment and how it behaves during the process, with this, it will be defined what will be included in the model, where it will start and end, in addition to make clear what will not need to be explored.

**Step 2: Listing the elements present in the clipping –** After defining the clippings, it is important to list all the items and elements that make up the analyzed process, which are spaces, buildings, equipment, human-machine interactions, objects, inputs and outputs. This listing will help in compiling the information, building the process flowchart, and eventually identifying any points that need attention or are being ignored for the next steps.

In the second step, a brief description of the behavior of the system can also be made, listing the movements, transformations and events that happen along with the dynamics of the process over time, already relating these points to the items and elements listed.

**Step 3: Compiling system information –** With the list finalized, the objective becomes to identify what information the real environment is already capable of generating and sending to the virtual model, as well as identifying gaps and points that may need adjustments so that there are sufficient inputs for the virtual model work as expected.

**Step 4: Creating a flowchart that represents the system –** After defining the cutout, elements and compilation of data and information that the physical dimension already generates, it is necessary to build a process flowchart, highlighting the above points and pointing out points of attention regarding the needs for information exchange.

The flowchart created can also be used as a source of consultation in identifying gaps for the preparation of the model, we can point out in it, points that should generate an input in the system, but which are not operational at the moment. The correction of these points is a separate case, since we will have more than one possible cause or problem, making it impossible to survey a single correction strategy, because of this, the present text will not focus on the attempt to list possible methods for this treatment.

**Step 5: Modelling the 3D elements–** With all the initial study completed, we move on to the modeling stage of the elements that will compose the model. This step will have the help of Blender, here any physical entities that will be transferred to the Unity environment will be modeled, along with the creation of animations and the creation of textures.

It is worth remembering that the modeled elements must faithfully represent the physical elements, both in terms of shapes, dimensions, as well as functioning. The information already collected will guide this construction, it is important to pay attention to the movement and transformation points of the process, since they require some specificity when being modeled. As in the previous step, there are several ways and means of doing this process, it is not up to this framework to indicate an ideal.

**Step 6: Importing elements into Unity –** After modeling all the entities, it is necessary to create a project in Unity for the construction of the model, after creating the project, we will have access to the so-called “scenes”, which are the environments where we will import the elements modeled in Blender, also called assets. It is also necessary to position all the elements in the scene, respecting positions, dimensions and any other points of the real environment, in addition to preparing the elements that will be changed during the simulation.

To assist in the import of elements, construction of scenes and other settings, it is possible to use tutorials and courses available on the internet.

**Step 7: Creating the logic of moves and transformations –** With the scene built in Unity, there is a need to create movement logics and codes to simulate behaviors from data and information from the real environment. This scripting will be done using Visual Basic, which interprets the C# language and will involve a steeper learning curve for dealing with reading files and changing parameters.

A guide to know which points will need to be configured and which will communicate are the data collection in the first steps, in particular, the flowchart with the details of the studied process.

**Step 8: Establishing communication between dimensions –** From the constructed scene and the logic inserted in the scripts, it is necessary to establish a communication between the model in Unity and the data source, both to take points from the real environment to the digital, as well as from the digital to the real. A tip is to use the reading and writing of files within a fixed folder to avoid conflicts or the need to change paths for the model to work, but beyond that point, it is not possible to list a single method for all models. The communication process will depend on the programming done and also what the data sources are and what data these sources generate.

**Step 9: Configuring triggers in Unity –** So that the codes can control the model within Unity, it is necessary to define triggers and execution routines that will be activated from a signal, an event or some data received from the real environment, the trigger will be just an activator that will trigger an animation, a change of state or an event in the model in Unity so that there can be a correspondence between changes in the physical environment and the digital environment.

**Step 10: Checking the model's performance –** After configuring the triggers, it is necessary to test the model, observe that the communication established is working and that the model is behaving as expected, if necessary, you can review the previous steps to ensure that nothing has been left out and do the necessary changes. The model can also be built iteratively, a simpler version of the process can be made, with the main points, and over time, more detail is added to this already more stable and functional version.

**Step 11: Finalizing the model –** After the verification and the necessary corrections made, the visualizations and the creation of the interface are created, both points are secondary and do not interfere with the functioning of the model, only the visualization and user familiarization with it.

With this, the generic step-by-step for building a Digital Twin model within a game engine is finalized, these are the main steps that will be taken to build a model, but depending on the process, some changes may be necessary, as well as the insertion of more steps, aiming to behave what is expected within the Unity environment.

## 5 Conclusions

This study contributes to the attempt to collaborate with the dissemination of themes related to the Digital Twin models, both in the academic environment and in practical applications, aiming above all to seek ways at no cost to build and have access to these models. This study also highlights the growing number of publications on Digital Twin in recent years and the existence of different definitions regarding the characteristics, functionalities and uses of these models.

Despite this variety of definitions, a more general and aggregating definition can be sought, where a Digital Twin is a simulation model that seeks to reliably represent a real environment, enabling communication and data exchange between them in real time, while step that makes it possible to monitor performance, test scenarios and capture data and insights to support decision making.

Although there is an increasing number of articles in the literature, there are still a number of unexplored opportunities and gaps, in addition to a vast field for experimentation. It is evident that the first works elaborated in the area focused on the construction of models for products, its origin was in the study of the life cycle of a product, but the evolution and expansion of the uses of the models also encompassed the representation of processes and the use of these products/processes.

In addition to expanding the scope of what is sought to be represented, with the advancement of technology, there has also been an advance in the architectures used, from the most basic with three dimensions, to the most complex with five dimensions, which have a special focus on tools and data management. With the advancement of technology, there was also the addition of new ways of collecting data and states such as RFIDs, as well as ways of storing this data, in the case of Big Data.

The universe that surrounds the Digital Twin models is expanding more and more, and it is up to these models to adapt and evolve to continue to offer new solutions and increasingly refined uses.

It was also possible, throughout the work, to identify a gap and explore it, aiming at the dissemination of this knowledge, a conceptual framework was built, which sought to list steps for the construction of a Digital Twin model in a game engine, a software that was born fulfilling the function of being used for the elaboration of virtual games.

The framework developed was premised on helping to build a model that would represent a process, but could easily be adapted to represent a product. In addition, the desire to use software that does not require the use of purchased licenses was also adopted, since throughout the analysis of the sample, several of the mentioned software were not easily available for testing or with student licenses.

The framework considered the use of three software, Blender, Visual Studio and Unity, in addition to presenting 11 steps that can guide people with experience with these programs to build a Digital Twin. The framework is just a conceptual model and it can, and should, be refined and adapted in future work.

Future studies also related to the Digital Twin can be focused on areas not explored in depth here, such as the adaptation of the physical environment for its construction, since we may have elements that were not designed thinking that in the future we would have the possibility of capturing data from them, another very extensive field for studies is the data generated and used, in addition to having many possibilities for studies regarding the model itself.

The present study is limited in terms of the research methods used, as well as the way in which the sample was searched, important studies may not have fit the filters used and, therefore, were not part of these analyses. The databases used were ScienceDirect, Scopus and Web of Science. It is also possible to highlight that the most recent articles may also have been excluded from the sample used because they were published after collection or because they had a low chance of having a high number of citations.

## Acknowledgements

This work was developed through the Scientific and Technological Initiation Program of UFSCar with support from the Laboratory of Ergonomics, Simulation and Project of Productive Situations – PSPLab, through the SimuCAD Group, from DEP-UFSCar.

## References

Bambura, R., Solc, M., Dado, M., & Kotek, L. (2020). Implementation of digital twin for engine block manufacturing processes. *Applied Sciences (Basel, Switzerland)*, 10(18), 6578. http://dx.doi.org/10.3390/app10186578.

Bao, J., Guo, D., Li, J., & Zhang, J. (2018). The modeling and operations for the digital twin in the context of manufacturing. *Enterprise Information Systems*, 13(4), 1-23. http://dx.doi.org/10.1080/17517575.2018.1526324.

Dalenogare, L. S. (2018). *A Indústria 4.0 no Brasil: um estudo dos benefícios esperados tecnologias habilitadoras* (Dissertação de mestrado). Escola de Engenharia, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Eliana, F. M. C., Morales, A. M., & Hengler, C. C. B. (2019). Revisão bibliográfica sistemática sobre ambientalização curricular no ensino superior: Uma análise com foco no ensino jurídico. *POIÉSIS Revista do programa de pós-graduação em educação*, 13(23). https://doi.org/10.19177/prppge.v13e232019122-141.

Fritsch, D., & Kada, M. (2004). Visualisation using game engines. *Archiwum ISPRS*, *35*(5), 1-5.

Grieves, M. (2014). *Digital Twin: manufacturing excellence through virtual factory replication*. USA: Florida Institute of Technology.

Guizzi, G., Falcone, D., & Felice, F. (2019). An integrated and parametric simulation model to improve production and maintenance processes: towards a digital factory performance. *Computers & Industrial Engineering*, 137, 106052. http://dx.doi.org/10.1016/j.cie.2019.106052.

Hernandes, E., Zamboni, A., Fabbri, S., & Di Thommazo, A. (2012). Using GQM and TAM to evaluate StArt – a tool that supports Systematic Review. *CLEI Electronic Journal*, 15(1). http://dx.doi.org/10.19153/cleiej.15.1.2.

Jiang, H., Qin, S., Fu, J., Zhang, J. & Ding, G. (2020). How to model and implement connections between physical and virtual models for digital twin application. *Journal of Manufacturing Systems*, 58(B), 36-51. http://dx.doi.org/10.1016/j.jmsy.2020.05.012.

Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital Twin in manufacturing: a categorical literature review and classification. *IFAC-PapersOnLine*, 51(11), 1016-1022. http://dx.doi.org/10.1016/j.ifacol.2018.08.474.

Lee, J., Bagheri, B., & Kao, H. (2015). A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3, 18-23. http://dx.doi.org/10.1016/j.mfglet.2014.12.001.

Moyne, J., Qamsane, Y., Balta, E., Kovalenko, I., Faris, J., Barton, K., & Tilbury, D. (2017). A requirements driven digital twin framework. *Specification and Opportunities. IEEE*, 8, 107781-107801. http://dx.doi.org/10.1109/ACCESS.2020.3000437.

Negri, E., Berardi, S., Fumagalli, L., & Macchi, M. (2020). MES-integrated digital twin frameworks. *Journal of Manufacturing Systems*, 56, 58-71. http://dx.doi.org/10.1016/j.jmsy.2020.05.007.

Qi, Q., Tao, F., Hu, T., Anwer, N., Liu, A., Wei, Y., Wang, L. & Nee, A. Y. C. (2019). Enabling technologies and tools for digital twin. *Journal of Manufacturing Systems*, 58(B), 3-31. https://doi.org/10.1016/j.jmsy.2019.10.001.

Quinalha, E. (2018). *Gêmeos digitais, o futuro da indústria 4.0: estudo de caso* (Monografia de especialização). Universidade Tecnológica Federal do Paraná, Curitiba.

Rolle, R., Martucci, V., & Godoy, E. (2020). Architecture for digital twin implementation focusing on Industry 4.0. *IEEE Latin America Transactions*, 18(5), 889-898. http://dx.doi.org/10.1109/TLA.2020.9082917.

Rosen, R., Wichert, G., Lo, G., & Bettenhausen, K. D. (2015). About the importance of autonomy and Digital Twins for the future of manufacturing. *International Federation of Automatic Control*, 48(3), 567-572. https://doi.org/10.1016/j.ifacol.2015.06.141.

Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., & Sui, F. (2018). Digital twin-driven product design, manufacturing and service with big data. *International Journal of Advanced Manufacturing Technology*, 94(9-12), 3563-3576. http://dx.doi.org/10.1007/s00170-017-0233-1.

Tropia, C. E. Z., Silva, P. P., & Dias, A. V. C. (2017). Indústria 4.0: uma caracterização do sistema de produção. In *XVII Congresso Latino-Ibero-americano de Gestão Tecnológica*. Cidade do México.

Watanabe, A. (2020). *Modelagem de uma planta virtual de produção de PCBs via digital twin dentro do contexto da Indústria 4.0* (Dissertação de mestrado). Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, Florianópolis.

Weyer, S., Meyer, T., Ohmer, M., Gorecky, D., & Zühlke, D. (2016). Future modeling and simulation of CPS-based factories: an example from the automotive Industry. *IFAC*, 49(31), 97-102. http://dx.doi.org/10.1016/j.ifacol.2016.12.168.

Zhang, H., Liu, Q., Chen, X., Zhang, D. & Leng, J. (2017a). A digital Twin-Based approach for designing and multi-objective optimization of hollow glass production line. *Special Section on Key Technologies for Smart Factory of Industry 4.0*, 5, 26901-26911. https://doi.org/10.1109/ACCESS.2017.2766453.

Zhang, Q., Zhang, X., Xu, W., Liu, A., Zhou, Z., & Pham, D. (2017b). Modeling of digital twin workshop based on perception data. In Y. Huang, H. Wu, H. Liu, & Z. Yin (Eds.), *Intelligent Robotics and Applications. ICIRA 2017. Lecture Notes in Computer Science* (Vol. 10464). Cham: Springer. https://doi.org/10.1007/978-3-319-65298-6_1.