applied sciences

MDPI

Article

Use of a Software Application to Generate a Sequence for Simulation Model Creation

Martin Ďuriška [1], Gabriel Fedorko [1,*], Jana Fabianová [1], Vieroslav Molnár [2], Hana Neradilová [3] and Filip Dolák [1]

[1] Faculty of Mining, Ecology, Process Control and Geotechnologies, Technical University of Košice, Park Komenského 14, 040 01 Košice, Slovakia
[2] Faculty of Manufacturing Technologies, Technical University of Košice with a seat in Prešov, Bayerova 1, 080 01 Prešov, Slovakia
[3] The College of Logistics, Palackého 1381, 750 02 Přerov, Czech Republic
* Correspondence: gabriel.fedorko@tuke.sk; Tel./Fax: +421-55-602-3143

Abstract: The use of simulation models is currently a necessity, considering the complexity of the problems solved in many areas of engineering and scientific work (including logistics). This fact places high demands on their creation, use, and possible modification. When implementing simulation models, we often encounter limitations (depending on the software used), for which an effective solution is the application of the additional programming method. However, the method's implementation is often associated with problems, the solution to which is, for example, the use of third-party services. Ultimately, this is an effective but lengthy process. The paper presents a new approach to the method of additional programming, which is based on using an addressable software application as a tool for generating sequences according to defined input parameters. The research was carried out using the simulation software Tecnomatix Plant Simulation and the software tool Simtalk, which is part of it.

Keywords: simulation; additional programming; sequence; software application; generation

Citation: Ďuriška, M.; Fedorko, G.; Fabianová, J.; Molnár, V.; Neradilová, H.; Dolák, F. Use of a Software Application to Generate a Sequence for Simulation Model Creation. *Appl. Sci.* **2023**, *13*, 5433. https://doi.org/10.3390/app13095433

Academic Editor: Arkadiusz Gola

Received: 31 March 2023
Revised: 19 April 2023
Accepted: 25 April 2023
Published: 27 April 2023

1. Introduction

Planning, organization, and management of any logistics system or part of it are often very demanding in terms of available sources of relevant information. Therefore, it is advisable, not only at the beginning of the whole process but also during its course, to find the most effective way to secure available sources of information with regard to their informative value and subsequent application possibilities [1]. One of the tools that allows us to find a suitable and effective way of using information resources is a computer simulation, which is already an integral part of logistics in the era of Industry 4.0 [2].

The simulation model is a powerful tool to optimize and identify bottlenecks in logistics processes [3]. The method of modeling and simulation of production logistics systems, whose main objective is to find and eliminate bottlenecks, was presented in [4]. The bottleneck analyzer in TX Plant Simulation software was used, and a method of optimizing logistics flows was tested by the Kanban pull system in [5]. A simulation can be defined as the investigation of a real process based on its model in a simulation system. The simulation results can be applied to every element of the real logistics system and at all stages of the process [6]. In contrast, the use of mathematical models to optimize complex processes is not very efficient, and sometimes it is even impossible [7]. Similarly, experimentation on a running live system is not the best solution. In the search for the optimal solution to the studied process, it is possible to make rapid parameter changes in the simulation system, experiment with elements not yet present in the real system, and investigate the effects of these changes immediately [8]. In this way, we can explore multiple solution alternatives in a short time and then apply the correct and most advantageous solution

to the real system [9]. Therefore, the use of a simulation approach through appropriate software tools is a suitable and efficient choice to solve the problem of optimizing logistic processes and flows [10].

Before creating a simulation model, it is necessary to determine the depth to which the process needs to be investigated [11]. The accuracy and correctness of the simulation results are based on the agreement of the parameters of the real system elements with the simulation model. When using the simulation method, the process under study is successively composed of interconnected sub-processes that build upon and influence each other [12].

The choice of the right simulation software is also an important decision. There is now a large variety of high-quality simulation software to choose from. Each simulation software has its advantages and disadvantages. Simulation software developers often try to focus on a specific problem, so they create specialized simulation software, which can be narrowly focused or designed for a wide range of applications.

Additionally, simulation systems are rapidly developing and improving [13]. They include new functions, new adjustable parameters, or new blocks so that it is possible to create the required simulation model in the simplest possible way [14]. Currently, simulation systems have reached such a level that it is possible to create a large spectrum of diverse simulations by connecting pre-programmed blocks. In this way, the user of the simulation system can simulate various situations, cases, and processes without a deep knowledge of programming or programming languages, based only on their ability to control the simulation software.

However, sometimes we may encounter a specific process whose model cannot be created using the block joining method. It is where simulation software developers came up with a solution that allows the user to reprogram their functions using a programming software tool. This opens up a vast simulation potential for users that is practically limited only by the user's programming abilities. On the contrary, an experienced programmer should be able to create and program almost anything, as presented in [15], where the authors described how the simulation mechanism SimTalk could be built into the Internet of things platform.

However, on the basis of the mentioned facts, the question arises as to whether it is possible to use the method of additional programming as a tool for streamlining the work with the simulation tool even without in-depth knowledge of programming. As part of the presented research, an answer to the above question was sought. The research was carried out on the simulation platform of Tecnomatix Plant Simulation, which is primarily designed for logistics and logistics processes. In Tecnomatix Plant Simulation, simulation models are created by connecting blocks that have their own functions and adjustable parameters. At the same time, the program offers an additional programming method to increase the functionality and analytical possibilities of simulation models and the process of their creation and use.

Furthermore, the presented research was not focused only on unilaterally and a specific type of simulation platform. The aim is to generalize the knowledge obtained with regard to its possible use in other existing simulation tools.

The aim of the research and the subsequent paper is to fill the gap related to the investigation of the possibility of simplifying the creation and modification of simulation models. It supports the broader use of computer simulation as a tool of common engineering practice for reassessing decision-making processes. The intention, based on the context of Industry 4.0, presents computer simulation as one of its main methods. The article attempts to point out that even with basic knowledge of working with simulation software, it is possible to quickly and efficiently create different types of simulation models through a simple address application and then actively use them.

2. Materials and Methods

The solution of various engineering tasks currently places increased demands on the effective use of software tools in order to deliver fast and accurate results. The difficulty and variety of individual problems and assignments are directly related to the requirements for the desired results. Often, already in the initial stages of the project, the conclusion is reached that the commonly available functionalities and modules of the individual software used are insufficient to achieve the expected result, or that, thanks to them, it is possible to achieve only a limited (inaccurate) series of results, often not corresponding to reality.

Currently, it is possible to identify the three most commonly used approaches in the field of logistics to create and use simulation models in the field of logistics with the help of an external specialist. Specifically, these are the classical, modified, and progressive approaches. In addition, there are, of course, other approaches. However, within each approach, the method of additive programming plays an important role.

2.1. Selected Approaches to the Use of Simulation Models in the Field of Logistics

The classical approach (Figure 1) to the use of simulation models in logistics is based on the utilization of a specialist in simulation model development. The whole model creation process from the beginning, every model change, and the implementation of individual analyses and simulation experiments are carried out by a specialist in a computer simulation. The client (logistics specialist) only enters the requirements and processes the delivered results.

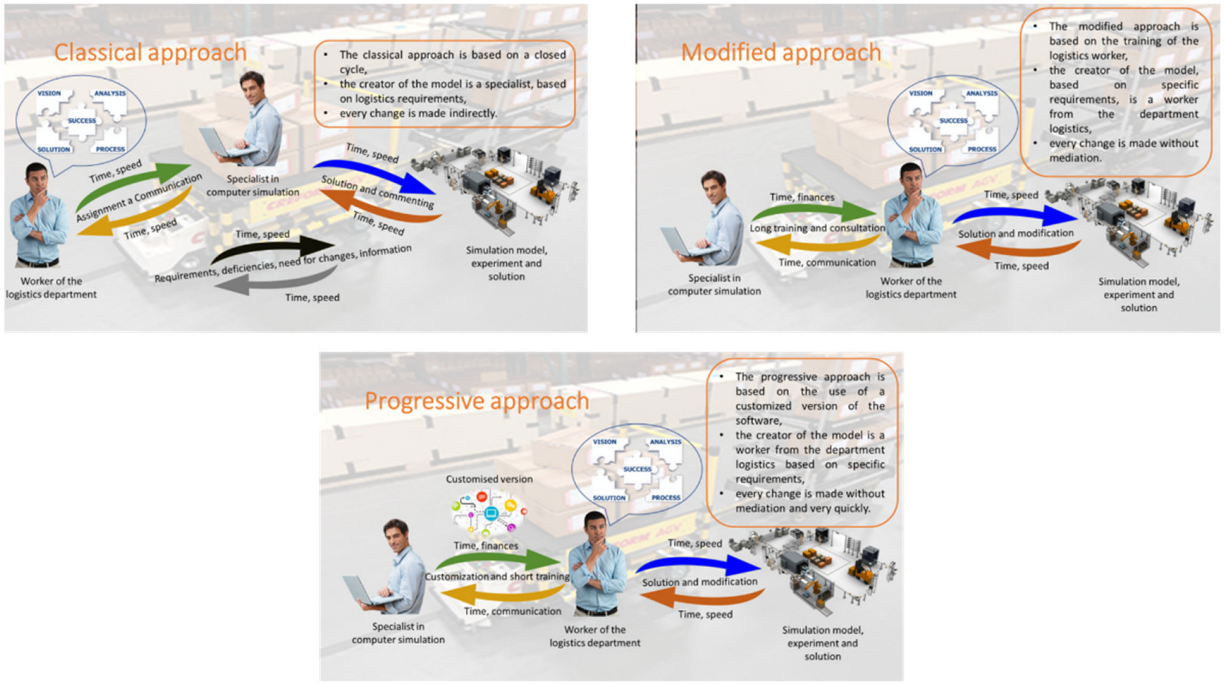

Figure 1. The principle of classic, modified, and progressive approach to the use of simulation models in the field of logistics.

The modified approach is based on an approach in which a logistics worker is trained to work in a simulation program. Subsequently, he can use the simulation tool to create his own models and perform analyses. Problems and ambiguities can be consulted with a computer simulation specialist if necessary.

The progressive approach is based on a more sophisticated method when a specialist in the field of computer simulation adjusts the software version in accordance with the needs of a worker in the field of logistics. At the same time, it adapts individual control elements to the needs in terms of model creation and the realization of simulation experiments. Nevertheless, it is possible to consult the computer simulation specialist again if necessary.

As already mentioned, in all of the above cases, the method of additive programming is extensively used. It is most often implemented in the form of text commands in the respective programming software, which create a program sequence. This method, in order to be used effectively, requires detailed knowledge of the individual commands, the form of their notation (transcription), and above all, very often, at least a basic or advanced knowledge of programming.

When creating more complex simulation models requiring specific functions, we may encounter limitations that hinder obtaining the desired results. That means the user cannot implement the simulation model and subsequent simulation experiments according to the initial requirement using the built-in modules of the program. It is because the built-in modules are insufficient for solving specific problems. In such situations, additional programming is an effective solution. The advantages and disadvantages of approaches to the use of simulation models in the field of logistics are listed in Table 1.

Table 1. Advantage and disadvantage approaches to the use of simulation models in the field of logistics.

Approach	Advantage	Disadvantage
Classical	- expertise in model creation - no training required - minimal investment costs	- the necessity of communication - a long time for modification - the necessity of providing info
Modified	- speed of model creation - professional knowledge - minimal costs	- training investment - time needed to gain experience - consultation with an expert
Progressive	- speed of modification - minimal production costs - simplicity and variability	- initial investment - defining requirements - suitable software

2.2. Additional Programming Method

Additional programming is a method that enables the definition of new functionalities and conditions for the need to create simulation models, which the basic user version of the simulation software does not offer or does not allow. However, the simulation program must be adapted to its application and enabled. The most common way to implement this is to use the appropriate programming software tool that is part of the simulation tool.

The method of additional programming significantly increases the functionality and possibilities of using the simulation software, simplifies work with the simulation model, increases simulation accuracy, and helps define complex logistic processes. It can be stated that it makes the model more complex and sophisticated.

However, not every user of the simulation software has programming experience; therefore, applying the additional programming method may be complicated or directly unavailable for them. However, if he needs to use a function, define a condition, or make work with a simulation model more efficient, it is necessary to use the additional programming method. In such a case, there is only one option left for the completion and correctness of the construction of the simulation model, which is the use of a third-party service, e.g., in the form of an experienced user or programmer who can program the necessary functionality.

However, before implementing additional programming (Figure 2), it is necessary to precisely specify the necessary functionalities and parameters that need to be implemented in the simulation model. When the method is created and implemented in the simulation model, it should no longer be a problem for an ordinary user without knowledge of programming languages to fully use this method in the implementation of simulation experiments.

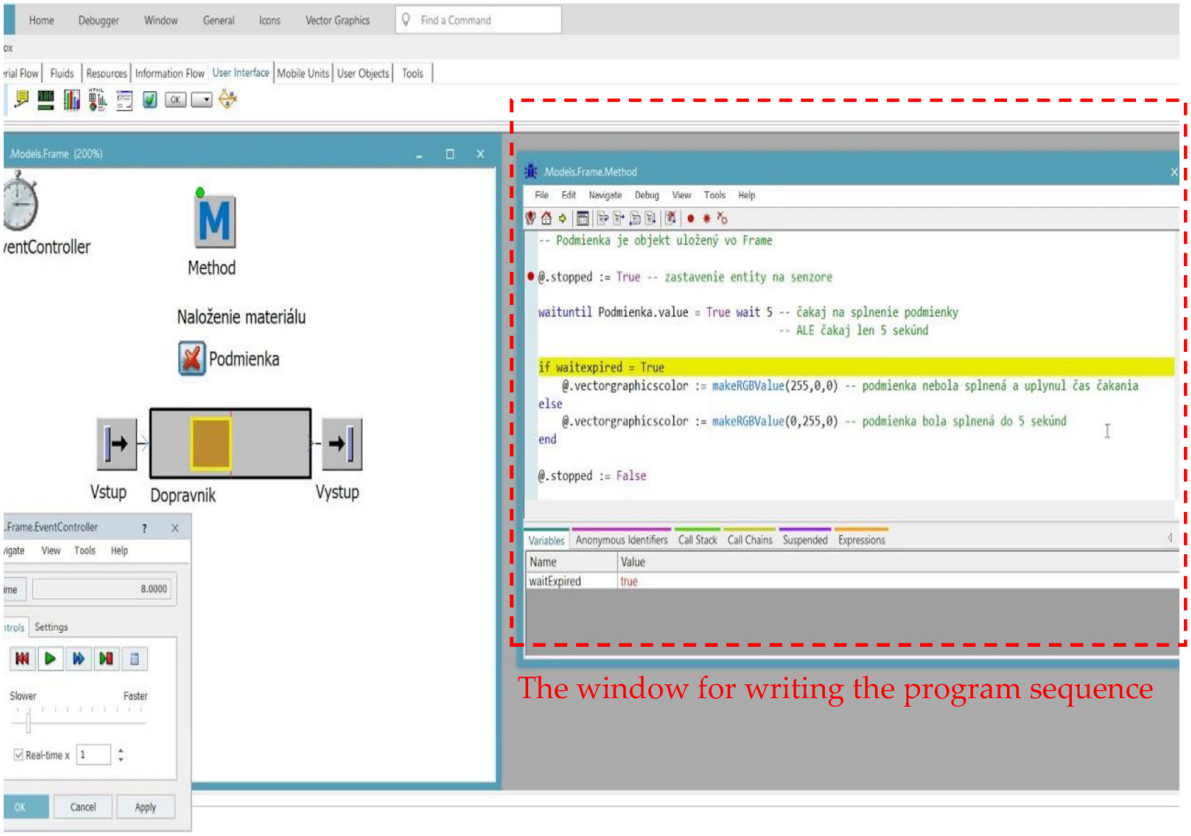

Figure 2. Example of additional programming in the simulation program Tecnomatix Plant Simulation.

The above parameters are subsequently written via a program sequence to a separate window or to the appropriate part of the simulation tool block, which most often takes the form of a notebook. The writing of the program sequence has well-defined rules, structure, and transcription.

In addition, what if it is necessary to adjust, modify, or change the basic concept of the model? Again, there is a need to master the additional programming method or use expert help, which can sometimes be inefficient and lengthy. For this reason, it is necessary to look for a different, more effective, and accessible approach (Figure 3).

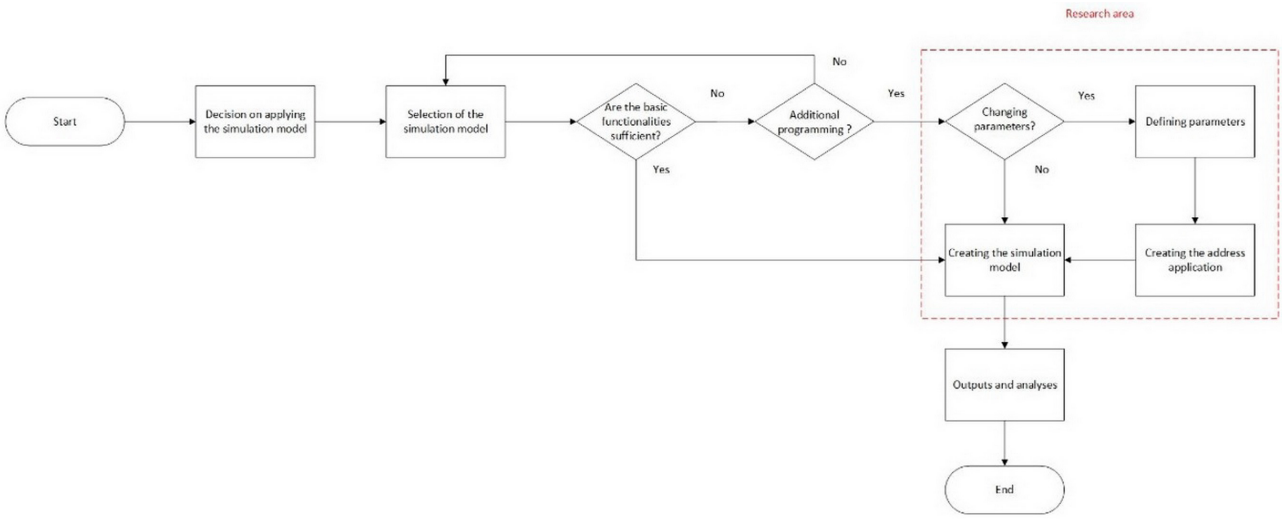

Figure 3. The methodology of applying the proposed approach for the creation of a simulation model.

3. Results and Discussion

On the basis of the mentioned facts, research was carried out, the goal of which was to find a more effective way that would allow the creation (generation) of a program sequence and its change for the needs of additional programming. In the context of previous experiences in logistics process simulation, the Technomatix Plant Simulation program was chosen as the research object, which has the SimTalk programming language for additional programming needs.

Tecnomatix Plant Simulation uses a special object named method (Figure 4) to apply the additional programming method. This object can be briefly characterized as a notebook in which an additional program (program sequence) in the SimTalk programming language can be written. Subsequently, the object method is connected to the required place in the simulation model, i.e., where the created program needs to be applied. When starting the simulation, the program identifies the use of the object method and, based on the relevant link, loads and then executes the prescribed program. In a similar way, it is possible to apply the additional programming method directly to individual objects. The functionality is exactly the same as when using the object method.

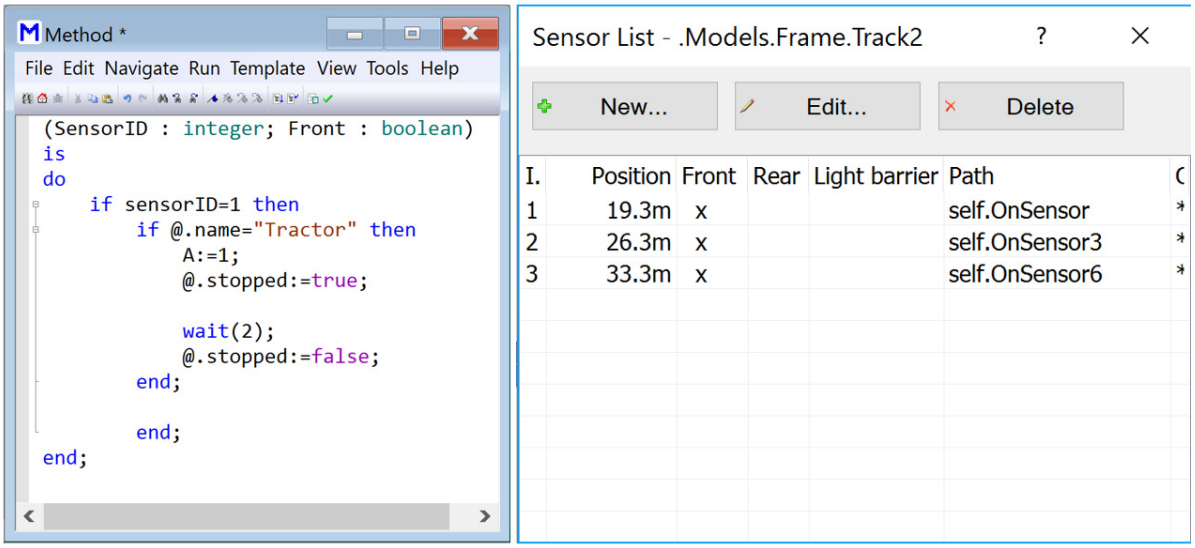

Figure 4. An example of additional programming using the object method.

The size and range of the program that can be created within the object method or dialog box are practically unlimited. In addition to defining common logical conditions, it is possible to program complex technological processes. That significantly reduces the number of classic blocks used. With the help of additional programming, it is possible to implement functions not ordinarily offered in the standard objects. At the same time, it is thus possible to integrate several blocks into one block from a functional point of view using a wide range of simple commands and program sequences. Therefore, it is possible to state that the mentioned approach extends the capabilities of the simulation tool to infinity, and practically, its possibilities are limited only by the capabilities of its user.

At the same time, this functionality enables the creation of user-defined and, as much as possible, customized user libraries of objects that can be created for specific operating conditions. This will significantly increase the efficiency of using simulation models and reduce the time required for conducting new simulation experiments to a minimum.

As part of the research, a suitable software solution was sought that would enable the generation of source code with commands in the SimTalk language directly implemented into the simulation model through the object method. At the same time, it was decided that the software solution should be in an addressable application form (console or with a graphical interface), which allows setting the required parameters of the program sequence and then generating it in the SimTalk language. In essence, the research aimed to create

a single-purpose translator that generates the required method in the SimTalk language based on the selected parameters. The goal of such a solution was to facilitate the creation and generation of the program sequence. At the same time, it would be a solution that would make it possible to use the reprogramming method even for less proficient users of the simulation program or for users without much programming knowledge.

3.1. Console Application

The first form of the software solution sought, validated for generating a program sequence for additional programming, was a console application.

A console application is a computer program that does not use a graphical interface but a computer console, system console, terminal, or emulator. The console application uses the computer keyboard for control. The disadvantage of a console application is that it is intended primarily for the execution of automated tasks that do not require direct interaction with the user.

The aim of this step was to verify the possibility of generating the code itself in the SimTalk program and applying it to the needs of the simulation model in the Technomatix Plant Simulation program. The console application was based on a simple flow diagram (Figure 5).

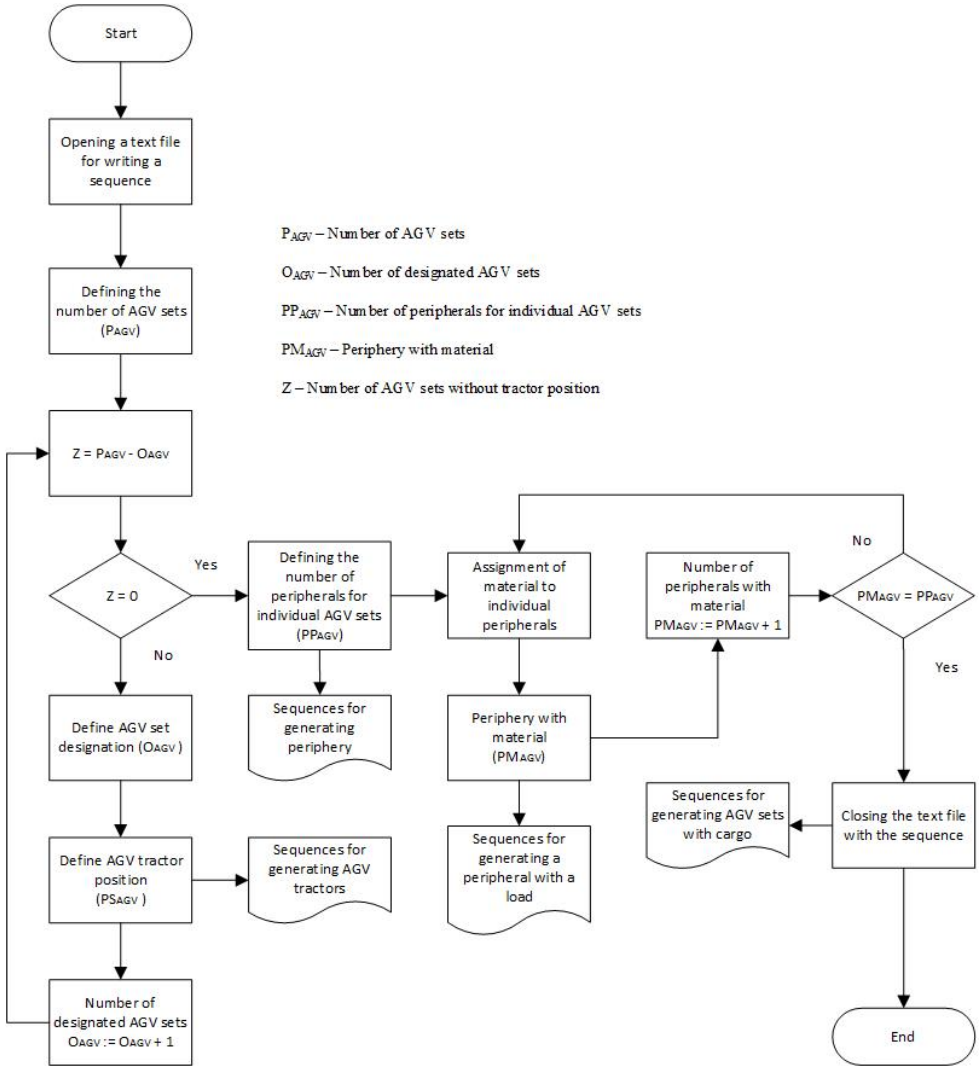

Figure 5. Console application flow chart.

The application thus created was designed to gradually guide the user through the input of the required data to the final generation of the sequence for the needs of additional programming in the form of a text file, which will consist of a program sequence in the SimTalk language. At the same time, the main advantage of the console application was exploited here, namely the creation of a large number of variables compared to the graphical solution. The whole application was built on a loop where the control variables (number of AGV sets, number of peripherals, and material transported) were set at the beginning. Subsequently, their unique names and other critical information had to be defined. As this was a console application, the resulting solution was not limited by space, but on the other hand, this solution offers less clarity.

The console application was designed so that as soon as the application is launched. The codes are continuously written to these text files based on the defined requirements. The text files are continuously updated with the required sequences throughout the console application's runtime and are only finalized when the console application exits. In that way, the required program sequence was generated sequentially based on the user's requirements while maintaining the required structure and transcription of the SimTalk language. The application was developed in the Eclipse IDE for Java developers.

The console application was created to allow a simple sequence to generate a certain number of AGV sets with a defined number of peripherals. Specifically, an enterprise virtual plant model was used to demonstrate the use of the console application to create a programming sequence for the purposes of additive programming. The model was created in part using the built-in functions of the individual blocks of Tecnomatix Plant Simulation ver. 16.1 and partly by additional programming in the SimTalk programming language.

The simulation model consisted of a garage where the logistic trains are located, a warehouse where the material load for production takes place, and a production area where the material is unloaded either on a conveyor truck or to the production equipment. A 3D view of the virtual enterprise operation model is shown in (Figure 6).

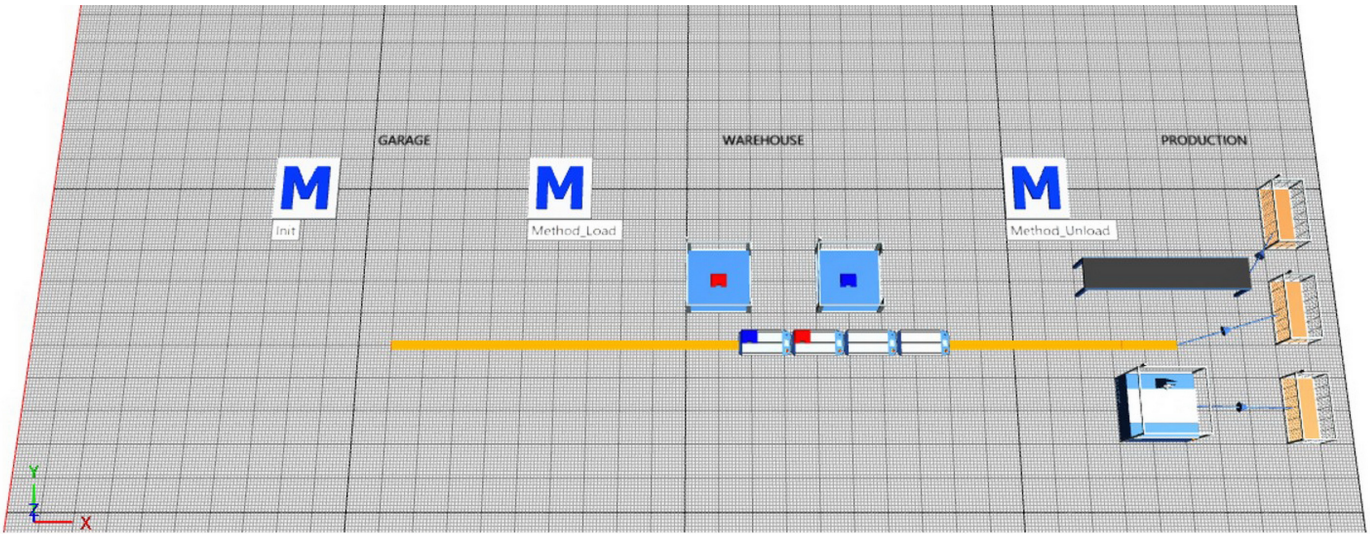

Figure 6. 3D model of virtual enterprise operation.

From Figure 5, it can be seen that the simulation process starts by generating the mobile units, namely one tractor, three trolleys, and two types of material (red and blue) in the quantity of two, through a sequence written in the method block named Init (Figure 7). For this operation, the sequence was generated through the console application.

After running the simulation, the logistics train dispatched to the warehouse loaded one unit of the material "red" on the 2nd cart and one "blue" on the 3rd cart. Such a simple model is difficult to create using only the blocks and functions that Tecnomatix offers by default, such that all the specific requirements are met. Therefore, it was necessary to

apply additional programming activities via the object method, the container for miniature programs inserted into the model.

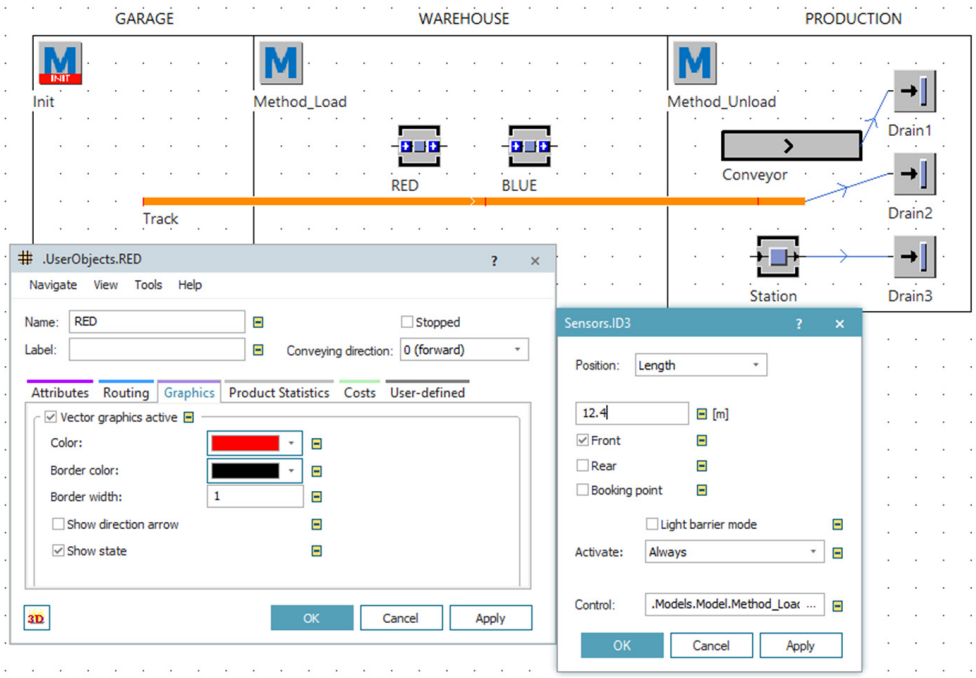

Figure 7. 2D model and object windows.

The basic settings of the elements in the model, which did not need to be programmed, were defined through the object windows. It was about the graphic parameters of the materials, the sensor settings, the logistic train speed, and the processing time for the workplace. An example of defining parameters through object windows and built-in functionalities is shown in Figure 7.

The sequences for the additional reprogramming activities were inserted into the methods block. In our model, operations that generated mobile elements and programmed the loading and unloading activities were implemented through additional programming. The sequences for the additional programming are shown in Figure 8. As mentioned before, for the Methods block labeled Init, the sequence was generated through the console application.

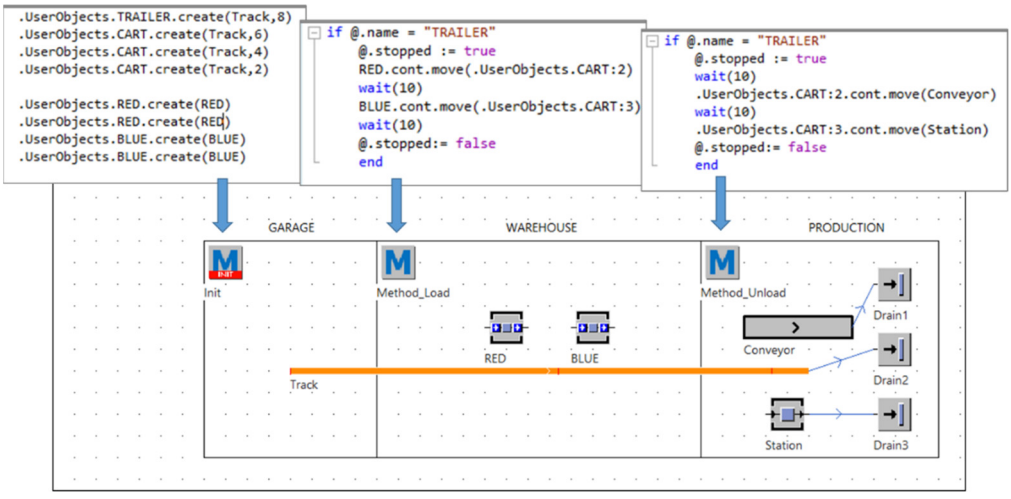

Figure 8. Codes for programmed actions inserted into methods.

First of all, the results of the first phase of the research confirmed that it is possible to generate the required program sequence in this way. Regardless, at the same time, it was found that the lack of a friendly graphic interface makes this solution unattractive and less clear for the future user.

3.2. Application with a Graphical Interface

The first phase of the research thus confirmed the feasibility and functionality of the sought-after solution of generating a program sequence for additional programming through an address application. However, the solution through the console application was not ideal, so the second phase followed.

In the second phase of the research, an application with a simple graphical user interface was developed. The application was created in the Eclipse IDE for Java Developers development environment. The JavaFX application platform was used to create the graphic pages of the applications.

For this application, it was decided that it would be controlled by a graphical form created using the JavaFX framework (Figure 9). All the necessary information for generating the sequence for additional post-programming, such as entity names, data, and output text file names, was determined using the frames in the created graphical view. After all the necessary parameters to be included in the generated sequence are filled in, everything is exported to text files using a one-click button. With the help of the data entered in this form, the code for additional programming of the required sequence into the simulation model is then generated using the created address application.

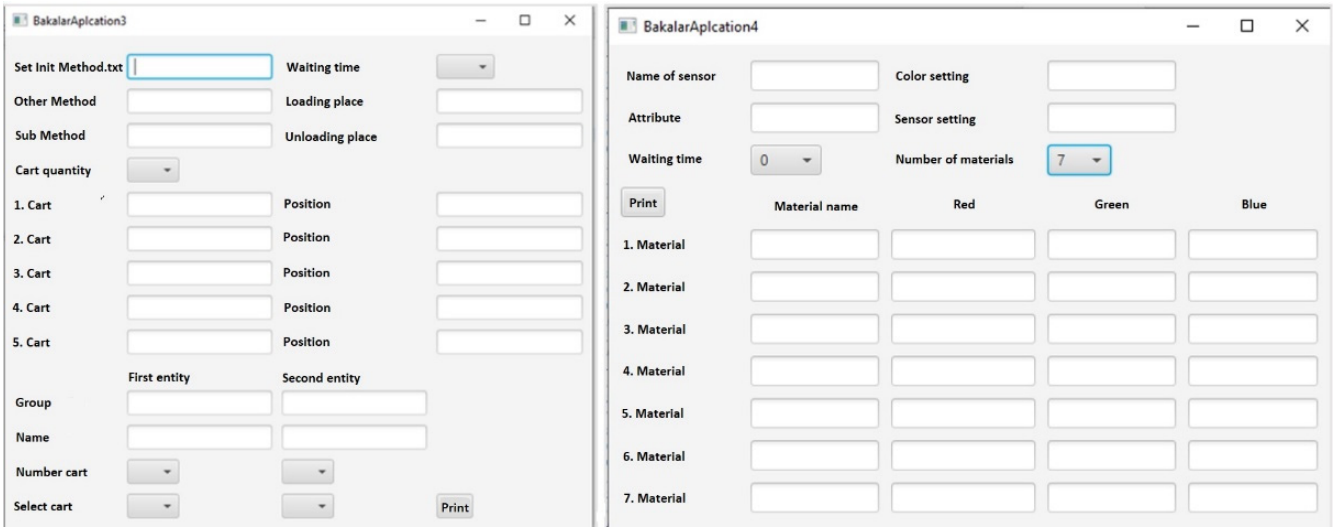

Figure 9. Preview of the graphical interface for defining sequence generation parameters.

The above solution is comfortable and fast. It provides an overview of the defined parameters. The advantage of the solution is that, especially for long sequences, there is a 100% guarantee of writing the correct structure of the program sequence.

As part of this phase of the research, a second addressable application was created to generate code for modeling the AGV set that supplies a certain number of workplaces within the MilkRun system. It was the same simple example implemented in the first phase of the research. Part of the source code of the created address application is shown in Figure 10.

The application created in this way offered much greater comfort for defining individual parameters necessary for generating program sequences. It also allowed the generation of several sequences simultaneously, where it was possible, for example, to share information that was common to several program sequences. Another advantage of the application's graphical interface was that it simply and transparently provided an

overview of the number and range of defined parameters that needed to be entered into the program sequence.

```
import javafx.geometry.Insets;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import javafx.application.Application;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;

import javafx.stage.Stage;

public class Main extends Application{
        Stage window;
    public void start(Stage primaryStage) throws Exception {
        int pocet = 400;

        GridPane grid =new GridPane();
        grid.setAlignment(Pos.CENTER);
        grid.setVgap(10);
        grid.setHgap(10);
        grid.setPadding(new Insets(10));
        //ComboBox comboBox;
        Button btn1=new Button("Tisk");

        Label lblMUs = new Label("Názav MUs");
        grid.add(lblMUs,0,6);
        Label lblDis = new Label("Délka cesty");
        grid.add(lblDis,0,7);
        Label lblOdkud = new Label("Pocatecni entita");
        grid.add(lblOdkud,0,8);
        Label lblKam = new Label("Koncova entita");
        grid.add(lblKam,0,9);

        Label lblInit = new Label("Nazev init txt");
        grid.add(lblInit,0,1);
        Label lblMethod = new Label("Nazev method txt");
        grid.add(lblMethod,0,2);
        Label lblMethod2 = new Label("Nazev method1 txt");
```

```
File myObj = new File(Method2);
                if (myObj.createNewFile()) {
                  System.out.println("File created: " + myObj.getName());
                } else {
                  System.out.println("File already exists.");
                }
              } catch (IOException e) {
                System.out.println("An error occurred.");
                e.printStackTrace();
              }
            try {
                FileWriter myWriter1 = new FileWriter(Init);
                FileWriter myWriter2 = new FileWriter(Method);
                FileWriter myWriter3 = new FileWriter(Method2);

               myWriter1.write(".MUs."+MUs+".create(Track, "+Dis+")");
                myWriter1.write(System.getProperty("line.separator"));
                myWriter1.write(".MUs.Entity.create("+Dis+")");
                myWriter1.write(System.getProperty("line.separator"));

                myWriter2.write("@.stopped := true");
                myWriter2.write(System.getProperty("line.separator"));
                myWriter3.write("waint("+WaitInt+")");
                myWriter2.write(System.getProperty("line.separator"));
                myWriter2.write(Odkud+".cont.move(@)");
                myWriter2.write(System.getProperty("line.separator"));
                myWriter2.write("@.stopped := false");
                myWriter2.write(System.getProperty("line.separator"));

                myWriter3.write("@.stopped := true");
                myWriter3.write(System.getProperty("line.separator"));
                myWriter3.write("waint("+WaitInt+")");
                myWriter3.write(System.getProperty("line.separator"));
                myWriter3.write("@.cont.move("+Kam+")");
                myWriter3.write(System.getProperty("line.separator"));
                myWriter3.write("@.stopped := false");
                myWriter3.write(System.getProperty("line.separator"));

                myWriter1.close();
                myWriter2.close();
                myWriter3.close();
```

Figure 10. Sample of the source code of the created application.

On the contrary, within the application, specifically in its source code, the required commands for the SimTalk programming language were defined, which, after supplementing with individual parameters, created a program sequence.

4. Conclusions

Based on the facts and examples presented, it can be concluded that the method of additional programming within simulation software significantly increases the possibilities for creating simulation models of logistical and technological processes. This is a highly specialized activity that often requires advanced programming knowledge. With its help, it is possible to simulate even the most complex logistical and technological processes and their parts. The creation of additional sequence programs and their subsequent application enables the creation of a more accurate simulation model and the obtaining of more valid results. Although additional programming requires at least a basic knowledge of programming for its use, it does not discriminate against those who do not have it.

Creating and modifying simulation models for ever-changing business needs or conditions is a reality for many businesses. It can be implemented by permanently using the services of external specialists. However, using the services of specialists for programming and creating simulation models means increased financial and time costs. Therefore, one solution is to use an addressable application with an acceptable graphical interface, which can subsequently be effectively used for quick modification and adjustment of existing models, for example, in the form of sequence generation in the programming language of the given simulation software tool.

An appropriate application can be designed to enable the fast and reliable generation of program sequences in the proper programming language. It is necessary to use the services

of a programmer to create it, but further use is most intuitive, and the generated program sequences can be used for quick modification and adjustment of simulation models. The presented research confirmed the research hypothesis and, at the same time, pointed out that it is more efficient to use an addressable application with a graphic interface. The obtained results confirmed the validity of the approach for the Tecnomatix Plant Simulation program. As part of further research, it is necessary to verify the validity of the method with other simulation programs. On the contrary, the presented approach can also be used with other simulation software to increase the efficiency of using simulation models. By using it, computer simulation can be turned into a working tool of daily use for a wide range of industrial areas, including logistics.

The paper focuses on the use of simulation models in the field of logistics. Its goal was to verify a method that would allow easier creation and modification of simulation models. It is a method based on using an address application for generating a program sequence for additional programming within the creation of simulation models in the Tecnomatix Plant Simulation program. In the presented research, a methodology was proposed, which is shown in Figure 3. The methodology mentioned has been successfully validated in Tecnomatix Plant Simulation. Further research is planned to validate it in other simulation tools supporting the additional programming method, such as Wittness or Flexim. The goal was to find a way to enable the use of computer simulation on a day-to-day basis. No special knowledge of working with a simulation program would be needed to master it. Further research is aimed at optimizing the creation of sequences for complementary programming. This paper presents the first phase of research aimed at supporting and extending the use of the computer simulation method in everyday industrial practice.

The paper sought to highlight the possibility of utilizing the creation of an address application for simulation model creation and sequence generation for additional programming. Our effort is to point out that it is possible to create simulation models with an approach not directly offered by their authors. Within the presented research, the authors' intention has been verified using Tecnomatix Plant Simulation, one of the most widely used tools in logistics. At the same time, we are fully aware of the pitfalls of the above approach, which requires the work of a programmer to create an address application. However, the importance of this approach increases if the application is created to support a simulation model used daily, e.g., for production planning, sales, warehousing, and other logistics processes.

Addressing application development and management is meaningful if the simulation model is used daily or on a regular basis. With its help, it is possible to make production and logistics processes significantly more efficient. The creation of applications for one-time simulation projects has practically no significance in terms of time because the time invested in their development and debugging represents an unnecessary and ineffective solution. On the contrary, the regular, frequent application and modification of the simulation model have their meaning. Furthermore, it can be beneficial from an overall economic and time point of view. However, these assumptions must be confirmed by further scientific research in this area.

Author Contributions: Conceptualization, M.Ď.; methodology, G.F. and J.F.; investigation, M.Ď. and V.M.; data analysis, G.F. and J.F.; writing—original draft preparation, F.D. and H.N.; writing—review and editing, J.F. and V.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work is a part of these projects VEGA 1/0101/22, VEGA 1/0600/20, KEGA 005TUKE-4/2022, KEGA 018TUKE-4/2022, APVV-21-0195, ITMS: 313011T567.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available from the authors upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Čujan, Z. Simulation of production lines supply within internal logistics systems. *Open Eng.* **2016**, *6*, 470–475. [CrossRef]
2. Agalianos, K.; Ponis, S.T.; Aretoulaki, E.; Plakas, G.; Efthymiou, O. Discrete event simulation and digital twins: Review and challenges for logistics. *Procedia Manuf.* **2020**, *51*, 1636–1641. [CrossRef]
3. Straka, M.; Spirkova, D.; Filla, M. Improved efficiency of manufacturing logistics by using computer simulation. *Int. J. Simul. Model.* **2021**, *20*, 501–512. [CrossRef]
4. Pawlewski, P.; Fertsch, M. Modeling and simulation method to find and eliminate bottlenecks in production logistics systems. In Proceedings of the 2010 Winter Simulation Conference, Baltimore, MD, USA, 5–8 December 2010; IEEE: New York, NY, USA; pp, 1946–1956. [CrossRef]
5. Pekarcikova, M.; Trebuna, P.; Kliment, M.; Dic, M. Solution of bottlenecks in the logistics flow by applying the kanban module in the tecnomatix plant simulation software. *Sustainability* **2021**, *13*, 7989. [CrossRef]
6. Wenzel, S.; Boyaci, P.; Jessen, U. Simulation in Production and Logistics: Trends, Solutions and Applications. In *Lecture Notes in Business Information Processing, Proceedings of the Advanced Manufacturing and Sustainable Logistics: 8th International Heinz Nixdorf Symposium, IHNS 2010, Paderborn, Germany, 21–22 April 2010*; Dangelmaier, W., Blecken, A., Delius, R., Klopfer, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 46, pp. 73–84.
7. Xu, Z. Logistic Regression Based on Individual-Level Predictors and Aggregate-Level Responses. *Mathematics* **2023**, *11*, 746. [CrossRef]
8. Straka, M.; Khouri, S.; Besta, P.; Drevko, S. Development of computer simulation and its use for the needs of logistics. In Proceedings of the Carpathian Logistics Congress (CLC 2017), Liptovsky Jan, Slovakia, 28–30 June 2017; pp. 86–92.
9. Rotunno, G.; Lo Zupone, G.; Carnimeo, L.; Fanti, M.P. Discrete event simulation as a decision tool: A cost benefit analysis case study. *J. Simul.* **2023**, 1–17. [CrossRef]
10. Daroń, M. Simulations in planning logistics processes as a tool of decision-making in manufacturing companies. *Prod. Eng. Arch.* **2022**, *28*, 300–308. [CrossRef]
11. Qiao, P. Simulation of Logistics Delay in Bayesian Network Control Based on Genetic EM Algorithm. *Comput. Intell. Neurosci.* **2022**, *2022*, 1–13. [CrossRef] [PubMed]
12. Straka, M.; Sofranko, M.; Vegsoova, O.G.; Kovalcik, J. Simulation of homogeneous production processes. *Int. J. Simul. Model.* **2022**, *21*, 214–225. [CrossRef]
13. Lin, Z. Understanding and Simulating Software Evolution. In Proceedings of the 35th International Conference on Software Engineering (ICSE 2013), San Francisco, CA, USA, 18–26 May 2013; Notkin, D., Cheng, B.H.C., Pohl, K., Eds.; IEEE: New York, NY, USA, 2013; pp. 1411–1414.
14. Ali, S.M.; Doolan, M.; Wernick, P.; Wakelam, E. Developing an agent-based simulation model of software evolution. *Inf. Softw. Technol.* **2018**, *96*, 126–140. [CrossRef]
15. Lin, Y.-W.; Lin, Y.-B.; Yen, T.-H. SimTalk: Simulation of IoT applications. *Sensors* **2020**, *20*, 2563. [CrossRef] [PubMed]