JAM CM Journal of Applied Mathematics and Computational Mechanics 2026, 25(1), 31-42
www.amcm.pcz.pl p-ISSN 2299-9965
DOI: 10.17512/jamcm.2026.1.03 e-ISSN 2353-0588

SIMULATION MODEL OF A CNC CELL

Milan Jus, Alena Breznická, Pavol Mikuš, Marcel Kohutiar, Michal Krbaťa Maroš Eckert

Faculty of Special Technology, Alexander Dubcek University of Trenčín
Trenčín, Slovakia
milan.jus@tnuni.sk, alena.breznicka@tnuni.sk, pavol.mikus@tnuni.sk, marcel.kohutiar@tnuni.sk
michal.krbata@tnuni.sk, maros.eckert@tnuni.sk

Received: 19 September 2025; Accepted: 5 December 2025

Abstract. The article deals with the creation of a simulation model of a CNC production cell. Two variants of the model are presented, where one is with an operator and the second model is automated with a robotic manipulator. The robotic model of the CNC cell was also modified with a reduction in the number of CNC cantilevers. All models were simulated, and the obtained data are presented and compared in the article.

MSC 2010: *68N19, 03C99*
Keywords: *simulation model, CNC, simulation, modeling, robotization, automation mechanics*

1. Introduction

This article presents simulation models of a CNC machining center. The first model consists of 3 single-purpose CNCs, where each CNC performs only one operation with a single tool. The operator is a worker whose task is to attach/detach material and transfer this material. The machining process consists of sequential machining, where substitution is not possible. By modifying this first model, a second model was created where the worker is replaced by a robot. The third model consists of two universal CNCs, which can simultaneously perform all 3 operations from the first model, and a worker is needed to move the material. By replacing the worker with a robot, a fourth model was created from the third model. The simulation of the machining center was performed with all four models for a simulation time of 8 hours. The simulation outputs of production and utilization of machining centers were evaluated and compared.

Simulation models and their simulation were implemented in Tecnomatix Plant Simulation [1, 2].

2. Mathematical model of system

A dynamic system can generally be characterized by an input vector $\boldsymbol{X}$, an output vector $\boldsymbol{Y}$, and the system itself can be expressed in several ways, e.g. by a differential equation.

If we are going to interconnect individual systems, then if we consider:

- Inputs $\boldsymbol{X}_{ci}$, where c is the number of components and i is the number of coupling,
- Outputs $\boldsymbol{Y}_{pj}$, where p is component part number of components; i is the number of coupling; and p, q, i and $j \in \mathrm{N}$.

The mathematical relation describing the coupling of two systems in series is characterized by relation (1), in which $p = 1,m$; $q = 1,n$; $(y(1)_q = x(2)_p)$ [3]:

$$\begin{aligned} y_1 &= y_1(y_{11}, y_{12}, ..., y_{1q}, ..., y_{1n}) \\ x_2 &= x_2(x_{21}, x_{22}, ..., x_{2q}, ..., x_{2n}) \end{aligned} \tag{1}$$

We will express the relationships between systems using relationship matrices. If we connect two systems, S_1 and S_2, where the outputs of S_1 are connected to the inputs of S_2, the coupling matrix $\boldsymbol{K}_{ei}$, will be an $m \times n$ sized matrix. The matrix elements are defined binary variables:

$$e_{pq} = e_{ij} : B = \{0, 1\} \rightarrow B, \tag{2}$$

where $e_{pq} = 1$ when the connection between $p \rightarrow q$ is defined and $e_{pq} = 0$ when connection $p \rightarrow q$ is undefined.

For a system of R order consisting of N subsystems of order R+1, the maximum number of links will be $N(N–1)$. Since many links are identical and repetitive, the number of links is actually small. The structure of a system S of R order is given by the type and number of component subsystems, as well as the type and number of relationships between them [3].

Simulation techniques and simulation software were used, where it is possible to visualize and analyze subsystem processes with respect to interactions between them and the environment. The evolution of the system state over time could therefore be described as a dynamic system, i.e. a system operating in a real environment.

3. Simulation models

In the next section, 4 simulation models of CNC workstations with a worker or an automated robot will be presented.

3.1. First simulation model

This simulation model includes the supply of material for machining using the *Source* and *Buffer* blocks. It also contains 3 dedicated CNCs, which are labeled

CNC_1, *CNC_2*, and *CNC_3*. The finally processed material leaves the model through the Buffer1 output warehouse and the *Drain* block. A worker is inserted into the model, who enters from the *WorkerPool* block, and *Workplace* stations and routes along which the worker moves are created, while his control is performed by the *Broker* block. Figure 1 shows a 2D simulation model [4].

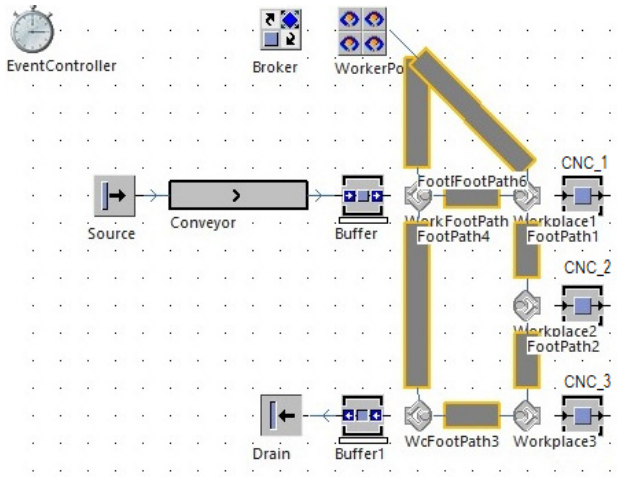

Fig. 1. First 2D simulation model

The machining process is in the sequence: sequential machining at *CNC_1*, continuation at *CNC_2* and completion at *CNC_3*. Machining times are listed in Table 1.

Table 1. The machining times of individual CNCs are given

	CNC_1	*CNC_2*	*CNC_3*
Processing time	25 min	20 min	15 min

If the worker brings the material to be machined on the *CNC*, it needs to be fixed. If the worker removes the material from the *CNC*, it needs to be released again. The clamping and releasing of the material follow a normal distribution within the time range of 4 minutes 45 seconds to 5 minutes 15 seconds. This setting illustrated in Figure 2 for *Workplace1*, also applies to *Workplace2* and *Workplace3* [5].

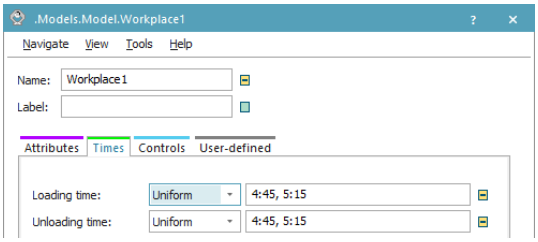

Fig. 2. Setting the material clamping and releasing times

The transfer of processed material from the *Buffer* warehouse to the first processing (*CNC_1*) is set in the *Importer-Transport* tab, which is shown in Figure 3.

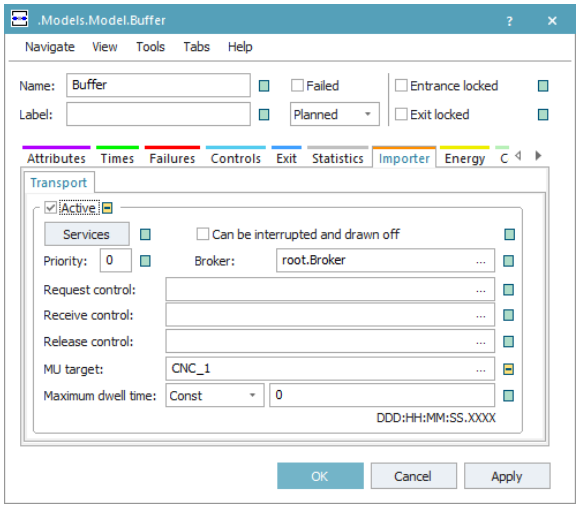

Fig. 3. Material transfer settings

The transfers of already processed material from *CNC_1* to *CNC_2*, from *CNC_2* to *CNC_3*, and finally from *CNC_3* to the output warehouse *Buffer1* are configured in the same way.

It is necessary to modify the worker's message so that it does not start moving material from the intermediate warehouse (*Buffer*) to *CNC_1* if *CNC_1* is processing. For this reason, it was necessary to insert a *Method* for the *Broker* that implements such worker management. It is inserted into the *Importer's request* file and its script/code is in Figure 4.

```
param obj: object, -- Importer
type: integer -- Importer type (0=failure importer, 1=setup
importer, 2=processing importer, 3=transport importer)

if obj.name = "Buffer"
    if CNC_1.empty = true//CNC_1.empty = true or Variable = true
        ?.doStandardImport(obj, type)
    end
else
    ?.doStandardImport(obj, type)
end
```

Fig. 4. Script of *Method_Broker*

The data obtained from the simulation are presented in Table 2.

Table 2. First model data obtained from simulation

	Portions of the states			Material flow properties	
	Working	Waiting	Blocked	Number of entries	Number of exits
CNC_1	62.50 %	37.17 %	0.33 %	12	12
CNC_2	45.83 %	54.07 %	0.10 %	11	11
CNC_3	32.96 %	66.50 %	0.54 %	11	10

In the simulated time of 8 hours, 10 pieces of material were processed. From the table, it can be seen that one piece of material is still being processed on *CNC_3*, and 12 pieces have left *CNC_1*, but only 11 pieces have arrived at *CNC_2*, so one piece is in the process of being transferred.

3.2. Second simulation model

In the second simulation model, the worker was replaced by a robot that moves the material. The robot also performs the fixation and release of the material in 4 minutes. The 2D simulation model is shown in Figure 5.

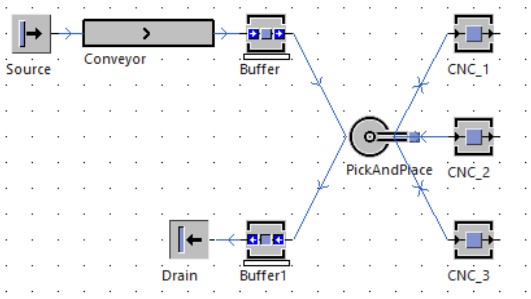

Fig. 5. Second 2D simulation model

In this case, it is necessary for the robot to perform tasks in the same technological process, as it was in the first model.

A *Part* is an element representing the material to be machined and is assigned a user variable *LastStation* in string format, which will carry information about where the last machined material was located. Initialization is performed to the value *Buffer*. The definition and initialization of the variable *LastStation* is shown in Figure 6.

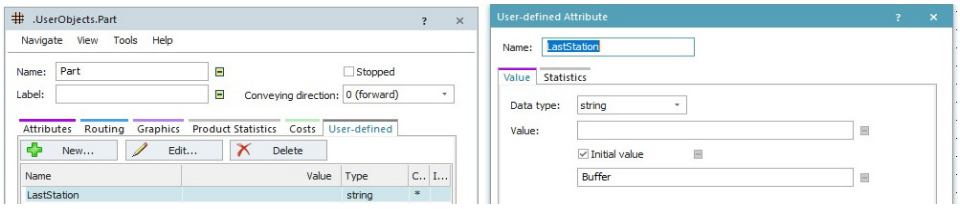

Fig. 6. The definition and initialization of the variable *LastStation*

In addition, the CNC processing times were adjusted to account for material fixation and release, so in this case the entered values are given in Table 3.

Table 3. The machining times of individual CNCs for the second model are given

	CNC_1	*CNC_2*	*CNC_3*
Processing time	33 min	28 min	23 min

A method *Set_LastStation* (*@.LastStation := ?.Name*) has been created, which changes the *LastStation* variable to the value *CNC_1*, *CNC_2* or *CNC_3* depending on where the processed material is located. The method is executed when the material is loaded into the CNC, which is ensured by inserting the *Entrance* item on the *Controls* tab, shown in Figure 7 only for *CNC_1* (*CNC_2* and *CNC_3* have the same setting).

In order for the robot to function properly, its actions must be controlled. Therefore, two additional methods were created: *Pull_Control*, which determines where the material will be taken from, and *Target_Control*, which specifies where the material will be inserted. The insertion of methods is shown in Figure 8.

The script code of the *Pull_Control* method is shown in Figure 9. The request from *CNC_1*, *CNC_2*, and *CNC_3* is always executed, but if the request comes from *Buffer*, it is executed only if *CNC_1* is not processing anything.

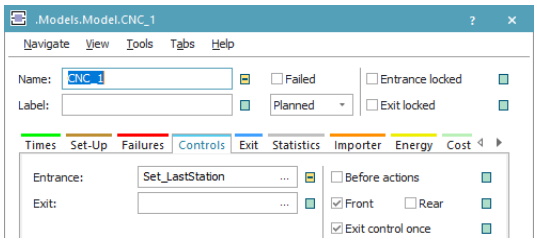

Fig. 7. The definition and initialization of the variable *LastStation*

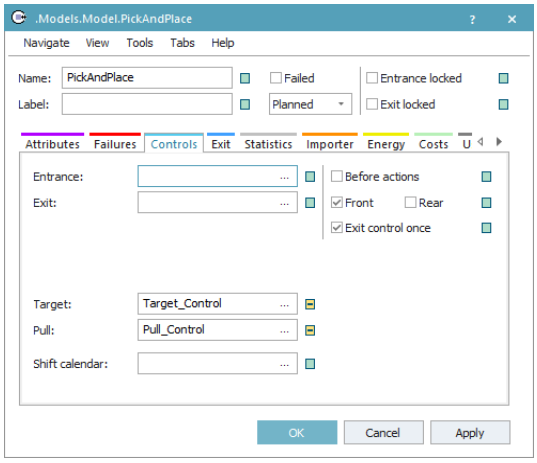

Fig. 8. Embedded *Pull_Control* and *Target_Control* methods

```
   var blockList := ?.fwBlockList
for var i:=1 to blockList.dim
  if blockList[i].~ = Buffer and CNC_1.occupied=false
    ?.unblock(blockList[i])
    return
  elseif blockList[i].~ = CNC_1
    ?.unblock(blockList[i])
    return
  elseif blockList[i].~ = CNC_2
    ?.unblock(blockList[i])
    return
  elseif blockList[i].~ = CNC_3
    ?.unblock(blockList[i])
    return
  end
next
```

Fig. 9. The script code of the *Pull_Control* method

The script code of the *Target_Control* method is shown in Figure 10. It evaluates where the material to be transferred was last located and sets the target location accordingly.

```
   if @.LastStation = "Buffer"
    ?.setDestination(CNC_1, false)
elseif @.LastStation = "CNC_1"
    ?.setDestination(CNC_2, false)
elseif @.LastStation = "CNC_2"
    ?.setDestination(CNC_3, false)
elseif @.LastStation = "CNC_3"
    ?.setDestination(Buffer1, false)
end
```

Fig. 10. The script code of the *Target_Control* method

The data obtained from the simulation are presented in Table 4.

Table 4. Second model data obtained from simulation

	Portions of the states			Material flow properties	
	Working	Waiting	Blocked	Number of entries	Number of exits
CNC_1	99.38 %	0.48 %	0.14 %	15	14
CNC_2	78.99 %	20.81 %	0.19 %	14	13
CNC_3	61.72 %	38.15 %	0.12 %	13	12

In the simulated time of 8 hours, 12 pieces of material were processed. By examining Table 4, it can be seen that one piece of material is still being processed on all CNC machines.

3.3. Third simulation model

This model, in comparison with the first simulation model, merges single-purpose *CNC_1*, *CNC_2*, and *CNC_3* into one CNC. Two such centers *CNC_A* and *CNC_B* are used in the model. Tool change in CNC takes 20 s, so the operating time of one center is 1 hour and 40 s. The movement, clamping, and release of the machined material are performed by a worker. The simulation model created in this way in 2D is shown in Figure 11.

In the model, it is necessary to ensure the transfer of material from *Buffer* to *CNC_A* and *CNC_B*, therefore it is necessary to dynamically change the material transfer target. For this purpose, the *Method_Broker* method is created, the script code of which is shown in Figure 12.

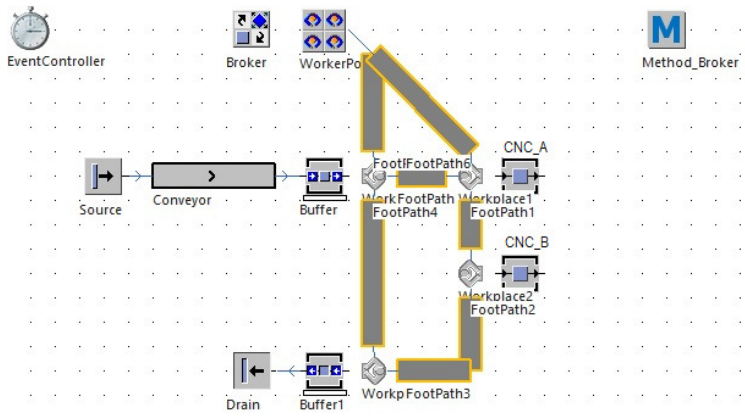

Fig. 11. Third 2D simulation model

The *Method_Broker* script code sets the target to move the material from the *Buffer* to *CNC_A* or *CNC_B*, whichever is idle. If the material is processed in *CNC_A* or *CNC_B*, the worker moves it to the *Buffer1* warehouse.

```
   param obj: object, -- Importer
type: integer -- Importer type (0=failure importer, 1=setup impor-
ter, 2=processing importer, 3=transport importer)

if obj.name = "Buffer"
    if CNC_A.empty = true or Variable = true
        ?.doStandardImport(obj, type)
    end
else
    ?.doStandardImport(obj, type)
end
```

Fig. 12. Script code of *Method_Broker*

A simulation was performed, and the data obtained from the simulation are shown in Table 5.

Table 5. Third model data obtained from simulation

	Portions of the states			Material flow properties	
	Working	Waiting	Blocked	Number of entries	Number of exits
CNC_A	86.21 %	13.76 %	0.03 %	7	6
CNC_B	83.87 %	14.90 %	1.22 %	7	6

In the simulated time of 8 hours, 12 pieces of material were processed. From Table 5, it can be seen that one material is still being processed on *CNC_A* and *CNC_B*.

3.4. Fourth simulation model

This simulation model is based on the third simulation model, where the worker is replaced by a robot. The assembled simulation model in 2D view is shown in Figure 13.

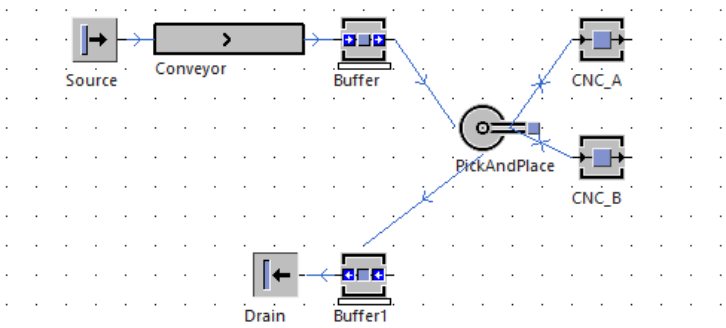

Fig. 13. Fourth 2D simulation model

As in the second simulation model, in this model, the fastening and releasing of the material is provided by a robot, therefore the processing time of *CNC_A* and *CNC_B* was adjusted to 1 hour 8 minutes 40 seconds, which is shown in Figure 14.

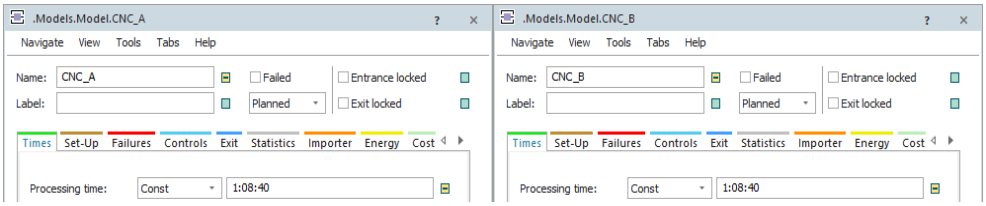

Fig. 14. Fourth 2D simulation model

The robot's action is controlled by the *Pull_Control* and *Target_Control* methods, which are shown in Figures 15 and 16.

```
   var blockList := ?.fwBlockList
for var i:=1 to blockList.dim
  if blockList[i].~ = Buffer and CNC_A.occupied=false
    ?.unblock(blockList[i])
    return
  elseif blockList[i].~ = Buffer and CNC_B.occupied=false
    ?.unblock(blockList[i])
    return
  elseif blockList[i].~ = CNC_A or blockList[i].~ = CNC_B
    ?.unblock(blockList[i])
    return
  end
next
```

Fig. 15. The script code of the *Pull_Control* method for fourth model

```
   if @.LastStation = "Buffer" and CNC_A.occupied=false
    ?.setDestination(CNC_A, false)
elseif @.LastStation = "Buffer" and CNC_B.occupied=false
    ?.setDestination(CNC_B, false)
elseif @.LastStation = "CNC_A"
    ?.setDestination(Buffer1, false)
elseif @.LastStation = "CNC_B"
    ?.setDestination(Buffer1, false)
end
```

Fig. 16. The script code of the *Target_Control* method for fourth model

After 8 hours of simulation, the data obtained were shown in Table 6.

Table 6. Third model data obtained from simulation

	Portions of the states			Material flow properties	
	Working	Waiting	Blocked	Number of entries	Number of exits
CNC_A	99.67 %	0.30 %	0.04 %	7	6
CNC_B	99.62 %	0.33 %	0.05 %	7	6

In the simulated time of 8 hours, 12 pieces of material were processed. From the table, it can be seen that one material is still being processed on *CNC_A* and *CNC_B*.

4. Summary

The data obtained from simulations of all simulation models and presented in Tables 2-6 are clearly processed into a graphical output, which is shown in Figure 17.

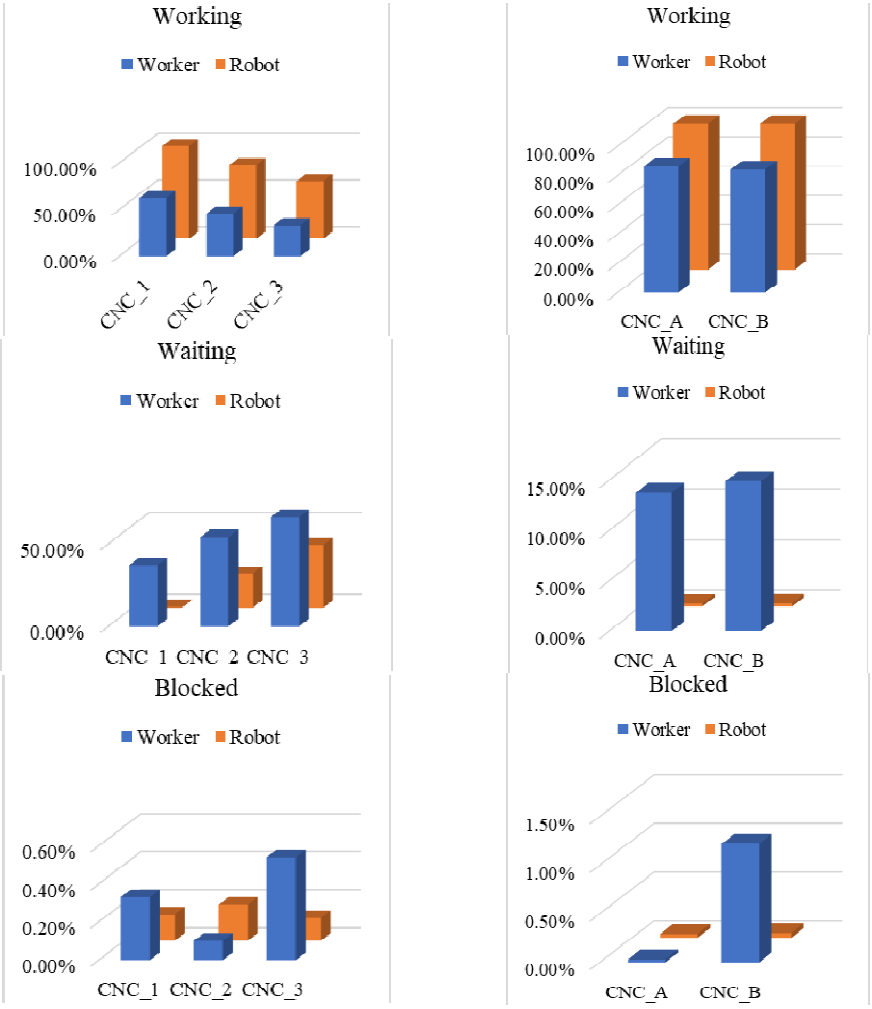

Fig. 17. Graphical display of simulation data

5. Conclusions

In this article, a machining simulation model was created that included 3 single--purpose CNCs. Subsequently, the original model was modified to a model with CNCs that combined the original 3 machining operations into one CNC. In the modified model, 2 such CNCs were used. In the first and second cases, the operator was a worker. Subsequently, the worker was replaced by a robot, so two more models were created.

Simulation data were obtained and compared from simulations performed over 8 hours. In the case of single-purpose CNC machines, replacing the worker with a robot increased production, while in the case of universal CNC machines, production

remained the same. In general, it can be stated that the robot in the models had an impact on increasing utilization and reducing waiting intervals and blocking time.

Acknowledgement

Publishing this article was made possible through the support of the project KEGA 005TnUAD-4/2024 Creation of interactive teaching texts for education using E-learning on Faculty of Special Technology, Alexander Dubček University in Trenčín.

References

[1] Jamal, A., Malallah, A., & AlShatti, M. (2024). *Manufacturing Facility Simulation Using Tecnomatix PLM by Siemens*. 7th European Conference on Industrial Engineering and Operations Management, Augsburg, Germany, July 16-18, 2024, IEOM Society International. DOI: 10.46254/EU07.20240220.

[2] Nickpasand, M., Jensen, S.W., Jensen, C.A., & Gaspar, H.M. (2025). Offshore wind turbine factory simulation as enabler for agile manufacturing. *Journal of Simulation*, 1-17. DOI: 10.1080/17477778.2025.2496304.

[3] Florescu, A., & Barabas, S.A. (2020). Modeling and simulation of a flexible manufacturing system A basic component of industry 4.0. *Applied Sciences*, *10*, 8300. DOI: 10.3390/app10228300.

[4] Duplák, J., Yeromina, M., Dupláková, D., Mikuláško, S., & Mitaľová, Z. (2023). Application of software tools in the identification of robotic workplace cooperation in the design phase. *SAR Journal*, *6*, 4, 221-229. DOI: 10.18421/SAR64-01.

[5] Fedorko, G., Molnár, V., Strohmandl, J., Horváthová, P., Strnad, D., & Cech, V. (2022). Research on using the tecnomatix plant simulation for simulation and visualization of traffic processes at the traffic node. *Applied Sciences*, *12*(23), 12131. DOI: 10.3390/app122312131.