BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 71(2), 2023, Article number: e144621
DOI: 10.24425/bpasts.2023.144621

CONTROL AND INFORMATICS

Modular distributed models of production systems: a Petri net-based approach

Reggie DAVIDRAJUH[1]*, **Damian KRENCZYK**[2], and **Bozena SKOLUD**[2]

[1] University of Stavanger, Stavanger, Norway
[2] Silesian University of Technology, Gliwice, Poland

Abstract. Modeling and simulation are key performance analysis and control techniques to optimize decision-making as well as design and operate complex production systems. They are also indicated as one of the technological pillars of modern industry and IT solutions supporting the implementation of the roadmap toward Industry 4.0 in the areas of digital transformation and automation. In the context of the required rapid transformation of today's enterprises, it becomes extremely important to look for solutions that allow the use of the existing infrastructure, information, and energy, so as to minimize the negative impact of new technologies and the transformation process itself on the environment. The article presents an approach to modeling large and complex production systems with the use of distributed Petri net models allowing the use of the possessed IT infrastructure as consistent with the idea of sustainable development in the activities of enterprises. This eliminates two major problems that render traditional models unusable. The first is related to the difficulties in analyzing and verifying models of enormous size and infinite space of states. The second is related to the required computing power, if such analyzes are to be performed on one computing unit, which would force the producers to replace the IT infrastructure. For this purpose, modular Petri nets are introduced. Other benefits of modularization, such as smaller components that can be independently analysed, are also presented in the paper. The proposed modular Petri net has been implemented in the proprietary GPenSIM software. The paper is complemented by a practical example of industrial modeling of production systems with automated guided vehicles (AGVs) using the modular model with intelligent Petri modules.

Key words: production engineering systems; sustainability; modular Petri nets; Petri modules; gpensim.

1. INTRODUCTION

Manufacturing facing new challenges, including shorter product life cycles and increasing competition. Manufacturers are striving to rationalize the preparation of production in every aspect of this stage, so designing, manufacturing, and supporting processes, and producing a wide range of products at lower costs, the focus is on modularity in various aspects. One approach is to understand modularity as building a complex product or process from smaller subsystems that can be designed independently, this approach is represented by Baldwin and Clark [1]. Kusiak in [2] expanded the understanding that modularity refers to products, processes, and resources that fulfill various functions, and are combined for a common goal, effective production. Modularity allows for production in a distributed environment and the outsourcing of resources used for the production of individual modules.

Such a solution is beneficial from the point of view of sustainable development, as it allows for the appropriate allocation of production sites, for example, close to material resources or close to the market. The solution minimizes the necessary logistic activities and reduces the consumption of energy necessary for transport associated with them.

Each complex production is preceded by the preparation stage. These include modeling, which is responsible for the elimination of disturbances, in particular related to the qualitative approach, answering the question: of whether the planned production and logistic flows can be realized, and whether there will be deadlocks or starvation. Various mathematical models are used for this purpose. The more popular ones include Markov networks, Petri networks, and simulation modeling.

The solution proposed in this paper allows for the dispersion of calculations, and thus it will reduce the requirements for the capabilities of computers. In turn, the possibility of using currently operated equipment may affect the real reduction of electronic waste. Thus, the practices of efficient use of computing resources are also supported in this way, in accordance with the postulates of the so-called "Green Computing". Green Computing is defined as the effective and efficient use of computers and IT technology by people in an environmentally friendly manner. It also places great emphasis on computer resource reuse and virtualization, which refers to the abstraction of computer resources, such as the process of starting two or more computer systems on one set of physical hardware [3, 4]. With virtualization, physical systems can be merged into virtual machines, thus disconnecting the original hardware and reducing energy and cooling consumption. It is also a technology that has the ability to logically separate the physical resources of a server and use them as different isolated machines, called virtual machines [5].

The problem turns out to be serious when it comes to complex production systems, which are very extensive and often

*e-mail: Reggie.Davidrajuh@uis.no

Manuscript submitted 2022-09-19, revised 2023-01-26, initially accepted for publication 2023-01-30, published in April 2023.

oversized. In order to be able to evaluate and find deficiencies (e.g. bottlenecks, critical resources), it is necessary to develop and apply mathematical models. However, mathematical models of these tend to be huge, so analyzing these for performance improvement becomes impossible. Furthermore, attention should be paid to the fact that some production processes are inherently discrete (e.g. car production). Others, in turn, are continuous processes that must be discretized (e.g. food production or robotics). For this reason, mathematical modeling also requires more and more computational possibilities, and thus the purchase of new hardware and software. In the era of the rapid pace of the development of electronics and information technology, it has become a bane of enterprises, as computers that are still quite functional are withdrawn from operation and replaced with new ones. From a different perspective, modeling and simulation issues are important in the context of guaranteeing ecological and social assumptions and harmonious technological development [6].

Furthermore, decentralization and virtualization are some of the main drivers and fundamental concepts of Industry 4.0, in addition to flexibility, product personalization, and corporate social responsibility. Modularity and virtualization are also listed as one of the six main design principles for Industry 4.0 implementation [7]. In this context, modularity is combined with the modular simulation strategy, used to reduce the complexity of the model building, with the ability to reuse and share sub-models, and is an important feature of agent systems [7]. For complex manufacturing systems, which consist of many sub-systems that operate simultaneously and interact with each other in different manufacturing phases, the current discrete event simulation (DES) frameworks are often proposed. However, many simulators are not designed to be easily extensible and to have features such as network transparency and an open and well-defined interface, and to serve a wide range of discrete event simulation tasks [8].

The main research question this paper is trying to solve: In this paper, Petri nets are proposed as a very useful method for modeling discrete production systems due to their graphical self-documentation property [9]. Petri nets use some mathematical basis, and therefore there are many analytical tools [10, 11]. The problem with implemented Petri net models is their enormous size and the infinite space of states [12]. To avoid these problems, this paper proposes modular networks in which modules can be run on different computers.

This focuses on modular Petri nets; Section 2 provides a survey on modular Petri nets. Section 3 discusses a new modular Petri net, proposed by the first author of this paper [13]. Section 4 is devoted to the presentation of the modular model with distributed Petri modules. Section 5 summarizes the advantages of modularization and distributed model run, e.g., compact modules that are robust, independently developed, and analysed.

2. PETRI NETS

This section provides a formal definition of Petri nets, followed by a brief literature review on modular Petri nets.

2.1. P/T Petri nets

Petri net formalism consists of many classes of Petri nets. The simplest and original one is the P/T (Place-Transition) Petri nets.

The place-transition Petri net (P/T Petri net, for short) is defined as a four-tuple [14]:

$$PTN = (P, T, A, M_0),$$

where,

- P is a set of places, $P = \{p_1, p_2, \ldots, p_{n_p}\}$.
- T is a set of transitions, $T = \{t_1, t_2, \ldots, t_{n_t}\}$. $P \cap T = \emptyset$.
- A is the set of arcs (from places to transitions and from transitions to places). $A \subseteq (P \times T) \cup (T \times P)$. The default arc weight W of a_{ij} ($a_{ij} \in A$, an arc going from p_i to t_j or from t_i to p_j) is one, unless noted otherwise.
- M is the row vector of markings (tokens) on the set of places. $M = [M(p_1), M(p_2), \ldots, M(p_{n_p})] \in N^{n_p}$, M_0 is the initial marking. Due to the markings, a $PTN = (P, T, A, M)$ is also called a **marked P/T Petri net**.

2.2. Literature review

Modular Petri nets have already been described in [15] and [16]. The methodology of compressing the Petri network modules was presented there. In [15] and [16], a module must be an event graph, and the input and output ports must be transitions. [17] is also one of the early papers on modular Petri nets. In this study, the module possesses a clear-cut interface. [18] and [19] propose "Object-oriented Petri nets" to make use of the benefits of object-oriented programming.

[20] presents a modularization technique using "fusion places" and "fusion transitions." This study also discusses property preservation due to modularization. [21] focuses on manufacturing systems, and uses reusable generic modules. Ref. [22] focuses on tackling models' uncertainties using reconfigurable modules.

[23] presents software for modular Petri nets known as "Exhost-PIPE". [24] focuses on modeling and simulation of molecular networks. [25] focuses on modeling "Spanish National Health System" as a modular model. [26] focuses on analyzing traffic signal control of a network of intersections. Also, [27] uses transitions as interfaces for inter-modular communication.

[28] models eliminating redundancy in virtual enterprises using a modular Petri net. This study uses colored Petri and transitions as the input and output ports of modules. Finally, [29] presents a model of Smart Factory Networks using modular Petri nets.

Already in the 1990s, some papers proposed model size reduction as a way of reducing simulation time. For example, Savi and Xie [15], and Claver *et al.* [16] proposed a reduction theorem in which event graphs (a particular class of Petri nets) can be compressed to a primitive module in which transitions function as input and output ports. However, it will be challenging to model an eventual manufacturing system as an event graph in which every place has exactly one input and one output transition. Usually, the rules for reducing model size do not apply to general manufacturing systems [13].

3. MODULAR PETRI NETS

The classic Petri net did not adequately support modularization. Work on the methodology for creating modules within the Petri nets was carried out in relation to three main aspects – all closely related to the previously mentioned problem of model size explosion (especially about modeling production systems). The first was related to the provision of methods facilitating modeling. These works concerned, among others, the areas of module compression, transparent interfaces, object-oriented Petri nets, functionality-based decomposition of Petri nets into modules, or flexible manufacturing systems as systems with different subsystems. The second aspect focused on facilitating the analysis of complex and huge Petri nets. In this case, the research focused on the following thematic areas: state space analysis, a reusable module for ease of analysis, and reconfigurable modules to tackle uncertainties associated with models or managerial implications of modularization. The third aspect, in turn, concerned the development of tools and areas of possible application of modular Petri nets. A more detailed description of the work in this area can be found at [13]. The conclusions from the aforementioned research indicate both the benefits that modularization is to derive (flexibility – the ability to add or change functionality, intelligibility – legibility of models, reduction of development time and reliability), as well as the ability to effectively model large systems as modular models. It should also be noted that classic monolithic paths are not suitable for inter-module communication, and the designed interface must exist at every level of the module for the module to react independently of the rest of the system.

The author's research [12, 13, 30], mentioned in the introduction, proposes the newest modular Petri nets meeting the above requirements. The modular Petri nets proposed in these two works are implemented in the general-purpose Petri net simulator (GPenSIM) [31, 32]. Hence, real-life systems can be modeled, simulated, and analyzed with GPenSIM as modular Petri net models. This subsection presents a summary of the newest modular Petri nets.

modular Petri nets = Petri modules + inter-modular connectors.

A modular Petri net consists of one more Petri module and zero or more inter-modular connectors (IMC).

Formal definition of modular Petri net:

A **modular Petri net** is defined as a two-tuple:

$$MPN = (\mathbb{M}, \mathscr{C})$$

where,

- $\mathbb{M} = \sum_{i=0}^{m} \Phi_i$ (one or more Petri modules),
- $\mathscr{C} = \sum_{j=0}^{n} \Psi_j$ (zero or more inter-modular connectors).

Formal definition of Petri module:

A **Petri module** is defined as a six-tuple:

$$\Phi = (P_{L\Phi}, T_{IP\Phi}, T_{L\Phi}, T_{OP\Phi}, A_\Phi, M_{\Phi 0}),$$

where

- $T_{IP\Phi} \subseteq T$: $T_{IP\Phi}$ is known as the input ports of the module.
- $T_{L\Phi} \subseteq T$: $T_{L\Phi}$ is known as the local transitions of the module.
- $T_{OP\Phi} \subseteq T$: $T_{OP\Phi}$ is known as the output ports of the module.
- $T_{IP\Phi}$, $T_{L\Phi}$, and $T_{OP\Phi}$, are all mutually exclusive: $T_{IP\Phi} \cap T_{L\Phi} = T_{L\Phi} \cap T_{OP\Phi} = T_{OP\Phi} \cap T_{IP\Phi} = \emptyset$.
- $T_\Phi = T_{IP\Phi} \cup T_{L\Phi} \cup T_{OP\Phi}$ (the transitions of the module).
- $P_{L\Phi} \subseteq P$ is known as the set of local places of the module. Since a module has only local places, $P_\Phi \equiv P_{L\Phi}$.
- $\forall p \in P_{L\Phi}$,
  - $\bullet p \in (T_\Phi \cup \emptyset)$. Only transitions of the module can be input transitions of the local place. A local place can be a source too (no input transitions).
  - $p\bullet \in (T_\Phi \cup \emptyset)$. Only transitions of the module can be output transitions of a local place. A local place can be a sink too (no output transitions).
- $\forall t \in T_{L\Phi}$,
  - $\bullet t \in (P_{L\Phi} \cup \emptyset)$. Only local places can be a source too (no input places).
  - $t\bullet \in (P_{L\Phi} \cup \emptyset)$. Only local places can be output places of local transitions. A local transition can be a sink too (no output places).
- $\forall t \in T_{IP\Phi}$
  - $\bullet t \in (P_{L\Phi} \cup P_{IM} \cup \emptyset)$. (input places of input ports can be local places or places in inter-modular connectors or can be even an empty set).
  - $t\bullet \in (P_{L\Phi} \cup \emptyset)$. (output places of input ports can only be local places or an empty set).
- $\forall t \in T_{OP\Phi}$
  - $\bullet t \in (P_{L\Phi} \cup \emptyset)$. (input places of output ports can be local places or an empty set).
  - $t\bullet \in (P_{L\Phi} \cup P_{IM} \cup \emptyset)$. Local places, IM-places, or an empty set can be output places of output ports.
- $A_\Phi \subseteq (P_L \times T_\Phi) \cup (T_\Phi \times P_L)$: where $a_{ij} \in A_\Phi$ is the internal arcs.
- $M_{\Phi 0} = [M(p_L)]$ is the initial markings in the local places.

Formal definition of inter-modular connector:

An **inter-modular connector (IMC)** is defined as a four-tuple:

$$\Psi = (P_\Psi, T_\Psi, A_\Psi, M_{\Psi 0})$$

where

- $P_\Psi \subseteq P$: P_Ψ is the set of places in the IMC (known as the IM-places). $\forall p \in P_\Psi$,
  - $\bullet p \in (T_{OP} \cup T_\Psi \cup \emptyset)$ (output ports of modules, IM transitions of this specific IMC, or none can be the input transitions of IM places).
  - $p\bullet \in (T_{IP} \cup T_\Psi \cup \emptyset)$ (input ports of modules, IM transitions of this specific IMC, or none can be the output transitions of IM places).

In other words, IM places are not allowed direct connections with either local transitions of modules or IM transitions of other IMCs.

- $\forall p \in P_\Psi,\ \ \forall i\ \ p \notin P_{\Phi_i}$ (a local place of a module can not be an IM-place).
- $T_\Psi \subseteq T$: T_Ψ is the transitions of the IMC (*aka* IM-transitions). $\forall t \in T_\Phi$,
  - $\bullet t \in (P_\Psi \cup \emptyset)$ (only the IM-places of this specific IMC or none can be the input places of IM-transitions).
  - $t \bullet \in (P_\Psi \cup \emptyset)$ (only the IM-places of this specific IMC or none can be the output places of IM-transitions).
- $\forall t \in T_\Psi,\ \ \forall i\ \ t \notin T_{\Phi_i}$ (an IM-transition is not allowed to be a member of any modules).
- $A_\Psi \subseteq (P_\Psi \times (T_\Psi \cup T_{IP})) \cup ((T_\Psi \cup T_{OP}) \times P_\Psi)$: where $a_{ij} \in A_\Psi$ is the IMC arc.
- $M_{\Psi 0} = [M(p_\Psi)]$ is the initial markings in the IM-places.

3.1. Application example: a production systems with AGVs

Figure 1 shows a production system that uses Automated Guided Vehicles (AGV). There are six workstations ('S1'–'S6'). The paths between the workstations are labeled as 'T12' to 'T61'.

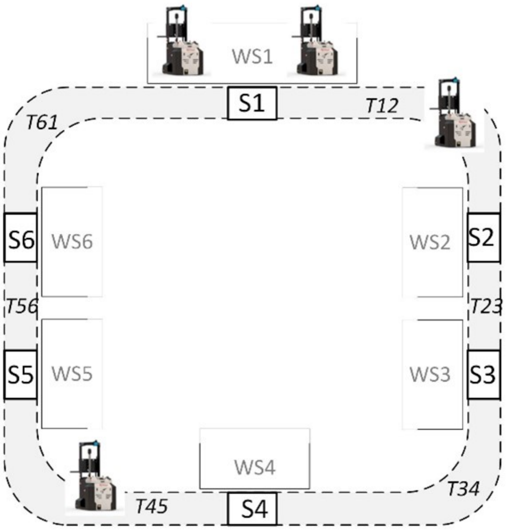

Fig. 1. Production system involving AGVs

The reason for modeling this system is to find out the number of AGVs needed for optimal operation. A similar problem is described in [33] and [34]; however, these works do not use modular models and do not provide a solution (model). The characteristics of the system:

- AGVs move in both directions (clockwise and counter-clockwise).
- There is only one path between two stations. This means only one AGV can move on the path. However, multiple AGVs can be parked in any workstation.
- Only at workstation S1, an AGV can change its direction.
- Also, only at station S1, an AGV can be decommissioned.

3.2. P/T Petri net (monolithic model)

A non-modular Petri net model of the production system is shown in Fig. 2. Details:

- Place Vpool contains the initial tokens (n number of vehicles). At workstation S1, Cin introduces AGVs in a clockwise direction into the system, whereas Ain anti-clockwise direction.
- Two places represent each workstation (e.g., SiC and SiA represent Si); SiC for vehicles clockwise, and SiA, is anti-clockwise.
- TmnC and TmnA represent the movement of a vehicle between workstations Sm and Sn.
- pmn is a locking mechanism for the lane Lmn, so that either T34A or T34C fires (moves an AGV), making use of the track T34.
- Finally, transition Cout removes AGVs in the clockwise movement from further operations, whereas Aout in anti-clockwise movement.

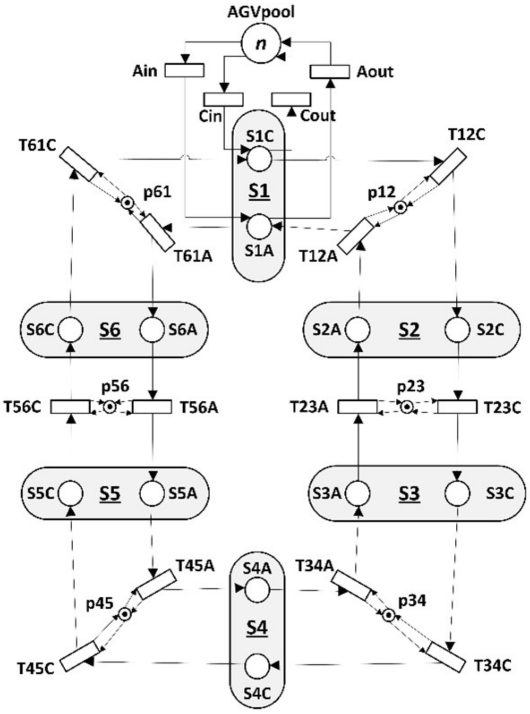

Fig. 2. Non-modular Petri net model

3.3. Colored Petri net (monolithic model)

The model in Fig. 2 can be simplified by the use of token colors:

- Ain and Cin can become one (Tin): Tin introduces AGVs in both directions by injecting tokens with the color 'A' or 'C'. In the same manner, Aout and Cout can become one (Tout).
- SiA and SiC can become one (Si). In other words, place Si represents the station Si.

The non-modular Petri net shown in Fig. 2 has two problems:

- **Extensibility**: The model is not extensible; for example, if more stations need to be added, the model must be re-designed.
- **Comprehensibility**: The model is not comprehensible due to many elements.

- **Robustness**: The model possesses vague assumptions. For example, one of the assumptions is that once an AGV starts from one workstation, it will somehow end up in the following workstation. In this model, it is not possible to include delays and stoppages in traveling.

3.4. Modular Petri net model

Figure 3 presents a Petri module. The Petri module is for the movement of AGVs between any two neighboring workstations (e.g., S0 and S1). The internal part of this module includes delays and stoppages. The advantage of this module is that any other details can be independently added without affecting the other modules. Hence, this Petri module is extensible (because more details can be independently added), and robust (to cope with changes).

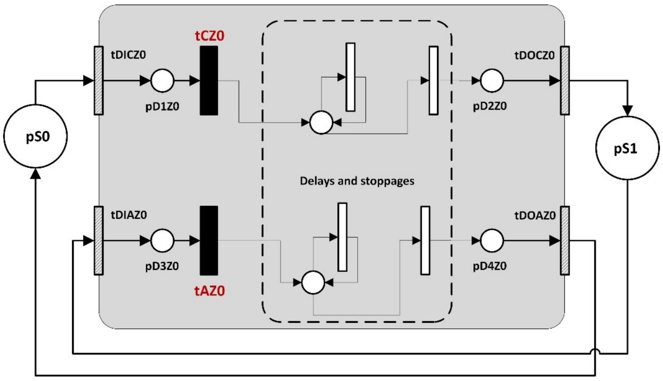

Fig. 3. A **Petri module** that represents the movement of AGV between two workstations

The overall model is shown in Fig. 4. This model is obtained by putting the Petri modules together. Since the workstations stand outside the Petri modules, they become the IMCs. The overall model is comprehensible (easy to understand).

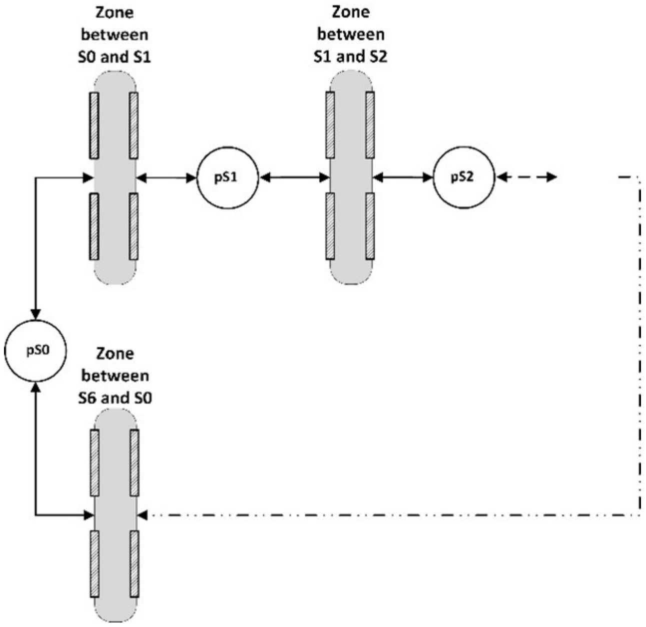

Fig. 4. The modular Petri Net model

4. MODULAR MODEL WITH DISTRIBUTED PETRI MODULES

Though Fig. 4 seemingly satisfies the three characteristics of modular models, the whole model must be run on a single computer. This (running the whole model on a computer) has a serious flaw; namely, the simulation would be time-consuming and computationally intensive (especially for complex production systems). Hence, running the whole model on a computer is not a good idea.

Figure 5 shows a unique way to solve the problem of investing in new high-performance computers, which eliminates the problem of large simulation time. In Fig. 5, each module (or a small group of modules) runs on a computer, and these modules are connected through an inter-computer communication facility (e.g., TCP/IP connection). In the distributed model, the underlying communication infrastructure replaces the buffering places and acts as the IMC. Additionally, such a simulation can be carried out on older infrastructure, thus reducing the potential amount of e-waste. This solution is also a type of virtualization that allows for concurrently running of tasks for oversized models (systems).

Though the distributed modules tend to reduce the simulation times and also solve the problem of the inability to use existing infrastructure with lower computing power, as the modules are run on different computers, inter-modular communication may become another problem. For example, if the whole model is run on a single computer, two adjacent modules may communicate via a buffering place, not taking much time. However, if two modules are run on different computers, the communication between these two modules has to go through many layers, including GPenSIM core, MATLAB, and the TCP/IP layer. Hence, in the distributed modular model, it is also necessary to minimize inter-modular communications; the following subsection suggests one way of minimizing inter-modular communication – by making the modules more intelligent.

4.1. Modular model with intelligent Petri net modules

The previous subsection mentions that high interactions between the modules running on different computers can cause some simulation time due to communication delays. One way to reduce the high interactions between the modules is the introduction of a supervisor; see Fig. 6. The supervisor is updated with the status of the other modules. However, there is a drawback to suggesting a "centralized" supervisor node. If this supervisor is out of function for any reason (e.g., problems in the network at the computer hosting the supervisor), then a complete model breakdown will result. Therefore, additional solutions should be introduced so that the supervisor does need not to collect all the necessary data from the other modules, as some data can be estimated or learned from the historical data.

4.2. Intelligent supervisor

To minimize the design complexities, the designer may opt for a master-slave paradigm in which the supervisor functions as the master and the other modules represent the zones. In this way, we can minimize inter-modular communication between the zone modules and force the zone modules to communicate

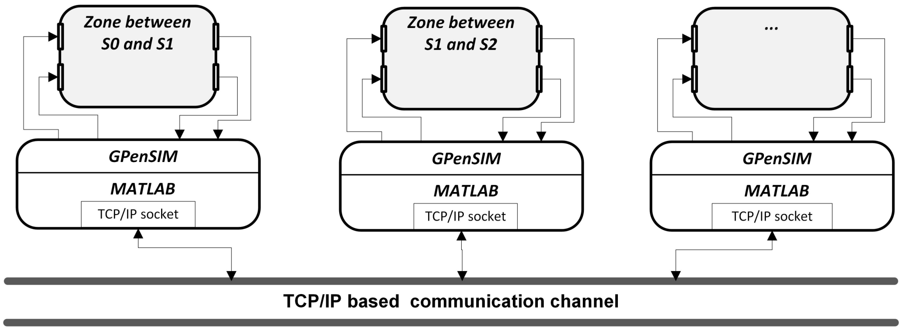

Fig. 5. The modular Petri Net model with distributed modules (internals of the modules are not shown)

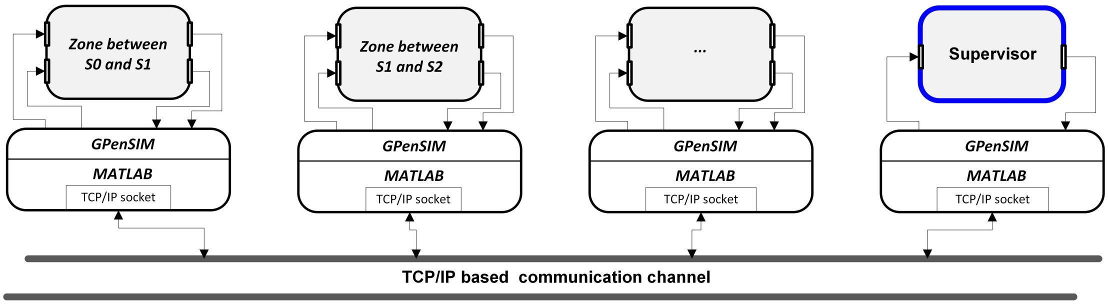

Fig. 6. The modular Petri Net model with a supervisor module

with the supervisor. Also, the supervisor can become more intelligent. For example, it need not expect all data from the modules, as it can estimate some data either from the historical data or using an estimator, e.g., Kalman filter [35]. Though some papers use AI and machine learning (e.g., [36, 37]) and digital twins (e.g., [38]) to add intelligence to Petri net models, this paper proposes the Kalman filter for predicting missing data as it is much simpler.

4.3. Application example

For the example in Fig. 3, **tDICZ0** is to take an AVG from the station **pS0** and make it ready to travel (in the counterclockwise direction to the station **pS1**) by placing it in the buffer **pD1Z0**. At the same time, let us say that there is an AGV approaching **pS0** with high priority, and this AGV is the one that is supposed to travel to **pS1**. However, the module cannot prioritize the incoming AGV as it does not possess any information about it. On the other hand, the supervisor can instruct **tDICZ0** whether to take on AGV or wait until another one with higher priority arrives.

The supervisor estimating arrival times of AVGs:
To avoid too many messages from the zone modules, the supervisor can limit these messages only to an absolute minimum and estimate the (missing) other values. For example, the supervisor can estimate the arrival times of different AGVs at different stations using Kalman filters.

Let us assume that AGVs move at a constant velocity; estimating position with velocity is a frequently used example for the Kalman filter ([39]), and the implementation (code) is so simple to be incorporated into the supervisor. The Kalman filter effectively removes the noise (variations in the velocity) and the sensitivity to the variation. Also, the Kalman filter manages missing inputs from the zone modules regarding the position and velocity of the AGVs.

The Kalman filter for estimating the position of an AVG:

```
    % *** INITIAL STEP ***
% x: state variable (position)
$x = 0;$
% P: error covariance
$P = 5 \cdot eyes(2);$
% A: system matrix
$A = [1\ \ dt;\ \ 0\ \ 1];$
% Q: covariance of state transition noise
$Q = [1\ \ 0;\ \ 0\ \ 10];$
% H: state to measurement matrix
$H = [0\ \ 1];$
% R: covariance of measurement noise
$R = 20;$
% *** PREDICTION STEP ***
$x = A \cdot x;$
$P = A \cdot P \cdot A^T + Q;$
% *** COMPUTING THE KALMAN GAIN ***
$K = P \cdot H^T \cdot inv(H \cdot P \cdot H^T + R);$
% *** FINALLY, ERROR COVAR FOR NEXT ROUND ***
$P = P - (K \cdot H \cdot P)$
```

The proposed solution allows for the elimination of the problem of inter-module communication and significantly extends the possibilities of using models based on Petri networks for large systems.

Based on the application example presented in this section, a prototype was made for a master's thesis project as a proof-of-concept [40]. The application example, as a proof-of-concept, is purposely chosen to be simple and short for brevity. However, the application definitely shows the benefits of the presented theory; namely, the modular Petri nets obtained by the proposed theory present smaller (compact) modules that can be run on different (geographically distributed) CPUs. Thus, the distributed model runs faster and less complex.

Also, this paper does not present the analytical details such as preservation of the properties. Davidrajuh [41] presents some analytical proofs and clarifications based on P- and T-invariants. Due to brevity, the analytical details are omitted in this paper.

A final word on simulation time: During the simulations of Petri nets, simulations (iterations) use incidence matrix multiplication by the vector representing the firing transitions. However, if the incidence matrix is extremely large (as there are many places and transitions), the matrix multiplications will be costly in terms of CPU computation time. That is why we propose dissecting large Petri net models into smaller modules. Thereby, the incidence matrices of the modules are much smaller. When we compute matrix multiplications using these smaller incidence matrices also on different CPUs, the computation time will be drastically reduced.

A final word on modular models: Since the 1990s, researchers have proposed modular models for modeling and simulating large manufacturing systems (e.g., [21, 42]). However, this paper not only proposes modules but also shows a way to run these modules on different computers that are close to the manufacturing operations.

5. CONCLUSION

Ensuring the continuous development of numerical methods supporting the management of production systems driving the digitization of enterprises, along with methods of their effective implementation, must meet the requirements of sustainable development. Solutions should be developed that will minimize the growing demand for information and energy, and therefore also, IT technologies being developed should take into account the requirements of, inter alia, regarding energy efficiency and optimal use of available computing resources. These requirements are met by the proposed method of creating modular distributed models of production systems based on the Petri nets approach. The proposed solution directly affects the environment by limiting e-wastes as well. Issues related to intermodular communication were identified as related to the requirement for communication between modules through many layers, including the simulation platform and the TCP/IP layer. In order to eliminate the above problems, it was proposed to use the so-called modular model with intelligent Petri modules, introducing a supervisor that was updated on the status of the other modules and a mechanism allowing for estimating the missing values based on the Kalman filter.

The subject of further research in this area will be the analysis of other properties of creating modular Petri net models, such as module independent development, as well as analysis and testing of the modules independently.

REFERENCES

[1] A. Shaik, V. Rao, and C. Rao, "Development of modular manufacturing systems – a review," *Int. J. Adv. Manuf. Technol.*, vol. 76, pp. 789–802, 02 2015, doi: 10.1007/s00170-014-6289-2.

[2] A. Kusiak, "Integrated product and process design: A modularity perspective," *J. Eng. Des.*, vol. 13, no. 3, pp. 223–231, 2002, doi: 10.1080/09544820110108926.

[3] S. Singh, "Green computing strategies & challenges," in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015, pp. 758–760, doi: 10.1109/ICGCIoT.2015.7380564.

[4] W. Binder and N. Suri, "Green computing: Energy consumption optimized service hosting," in *SOFSEM 2009: Theory and Practice of Computer Science*, M. Nielsen, A. Kučera, P.B. Miltersen, C. Palamidessi, P. Tůma, and F. Valencia, Eds. Springer Berlin Heidelberg, 2009, pp. 117–128.

[5] N. Almurisi and S. Tadisetty, "Cloud-based virtualization environment for iot-based wsn: solutions, approaches and challenges," *J. Ambient Intell. Humaniz. Comput.*, pp. 1–23, 03 2022, doi: 10.1007/s12652-021-03515-z.

[6] I. Rojek, M. Macko, D. Mikołajewski, M. Sága, and T. Burczyński, "Modern methods in the field of machine modelling and simulation as a research and practical issue related to industry 4.0," *Bull. Pol. Acad. Sci. Tech. Sci.*, vol. 69, no. 2, p. e136717, 2021, doi: 10.24425/bpasts.2021.136717.

[7] W. de Paula Ferreira, F. Armellini, and L.A. De Santa-Eulalia, "Simulation in industry 4.0: A state-of-the-art review," *Comput. Ind. Eng.*, vol. 149, p. 106868, 2020, doi: 10.1016/j.cie.2020.106868.

[8] D. Wangerin, C. DeCoro, L. Campos, H. Coyote, and I. Scherson, "A modular client-server discrete event simulator for networked computers," in *Proc. 35th Annual Simulation Symposium*, 2002, pp. 125–133, doi: 10.1109/SIMSYM.2002.1000138.

[9] J. Patalas-Maliszewska, R. Wiśniewski, M. Topczak, and M. Wojnakowski, "Design optimization of the petri net-based production process supported by additive manufacturing technologies," *Bull. Pol. Acad. Sci. Tech. Sci.*, vol. 70, no. 2, p. e140693, 2022, doi: 10.24425/bpasts.2022.140693.

[10] J.L. Peterson, *Petri net theory and the modeling of systems*. Prentice Hall PTR, 1981.

[11] B. Skolud, D. Krenczyk, and R. Davidrajuh, "Solving repetitive production planning problems. an approach based on activity-oriented petri nets," in *International Joint Conference SOCO'16-CISIS'16-ICEUTE'16*. Springer, 2016, pp. 397–407.

[12] R. Davidrajuh, "Design of a new modular petri nets," in *Petri Nets for Modeling of Large Discrete Systems*. Springer, 2021, pp. 95–105.

[13] R. Davidrajuh, "A new modular petri net for modeling large discrete-event systems: a proposal based on the literature study," *Computers*, vol. 8, no. 4, p. 83, 2019.

[14] T. Murata, "Petri nets: Properties, analysis and applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[15] V. M. Savi and X. Xie, "Liveness and boundedness analysis for petri nets with event graph modules," in *International Conference on Application and Theory of Petri Nets*. Springer, 1992, pp. 328–347.

[16] J. Claver, G. Harhalakis, J. Proth, V. Savi, and X. Xie, "A stepwise specification of a manufacturing system using petri nets," drum.lib.umd.edu, Tech. Rep., 1992.

[17] G.G. De Jong and B. Lin, “A communicating petri net model for the design of concurrent asynchronous modules,” in *31st Design Automation Conference*. IEEE, 1994, pp. 49–55.

[18] L.-C. Wang, “Object-oriented petri nets for modelling and analysis of automated manufacturing systems,” *Comput. Integr. Manuf. Syst.*, vol. 9, no. 2, pp. 111–125, 1996.

[19] L.-C. Wang and S.-Y. Wu, “Modeling with colored timed object-oriented petri nets for automated manufacturing systems,” *Comput. Ind. Eng.*, vol. 34, no. 2, pp. 463–480, 1998.

[20] S. Christensen and L. Petrucci, “Modular analysis of petri nets,” *Comput. J.*, vol. 43, no. 3, pp. 224–242, 2000.

[21] G.J. Tsinarakis, N. Tsourveloudis, and K.P. Valavanis, “Modular petri net based modeling, analysis, synthesis and performance evaluation of random topology dedicated production systems,” *J. Intell. Manuf.*, vol. 16, no. 1, pp. 67–92, 2005, doi: 10.1007/s10845-005-4825-5.

[22] H. Lee and A. Banerjee, “A modular petri net based architecture to model manufacturing systems exhibiting resource and timing uncertainties,” in *2009 IEEE International Conference on Automation Science and Engineering*. IEEE, 2009, pp. 525–530.

[23] O. Bonnet-Torrès, P. Domenech, C. Lesire, and C. Tessier, “E xhost-pipe: Pipe extended for two classes of monitoring petri nets,” in *International Conference on Application and Theory of Petri Nets*. Springer, 2006, pp. 391–400.

[24] M.A. Blätke, S. Meyer, and W. Marwan, “Pain signaling-a case study of the modular petri net modeling concept with prospect to a protein-oriented modeling platform,” in *Proc. of the International Workshop on Biological Processes & Petri Nets (BioPPN), satellite event of Petri Nets*. Citeseer, 2011, pp. 117–134.

[25] C. Mahulea, J.-M. García-Soriano, and J.-M. Colom, “Modular petri net modeling of the spanish health system,” in *Proc. of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*. IEEE, 2012, pp. 1–8.

[26] M. dos Santos Soares and J. Vrancken, “A modular petri net to modeling and scenario analysis of a network of road traffic signals,” *Control Eng. Practice*, vol. 20, no. 11, pp. 1183–1194, 2012.

[27] A. Słota, J. Zając, and M. Uthayakumar, “Synthesis of petri net based model of a discrete event manufacturing system for nonlinear process plan,” *Manag. Prod. Eng. Rev.*, vol. 7, 2016, doi: 10.1515/mper-2016-0018 .

[28] R. Davidrajuh, “Distributed workflow based approach for eliminating redundancy in virtual enterprising,” *J. Supercomput.*, vol. 63, no. 1, pp. 107–125, 2013.

[29] S. Berger, M. Bogenreuther, B.S. Häckel, and O. Niesel, “Modelling availability risks of it threats in smart factory networks - a modular petri net approach,” in *European Conference on Information Systems (ECIS)*, 2019.

[30] R. Davidrajuh, D. Krenczyk, and B. Skolud, “Modeling production systems as modular systems: A petri nets based approach,” in *Advances in Manufacturing III. Lecture Notes in Mechanical Engineering*, J. Trojanowska, A. Kujawińska, J. Machado, and I. Pavlenko, Eds. Springer International Publishing, 2022, pp. 3–12.

[31] R. Davidrajuh, *Modeling discrete-event systems with gpensim: An introduction*. Springer, 2018.

[32] GPenSIM, “General-purpose Petri net simulator,” http://www.davidrajuh.net/gpensim, Tech. Rep., 2019, (accessed: 20 July 2020).

[33] B. Hrúz and M. Zhou, *Modeling and control of discrete-event dynamic systems: With petri nets and other tools*. Springer, 2007, vol. 59.

[34] R. Davidrajuh, “Models of real-life systems,” in *Petri Nets for Modeling of Large Discrete Systems*. Springer, 2021, pp. 29–50.

[35] G. Bishop, G. Welch *et al.*, “An introduction to the kalman filter,” *Proc of SIGGRAPH, Course*, vol. 8, no. 27599–23175, p. 41, 2001.

[36] H. Li, D. Yang, H. Cao, W. Ge, E. Chen, X. Wen, and C. Li, “Data-driven hybrid petri-net based energy consumption behaviour modelling for digital twin of energy-efficient manufacturing system,” *Energy*, vol. 239, p. 122178, 2022.

[37] J. Friederich, D.P. Francis, S. Lazarova-Molnar, and N. Mohamed, “A framework for data-driven digital twins for smart manufacturing,” *Comput. Ind.*, vol. 136, p. 103586, 2022.

[38] G. Tsinarakis, N. Sarantinoudis, and G. Arampatzis, “A discrete process modelling and simulation methodology for industrial systems within the concept of digital twins,” *Appl. Sci.*, vol. 12, no. 2, p. 870, 2022.

[39] Y. Kim and H. Bang, *Introduction to Kalman filter and its applications*. A-JIN, Korea, 2018.

[40] A.M. da Rocha Pinto Rolim Marques, “Modeling and simulation of large discrete-event systems as Petri modules: An approach based on GPenSim,” University of Minho, Portugal, Tech. Rep., 2021, master Thesis.

[41] R. Davidrajuh, *Petri Nets for Modeling of Large Discrete Systems*. Springer, 2021.

[42] G. Rogers and L. Bottaci, “Modular production systems: a new manufacturing paradigm,” *J. Intell. Manuf.*, vol. 8, no. 2, pp. 147–156, 1997.