Petri net implementation in programmable logic controllers: methodology for development and validation

Igor Azkarate Fernández, Juan Carlos Mugarza Cortabarría, and Luka Eciolaza Echeverría
Department of Electronics and Computing, Mondragon Unibertsitatea
{iazkarate,jcmugarza,leciolaza}@mondragon.edu
Arrasate-Mondragón, Spain

***Abstract*—Lead times are key to good market positioning of providers of automated solutions based on a programmable logic controller (PLC). Testing control software against a digital twin (DT) of the process, any programming errors that may have incurred are detected before commissioning, which reduces project duration. This work raises the possibility of reducing that probability of error when programming discrete event dynamic systems (DEDS), by implementing a Petri net (PN) managing algorithm. A framework is presented which combines the use of this algorithm, by means of pre-incidence and post-incidence matrices and initial marking vector of a net, with code validation through emulation. A use case is brought forward in which the control program of a sequential process with parallel operations is implemented, with both virtual (VC) and real commissioning.**

***Index Terms*—Agile software development, digital twin, discrete-event systems, manufacturing automation, Petri nets, software algorithms, virtual commissioning**

I. Introduction

Lead times for industrial automation projects, both new and reconditioning of old systems, are key to good market positioning of machine manufacturers and system integrators, whose reputation and future contracts may be at stake. Numerous studies focus on how to shorten deadlines, aiming at the final phase, the commissioning of automated solutions [1]. Control device software verification and validation has generally been carried out at this stage. Emulation tools make it possible code testing before the mentioned phase, making it more agile.

After reviewing code testing against real process or previously through emulation, this section focuses on its development, source of potential errors. An approach of using Petri net (PN) implementations for semi-automated code generation on control devices is proposed, in order to reduce bugs that need to be eliminated during virtual and/or real commissioning.

A. Software validation during commissioning

Code is tested in the development environment of the controller itself, commonly a programmable logic controller (PLC) [2], without observing its effect on the behaviour of the process to be automated. An inadequate test procedure can lead to errors being overlooked. Later, in the final phase of the project, the system is commissioned, with the equipment already assembled. It can be up to 25% of total duration [3]. It is exposed to unforeseen events such as collisions, material and/or personal damage, and extra travel and accommodation costs. Resulting downtime affects developers' prestige.

B. Virtual commissioning

The increasing capacity of computing equipment, both in memory as well as in CPU power, combined with the emergence of new emulation tools or digital twins (DT) that virtually reproduce industrial systems and processes, make it possible to test PLC programs before commissioning. Known as virtual commissioning (VC) [4], allows to shorten development and validation phases [5]; a more reliable validation by seeing how a virtual model works, and higher quality results [6]; considering and testing different scenarios [7]; and a reduction in commissioning manpower [8].

C. Implementation of sequential systems in a PLC

Once there are tools for detecting coding errors before commissioning, the research question on which this work is based arises. Is it possible to generate PLC programs with a minimum probability of containing development errors, avoiding the expenses incurred when detecting them later?

Industrial sequential processes can be represented by diagrams according to GRAFCET methodology. This graphic method is a particular case of a PN [9], which was created to model discrete event dynamic systems (DEDS) [10]. GRAFCET or PN implementation in PLC has traditionally been done by means of a boolean variable or a counter for each step or place [11] [12], with its status evaluated at every scan cycle of the device. Currently, development environments include Sequential Function Chart (SFC), a graphic modeling and description method for sequential automation systems, suitable for GRAFCET and standardised in IEC 61131-3.

Literature evidences methodologies of implementing PNs, as a more general case, into standard PLC languages [13] [14] and in an interpreted basis, i.e. with the structural management integrated into the code itself. [15] incorporates the possibility of generating instruction lists automatically by means of the net description. Complete developments involve programming workload, and modifications necessarily affect the code.

D. What is proposed

Focusing on PNs and using their matrix representation [16] and evolution rules, allowing tokens moving among net places as transitions firing consequence, opens a new perspective to the programming of this type of processes. Model validation abilities are much more powerful than those proposed for GRAFCET, mainly related with divergence-convergence managing [17]. Instead, for PN there is a wide theory for model structural analysis and property checking [16], such as liveness, cyclicity, limitation and invariants.

This work proposes a methodology for the development and commissioning of sequential systems controlled by PLC, based on a semi-compiled approach. This approach can be considered semi-compiled because it manages the evolution of any PN, by means of their pre-incidence and post-incidence matrices and initial marking vector. Compared to a GRAFCET-based development, programming workload and the possibility of errors can be reduced, and combined with a virtual validation supported by a DT, they can make commissioning a simple procedure.

II. Methodology

In this section a methodology is proposed, which combines an algorithm that manages, in a semi-compiled way, the structure and evolution of the marking of a PN, with the use of a DT to support the development and commissioning of PLC controlled sequential systems. Previous preparation of data according to net structure is required, and the coding of its interpretation. The specifications that the program meets, and the phases of implementation are described.

A. Algorithm for PN marking management

The algorithm kernel implements the transition firing and marking updating rules for a general PN. Obviously, net interpretation relating coding depends on each particular system. This is why the implementation of the algorithm is called semi-compiled. Note that it is coded in structured language, in order to operate more simply with matrices and for an easier portability to other manufacturers' devices.

1) Startup: initial marking value is given to the current one, and incidence matrix is calculated from the pre-incidence and post-incidence ones. It is executed only in the first scan cycle.

2) Enabling: determine transitions enabled by current marking. It is executed only if a transition has been fired and consequently current marking has changed. The latter is compared with each column of the pre-incidence matrix to determine whether marking enables the transition. If so, it is recorded in an array of enabled transitions.

3) Firable: check whether currently enabled transitions are likely to be fired. According to net interpretation, it evaluates fulfillment of transitions firing conditions, writing them down in an array. Subsequently, it is evaluated if they are liable to be fired, reflecting it in another array. The number of transitions that can be fired from current marking is also noted.

4) Make a decision: implementation of a rule. A firable transition must not have to be fired at the moment. It could be posponed or even no longer allowed after any (some) other(s) transition(s) firing. This PN property is particularly important and interesting in the modeling of sequencing problems: order control in the sequence of transitions firing. A function is introduced for deciding the transition to be fired among the firable ones, as for transitions in conflict. In this case, the last of those in the firable transitions list.

5) Marking update: recalculate. As a consequence of the firing of previously selected transition, among firable ones, net marking is computed. It implies updating it by adding the incidence matrix column related with the fired transition.

6) Actions: places currently marked. This function updates actions related with places, as net interpretation determines, accordind to the achieved new places markings.

Note that, provided with any particular PN incidence matrix and initial marking vector, this algorithm allows token movements among places managing.

B. Phases of the methodology

In a comparison between the conventional procedure and the proposed one (see Table I), specifications, wiring and commissioning are obviously common to both. It is the development of the intermediate phases which makes them different, being these the ones described in more detail.

1) Specifications: key signals and desired operation.

2) PN model development (start of DT modeling): once specifications are defined, the PN structural design is carried out for its subsequent digitalization and analysis. Interpretation is added, associating conditions and actions with PN structure transitions and places, respectively. Furthermore, the process DT can be developed, with the support of an emulation software, for VC.

3) PN structural analysis: consists of net graphical edition and token moving among places simulation. It includes PN properties analysis and representative matrices generation. It is useful a software tool such as that shown in Fig. 1, Platform Independent Petri net Editor 2 (PIPE2) [18]. It provides net invariants computing, allowing testing strctural properties as boundness, liveness and deadlock freeness. Finally, a .html file is generated with the matrix representation of the net structure.

4) Hardware and variable definition (internal and I/O): in the PLC programming environment, the project containing the algorithm is opened and hardware and variables are set.

5) Structure implementation, transfer of matrices data to PLC memory: from the .html file generated by PIPE2, the matrix representation is transferred to a data block in PLC memory. A parser can be used to adequate the information provided by PIPE2 to the PLC's particular syntax.

6) Interpretation implementation (end of DT modeling): the aim is to implement II-A3 and II-A6 points, according to the above added interpretation (II-B2). Each case requires corresponding code development, which affects program memory. Enabling and actions functions are edited, associating states of digital inputs and/or internal variables to transitions, and places to digital outputs and/or internal variables.

TABLE I: Conventional peocedure versus proposed procedure

Conventional procedure	Proposed procedure
- Specifications.	- Specifications.
- Design of the GRAFCET diagram.	- PN model development (start of DT modeling). - PN structural analysis: graphic edition, simulation (movement of tokens around places), validation and matrix generation.
- Interpreted implementation of the GRAFCET diagram.	- Hardware and variable definition (internal and I/O). - Structure implementation: transfer of matrices data to PLC memory. - Interpretation implementation (end of DT modeling).
- PLC emulation in its development environment.	- Virtual commissioning.
- Wiring.	- Wiring.
- Commissioning.	- Commissioning.

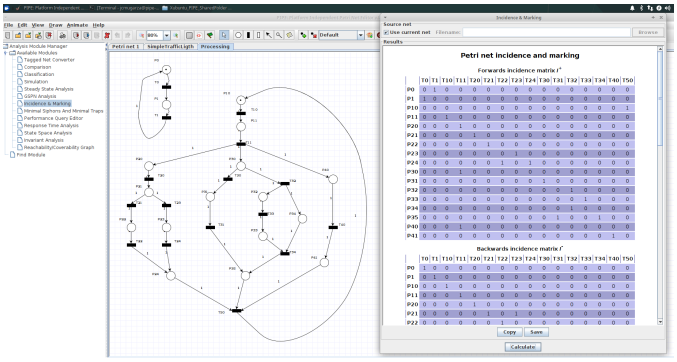

Fig. 1: PN structure and associated matrices in PIPE2.

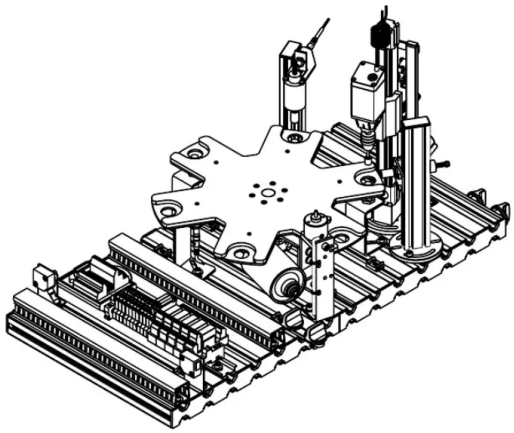

Fig. 2: System under study [19].

7) Virtual commissioning: the DT must be available at this point. This step implies verifying the correct behavior of the process, according to specifications and with all virtual sensors and actuators connected to the real or emulated PLC, commonly using OPC UA standard.

8) Wiring: between PLC and process sensors and actuators.

9) Commissioning: process performance testing.

III. Use case

This section presents a first and simple use case of the algorithm and the proposed methodology. Their application has as process the didactic station shown in Fig. 2, of which a couple of physical units are available in Mondragon Unibertsitatea (MU), for teaching and research purposes. The work to be done consisted on programming a PLC based on specifications for the automation of the mentioned sequential process with parallel tasks and synchronization among them.

A. Process

The process to be automated was Processing Station of Festo Didactic, global leader in technical training solutions for industrial and process automation. It was chosen because, on the one hand, its PN has exclusively binary marking and there are not shared resources, thus it is a simple case to represent both by GRAFCET and PN. On the other hand, the manufacturer provides an emulator including a station's DT.

The operations carried out on this station are to check the correct positioning of workpieces, to test if they already have a central hole; to machine them; and to supply them to a subsequent station.

In the desired behavior, workpieces are tested and processed on a rotary indexing table driven by a direct current (DC) motor. The table is positioned by a relay circuit, with its position being detected by an inductive sensor. Workpieces are tested and drilled in two parallel processes. A solenoid actuator with an inductive sensor checks that the items are inserted in the correct position. During drilling operation, the workpiece is clamped by a solenoid actuator. Finished items leave the system by means of an electrical ejector. The modules can work simultaneously and in a coordinated way, for a shorter cycle time.

B. Work to be done

The work to be done was to automate the described process. Both procedures in Table I were applied, for a first comparison. The first one was the most usual: a fully interpreted GRAFCET diagram. It was considered that design and implementation errors were probable. The second option was the semi-compiled approach in section II, supported by emulation. The algorithm, mainly fixed, manages structure and marking of a PN, whose

particular case is a GRAFCET, as a matrix data structure. Programming workload is limited to incorporating interpretation to that structure: associating conditions to transitions and actions to places. Coding error probability was lower. Even more, PN structure was previously validated and matrix data loading automated.

C. Resources

The material used consisted of the station, a Siemens S7-1500 series PLC, and a laptop with the following software:

- PIPE2 tool, for creating and analyzing PN structures, and generating associated matrices.
- A propietary application for converting data from HTML to text format, suitable for an easy insertion in variable declaration modules of various environments.
- TIA Portal V15.1 environment for Siemens control devices, with the algorithm already available; and S7-PLCSIM Advanced V2.0 SP1, Siemens PLC emulator.
- CIROS® Education 6.2, virtual learning environment for PLC controlled systems.
- EzOPC 5.6, an OLE for Process Control (OPC) server that allows to control DTs through an external PLC.

D. Implementation based on conventional procedure

The steps listed in the left column of Table I were followed in the usual procedure for programming a DEDS applying GRAFCET in a PLC. The development environment has a specific editor for designing the code in a visual way.

E. Implementation based on proposed procedure

Defined specifications (II-B1) in terms of desired operation and key variables and signals, the PN structure of Fig. 3 was twofolded designed (II-B2). The first one manages start and stop situations. The second reflects the normal operation of the process. It consists of a branch for table rotation, in case the modules have concluded their operations, and a divergence in three simultaneous branches for the respective modules.

Afterwards, net interpretation was defined, associating (boolean) conditions to transitions and actions to places. For example, the actions of P_{21} are the activation of the workpiece test mechanism, and the launching of a timer for this operation. On the other hand, the condition of T_{23} queries whether the testing time has expired without a signal that the workpiece is correct, or whether reset button is pressed.

Once model structural analysis (II-B3) was carried out using PIPE2, pre-incidence, post-incidence and initial marking matrices were generated, as already shown in Fig. 1.

PN implementation was carried out during corresponding steps (II-B4, II-B5 and II-B6). At this point, the DT of the system was already available, to face next phase, that of VC (II-B7). Note that a virtual model of the station is available in CIROS® Education, so all that had to be done was to add an OPC client object and establish virtual connections between sensors (actuators) and digital inputs (outputs) of the PLC.

A software-in-the-loop (SIL) type configuration was defined through OPC UA standard. EzOPC, which allows to control

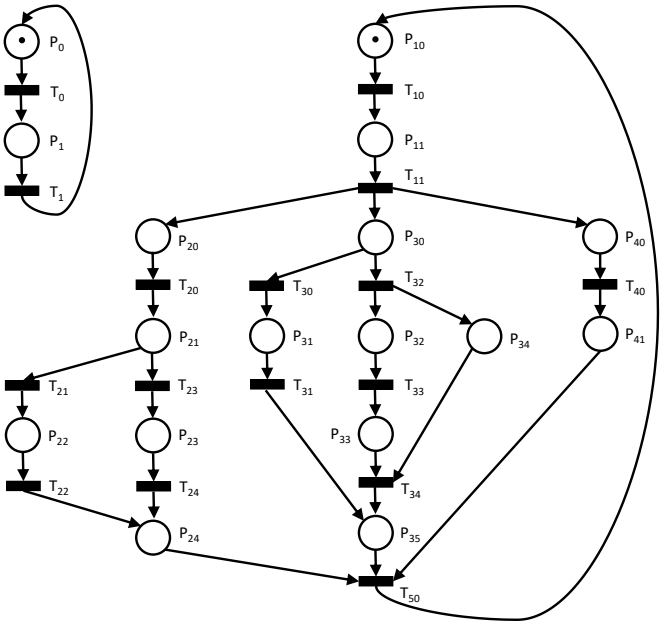

Fig. 3: PN structure

processess in CIROS via an external PLC, acted as a server; and process (CIROS) and PLC (S7-PLCSIM Advanced) emulators operated as clients. VC consisted of verifying correct process operation in its DT (Fig. 4), according to specifications and with all sensors and actuators connected to the emulated Siemens PLC I/O signals using OPC UA.

Prior commissioning the control system, real PLC and real process were wired (II-B8), in the same terms as the virtual one. It was carried out through the terminal block of the educational station. Once the project was uploaded to the device, it was checked whether the real process worked according to specifications (II-B9).

IV. Results

By means of the proposed procedure, a PN has been implemented in PLC in an easy and agile way, with low coding workload and probability of errors (and whose detection is supported by DT), and with an simple final commissioning.

Error-prone points are reduced to one relating to code development and three as a consequence of the paradigm shift:

- In contrast to a GRAFCET, the critical point in programming sequential parts is minimal: manual addition of conditions and actions, which is the only non-fixed part. The algorithm can be considered easy to adapt to any application. For other functions such as PID controllers the workload and potential errors remain the same.
- A PN needs to be set up: there can be misinterpretations of the non-formal specification. This is something to which the design of an interpreted GRAFCET is also exposed. It should also be noted that PIPE2 is a tool which is easy to use. Any technician who knows how to program a GRAFCET diagram in a PLC can adapt it to the edition of a PN and its structural analysis.

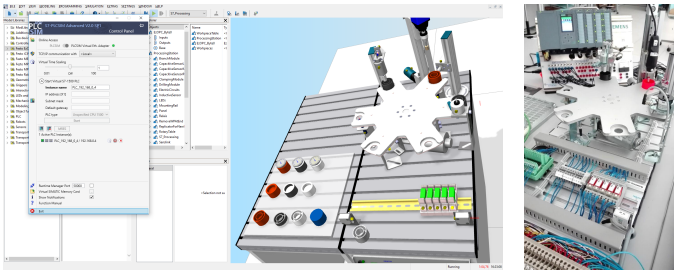

Fig. 4: Virtual and real commissioning.

- Matrices need to be mapped with existing variables: this operation has been automated.
- Design of a DT and its communication with PLC: despite the lack of skilled professionals a few years ago [20], modeling is already present in educational [4] and industrial environments [5]. In this case, a DT of the didactic station is available in libraries of the emulation tool supplied by the manufacturer.

V. Conclusions

The proposed approach makes it possible to program PNs in PLCs with the following features:

- It is semi-compiled. PN structure is previously validated.
- Only PN interpretation is error-prone in coding.
- Standard methodology. It consists of transferring the mainly fixed algorithm to the desired PLC.
- The inclusion or modification of a PN is simple.

The use of other PLCs or DTs is possible, if they fulfill OPC UA compatibility. The modeling effort will be justified in cases of greater complexity or with more non-sequential parts that cannot be represented by a PN. The latter will be programmed by the technical staff, supported by a DT to test and validate the developed code, as well as the sequential parts managed by the algorithm.

The presented has been a first approach, of simple features and with the real process available for a complete implementation. It is in more demanding applications that the advantages of this methodology over GRAFCET will become more apparent. As future work, it is planned to experiment with other PLCs and more complex industrial developments, normally through DT. On the one hand, shared resources (e.g. manipulating robots), synchronization, and limited buffers (e.g. temporary storage) will be managed integrated in the PN's own structure. The latter, for example, would require counters in GRAFCET, as the variable associated with each step is boolean. On the other hand, it should be noted that for several nets the treatment is identical to that of the use case: they are treated as a single one, in which some arcs are considered to be zero weight. In addition, this approach includes the concept of hierarchy.

An evolution and improvement of this work can be the parameterization of a program block encapsulating the algorithm. In this way it can be used more easily in terms of associating the structure of the PN and its interpretation.

References

[1] S. Makris, G. Michalos, and G. Chryssolouris, "Virtual commissioning of an assembly cell with cooperating robots," *Advances in Decision Sciences*, vol. 2012, pp. 1–11, sep 2012.

[2] E. M. Pérez, J. M. Acevedo, and C. F. Silva, *Automatas programables y sistemas de automatizacion/PLC and Automation Systems*. Marcombo, 2009.

[3] G. Reinhart and G. Wünsch, "Economic application of virtual commissioning to mechatronic production systems," *Production Engineering*, vol. 1, no. 4, pp. 371–379, Dec 2007. [Online]. Available: https://doi.org/10.1007/s11740-007-0066-0

[4] I. Azkarate Fernández, M. Ayani Eguía, and L. Eciolaza Echeverría, "Virtual commissioning of a robotic cell: an educational case study," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 820–825.

[5] M. Ayani, M. Ganebäck, and A. H.C. Ng, "Digital twin: Applying emulation for machine reconditioning," *Procedia CIRP*, vol. 72, pp. 243–248, 01 2018.

[6] M. Schamp, S. Hoedt, A. Claeys, E. Aghezzaf, and J. Cottyn, "Impact of a virtual twin on commissioning time and quality." *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1047 – 1052, 2018, 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896318315970

[7] M. Dahl, K. Bengtsson, M. Fabian, and P. Falkman, "Automatic modeling and simulation of robot program behavior in integrated virtual preparation and commissioning," *Procedia Manufacturing*, vol. 11, pp. 284 – 291, 2017, 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2351978917303116

[8] N. Shahim and C. Møller, "Economic justification of virtual commissioning in automation industry," in *Proceedings of the 2016 Winter Simulation Conference*, ser. WSC '16. Piscataway, NJ, USA: IEEE Press, 2016, pp. 2430–2441. [Online]. Available: http://dl.acm.org/citation.cfm?id=3042094.3042396

[9] M. Silva and E. Teruel, "Petri nets for the design and operation of manufacturing systems," *European journal of control*, vol. 3, no. 3, pp. 182–199, 1997.

[10] ——, "A systems theory perspective of discrete event dynamic systems: The petri net paradigm," 08 1996.

[11] G. Cansever and I. B. Kucukdemiral, "A new approach to supervisor design with sequential control petri-net using minimization technique for discrete event system," *The International Journal of Advanced Manufacturing Technology*, vol. 29, no. 11-12, pp. 1267–1277, sep 2005.

[12] M. Stanton, W. Arnold, and A. Buck, "Modelling and control of manufacturing systems using petri nets," *IFAC Proceedings Volumes*, vol. 29, no. 1, pp. 4754–4759, jun 1996.

[13] M. Uzam and A. Jones, "Discrete event control system design using automation petri nets and their ladder diagram implementation," *International Journal of Advanced Manufacturing Technology*, vol. 14, pp. 716–728, 01 1998.

[14] S. Korotkin, G. Zaidner, B. Cohen, A. Ellenbogen, M. Arad, and Y. Cohen, "A petri net formal design methodology for discrete-event control of industrial automated systems," in *2010 IEEE 26-th Convention of Electrical and Electronics Engineers in Israel*, 2010, pp. 000 431–000 435.

[15] G. Frey, "Automatic implementation of petri net based control algorithms on plc," vol. 4, 02 2000, pp. 2819 – 2823 vol.4.

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

[17] R. David and H. L. Alla, *Du Grafcet aux réseaux de Petri*, 1992.

[18] N. J. Dingle, W. J. Knottenbelt, and T. Suto, "Pipe2: A tool for the performance evaluation of generalised stochastic petri nets," *SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 4, p. 34–39, Mar. 2009. [Online]. Available: https://doi.org/10.1145/1530873.1530881

[19] "03 processing grafcet — computer engineering — computer programming," https://es.scribd.com/document/140248558/03-Processing-Grafcet, 24-01-2020.

[20] Z. Liu, C. Diedrich, and N. Suchold, *Virtual Commissioning of Automated Systems*. INTECH Open Access Publisher, 2012. [Online]. Available: https://books.google.es/books?id=YeTmoAEACAAJ