

## **TEACHING SYSTEM MODELLING AND SIMULATION THROUGH PETRI NETS AND ARENA**

Jaume Figueras i Jové

Automatic Control Department  
Universitat Politècnica de Catalunya  
Rambla Sant Nebridi, 10  
Terrassa, 08222, SPAIN

Antoni Guasch i Petit  
Pau Fonseca i Casas  
Josep Casanovas-Garcia

inLab FIB  
Universitat Politècnica de Catalunya  
Edifici B5-S102 C/ Jordi Girona, 31  
Barcelona, 08025, SPAIN

### **ABSTRACT**

This paper describes our experience teaching discrete-event simulation to several Engineering branches and Computer Science students. In our courses we emphasize the importance of conceptual modelling rather than the simulation tools used to build a model. We think that in discrete-event simulation university courses it is more important to provide knowledge to students to develop and analyze conceptual models than focusing in a specific simulation tool that will be industry dependent. Focusing in conceptual modelling with the support of a well-known simulation software provide the student with skills to create a model and then translate it to any simulator. Our courses use the Petri Net (PN) methodology by incrementing the complexity of models and PN using: Place-Transition PN, Timed PN and Colored Timed PN. A PN simulator is used to analyze the conceptual model and different rules and procedures are provided to match PN conceptual model to Arena simulation software.

### **1 INTRODUCTION**

Conceptual Modelling (CM) is the process of abstracting a model from a real or proposed system after a conceptualization process in the mind. It is almost certainly the most important aspect of a simulation project (Robinson 2008). As education in CM has the prevalent idea of being more an art than a science and mostly dependent on teacher skills (Van der Zee, Kotiadis, Tako, et al. 2010) we propose a methodology based on Petri Nets to develop a mathematical conceptual model that can be analyzed and simulated. We provide a mapping technique to convert the conceptual model to a simulation model written in Arena.

Petri Nets (PN) are one of several mathematical modeling languages that can be used for the description of distributed systems, DEVS (Zeigler, Praehofer, & Kim, 2000) or SDL (ITU-T, 2012) can be others; see (Fonseca i Casas, 2013) for a description of some of the formal languages that can be used for discrete simulation. A Petri net is a directed bipartite graph, in which the nodes represent transitions (i.e. events that may occur, signified by bars) and places (i.e. conditions, signified by circles). A PN model of a system describes the states which the system may be in and the transitions between these states (Kristensen, Christensen, Jensen 1998). PN offer both a matrix and a graphical representation for stepwise processes that include choice, iteration, and concurrent execution (Petri 2008). Petri Nets have an exact mathematical definition of their execution semantics, with a well-developed mathematical theory for process analysis.

Petri Nets (PN) is a valuable formalism to model manufacturing systems (Silva, Teruel 1997) and have been used in a plethora of industrial problems such as: Construction of schedules (Sawhney 1997); Scheduling and control of semiconductor manufacturing systems (Zhou, DJeng 1998); Model Verification

and Validation (Samkari, Franz 2012); Manufacturing systems and deadlock prevention (Ezpeleta, Colom, and Martínez 1995) and (Li, and Zhou 2004); Supervision (Lee, Zhou and Hsu 2005); and many other.

The usage of PN for teaching discrete event simulation to future industry engineers meet the requirements of being a mathematical proven formalism and applicable to industrial problems making this formalism ideal for teaching CM. PN is also useful for us to introduce the concept of Verification and Validation prior to model coding using a specific simulator.

Students also have to be skillful with specific simulation software in order to be able to learn discrete event simulation results analysis. Arena simulation tool is being used to code CM into runnable simulation models. A mapping between the conceptual model and the simulation model has to be provided to show the student the importance of coding a simulation model from a validated CM instead of just teaching the use of certain software. One of the weaknesses of education in discrete event simulation is to teach the use of software instead of modelling. This leads to a complete misunderstanding of what is a conceptual model and the needed differentiation between the model and an implementation of this model.

## 2 STUDY CASES

To introduce the conceptual modeling we use two different models, first a classical M/M/1 to introduce the Place Transition Petri Net (PTPN) and Timed Petri Net (TPN) concepts and a second example, the manufacturing process proposed in (Carrie 1988), to introduce the Colored Timed Petri Net (CTPN). Both examples are useful to develop the key skill to conceptual modelling as they can explain the basic concepts in industrial processes, such as choice, concurrency, reachability, boundedness and deadlocks.

### 2.1 The vending machine example

The vending machine example is a simple system to teach the basics of PN and Discrete Event Systems (DES). In this example a client arrives at the vending machine to purchase products as an M/M/1 system.

### 2.2 The process sequence example

In the process sequence example, twelve different types of products arrive to the system where a worker has to place them to a loading/unloading station. Each type of product has its own station. An AGV moves the product from the loading/unloading station to one of the three identical machines; the product is then processed in the machine. Once the process is finished, the AGV takes the product back to the station where the worker removes it from the system. Figure 1 shows the conceptual scheme of the system provided to the students.

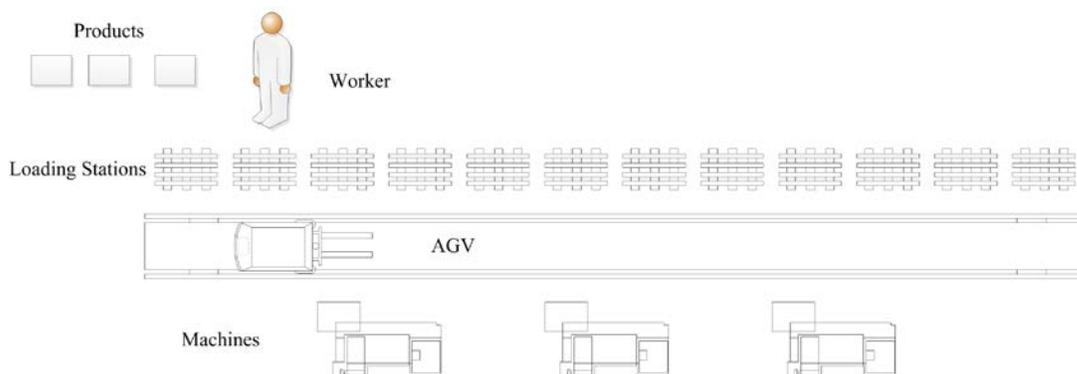


Figure 1: Scheme of the process sequence.

### 3 CONCEPTUAL MODELLING

Our first step in teaching CM is to show which roles take the different elements and structures of a PN into DES. A PN transition represent an event in DES, transitions can be unconditional and conditional, depending on the inbound arcs. PN places represent the state of the net they also determine the state of a DES model, besides they represent the state in DES they have a specific meaning: queues, activities and resources. PN tokens represent the entities of a DES model both temporal entities and permanent entities.

#### 3.1 Place Transition PN

The first model our students have to face is a simple M/M/1 model. It represents a vending machine where people go to purchase different products. In this example we present the basics of conceptual modelling and PN. Figure 2 shows the complete model PTPN for this system and Table 1 explain the meaning of each PTPN element. First the unconditional transition T1; an unconditional transition is always active and the firing of this transition creates a token to the connected place(s) in DES. This transition represent the arrival of a client to the vending machine. This transition creates a token that represents a client. This unconditional transition models the arrival of temporal entities to a DES model.

The transition is connected to the place P1; this place represents the queue of the vending machine where the clients wait until the vending machine is free to be used.

After queuing, the clients want to use the vending machine, so in PN we can represent this behavior through synchronization T2, where the waiting entity has to synchronize with the free resource P4. In DES we call this event the *seize* of a resource. As the temporal entity (the client) has seized the permanent entity (the vending machine resource) it can perform an activity P2, in this case the purchase of a product. This activity is represented in PN as a place where the client remain until the activity finishes, thus the client get its product.

When the activity finishes the client has the requested product and the vending machine recover its free state and is available for serving other clients. In DES this is the *release* of a resource event T3.

Finally we add the last element of the DES model. The modelling of this last place P3 can be discussed if it is necessary or not for modelling the M/M/1 system. We decided to model it to represent the exit of the clients.

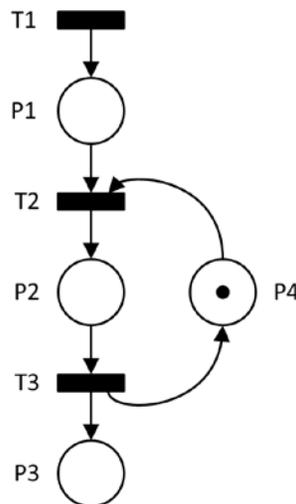


Figure 2: Complete PN of the M/M/1 example.

Table 1: Events, Activities, Queues and Resources of vending machine example.

Element	Meaning
T1	Arrival event, it creates the clients arriving to the vending machine
T2	Resource <i>seize</i> event, it synchronizes the client with the vending machine
T3	Resource <i>release</i> event, it is a concurrency in PN, so at the same time the resource recover its free state and the client leaves the system
P1	Queue
P2	Activity
P3	Exit
P4	Resource
[P1, P2, P3, P4]	represents the state of the system, allowing to know at any moment how many clients are waiting in the system, the state of the machine, if there is or not a client using the machine and how many clients have purchased a product in the vending machine
[0,0,0,1]	This is the initial state representing that no client has yet arrived to the vending machine and that the machine is free.

With this example we also use a PN simulator to introduce the active transition and the firing of a transition concepts, representing the possible events that can occur in the system and the event occurrence. We use the PIPE simulator; it is platform independent and can be used with any operating system.

With this example we also introduce the boundedness and reachability graph concepts. This particular PN is unbounded as the unconditional transition T1 can create infinite tokens. Limiting the capacity of the queue (P1) the reachability graph can be easily computed as shown in Figure 3.

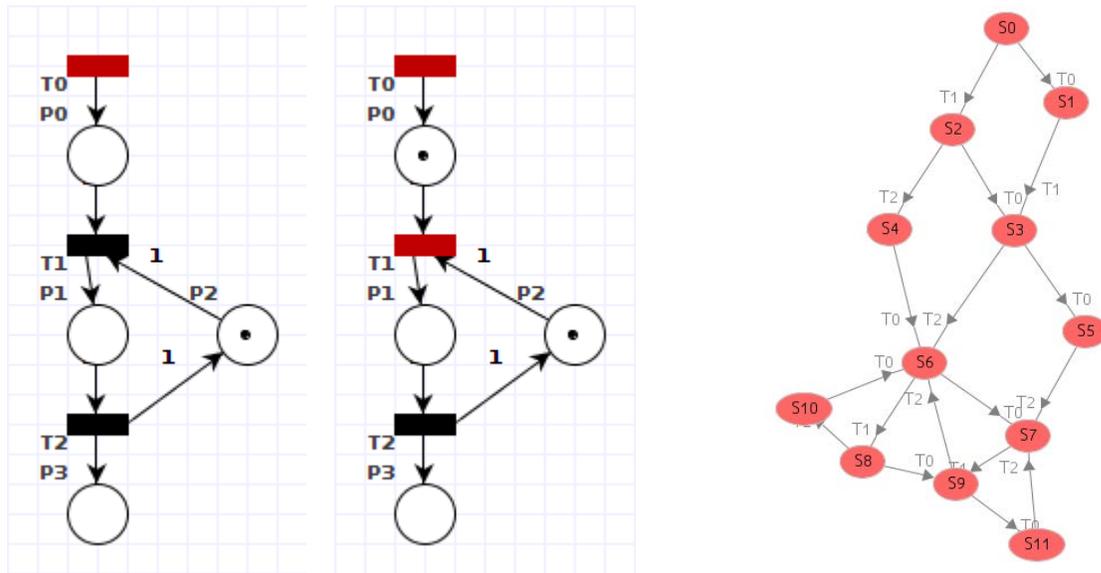


Figure 3: Different states of the M/M/1 PTPN simulation and its reachability graph.

### 3.2 Timed PN

Once the basic model is introduced we add the concept of time in the PN using the Timed Petri Nets formalism (TPN). In TPN there are two types of transitions the immediate transitions that are fired as the activation conditions are met and the timed transitions that are fired after a certain time while the activation conditions are met. In the examples the immediate transition is modelled with a black transition whilst the timed is modelled with a white transition. For the M/M/1 this means that transitions T1 and T3 are timed transitions, representing the time between arrivals for T1 and the processing time for T3.

Now the students are ready to work on the second example presented. This example is very useful because the students tend to create deadlocks when seizing or releasing the load-unload stations or the AGV. The description of the model is explained in Table 2 and Figure 4.

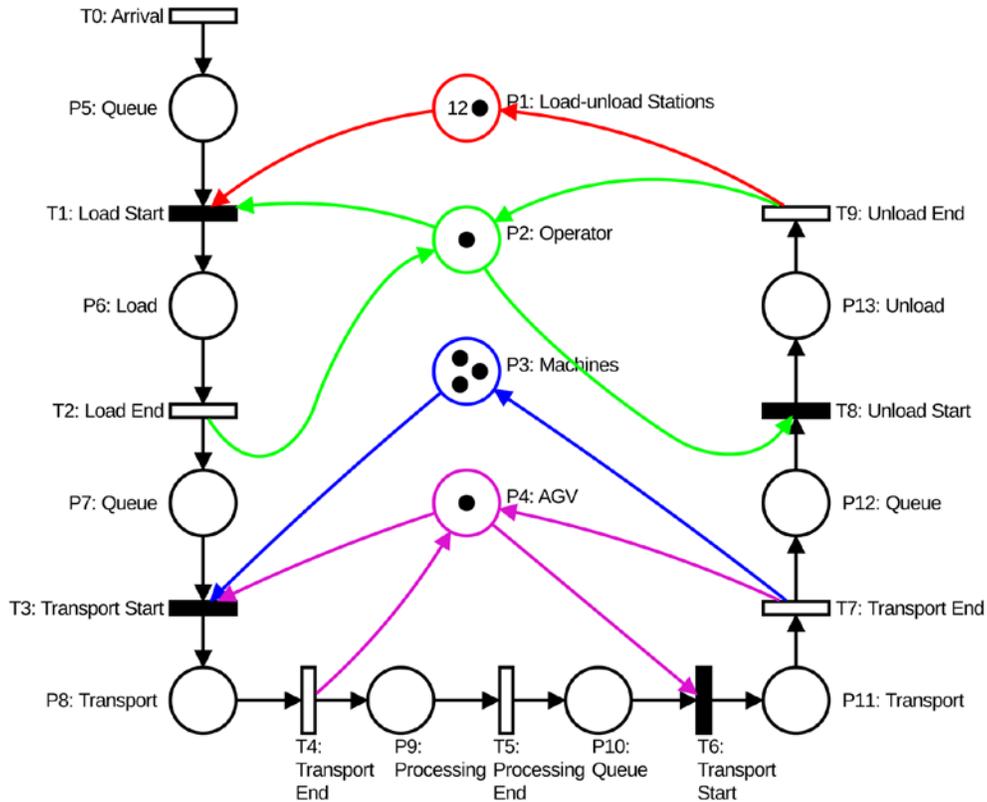


Figure 4: TPN of the process sequence example.

Table 2: Events, Activities, Queues and Resources of the process sequence example.

Events	
T0	Arrival of product to be processed, it is a timed transition with an arrival rate depending on desired production.
T1	Start of the load of an station done by the worker
T2	End of the station load process, it is a timed transition parameterized with the loading time of a station.
T3	Start of the transport of a product to the machine.
T4	End of the transport to a machine, it is a timed transition parameterized with the transport time of the AGV. It also represents the start of the machine processing as the machine can begin the processing immediately.
T5	End of the machine process. It is a timed transition parameterized by the machining time depending on the product type.
T6	Start of the transport back to a station.
T7	End of the transport back to the station, it is a timed transition parameterized with the transport time of the AGV.
T8	Start of the unload of a station done by the worker.
T9	End of the station load process, it is a timed transition parameterized with the unloading time of a station.



Table 3: Colors, Guards and Arc Expressions of the process sequence example.

Colors	
pr	Product type {1..12}
st	Station type {1..12}
op	Worker
m	Machine
Guards (extract)	
T1	[pr = st] The product type must be equal to the station type, matching the product to a certain station
Arc expressions (extract)	
T0-P5	$1'(pr)$ The arriving product to be processes is assigned with a product type
T9-P1	$1'(st=pr)$ The station to be released is equal to the product type (this equivalence is assured by guard T1)

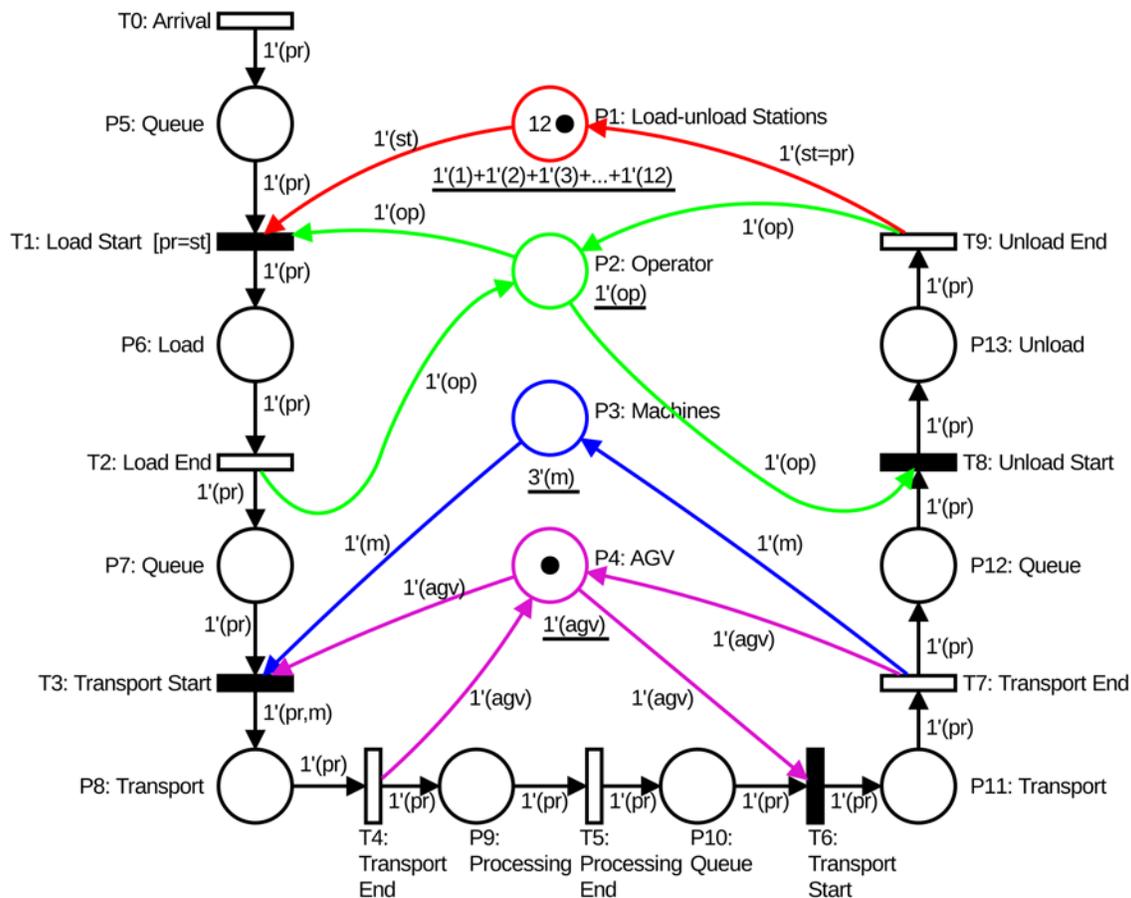


Figure 6: CTPN model of the process sequence example.

### 3.3 Hierarchical PNs

Although there is no time to study the hierarchical extensions of PNs (Huber, Jensen and Shapiro 1990) or (Fehling 1993) this concept is introduced to show the capabilities of PNs to reduce the complexity of a system. Hierarchical elements can be modelled and understood by all levels of any organization. The hierarchies can be modelled as submodels in any DES simulator and have many parallelisms in computer science and engineering, such as hierarchical networks, software development or electronics.

## 4 MODEL TO SIMULATOR MAPPING

Finally, after the conceptual modelling of DES using PN the students move to lab classes to create simulation models with Arena. We focus on the mapping between the CM and Arena. This section explains the development of the simulation model of the process sequence example represented in Figure 4 and Figure 6.

### 4.1 Timed PN

The timed T0 transition is an arrival so the natural arena block is a Create Block. The parameters of the block are taken from the PN. The interarrival time is the time associated with the transition and the entities per arrival parameter is the weight of the arc to P5. Figure 7 shows the Create Block.

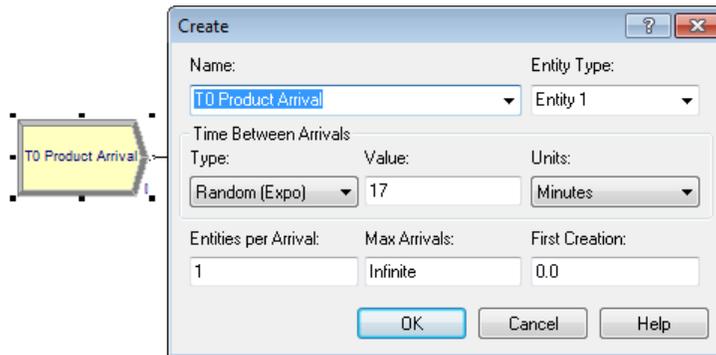


Figure 7: The Create Block.

The place P5 and the transition T1 are modelled with a Queue and a Seize block. The resources to seize are determined by the topology of the PN, following the arcs arriving to T1 the student knows which resources have to be seized. The quantity of resources to seize is determined by the arc weights. Queues can be omitted since Arena creates automatically a queue for each seize bloc. But for a better understanding we use the two blocs. Later, when the students have a better knowledge the queue step is not used anymore. Figure 8 shows the queue and Seize Block.

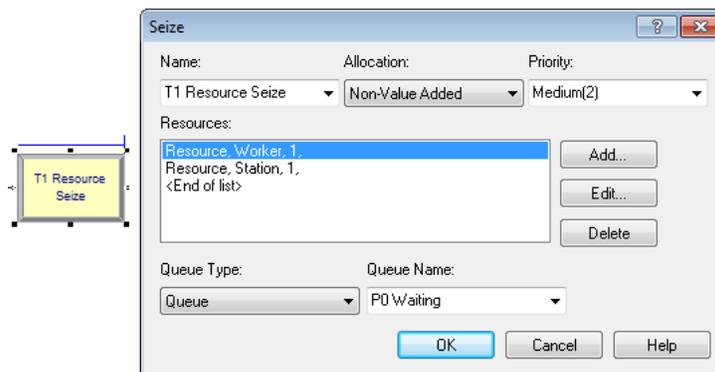


Figure 8: Seize Block.

The place P6 is modelled with a Delay block. The product stay at P6 while the time set in the T2 has not passed. So the block is parameterized using the time on the timed transition T2. Figure 9 shows the Delay Block.

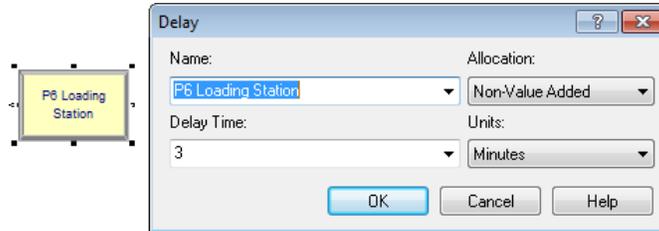


Figure 9: Delay Block.

The T2 transition is the release of the used resources, using a Release Block in Arena we can model this behavior, The quantity of resources to release is determined by the weight of the connected arcs. Which resources have to be released is determined by the connected places of the transition. Figure 10 shows the Release Block.

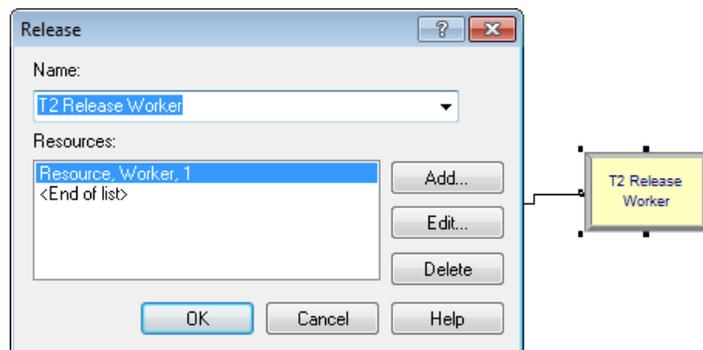


Figure 10: Release Block.

The following blocks can be easily deduced from the PN of the system. All of them are a mix of the Queue, Seize, Delay and Release blocks. At the end of the model a Dispose block is needed (it is omitted here due to its simplicity).

#### 4.2 Colored Timed PN

The colors of a CTPN are attributes of the entities in Arena. Following this assumption, some changes have to be introduced to model the system using CTPN. As shown in Table 3, there is an arc expression from T0 to P5 assigning a type to the product, the block used in Arena to model this assignment is an Assign block (Figure 11). The type assignment can be modelled using mathematical expressions (as in PN), for this example a discrete probability function is used.

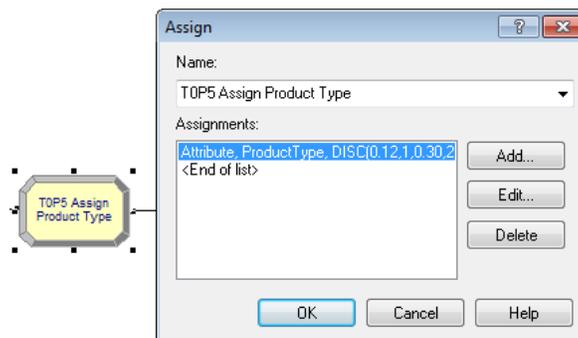


Figure 11: The Assign Block.

In Figure 6, the Stations resource is defined as twelve different stations. In Arena this is done the same way, defining twelve different resources. To model them as a unique place the Set block is the natural Arena block capable of grouping them in one place. Figure 12 shows the Set parameterization.

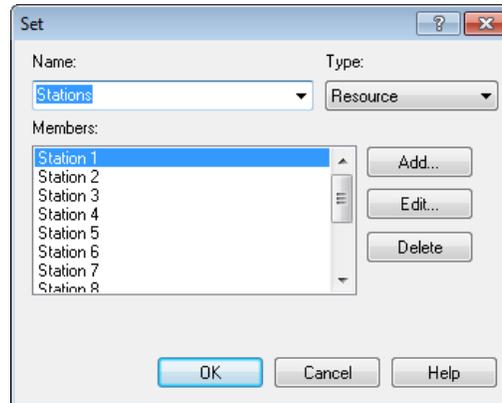


Figure 12: The Set Block.

Finally the seize has to be modified to meet the T1 guard, it forces the resource to be of the same type as the product. Figure 13 shows how to parameterize the seize block to seize a resource of the type of product.

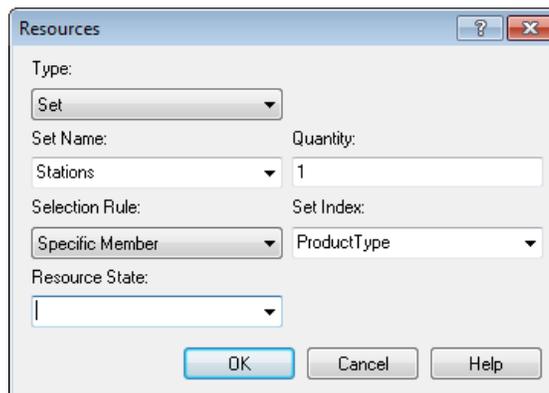


Figure 13: Seize block modifications.

## 5 CONCLUSIONS

To teach the basics of conceptual modelling and Petri Nets and the lab to practice the modelling with arena using the approach presented in this paper is about 16 hours. Depending if they are more from the industrial engineering branch or the computer science branch is more complex or easy to explain these concepts. In industrial engineering we need more time to explain Arena simulation software. Computer science student model more easily with Arena but we need more time to explain the conceptual modelling. At the end of these sessions the students understand clearly the difference between the system, the model and the implementation of this model. Also they understand that the formal representation of the model simplify the Validation and Verification process, thanks to the analysis of concurrency, reachability, boundedness and deadlocks done with PN.

## ACKNOWLEDGMENTS

This work has been funded by the Ministerio de Ciencia e Innovación, Spain. Project reference TIN2011-29494-C03-03.

## REFERENCES

- Ezpleta, J., J.M., Colom and J., Martínez, 1995. "A Petri-net Based Deadlock Prevention Policy for Flexible Manufacturing Systems". *IEEE Transactions on Robotics and Automation* Vol. 11.2:173-184.
- Fehling, R., 1993. "A Concept of Hierarchical Petri Nets with Building Blocks". In *Advances in Petri Nets 1993*, edited by G. Rozenberg, 148-168. Springer Berlin Heidelberg.
- Fonseca, P. ed. 2014. *Formal Languages for Computer Simulation: Transdisciplinary Models and Applications*. Hershey PA: IGI Global.
- Huber, P., K. Jensen, and R. M. Shapiro. 1990. "Hierarchies in coloured Petri nets". *Advances in Petri Nets 1990*, edited by G. Rozenberg, 313-341. Springer Berlin Heidelberg.
- ITU-T. "Specification and Description Language (SDL)". Accessed July 16, 2014. <https://www.itu.int/rec/T-REC-Z.100/en>
- Kristensen, L.M., S. Christensen, and K. Jensen. 1998. "The Practitioner's Guide to Coloured Petri Nets". *International Journal on Software Tools for Technology Transfer* 2(2): 98-132.
- Lee, JS, M.C. Zhou, and P.L. Hsu. 2005. "An application of Petri nets to supervisory control for human-computer interactive systems". *IEEE Transactions on Industrial Electronics* 52(5): 1220-1226.
- Li, ZW, and M.C. Zhou. 2004. "Elementary Siphons of Petri Nets and their Application to Deadlock Prevention in Flexible Manufacturing Systems". *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 34(1): 38-51.
- Petri, C. A., R. Valk, and K. M. van Hee. 2008. "On the Physical Basics of Information Flow". *Applications and theory of Petri Nets*, edited by K. M. van Hee, and R. Valk, 12 Springer Berlin Heidelberg.
- Robinson, S. 2008. "Conceptual Modelling for Simulation Part I: Definition and Requirements". *Journal of the Operational Research Society*, Vol. 59 Issue 3 (2008): 278-290.
- Samkari, K., and V. Franz. 2012. "A Petri Net Based Method for the Early Verification & Validation of a Simulation Study in Construction Management". *Proceedings of the 2012 Winter Simulation Conference*, edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Sawhney, A. 1997. "Petri Net Based Simulation of Construction Schedules". *Proceedings of the 1997 Winter Simulation Conference*, edited by S. Andradbttir, K. J. Healy, D. H. Withers, and B. L. Nelson, 1111-1118. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Silva, M., and E. Teruel. 1997. "Petri Nets for the Design and Operation of Manufacturing Systems". *European journal of control* 3(3): 182-199.
- Van der Zee, D., K. Kotiadis, A. Tako, M. Pidd, B. Osman, A. Tolk, and M. Elder. 2010. "Panel Discussion: Education on Conceptual Modeling for Simulation - Challenging the Art". *Proceedings of the 2010 Winter Simulation Conference*, edited by B. Johanson, S. Jain, J. Montoya-Torres, 290-304. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Zeigler, B. P., H. Praehofer, and T. G. Kim. 2000. "Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems". Academic Press.
- Zhou, M. 1998. "Modeling, analysis, simulation, scheduling, and control of semiconductor manufacturing systems: A Petri net approach" *IEEE Transactions on Semiconductor Manufacturing* 11(3): 333-357.

## **AUTHOR BIOGRAPHIES**

**JAUME FIGUERAS** born in 1974 had his degree in Computer Science in 1998. His research is in Automatic Control and Computer Simulation and Optimization. He has designed and developed CORAL, an optimal control system for sewer networks, applied at Barcelona (Spain); PLIO, an optimal control system and planner for drinking water production and distribution, applied at Santiago de Chile (Chile) and Murcia (Spain). Nowadays He participates in different industrial projects, like the power consumption optimization of tramway lines in Barcelona with TRAM and SIEMENS and the development of tooPath (<http://www.toopath.com>) a free web tracking system of mobile devices. He is also the local representative of OSM (<http://www.openstreetmap.org>) in Catalonia and participates in different FOSS projects. His email address is [jaume.figueras@upc.edu](mailto:jaume.figueras@upc.edu).

**ANTONI GUASCH** is a research engineer focusing on modelling, simulation and optimization of dynamic systems. He received his Ph.D. from the UPC in 1987. He is an Associate Professor in the department of "Ingeniería de Sistemas, Automática e Informática Industrial" in the UPC and head of Simulation and Industrial Optimization at inLab FIB (<http://inlab.fib.upc.edu/>). Since 1990, Prof Guasch has lead more than 40 industrial and research projects related with modelling, simulation and optimization of nuclear, textile, transportation, car manufacturing, water, pharmaceutical and steel industrial processes. His email address is [toni.guasch@upc.edu](mailto:toni.guasch@upc.edu).

**PAU FONSECA I CASAS** is a Professor of the Department of Statistics and Operational research of the Technical University of Catalonia, teaching in Statistics and Simulation areas. He obtained his Ph.D. in Computer Science on 2007 from Technical University of Catalonia. He also works in the InLab FIB (<http://inlab.fib.upc.edu/>) as a head of the Environmental Simulation area, developing Simulation projects since 1998. His research interests are discrete simulation applied to industrial, environmental and social models, and the formal representation of such models. His email address is [pau@fib.upc.edu](mailto:pau@fib.upc.edu).

**JOSEP CASANOVAS** Professor Josep Casanovas (Ph.D. in Computer Science, Industrial Engineer, MSc in Economics) is the head of inLab FIB, formerly called LCFIB, at Barcelona School of Informatics. He is a full professor of the Statistics and Operations Research Department at UPC. His main research areas are Modeling and Simulation, Internet and Information Systems. He is the author of numerous research articles and has collaborated in the development of many projects for the European Union and other companies and institutions. Between 1998 and 2004 he was dean of the Barcelona School of Informatics. In addition, Prof. Casanovas has been vice-rector of university policies of the Technical University of Catalonia (2006-2011). He was also responsible for ICT policies at UPC. Currently, Josep Casanovas is co-director of LogiSim (Centre of Simulation and Optimization of Logistic Systems) and coordinator of the Severo Ochoa Research Excellence Program in the Barcelona Supercomputing Center (BSC-CNS). His email address is [josepk@fib.upc.edu](mailto:josepk@fib.upc.edu).