Chapter 5
Simulation Examples

Antoni Guasch and Jaume Figueras

5.1 Introduction

Chapter 1 introduced the phases of a simulation project, emphasizing the importance of each one of them. In consequence, it would be a serious mistake to think that a simulation project is simply coding the model by using simulation software and then extracting the results.

Chapter 2 reviewed the important statistical aspects of simulation, and then Chap. 3 introduced Petri nets as a method for conceptual modeling of the process involved. Many simulation reference books put particular emphasis on the statistical aspects, for both the parameterization of the model and the validation, experimentation and analysis of the results. This book highlights the importance of conceptual modeling as a step prior to the construction of a good Simulation Model. This chapter gives a set of examples initially modeled using Petri nets and later coded with Simio.

One of the most complex phases of a simulation project is the validation of the model, in other words, proving that the model behaves like the real process involved. The best way to approximate the validation step is to suppose that the model is incorrect and do everything necessary to try to demonstrate this. If we cannot demonstrate that it is wrong, we can assume that the model is valid, although the possibility always exists that it is not. One of the mechanisms that can be used to try to validate the model is to contrast the results of a model with the results of another model constructed using different techniques and, if possible, using different work equipment. If the results coincide, the conviction that the simulation model is valid is reinforced in the conclusions.

Whatever the validation method used, we recommend doing a theoretical analysis of the process prior to simulation. In the simple examples given in this chapter, the results of the theoretical calculations would be expected to be very similar to those obtained by simulation. The differences between the results of the two models should be justified based on the different hypotheses used in the

I.F. De La Mota et al., *Robust Modelling and Simulation*,
DOI 10.1007/978-3-319-53321-6_5

construction of each model. If there is no justification, the differences could be a sign of an invalid simulation model or of wrong theoretical calculations.

5.2 Canal-Lock System

Ten barges are used to transport materials in a canal. The barges are loaded by one of the two cranes located on the low part of the canal and are unloaded by one of the two cranes to be found at the end of the high part of the canal. Therefore, two barges can be loaded or unloaded simultaneously, one at each end of the canal. The canal is divided into two sections, the low section and the high one, which are separated by a lock that can only raise or lower one barge at a time. These ten barges only work on the canal and never leave it, using the canal as if it were a closed circuit. At the start, it is assumed that the 10 barges are in the low part, queued for loading. Figure 5.1 shows how the lock works.

Table 5.1 shows the triangular distribution of the times associated with the different operations.

It is a good idea to do a theoretical analysis of the process before the simulation study.

In order to do the theoretical analysis, we will employ basic queue theory concepts. The parameters of a queuing process are:

λ mean frequency of arrivals: With a known (expected) mean time between arrivals $E(A)$, the mean frequency of arrivals is calculated as $\lambda = 1/E(A)$.

ω mean frequency of service: With a known (expected) mean time of service $E(S)$, the mean frequency of service is calculated as $\omega = 1/E(S)$.

s number of servers.

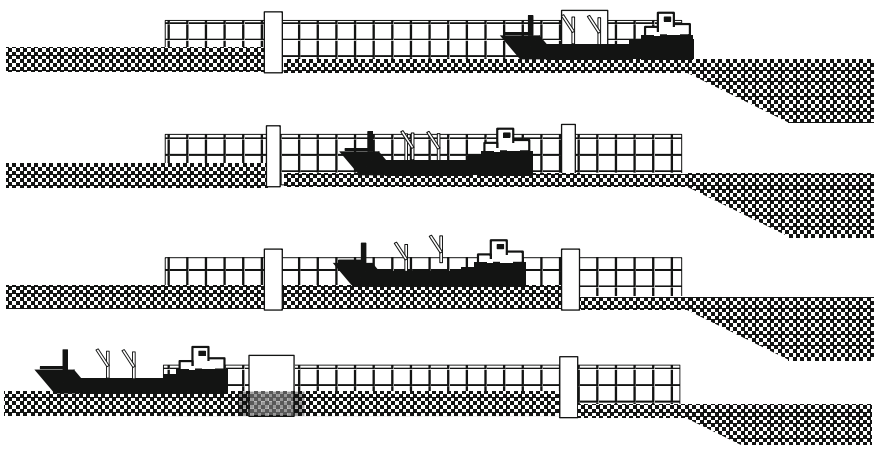


Fig. 5.1 Operational schema of how a lock works

Table 5.1 Process time (triangularly distributed and expressed in hours)

	Minimum		Mode	Maximum
Loading time	2.00		2.25	3.00
Unloading time	1.50		2.25	2.50
Lock time	0.40		0.50	0.6
Path of the low section				
	Raised	2.00	2.50	3.00
	Lowered	1.50	1.75	2.00
Path of the high section				
	Raised	1.00	1.25	1.50
	Lowered	0.50	0.75	1.00

The server utilization factor can be defined based on the aforementioned parameters:

$$\text{Server utilization factor}: \ \rho = \frac{\lambda}{\omega \cdot s} = \frac{E(S)}{E(A) \cdot s}$$

In this example we want to calculate the average utilization factor of the loading and unloading cranes as well as of the lock. For this calculation, we use the mean operating times of the above triangular distributions. The mean is calculated as (minimum + mode + maximum)/3. We do not know the mean time expected between the arrivals of the barges $E(A)$, but given that the 10 barges work in a closed circuit, we can hypothesize that $E(A)$ is equal to the time of one complete cycle of the barge divided by the number of barges

$$E(A) = \frac{11.75}{10} = 1.175$$

This makes it possible to calculate the utilization of the loading cranes:

$$\rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{2.416}{1.175 \cdot 2} = 1.02$$

The value of 1.02 indicates that with this hypothesis, the loading cranes will work at 100% and, therefore, they will limit the barges' arrival rate to the remaining resources. Thus, it seems more correct to suppose that the arrivals rate is the mean time of service $E(s)$ divided by the number of loading cranes

$$E(A) = \frac{E(s)}{s} = \frac{2.416}{2} = 1.208$$

And the utilization of the resources is:

$$\text{Loading cranes} : \rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{2.416}{1.208 \cdot 2} = 1$$

$$\text{Unloading cranes} : \rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{2.083}{1.208 \cdot 2} = 0.86$$

$$\text{Lock} : \rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{0.5 \cdot 2}{1.208 \cdot 1} = 0.83$$

Note that the lock works twice for every cycle of the barge.

Once the theoretical analysis has been done, we can continue with the simulation study. Before coding in Simio, we think it advisable to obtain the conceptual model of the process in Petri nets, given in Fig. 5.2.

If we look at the Petri net shown below, we note that the barges are not treated like resources. The barges are the temporary entities that flow through the process. The developed model is aimed at the process where the rising passage and falling passage through the lock are differentiated.

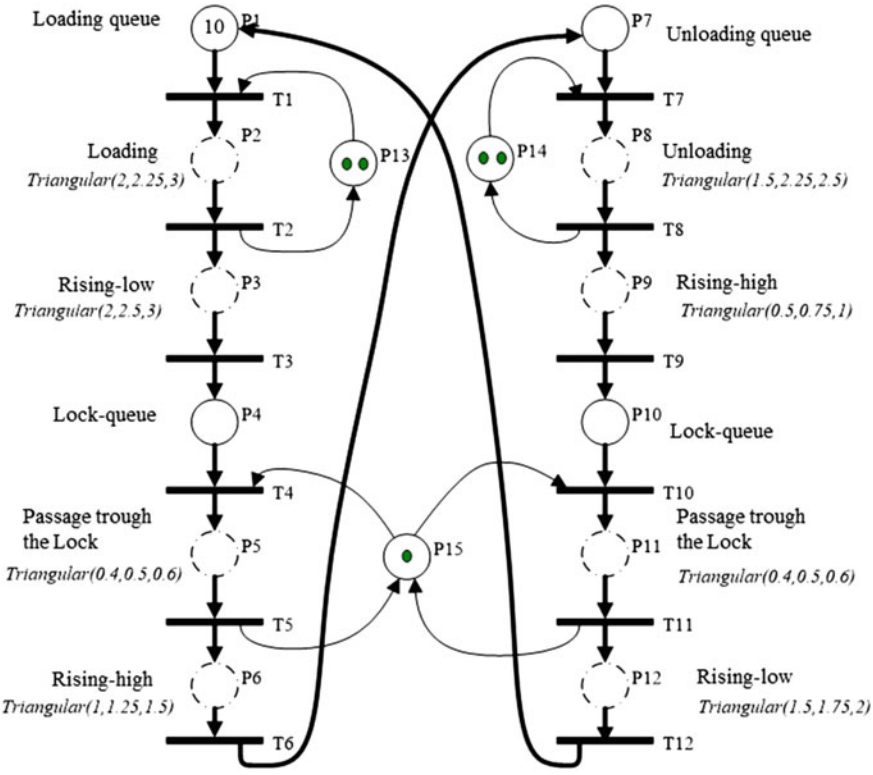


Fig. 5.2 Petri net model for the canal-lock system

With the places being:

- P1: queue of barges awaiting loading
- P2: loading activity
- P3: rising activity of the low section
- P4: queue in order to access the lock in the lowering direction
- P5: activity of passage through the lock in the rising direction
- P6: rising activity of the high section
- P7: queue of barges awaiting unloading
- P8: unloading activity
- P9: lowering activity of the high section
- P10: queue in order to access the lock in the lowering direction
- P11: activity of passage through the lock in the lowering direction
- P12: lowering activity of the low section

And the transitions:

- T1: start of the loading activity
- T2: end of the loading activity and start of the rising of the low section
- T3: end of the rising of the low section in order to queue up to enter the lock
- T4: start of the activity of passage through the lock in the rising direction
- T5: end of the activity of passage through the lock and start of the rising of the high section
- T6: end of the rising of the high section and start of the waiting for unloading
- T7: start of the unloading activity
- T8: end of the unloading activity and start of the rising of the low section
- T9: end of the rising of the low section in order to queue up to enter the lock
- T10: start of the activity of passage through the lock in the rising direction
- T11: end of the activity of passage through the lock and start of the rising of the high section
- T12: end of the rising of the high section and start of the waiting for unloading

Some interesting aspects that have, in a way, already been explained in the previous chapter but that are worth remembering, are:

- The thick lines represent the explicit flow of the temporary entities in Simio.
- Places with an interrupted line represent states whose length of time depends on explicit time functions.
- Places with a continuous line represent waiting states whose duration does not depend directly on the time functions.

Figure 5.3 shows the associated Simio code. The equivalent elements in the Petri net are given for each one of the main objects. The remarks are:

- The object Initial Barges of Source class creates only 10 temporary entities in the initial instant. This operation corresponds to the initialization of the P1 place.
- The movement times in the different sections are coded with objects of the TimePath class.

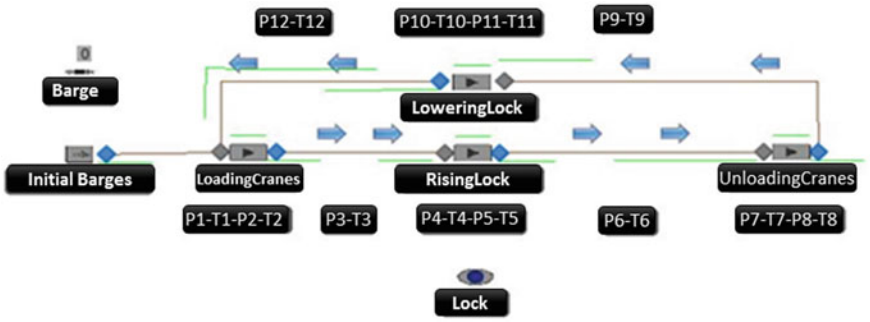


Fig. 5.3 Simio code for the canal-lock system

- The LoadingCranes, UnloadingCranes, LoweringLock and RisingLock objects belong to the Server class. The LoweringLock and RisingLock objects compete for the same Lock resource. This is clearly observed in the Petri net, but is implicit in the coding in Simio.

Ten replications were done with a warm-up time of 20 h for the purpose of having statistics in a permanent regime that make it possible to compare them with the expected theoretical results. The first result that surprises us is the observation that the average utilization of the loading cranes is 0.956, instead of the expected value of 1, because these cranes are a bottleneck in the process. The deviation is because the real cycle time is 11.75 h plus 0.89 h waiting in the queue. If we take into account that the real cycle time is 12.64, the utilization of the loading cranes would have to be:

$$\rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{2.416}{1.264 \cdot 2} = 0.956$$

I.e., the real cycle time is the factor that determines E(A). Table 5.2 shows the theoretical and real utilization for each one of the resources:

There is no point in increasing the number of barges, given that the system is almost saturated. The utilization factor of the loading cranes is at 95%. If we work with one more barge in the model, we see that the number of transports only rises from 70.6 to 73.6 transports in the complete period of 100 h.

Table 5.2 Utilization of resources

Resource	Theoretical utilization	Real utilization
Loading cranes	1	0.95
Unloading cranes	0.86	0.82
Lock	0.83	0.79

5.3 Two-Robot and 5-Machine Process

Two different types of pieces arrive at the subprocess of a manufacturing plant. These pieces, identified as type P1 and type P2 pieces, follow two different machining processes. See Fig. 5.4. The P1 pieces are processed by being machined by an M1 machine and afterward by an M2 machine; the P2 pieces are only processed in an M2 machine. These pieces arrive at an 8-setting entry buffer and are loaded into the corresponding machine by two robots. The R1 robot loads the M1 machines and the R2 robot loads the M2 machines. The piece (P1 or P2 type) that has been the longest waiting in the buffer is always processed. So, if a P1 type piece arrives, the subprocess follows the sequence given below:

1. If there is space in the buffer, it enters it.
2. The R1 robot takes the piece from the buffer and loads it into an M1 machine.
3. The piece is machined in the M1 machine.
4. The R1 robot leaves the piece in the buffer.

For the P2 type pieces or the P1 type ones already processed in M1, the processing sequence is simpler:

1. The R2 robot takes the P2 type piece or processed P1 type and loads it into an M2 machine.
2. The piece is machined in the M2 machine.
3. The R2 robot leaves the piece at the exit of the subsystem.

The work times are:

– The time between arrivals of P1 type pieces is exponential, with a mean of 10 min
– The time between arrivals of P2 type pieces is exponential, with a mean of 15 min

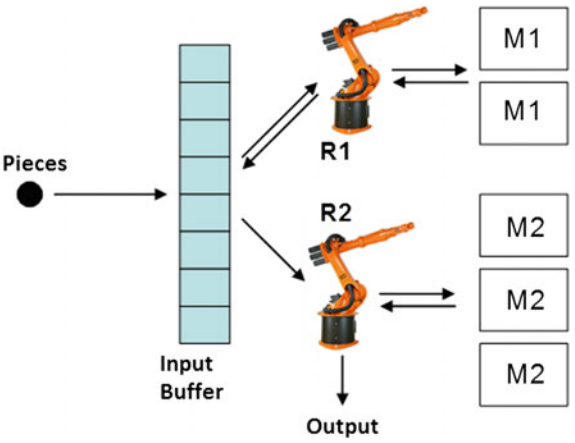


Fig. 5.4 2-robot and 5-machine system

– The total transport time of the robots is 10 s
– The process time of the M1 machine is 18 min
– The process time of the M2 machine is 16 min

And we have two M1 machines and three M2 machines.
The utilization of the resources is

$$\text{Robot1} : \rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{(10/60) \cdot 2}{10 \cdot 1} = 0.034$$

$$\text{Machines1} : \rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{18}{10 \cdot 2} = 0.9$$

$$\text{Robot2} : \rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{(10/60) \cdot 2}{6 \cdot 1} = 0.056$$

$$\text{Machines2} : \rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{16}{6 \cdot 3} = 0.89$$

The expected time between arrivals at the R2 robot and the M2 machine is

$$E(A) = \frac{10 \cdot 15}{10 + 15} = 6$$

Prior to the Petri net deduction process, it is a good idea to identify:

- The resources (or permanent entities) that are involved in the process: robot R1 (1), robot R2 (1), machine M1 (2), machine M2 (3), spaces in the buffer (8).
- The temporary entities: piece P1, piece P2.
- The activities: transport to M1 machine, process in M1 machine, return to buffer from M1 machine, transport to M2 machine, process in M2 machine and transport from exit of M2 machine in order to leave the subsystem.
- The transitions: arrival of piece P1, arrival of piece P2, entry into buffer, start of transport toward machine M1, …

Figure 5.5 shows the Petri net for the system. The places are:

– P1: waiting for space in the buffer for the P1 type pieces
– P2: P1 type pieces in the buffer waiting to go to the M1
– P3: P1 type pieces transported by the R1 to the M1
– P4: P1 type piece being processed in M1 machine
– P5: M1 machine blocked while waiting for the R1.
– P6: transport of the P1 piece to the buffer
– P7: waiting for space in the buffer for the P2 type pieces
– P8: P1 or P2 type pieces in the buffer waiting to go to the M2
– P9: P1 or P2 type pieces transported by the R2 to the M2
– P10: P1 or P2 type piece being processed in M2 machine

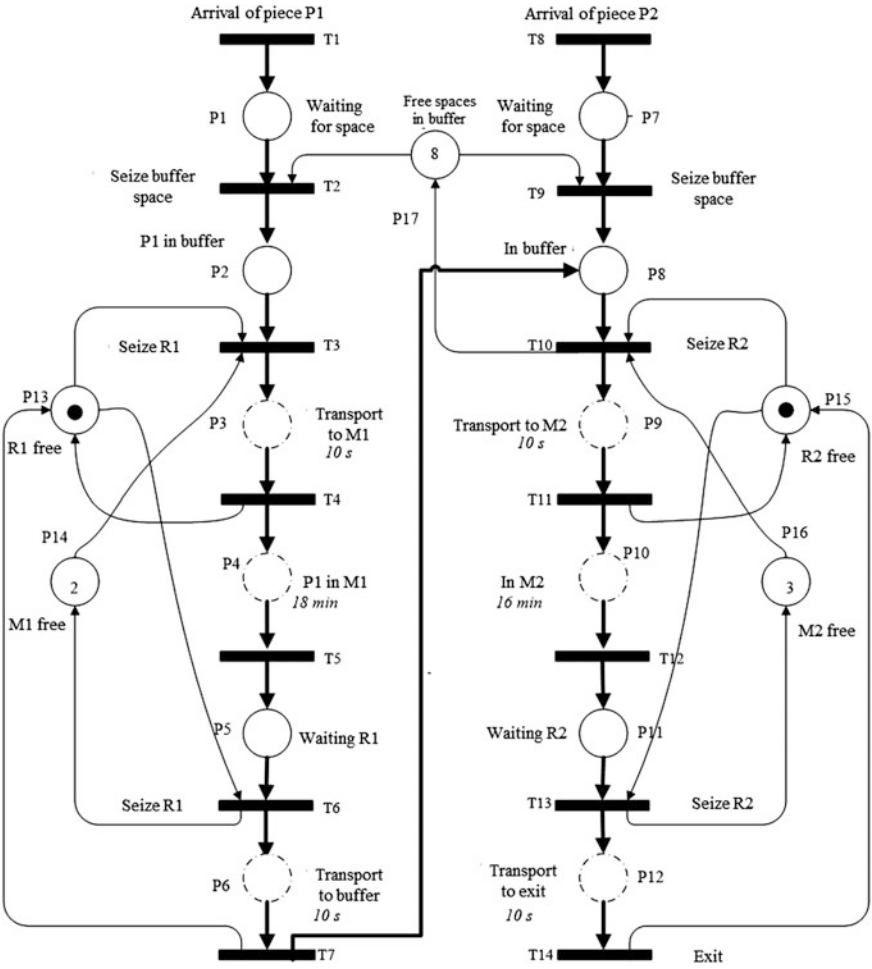


Fig. 5.5 Petri net model for the 2-robot and 5-machine system

- P11: M2 machine blocked while waiting for the R2.
- P12: transport of the P1 or P2 piece to the exit
- P13: R1 free
- P14: M1 free
- P15: R2 free
- P16: M2 free
- P17: Free spaces in the buffer

And the transitions:

- T1: arrival of P1 pieces
- T2: seize of buffer space
- T3: seize of the R1 and M1, before transport begins
- T4: end of the transport, release of the R1 robot and start of the process in the M1
- T5: end of process in the M1
- T6: seize of the E1 in order to return to the buffer and release of the M1
- T7: end of transport with the R1
- T8: arrival of P2 piece
- T9: seize of buffer space
- T10: seize of the R2 and M2 before transport begins
- T11: end of the transport, release of the R2 robot and start of the process in the M2
- T12: end of process in the M2
- T13: seize of the R2 in order to transport to exit and release of the M2
- T14: end of transport with the R2 and exit of the piece

The most important actions in a simulation model are related to the seizing (seize), utilization (delay) and use of resources (release or freeing in a context of scarce resources. The Petri net has the advantage of showing this set of actions explicitly and graphically. In relation to the seizing of resources, we stress that the space is not freed in the buffer when R1 takes the P1 piece; thus, we can guarantee space in the buffer for returning the piece once the process in the M1 is completed. Early release in transition T3 can cause the model notice blocked.

Figure 5.6 shows the Simio code associated with the former net. Transition T3 is triggered if we have a free R1 robot and a free M1 machine synchronously. The coding in Simio of the transitions where more than one resource is seized is a very delicate point, given that in its current version Simio does not in general guarantee the synchronous seizing of resources. This can cause undesirable effects and blockages that are not attributable to the model.

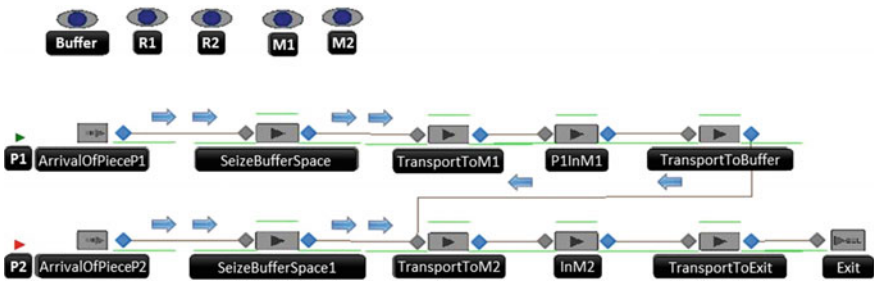


Fig. 5.6 Simio code for the 2-robot and 5-machine system

Figure 5.7 shows that in order to execute the transport it is necessary to seize the M1 machine "On entering" and in "Resources for Processing" robot R1. This coding guarantees that machine M1 is seized first, and robot R1 is seized second. If instead of seizing machine M1 "On entering", it is seized "Before Processing", the simulator may, and in fact does, become blocked if it first seizes robot R1 while the two machines M1 are working on other pieces. In order to release one of the two M1 machines, robot R1 has to be free, but robot R1 is seized by the piece that has to be transported to M1.

To conclude, this example stand out because the results of the simulation are very similar to the theoretical results we anticipated for the utilization of the resources.

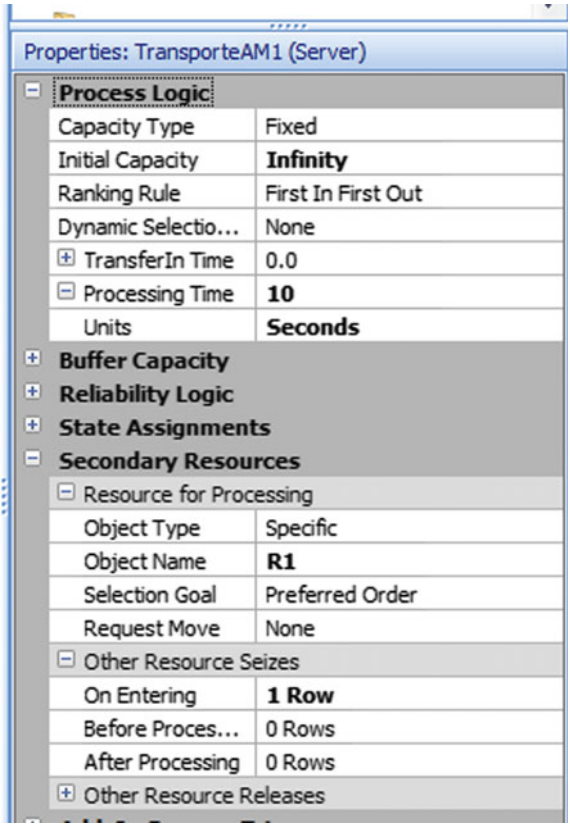


Fig. 5.7 Parameterization of the transporteam1 object of the server class

5.4 The Philosophers' Dinner

The problem of the philosophers' dinner is an illustration of a common problem in concurrent computing and a classic problem of multiprocess synchronization. Dijkstra (1971) established an exam question in a synchronization problem where five computers compete for access to five shared peripheral controller tapes. Very soon after this, Tony Hoare renamed this problem the problem of the philosophers' dinner.[1]

Five Chinese philosophers (f1, f2,…, f5) are seated at a round table. In the center of the table there is a rice bowl. Between each pair of philosophers there is one chopstick. Each philosopher alternates between meditating and eating. In order to eat, the philosopher needs two chopsticks, and he is only allowed to use the two close to him (to his left and right). Sharing the chopsticks in this way stops two neighbors from eating at the same time. This is illustrated graphically in Fig. 5.8.

This problem is often used to illustrate several problems that occur when multiple resources are competing for limited resources. The lack of available forks is an analogy to the problem of seizing shared resources in real programming in a computer, a situation known as concurrence.

Seizing a resource is a common technique for ensuring that the resource is accessible only by one program or piece of code at a time. When several programs are involved in resources seized, there may be mutual blocking depending on the circumstances. One way to prevent this blocking in the case of the dinner consists of insisting that both chopsticks (the one on the left and the one on the right) must be available simultaneously for one philosopher. Figure 5.9 shows a possible model using a Petri net.

The state of each philosopher can be represented by three places (Mi, Ei, Ci) that represent the states of meditation, and waiting to seize the chopsticks and eating respectively. The Pi places represent the available chopsticks. In order to be able to

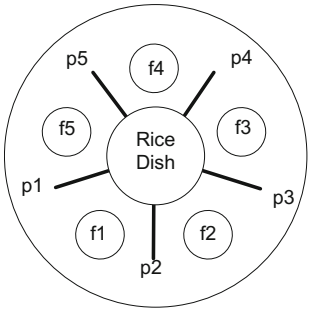


Fig. 5.8 The philosophers' table

[1]See http://en.wikipedia.org/wiki/Dining_philosophers_problem.

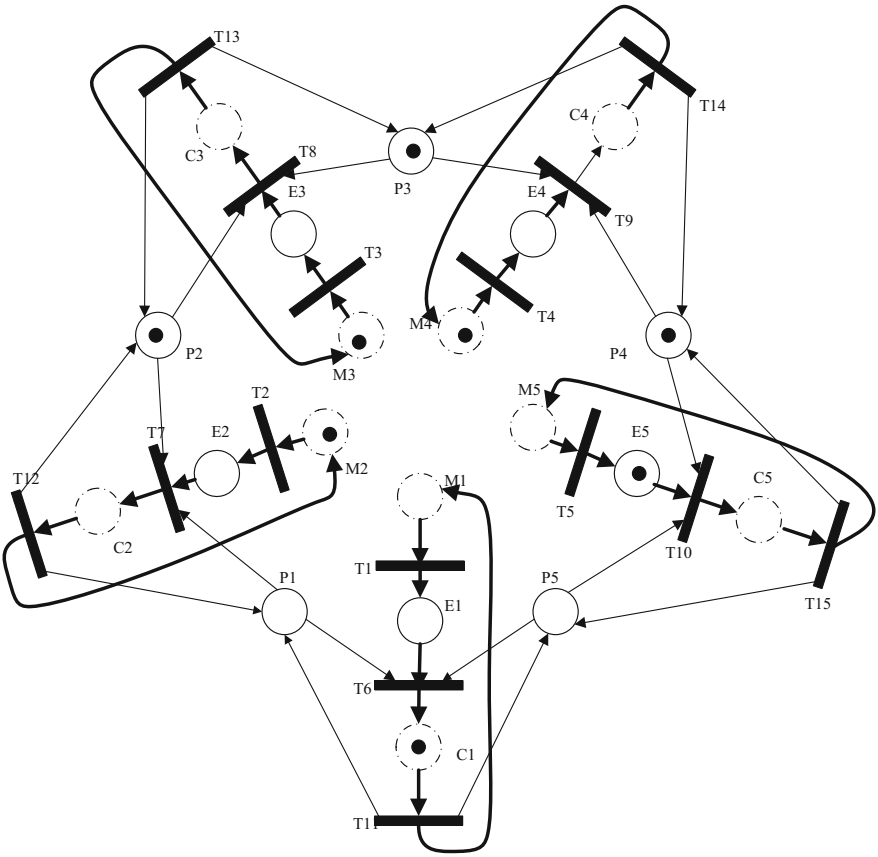


Fig. 5.9 Petri net for the philosophers' dinner problem

pass from the waiting state to that of eating, both chopsticks (the one on the left and the one on the right) have to be available.

The places are:

- M1–M5: meditation of each philosopher
- E1–E5: waiting of each philosopher until left and right chopsticks become available
- C1–C5: philosopher eating
- P1–P5: chopsticks free

And the transitions:

- T1–T5: end of meditation and start of waiting for the chopsticks
- T6–T10: seizing of the left and right chopsticks and start of meal
- T11–T15: end of meal and start of meditation

Figure 5.10 shows one of the five submodels that make up the complete model. In this example, the meditation time follows a uniform distribution U(1.10) minutes and the meal time also follows a uniform distribution U(1.10). The total simulation time is 10,000 min.

The following objects are employed in each submodel:

- The Source class object *InicializacionFilosofo* creates just 1 temporary entity in the initial instant that philosopher f1 represents.
- The Server class object M1 that codes the meditation time (Fig. 5.11).
- The C1 object of the Server class that codes the wait to seize chopsticks P1 and P5, the seizing of the chopsticks, the meal time and the release of the chopsticks (Fig. 5.12). Simio forces the synchronization of the resources seize, in this case when both resources are requested from the same field in the "Secondary Resources".

The synchronous seize of both chopsticks guarantees that the process will not have blockages. Some results obtained from 10 replications are:

– Utilization of the chopsticks: 0.91
– Average time of waiting to have the two chopsticks: 0.11 min

Another variant of the same problem consists of the philosopher trying to reach first the chopstick on his left, and then, the chopstick on his right. He releases both chopsticks simultaneously once he has finished eating and begins to meditate. In this approximation, the blockage is possible when the five philosophers have a chopstick in their left hand. This can happen when all the philosophers want to start

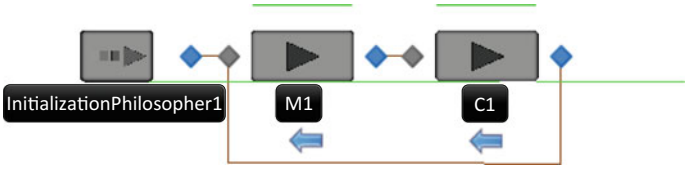


Fig. 5.10 Submodel in simio for the philosophers' dinner

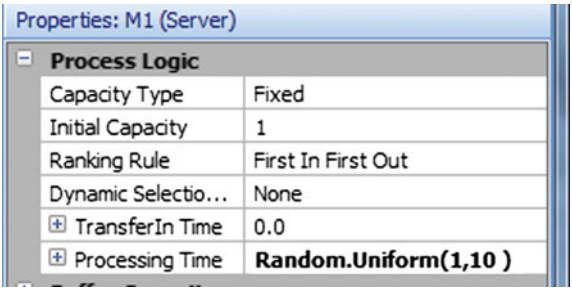


Fig. 5.11 Parameterization of the M1 object of the Server class

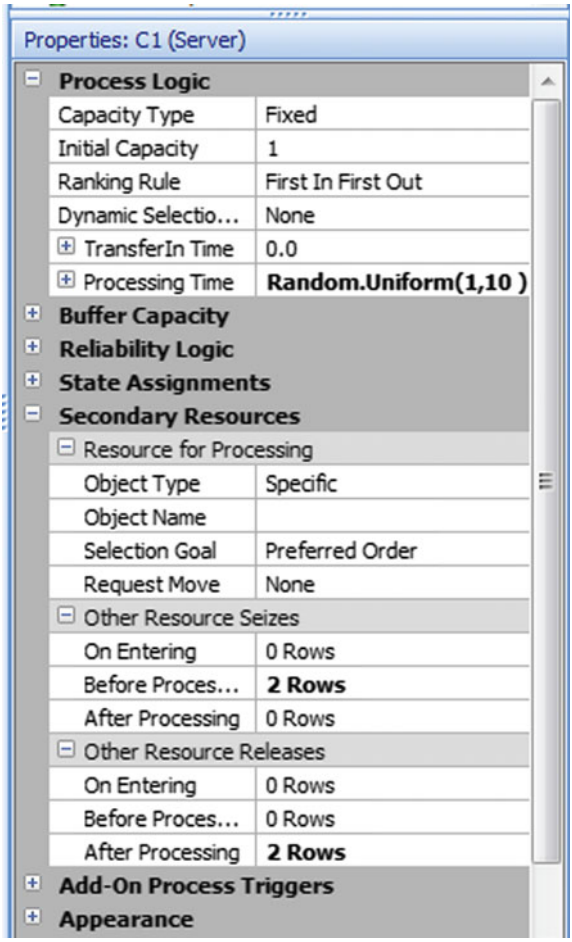


Fig. 5.12 Parameterization of the C1 object of the Server class

eating at the same time, take their left chopstick and then want to pick up the right one.

Unlike the previous Petri net, we can employ a much more compact representation by working with colored petri nets (RPC), as in Fig. 5.13. The statements for the colored petri net are:

- Color P = integer with 1…5
- Color F = integer with 1…5
- Attribute pi of color P
- Attribute pd of color P
- Attribute f of color F
- Previous function (f:F): pd = if f > 1 then f-1 else 5

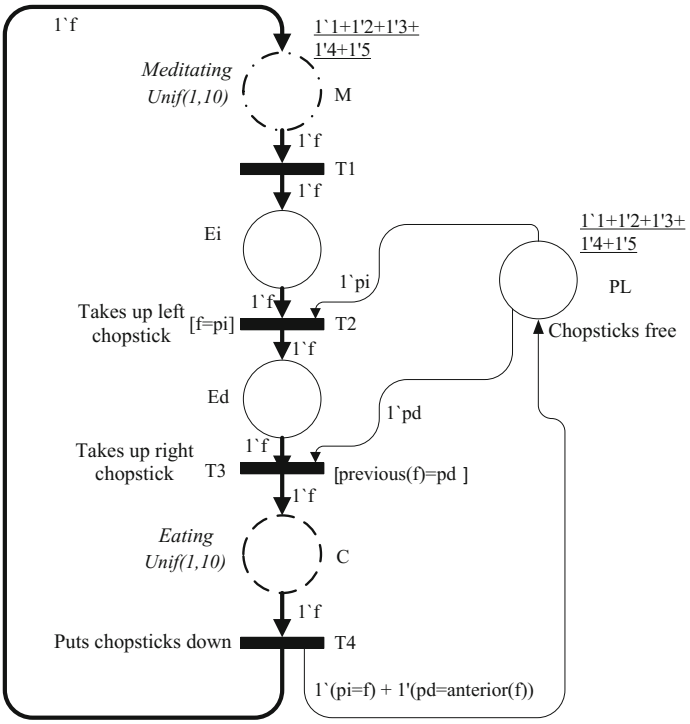


Fig. 5.13 CPN model for the philosophers' dinner

The central aspect of the colored petri net is the expression of guard T3, which indicates that any philosopher of value *f* who wants to eat, needs the chopstick of value *f* and the one of value *f + 1* with the exception of philosopher 1 who needs chopstick 1 and 5.

Below is the formal specification of the above CPN:

$$\Sigma = \{P, F\}$$

$$P = \{M, Ei, Ed, PL\}$$

$$T = \{T1, T2, T3, T4\}$$

$$A = \{MaT1, T1aEi, EiaT2, PLaT2, T2AEd, EdaT3, PLaT3, T3aC, CaT4, T4aM, T4aPL\}$$

$$N(a) = (ORIGIN, DESTINATION) \text{ if } a \text{ has an } ORIGIN \text{ to } DESTINATION \text{ format}$$

$$C(p) = \begin{cases} P & if\ p = PL \\ F & if\ p \in \{M, Ei, Ed\} \end{cases}$$

$$G(t) = \begin{cases} f = pi & if\ t = T2 \\ previous(f) = pd & if\ t = T3 \\ true & if\ t \in \{T1, T4\} \end{cases}$$

$$E(a) = \begin{cases} 1'f & if\ a \in \{MaT1, T1aEi, EiaT2, T2aEd, EdaT3, T3aC, CaT4, T4aM\} \\ 1'pi & if\ a \in \{PLaT2\} \\ 1'pd & if\ a \in \{PLaT3\} \\ 1'(pi = f) + 1'(pd = previous(f)) & if\ a \in \{T4aPL\} \end{cases}$$

Figure 5.14 shows the complete code at "Facility" level. The *In InitializationPhilosophers* object of the Source class creates five philosophers, and for each one of them, it initializes the value of the attribute for the philosopher's identifier (from 1 to 5). M1 is an object of the Server class with infinite capacity and a process time that follows the uniform distribution (1.10) min. Lastly, the object Ei_Ed_C of the Server class with infinite capacity codes the dinner time (uniform (1.10) min).

In the parameterization of the object Ei_Ed_C of the Server class (Fig. 5.15) we can see that we have the "Add-On Process Trigger" Ei_Ed_C_Processing that is executed before initiating the process and the Ei_Ed_C_Processed that is executed when the process is completed.

For each philosopher, the "Add-On Process Trigger" Ei_Ed_C_Processing calls up two consecutive Seize steps. The first being to seize the chopstick on the left and the second to seize the chopstick on the right. Note that the compacting of the Simio model in the "Facility" layer has been achieved by incrementing the complexity in the "Processes" layer. Also note, although it has no effect in this trigger, that the seize of multiple resources in a single Seize Step is not synchronic, Fig. 5.16.

Ei_Ed_C_Processed simultaneously frees the right and left chopsticks when the meal ends (Fig. 5.17).

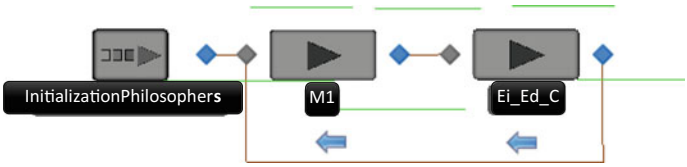


Fig. 5.14 Simio model for the philosophers' dinner

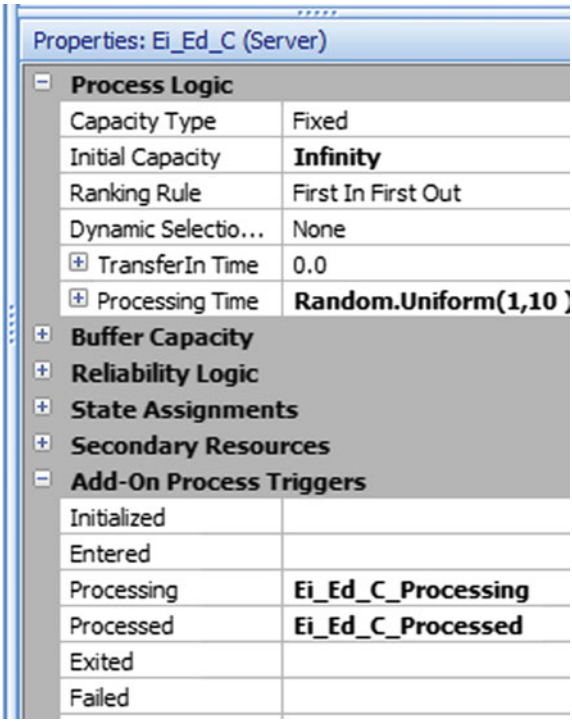


Fig. 5.15 Parameterization of the object Ei_Ed_C of the server class

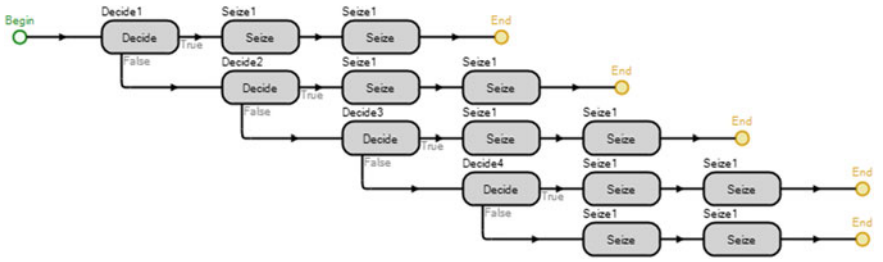


Fig. 5.16 Ei_Ed_C_Processing add-on process trigger

5.5 Manufacturing Process

This case is an adaptation of an example proposed by (Law 2000). Figure 5.18 shows an approach for the process involved. It consists of 5 work stations, one station for the entry/exit of pieces and one or more forklifts for transporting pieces between stations. Each work station has several identical machines and a queue without any size limit. The end goal of the study is to determine how many

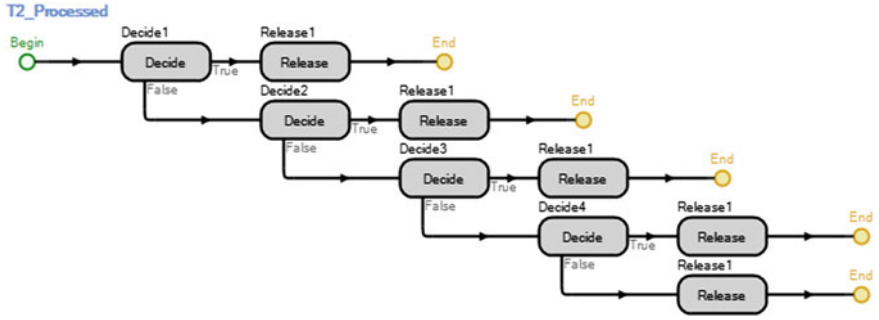


Fig. 5.17 Ei_Ed_C_Processed add-on process trigger

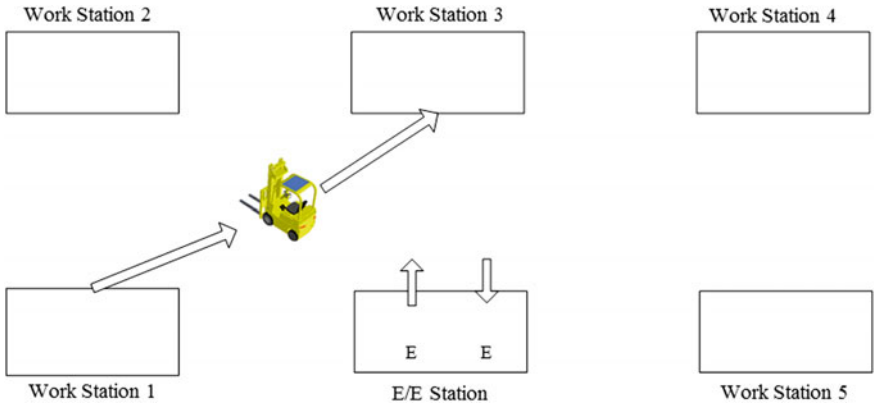


Fig. 5.18 Manufacturing process

Table 5.3 Circuits and mean service times

Type of piece	Circuit	Mean service time (hours)
1	3, 1, 2, 5	0.25, 0.15, 0.10, 0.30
2	4, 1, 3	0.15, 0.20, 0.30
3	2, 5, 1, 4, 3	0.15, 0.10, 0.35, 0.20, 0.20

machines are required in each station and how fast the forklift should be in order to appropriately meet the production specifications.

It is assumed that the pieces to be processed arrive at the entry/exit station depending on a mean exponential distribution of 1/15 h/piece. There are three types of pieces. The probability that a piece is type 1 is 0.3; type 2, 0.5 and type 3, 0.2. Each type of piece has a different circuit, exactly as represented in Table 5.3.

Given that the pieces enter and exit through the E/E station, the trips from the entry/exit station to the first work station and from the last work station to the

Table 5.4 Distance between stations, in meters

	1	2	3	4	5	E/E
1	0	45.5	65	102	91	45.5
2	45.5	0	45.5	91	102	65
3	65	45.5	0	45.5	65	45.5
4	102	91	45.5	0	45.5	65
5	91	102	65	45.5	0	45.5
E/E	45.5	65	45.5	65	45.5	0

entry/exit station must also be taken into account. The time required to process any piece in any machine follows a gamma distribution with a shape parameter of value 2 ($\alpha = 2$), whose mean depends on the type of piece and on the station being worked on. The above table also shows the mean work time per piece and station.

When a machine finalizes an operation, it is blocked until the forklift takes the piece and frees the machine. In this first analysis the forklift travels 1.5 m/s. Table 5.4 shows the distance between the different stations.

We can do a theoretical analysis in order to determine the initial design for the simulation. In order to calculate the number of machines in each work station, we start from the expected time of arrival of pieces at each station and the expected mean service time in each station.

For example, for station 2 we have

$$E(A) = (1/15)/(0.3 + 0.2) = 1/7.5 = 0.133 \text{ h}$$

And employing conditioned probabilities,

$$E(S) = \frac{0.3 \cdot 0.1 + 0.2 \cdot 0.15}{0.3 + 0.2} = 0.12 \text{ h}$$

Given that the utilization has to be under or equal to 1 we have

$$\rho = \frac{\lambda}{w \cdot s} = \frac{E(S)}{E(A) \cdot s} = \frac{0.12}{0.133 \cdot s} \leq 1$$

and

$$s \geq \frac{E(S)}{E(A)} = \frac{0.12}{0.133} = 0.9 \Rightarrow 1$$

The number of theoretical machines per station is summarized in Table 5.5.

A similar calculation can be done to evaluate if the forklift has sufficient capacity. Given that the pieces are type one with a probability of 0.3, the expected time between arrivals $E(A) = (1/15)/0.3 = 0.222$ h. The service time of the forklift is the travel time associated with the *E/E* circuit, 3, 1, 2, 5 and *E/E*

Table 5.5 Number of machines required in each station

Work station	$E(A)$ (hours/work)	$E(S)$ (hours/work) machine	Number of machines
1	0.066	0.215	$3.25 \Rightarrow 4$
2	0.133	0.120	$0.9 \Rightarrow 1$
3	0.067	0.265	$3.95 \Rightarrow 4$
4	0.095	0.165	$1.73 \Rightarrow 2$
5	0.133	0.220	$1.65 \Rightarrow 2$

Table 5.6 Number of forklifts needed

Type of piece	$E(A)$ hours/work	$E(S)$ (hours/work)	Number of forklifts
1	0.222	0.056	0.25
2	0.133	0.051	0.38
3	0.333	0.084	0.25
All the pieces			**0.88** $\Rightarrow 1$

$$E(S) = (45.5 + 65 + 45.5 + 102 + 45.5)\ \text{m}/1.5\ \text{m/s}\ *\ 1/3600\ \text{h/s} = 0.056\ \text{h}$$

and the number of forklifts needed for the type 1 pieces is

$$s = \frac{E(S)}{E(A)} = \frac{0.056}{0.222} = 0.25$$

The number of forklifts needed is summarized in Table 5.6.

These theoretical calculations suffer from two major defects:

1. They do not take into account the proportion of time when the machines are blocked.
2. They do not take into account the movements of the forklifts without a load.

Thus, it is advisable to do the simulation study in order to improve the theoretical design. Figure 5.19 shows the partial Petri net of the system. We can see that the machine ends the process and is blocked while waiting for the forklift. Place P4 considers the time it takes the forklift to move from where it is at that instant to station 2, and the time it takes it to move from station 2 to 5.

The places are:

– P1: pieces waiting to be processed in station 2
– P2: piece being processed in station 2
– P3: piece waiting for the forklift. The machine is blocked
– P4: movement of the forklift from where it is to station 2
– P5: transport to station 5
– P6: pieces waiting to be processed in station 5

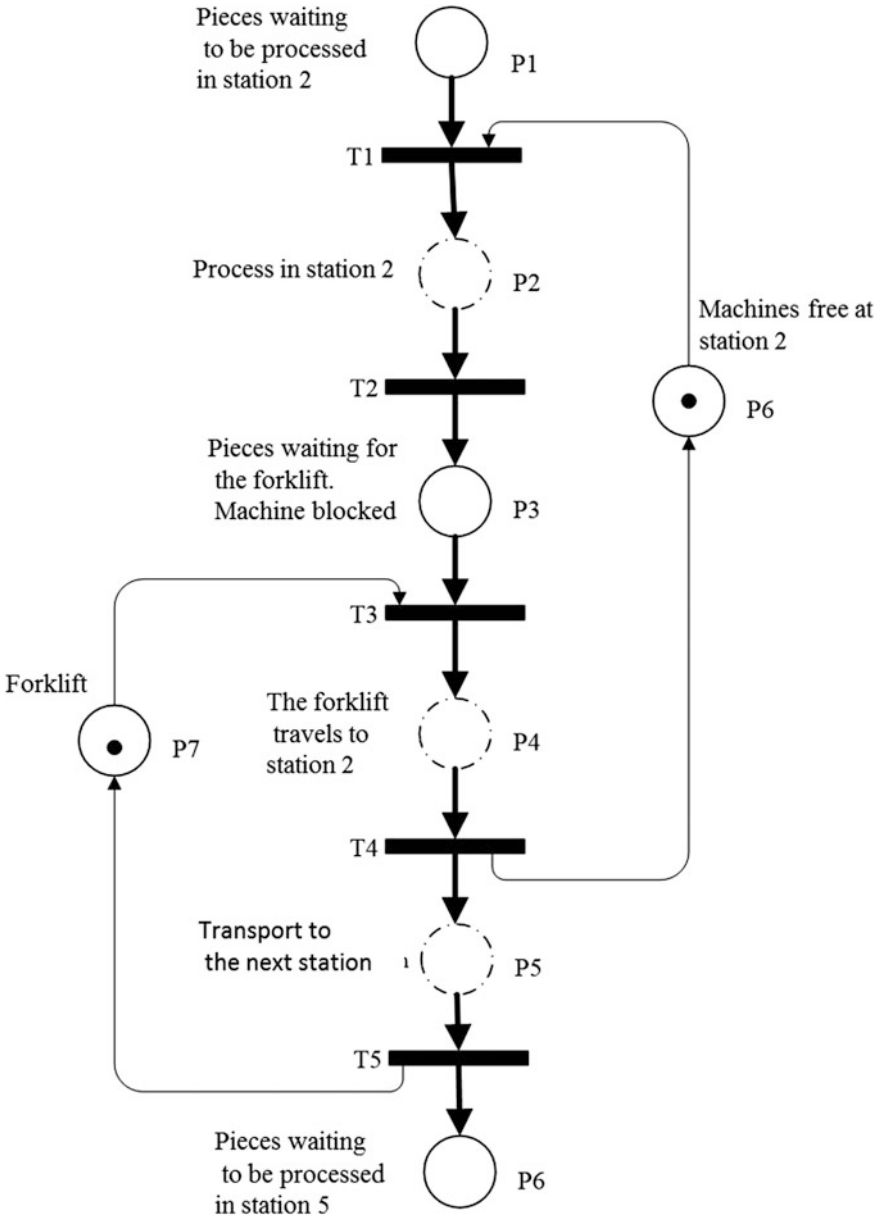


Fig. 5.19 Partial petri net for the manufacturing system

And the transitions,

- T1: seize of the machine of station 2 and start of process
- T2: end of process and start of blocking

– T3: seize of forklift
– T4: arrival at station 2 and release of machine
– T4: end of transport and release of R1 robot

In this example we opted not to get the Petri net of the complete system. In general, we use these nets as a support for modeling and later coding. The decision about what parts are to be modeled in Petri nets and the level of detail of the net itself is a choice that has to be made.

The Simio model is given in Fig. 5.20. It consists of a set of objects of the Server class (*W*1, *W*2, *W*3, *W*4, *W*5), an object of the Source class (*arrival*) and an object of the Sink class (*exit*), connected to each other by a transport network formed in its central rectangular zone by objects of the Path class whereby the *Forklift* object of the Vehicle class is moved. The entry (or exit from each work station *Wi*) TranferNodes are connected by Connectors to the TranferNode of the transport network. For example, the TransferNode *TN1o* is connected to the TransferNode *Input@Wi* and the TranferNode *TN1i* is connected to the TranferNode *Output@W1*.

For every arrival of a piece, the Source1_CreatedEntity Add-On Process Trigger is executed as shown in Fig. 5.21. The Decide Steps bifurcate the token along three paths with probabilities 0.3, 0.5 and 0.2.

The Steps Set Table specifies the sequence that the pieces have to follow according to their type. For example, Fig. 5.22 has defined the sequence for the type 1 piece and is formed by the set of TranferNodes to be visited. The Steps

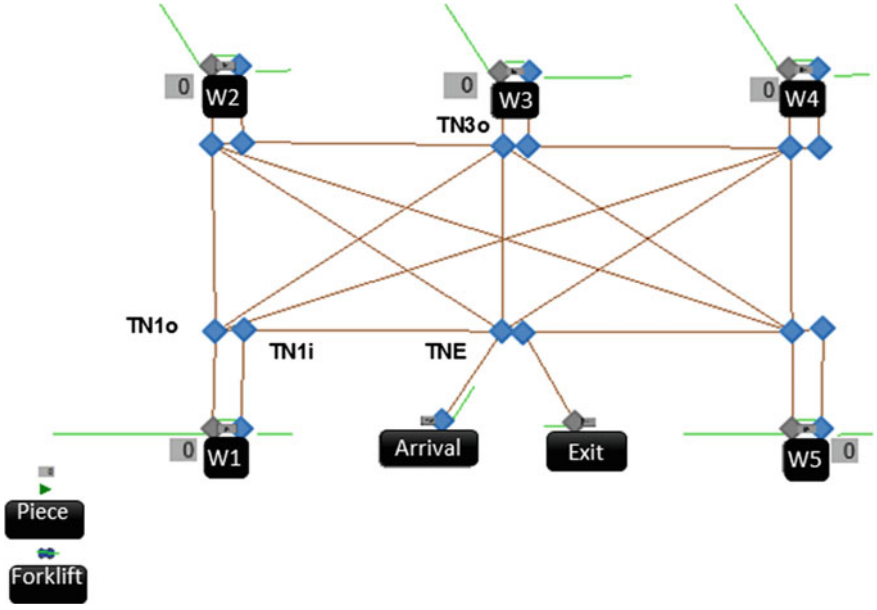


Fig. 5.20 Simio model for the philosophers' dinner

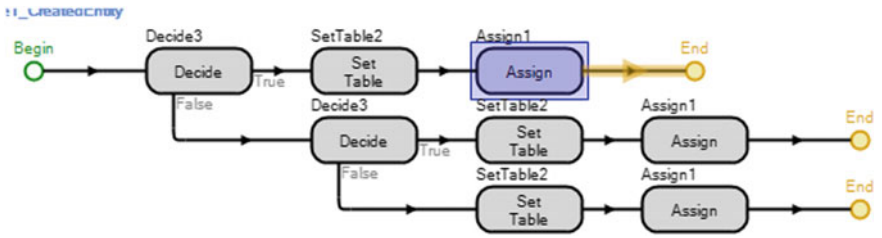
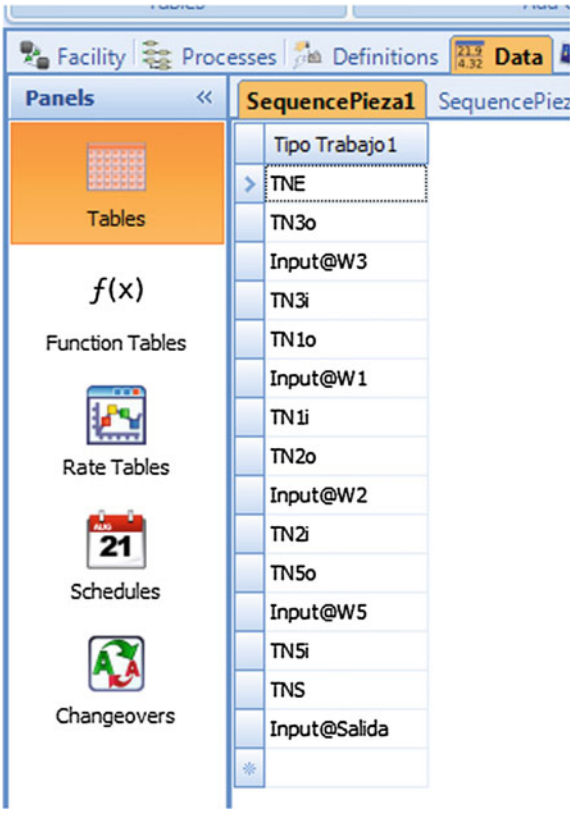


Fig. 5.21 Source1_CreatedEntity add-on process trigger

Fig. 5.22 Sequence of type 1 pieces



Assign memorize, in state variables associated with the piece, the runtime that this will have in each work station.

The last aspect to be highlighted is the parameterization of the TransferNodes (Fig. 5.23). In the OutputLlegada node we specify that the Destination of the entity (piece) is chosen depending on its sequence. In the TNE node we ask the forklift to go to the next Destination. In the TN3o node, the entity frees the forklift and

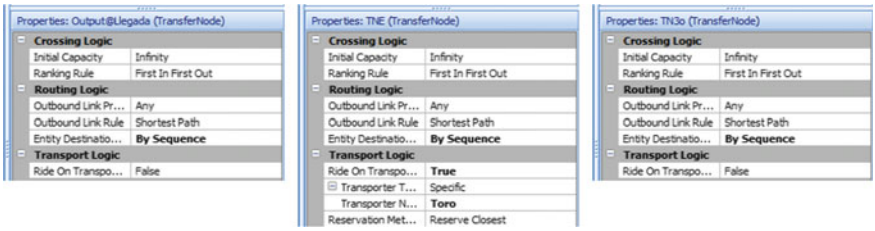


Fig. 5.23 Parameterization of the "transfernode"s Ouput@Llegada, TNE, TN3o

Table 5.7 Results of the simulation for design 1

Station	1	2	3	4	5
Number of machines	4	1	4	2	2
Proportion of machines occupied	70.36	67.29	83	72.61	60
Proportion of machines blocked	21.16	32.51	16.2	27	23
Average number of pieces in queue	2.79	295.2	199.8	179	1.29
Maximum number of pieces in queue	18	487	335.3	309.5	12
Daily average production	92.7				
Average time in the system	44.4				
Proportion of loaded forklift movement	75				
Proportion of unloaded forklift movement	24				

continues with the sequence that takes it to the Input@W3 node, for entry to the work station 3.

Table 5.7 shows the results of the simulation for the theoretical design. The study was done with 10 replications, 320 h of simulation and a warm-up time of 64 h. The daily production achieved with a value 92.7 pieces is far from the target production of 120. We can also see the high number of average and maximum pieces in the queue at stations 2, 3 and 4.

In design 2 a machine is added to stations 2, 3 and 4. The results (Table 5.8) are better, but the daily target production is not yet reached. One aspect that seems important is the high proportion of blocked machines. This is due to the time the

Table 5.8 Results of the simulation for design 2

Station	1	2	3	4	5
Number of machines	4	2	5	3	2
Proportion of machines occupied	74	43	75	54	69
Proportion of machines blocked	25	37	22	32	30
Average number of pieces in queue	138.3	1	21	2.2	163.3
Maximum number of pieces in queue	244.7	9.8	52	16.2	266.4
Daily average production	107.4				
Average time in the system	21.8				
Proportion of loaded forklift movement	83				
Proportion of unloaded forklift movement	16				

Table 5.9 Results of the simulation for design 3 (forklift speed at 2 m/s)

Station	1	2	3	4	5
Number of machines	4	2	5	3	2
Proportion of machines occupied	80	44	80	57	81
Proportion of machines blocked	14	10.8	10	15	16
Average number of pieces in queue	10.8	0.3	4.4	0.7	28.2
Maximum number of pieces in queue	43.5	7.3	27.1	10.8	62.7
Daily average production	120				
Average time in the system	3.9				
Proportion of loaded forklift movement	66				
Proportion of unloaded forklift movement	27				

forklift takes from when the request is made to extract the manufactured piece from the machine until the forklift loads it.

One way of making an improvement may be to increase the speed of the forklift from 1.5 to 2 m/s. The results of Table 5.9 show that the improvement is notable, with the daily target production being achieved. In the event that the maximum number of pieces in stations 1 and 5 is excessive, the possibility of incorporating another machine into stations 1 and 5 could be raised.

5.6 Automated Warehouse

This example is a simplification of the problem proposed in (Guasch et al. 2011). The system to be studied is an intermediate warehouse for paper spools managed by two pallet elevators that automatically feed several rotating pieces for printing newspapers. Figure 5.24 shows a diagram of the warehouse made up by a central rail along which the two pallet elevators move; 88 warehousing positions in the horizontal direction of the rail.

For each horizontal position we have, in general, 6 warehousing positions; 3 on both sides of the pallet elevator occupying three different positions in the vertical direction, see Fig. 5.25.

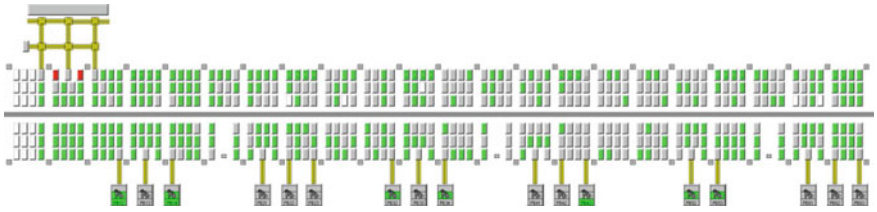


Fig. 5.24 Diagram of automated warehouse

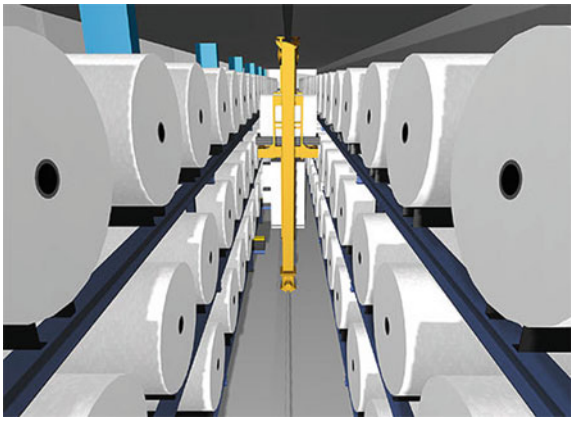


Fig. 5.25 View of the warehouse from the central rail

Given that the two pallet elevators circulate along the same central rail, the objective of the study is to determine the work zone assigned to each pallet elevator in order to distribute the work load equitably between the two. The working configuration using two pallet elevators is necessary given that just one is not enough to cover the work demand during the peak hours from 0 to 2 a.m.

Briefly, pallet elevator 1 works in the left zone; 2 in the right zone, and a block of 2 horizontal positions (12 warehousing positions) is reserved as a zone exclusively for the exchange of spools between the pallet elevators. To avoid the two pallet elevators hitting each other, the control system only allows the entry of one pallet elevator at a time into the exchange zone.

The travel time between two horizontal positions depends on the distance that is run in horizontal positions (dh)

$$Time = (dh - 5) * 0.97 + 15.7\,(\mathrm{s})$$

If $dh = 0$, the time is also 0. The travel time between two vertical positions depends on the distance that is run in vertical positions (dv) and whether it is up or down (Table 5.10). Since horizontal and vertical movements are performed in parallel, the movement time between two different positions is the maximum of the vertical and horizontal times.

The time taken to transfer a spool from the warehouse to the pallet elevator or from the pallet elevator to the warehouse is 21 s.

To do the study, we start from a file of real orders in the period from 0 to 2 a.m. during 7 consecutive days. These orders, although given in the working configuration with a single pallet elevator, are independent of the way of working. This is

Table 5.10 Vertical travel times

Dv	Direction	Time (s)
1	Up	10.2
2	Up	15
1	Down	12.2
2	Down	23

why it is unnecessary to manage the locations in the model, except for the locations in the exchange zone. Nevertheless, the management of said locations has been simplified, assuming that there is sufficient space and mean access times.
The most important aspects of the process to be simulated are:

- The resources (or permanent entities) that are involved in the process: pallet elevators T (2) and the exchange zone E (1). The set of locations in the exchange zone is not considered a resource. However, the simulation shows us that the number of spools there are in this zone never at the same time exceeds its maximum capacity.
- The temporary entities: orders O.
- The Set of activities and transitions that are made explicit in the colored petri net.

The statements for this type of net are:

- Color T: integer with 1…2
- Color H: integer with 1…88
- Color E: exchange zone
- Color O = HxH: order with a position of origin and a position of destination
- Attribute t of color T
- Attribute ha, ho, hd of color H, where ha is the current horizontal position of the pallet elevator, ho the position of origin where the spool to be picked up is to be found and hd the destination position of the spool to be placed.
- Attribute e of color E
- Function z(h:H): if h <=hi then 1 else 2

Figure 5.26 shows the Petri net of the system. The places are:

– P1: transport order with position of origin ho and destination hd
– P2: transport order awaiting pallet elevator 1 that is in the horizontal position ha
– P3: pallet elevator 1 moves from ha to ho in order to load the spool and from ho to hd in order to unload it
– P4 and P5: the same as P2 and P3, but for orders that only specify pallet elevator 2
– P6: the two free pallet elevators
– P7: the free exchange zone
– P8: order waiting for pallet elevator 1
– P9: pallet elevator 1 moves from ha to ho in order to load the spool and from ho to position hi-1 waiting to be able to enter the exchange zone
– P10: pallet elevator 1 waits for the exchange zone to be free
– P11: movement of pallet elevator 1 to the exchange zone in order to leave the spool and withdraw to position hi-1
– P12: waiting for pallet elevator 2 to be free
– P13: movement of pallet elevator 2 to position hi + 2, bordering on the exchange zone
– P14: pallet elevator 2 waits for the exchange zone to be free

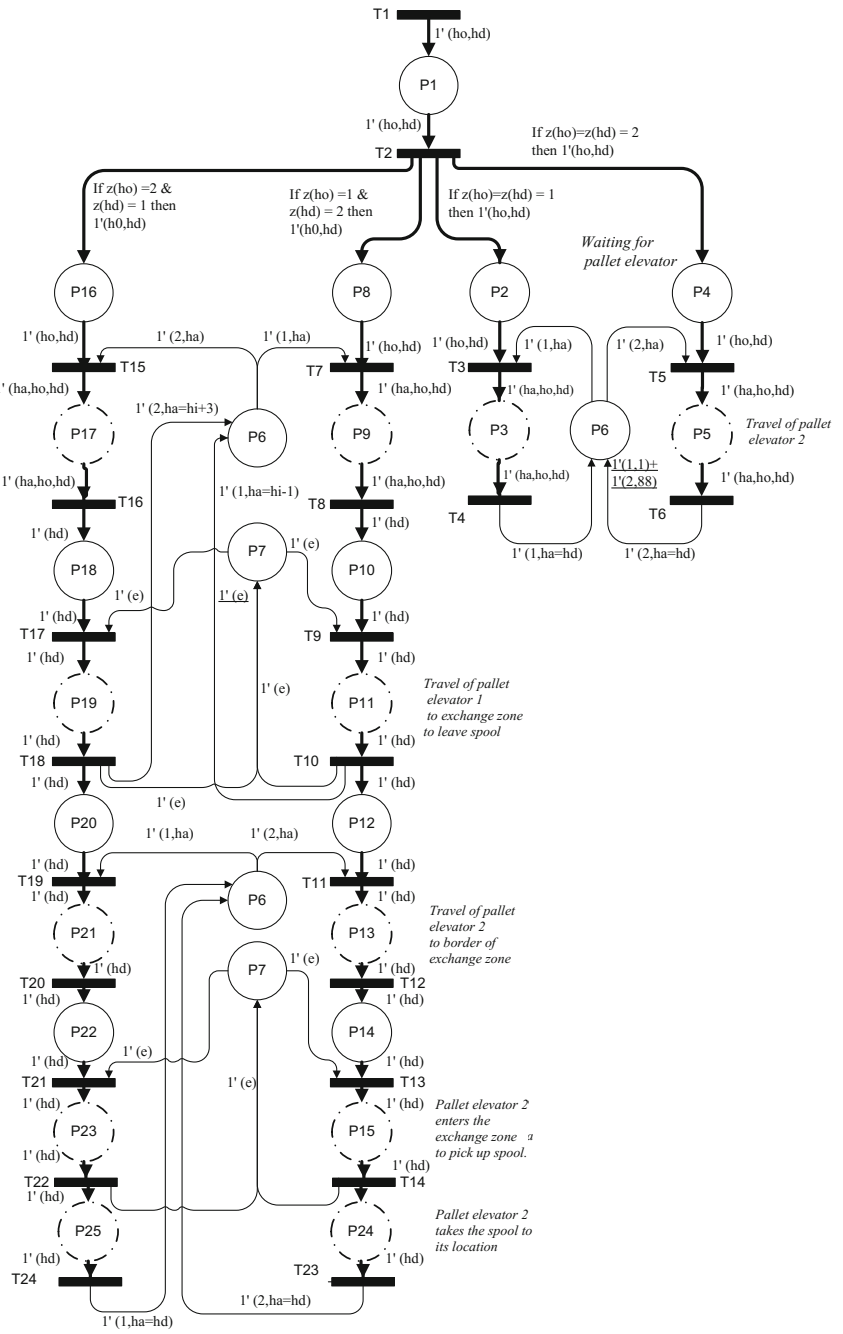


Fig. 5.26 Colored petri net of the model

- P15: pallet elevator 2 enters the exchange zone in order to pick up the spool and leave this zone
- P23: pallet elevator 2 takes the spool to its final location
- P16, P17, P18, P19, P20, P21, P22, P23 and P25: the same as P8, P9, P10, P11, P12, P13, P14, P15 and P24 but the first stage of the process is done by pallet elevator 2 and the second, pallet elevator 1.

And the transitions:

- T1: arrival of orders.
- T2: distribution of orders as a function of whether we only need pallet elevator 1 or 2, or a combination of both when the exchange zone is used.
- T3: seizing of pallet elevator 1 and start of movement.
- T4: end of operation and release of pallet elevator 1.
- T5 and T6: the same as T3 and T4, but for pallet elevator 2.
- T7: seizing of pallet elevator 1 and start of movement.
- T8: the pallet elevator has arrived at the position hi-1 before the exchange zone and stops, waiting for this zone to become free.
- T9: the exchange zone is seized and the movement starts in order to leave the spool in the exchange zone.
- T10: pallet elevator 1 has completed the operation and remains stopped in position hi-1.
- T11: seizing of pallet elevator 2 and start of movement.
- T12: pallet elevator 2 has reached the position hi + 2 waiting for the exchange zone to become free.
- T13: seizing of the exchange zone. Pallet elevator 2 starts the movement of entry to this zone.
- T14: pallet elevator 2 leaves the exchange zone and leaves it free.
- T23: pallet elevator 2 leaves the spool in its location and remains free in this position waiting for new orders.
- T15, T16, T17, T18, T19, T20, T21, T22 and T24: the same as T7, T8, T9, T10, T11, T12, T13, T14 and T24 but the first stage of the process is done by pallet elevator 2 and the second, pallet elevator 1.

Figure 5.27 shows the Simio code for the process to be simulated. It has a direct parallelism with the colored petri net. The initial stage consists of the pallet elevator going in empty to the position where the spool is and later moving on loaded. It has been coded with two objects of the Server class, for example P5a and P5b. The coded model includes both horizontal and vertical movement, although the CPN does not explain the attributes associated with the vertical positions. The times associated with each one of the different movements is calculated in the Add-On Process Triggers for the different objects of the Server class.

Table 5.11 shows the results for each one of the simulations performed. We can see that the exchange position 45 (and 46) is the one that has associated a shorter mean time in the system. This is the time that elapsed between order being received to transfer a spool and the order being executed. For this specific case, the

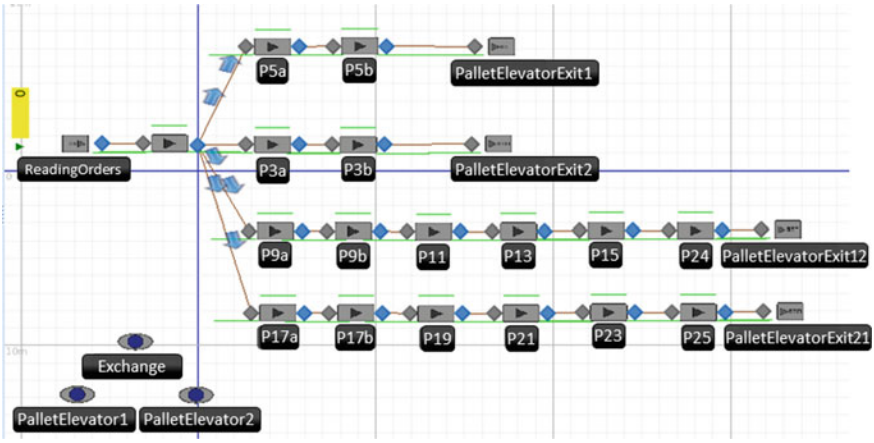


Fig. 5.27 Simio code for the warehouse model

Table 5.11 Results of the simulation study

Exchange position	Utilization tr1	Utilization tr2	Waiting time	Maximum wait	Time in system	Exchange factor
0	0	0.76	247	918	351	0
10	0.18	0.79	294	1373	472	0.21
20	0.25	0.73	205	1268	374	0.25
30	0.4	0.63	82	679	249	0.3
35	0.49	0.51	64	541	214	0.29
40	0.54	0.43	57	491	192	0.24
45	0.55	0.37	58	498	188	0.21
50	0.59	0.35	85	913	216	0.23
60	0.66	0.27	108	763	237	0.2
70	0.78	0.14	263	1107	406	0.17
80	0.78	0.11	289	1253	422	0.12
89	0.76	0	247	918	351	0

utilization of the two pallet elevators is 0.55 and 0.37 respectively; the mean queuing time of the orders is 58 s; the order that waited longest has taken 498 s to be served and 21% of the operations have employed the exchange zone.

Figure 5.28 shows the mean time of the orders in the system as a function of the position of the exchange zone.

The model coded in Simio is a simplification of the reality and it is possible to get better results with other management policies. In this model it is assumed that before entering the exchange zone we stop the pallet elevator in order to request permission to enter and on exiting we stop the pallet elevator in order to free the exchange zone. These stops have a significant associated loss of time. An

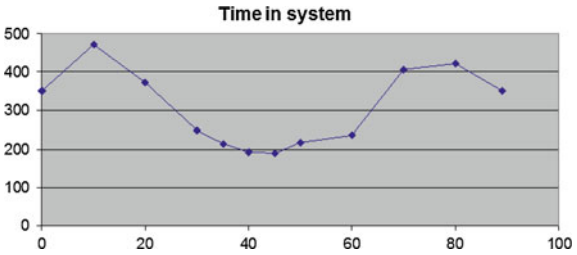


Fig. 5.28 Time in the system as a function of the position of the exchange zone

improvement is to release on exiting without stopping the pallet elevator, and, if the exchange zone is free, not to stop on entering it. Another possible improvement to be assessed would be to leave the pallet elevator in its central work zone after completing the operation if no orders are queued.

References

Dijkstra, E. W. (1971). Hierarchical ordering of sequential processes. *Acta Informatica, 1,* 115–138.

Guasch, A., Piera, M. A., & Figueras, J. (2011). Automatic warehouse modeling and simulation. *International Journal of Simulation and Process Modeling*, *6*, 288–296. ISSN: 1740-2123.

Law, A. M., & Kelton, W. D. (2000). *Simulation modeling and analysis*. New York: McGraw-Hill.