Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, eds.

APPLICATION OF NON-MARKOVIAN STOCHASTIC PETRI NETS TO THE MODELING OF RAIL SYSTEM MAINTENANCE AND AVAILABILITY

Pierre Dersin

ALSTOM Transport
48 Rue Albert Dhalenne
93482 St-Ouen, FRANCE

René C. Valenzuela

School of Aerospace Engineering
Georgia Institute of Technology
270 Ferst Dr., Atlanta, GA 30332, USA

ABSTRACT

With the increasingly stringent contractual requirements placed on system availability in urban and intercity passenger rail systems, and the emergence of public-private partnerships with maintenance contracts over periods of 25 years or more, rail system suppliers such as ALSTOM Transport now adopt an integrated logistic support (ILS) vision, where the entire support system (maintenance policy, crew scheduling, spare parts, tools, etc.) is modeled at the same time as the main system. The need to overcome the restrictive assumptions imposed by Markov models has led us to the use of non-Markovian stochastic Petri Nets, which in addition lend themselves to building decentralized, hierarchical models. The challenges that are addressed in this paper are how to deal with: deferred maintenance, aging, and different time scales (not all units in the rail system have the same mission profile). Comparisons are made with results obtained with Markov models.

1 INTRODUCTION

To succeed in the competitive market of urban passenger rail systems, suppliers must minimize the cost of the offered solution while still satisfying stringent performance and dependability requirements. Current contracts have evolved to include maintenance provision during the life cycle of the system thus shifting the operating and maintenance costs (and associated risks) to the equipment manufacturer or integrator. In these so called *Performance Based Contracts* one of the most appropriate measures on which to base the purchasing decision is the solution's life cycle cost (LCC). In its most coarse description LCC can be split into the sum of the cost of acquisition, the cost of operation and maintenance, and the cost of disposal (IEC 2005). In passenger rail systems the cost of operation and maintenance can be several times the acquisition cost. To offer competitive solutions it is fundamental to obtain an accurate and timely estimation of a set of feasible configurations and their associated cost of operation and maintenance.

Passenger rail systems are large scale systems made up of heterogeneous very reliable components. Components with both constant and increasing failure rate are part of the system. The system is spatially decentralized. To satisfy the reliability requirements redundancy at different hierarchical levels is used. Their maintenance policy includes corrective and preventive maintenance actions. Corrective maintenance actions are chosen based on the effect of the failure on the system. In an an urban train passenger system we have that for some of the subsystems only a portion of the operating time will produces revenue to the operator. We call this portion of time the revenue service time. To achieve high levels of availability the maintenance policy defines an immediate replacement action for components whose failure affects revenue service time. On the other hand, for components whose failure does not affect revenue service time but its immediate restoration does the maintenance policy defines a deferred replacement action. Depending on the component and on the failure mode some of the replaced components are repaired (at a workshop) or discarded and replaced by a new one. Periodical inspections (scheduled by groups of components) are used to detect failures of redundant components that have not been detected by their built-in test equipment.

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

Dependability measures of interest include steady state availability, interval availability and completed number of trips in a given period of time. To carry out the maintenance actions the necessary workers and spares need to be available. The benefits that can be obtained by including logistic considerations have been recently considered by some researchers. Volovoi and Peterson (2011) combine the results from a maintenance model and a logistics optimization model to iteratively calculate an optimal set of spares and availability. Tu, Ghoshal, Luo, Biswas, S., Jaw, and Navarra (2007) combine prognostic techniques with a higher-level reasoning engine to generate actionable tasks for the inventory and maintenance management decision support systems. It can assess different sparing allocation schemes and maximize availability within budget constraints.

Since the early work summarized in (Barlow and Proschan 1965) stochastic-process techniques have been used extensively to study the optimality of different maintenance policies. A review of the current state of such approach can be found in (Kobbacy and Murthy 2008). Due to the large scale and the complexity of the maintenance policy the analytical approach cannot be used as a modeling tool in this case. Discrete event simulation models based on different modeling frameworks have been used to model these systems. Shaikh and Mettas (2010) estimate availability, production and maintenance costs of a natural gas processing plant. Chang, Chang, and Zio (2010) study the effect of different maintenance strategies on the production availability of an off-shore facility. Both use an approach based on reliability block diagrams and Monte Carlo simulation. As an alternative to RBD and CTMC models, non-Markovian stochastic Petri nets have also been used to model maintenance processes of complex systems. Volovoi (2007) compared opportunistic and conditioned based policies with optimal age-based replacement policies. Zille, Brenguer, Grall, and Despujols (2010) used a hierarchical approach to model and study the performance of a turbo lubricating system. When using a simulation approach to estimate the dependability metrics of large scale very dependable systems a crude Monte Carlo (Asmussen and Glynn 2010) simulation approach becomes impractical (Nicola, Shahabuddin, and Nakayama 2001). Techniques such as importance sampling and splitting have been used to reduce the number of replications needed to achieve the desired confidence level of the estimator. More information on the use of importance sampling in context of calculation of dependability measures can be found in (Nicola, Shahabuddin, and Nakayama 2001).

In this paper, we discuss a modeling methodology based on non-Markovian stochastic Petri nets that can be used to systematically create a model of the failure and maintenance process of a large scale complex system including maintenance operators and spares inventory. We describe in particular, how non-Markovian stochastic Petri nets can be used to model deferred maintenance actions and aging. Furthermore, we include a discussion of how they are used to model the different time scales of the failure and restoration process.

The paper is structured as follows. In Section 2 we formalize the specific stochastic Petri net framework used in this work. In Section 3 we present the modeling methodology and describe how aging and deferred maintenance policies are handled.

2 STOCHASTIC PETRI NETS

Following (Bobbio, Puliafito, Telek, and Trivedi 1998), a marked Petri net PN is a tuple $PN = (P, T, I, O, M)$, where: P is the set of places (drawn as circles), T is the set of transitions (drawn as thin bars if immediate or thick bars otherwise), I, O are the input (output) functions and M is the marking of the PN. A transition is enabled in a marking if each of its input places contains at least as many tokens as the multiplicity of the input function I. An enabled transition fires by removing as many tokens as the multiplicity of the input function I from each input place, and adding as many tokens as the multiplicity of the output function O to each output place.

Stochastic Petri nets are PN in which the timing is stochastic. In this work, the firing time of any given transition can be either: immediate, deterministic or random (modeled with a continuous cdf with infinite support). Furthermore, inhibitor arcs, transition priorities and guards are the structural extensions with which the Petri net framework used in this work is enriched. An inhibitor arc disables the transition to which is connected if the number of tokens on the place on the other end of the arc is greater than zero.

Transition priorities allow the breaking of ties between the firing of transitions. If two or more transitions are scheduled to fire at the same time then the transition with the higher priority will fire first. Furthermore, if two or more transitions with the same priority are scheduled to fire at the time then the transition with the lower index will fire first. Guards are boolean conditions that may be associated with a transition. The transition will be enabled if and only if the appropriate number of tokens are on in its input places and the guards associated with it evaluate to true.

In the Petri nets framework the enabling of a transition is interpreted to be the starting of an activity. The firing of a transition corresponds to the completion of an activity. When the firing of a transition causes a previously enabled transition to become disabled it means that the corresponding activity was interrupted before being completed.

3 MODELING METHODOLOGY

The driving principles when designing the modeling methodology are ease of implementation and reusability. The inputs required to create the model are the system structure, FMECA (Failure Modes, Effects and Criticality Analysis), proposed maintenance policy including logistic support (structure) and the customer RAM (i.e. Reliability, Availability and Maintainability) requirements. As an output of this process a SPN model of the system will be created.

The SPN model will receive as inputs the reliability data of the components comprising the system and the parameters of the maintenance policy and will produce as outputs the estimation of the performance parameters and cost of the proposed maintenance policy. A scheme of this process is depicted in Figure 1

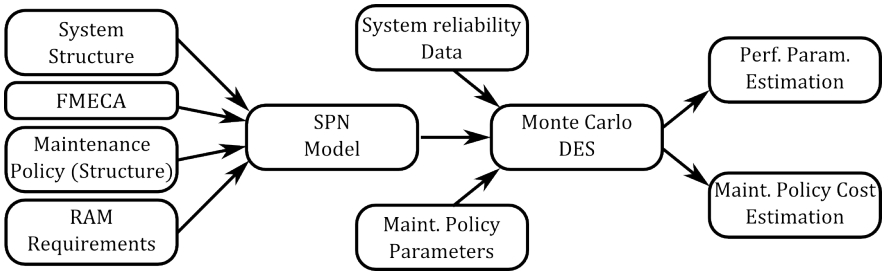

Figure 1: Modeling creation and usage flowchart.

The proposed modeling methodology is based on the modularization and data encapsulation of the code. Modularization will ease the implementation by allowing the reuse of previously defined *PN* subnets. With data encapsulation we ensure consistency between a specific level of the model hierarchy and the abstraction level of data available to it. The first step in this direction involves establishing a taxonomy of the process. To do that a description of the process under consideration follows.

3.1 System Description

Urban train signaling systems offered by Alstom share a common backbone which is then customized to ensure compliance with the customer needs and requirements. The system has a well defined hierarchical structure with redundant elements at different hierarchy levels. Its elements are spatially distributed on both fixed (servers in control rooms, point machines along the track) and moving locations (on-board equipment on rolling stock). Some of them base their operation on electronic principles and others on mechanical principles.

The system implements graceful degradation by allowing different operation types depending on the availability of some of its subsystems. Some of the components (servers) operate continuously while others only during revenue service (on-board products). Revenue service hours (period of time during which the

operation of the system is generating revenue) vary from customer to customer. Furthermore, there can be a difference between the number of products operating depending on the time of the day/week.

At the lowest hierarchical level a specific function is achieved either by a single component, two identical components in a 1oo2 (one-out-of-two) architecture or three identical components in a 2oo3 (two-out-of-three) architecture. At the component level there are two processes of interest: the failure process and the restoration process.

The failure process is intrinsic to the component and describes the paths which the component can follow to go from the operating "UP" state to the non-operating "DWN" state. In the components being considered a distinction must be made between detected and undetected failures. A detected failure is a failure which the self-test equipment of the component is able to identify. After a failure is detected the failed component can signal its own state to the higher levels of the system. The fault coverage of a component is an important design parameter and has an impact on the future maintenance operations. Consider, for instance, a product made up of N components arranged in series. In this setting, failure of one of the components will result in loss of the function of the product. If the component has suffered an undetected failure then to restore the product to an operational state a fault localization procedure has to be performed first. If, on the other hand, the failure of the component has been detected then the fault localization procedure is not required. In the case of a function being performed by a set of components in a redundant architecture detected failures are combined with deferred maintenance actions to minimize the possibility of service interruption.

The restoration process describes the set of actions that are available to a maintainer to analyze and restore a component to the operational state. To make it intrinsic to the component it is considered that this process consist of the actions that have to be taken only after the logistic prerequisites have been satisfied.

To maintain the high levels of the dependability required of the system there are two types of corrective maintenance actions implemented in the maintenance policy: immediate and deferred. Both of them will replace the failed element by a spare from the pool of available spares (new or repaired off-line previously). We refer to this process as an on-line replacement. The difference between the immediate and deferred maintenance action is the time at which it will be performed. An immediate corrective maintenance action is triggered immediately after failure and a deferred corrective maintenance action is triggered at the first (previously defined, fixed) planned scheduled maintenance after the failure has occurred. The selection on which one of the corrective maintenance actions is implemented depends on the effect of the failure on the performance of the system.

As a result of the restoration process, a failed spare is now part of the system. The process describing the procedure that brings the failed part to a state of an available spare part is dubbed the off-line restoration process. In the most general setting the failed spare is then taken to a workshop where it is fixed or replaced by a new part and then returned to the pool of available spares.

3.2 Lower Level Model Taxonomy

Based on the previous description of the system the following taxonomy (modularization) is proposed at the component hierarchical level (the idea being to split the net in atomic modules (subnets) each of which handles a specific set of actions). Six subnets are defined: the failure subnet, the logistics subnet, the on-line restoration subnet, the off-line restoration subnet, the system state subnet and the messaging subnet.

The logistics subnet describes the preconditions required for the restoration process to begin. It is derived from applying the system maintenance policy to the component. It is not intrinsic to the component because the same component can be subject to a different maintenance policy depending on the system on which is embedded.

The messaging subnet encapsulates the data and transmits it at the appropriate abstraction level. For instance, consider three consecutive hierarchical levels. Suppose, that an undetected failure occurs at the lowest level (component) which in turns results in the product (at the intermediate hierarchical level) to also

failing (due to loss of function). Although in the model it is known at the product level which component has failed, at the subsystem level (higher hierarchical level) it is only known that the product is failed.

The relationship between the state of the component and these subnets is shown in Figure 2. In the figure there are five states depicted. “UP” and “DWN” denote the corresponding states of the component. “Wrk” denotes the number of available workers. “SprF” and “SprA” denote the spares that are failed and the available spares.

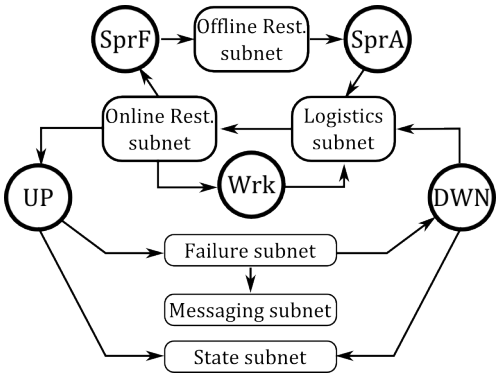

Figure 2: Component state and process subnets.

An example of the implementation of the on-line restoration subnet for a pair of identical components in a 1oo2 architecture is shown in Figure 3.

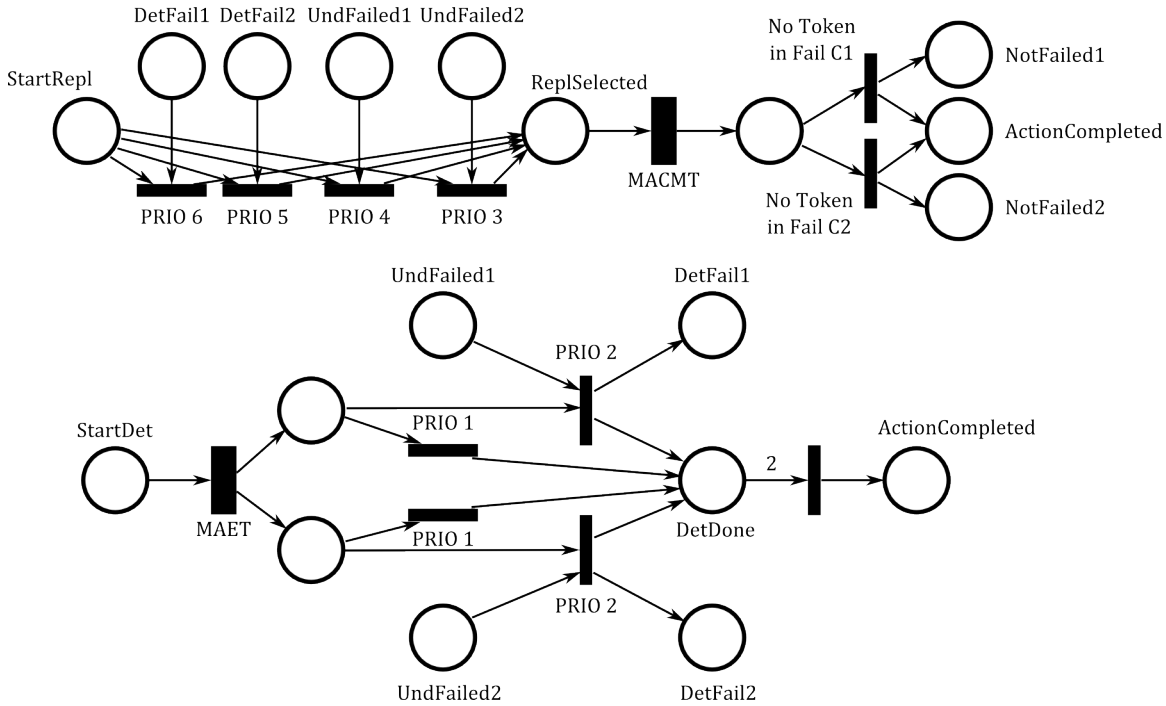

Figure 3: Restoration subnet for a 1oo2 architecture.

In this example the restoration subnet is composed of two subnets. One of the subnets models the replacement of a failed component (its entry point is the *StartRepl* place) and the other models the failure localization procedure (its entry point is the *StartDet* place). The exit point of both subnets is

the *ActionCompleted* place. At that point control of the simulation returns to the appropriate point in the logistics subnet. The algorithm for the replacement of failed part is as follows. The first step is to select the part to be replaced (this subnet assumes that there is a part that needs to be replaced. Checking of this assumption occurs on the corresponding logistics subnet). At this point there could be one or two parts that need to be replaced. To select only one part in the case that two of them are failed transition priorities are used. Once the selection is performed (note that the selection will remove the token representing the component from the failure subnet) the replacement action takes place. This is modeled by the transition labeled MACMT (Mean Active Corrective Maintenance Time). When using a Markov model this time is modeled as following an exponential distribution. Since this limitation is not imposed by the non-Markovian SPN model a different probabilistic model can be chosen here (uniform or lognormal are two possibilities). Finally, with the appropriate use of guards a token is returned to the *UP* state of appropriate failure subnet.

The use of lognormal distributions is justified for instance, when modeling the MACMT of a unit which we know is failed but it has several failure modes that need to be diagnosed before the restoration can occur. Most of the times the particular failure mode which has occurred is easy to diagnose leading to short restoration times. Other times, however, a failure which cannot be easily recognized occurs. In that case the maintenance operator will follow the standard diagnostic procedures which will fail to recognize the failure. Once that happens the restoration time can easily become very long (an expert might have to brought in to study the problem, the restoration is deferred to the next shift where a more experienced maintenance operator is at hand, etc.)

3.3 Immediate and Deferred Maintenance Policy Implementation

Figure 4 shows a simple implementation of the logistics subnet of a component. The entry point of the net is a request for a maintenance action (MA) coming from the failure subnet (through the DWN state). After the request is placed the MA is only triggered when the resources (workers and spares) are available. Once the resources become available and after the firing delay (specified in transition T_1) a MA is triggered in the restoration subnet (after the resources become available at the location of the component that has failed).

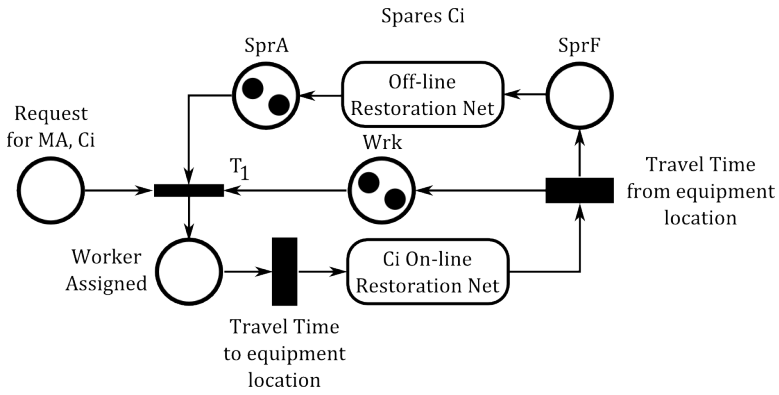

Figure 4: Integrated logistics support net *i*.

To specify whether an immediate or deferred MA is desired one only needs to set the firing delay of transition T_1 accordingly. For an immediate MA an instantaneous type must be used (as shown in Figure 4). For a deferred MA a periodic type (which only fires if its enabled at predefined time intervals) must be selected.

3.4 Operation Profiles

The discontinuous nature of the revenue service time interval and the fact that most of the performance metrics are based on operations during revenue service time imply that great care must be taken to ensure that at any given time (but in practice at any given failure time) the model can decide whether it is in a revenue service period or not. To illustrate consider a system which has a revenue service time interval of 18 hours a day (on weekdays) and a server that operates continuously (24 hours a day). Figure 5 shows a typical operating profile of an element of the system. Failure of the server induces a service interruption but only if the system is in a revenue service period. The probability of a server failure during non-revenue service time is non-negligible and must be considered when estimating the performance parameters. There are two issues involved in this example. First, the failure distributions of the elements in the system correspond to failures in operating time, which for elements operating non-continuously is different from calendar time. Second, a way for the model to decide whether a given time corresponds to a revenue service time needs to be implemented. We deal with the first issue in this section and leave the latter for the following section.

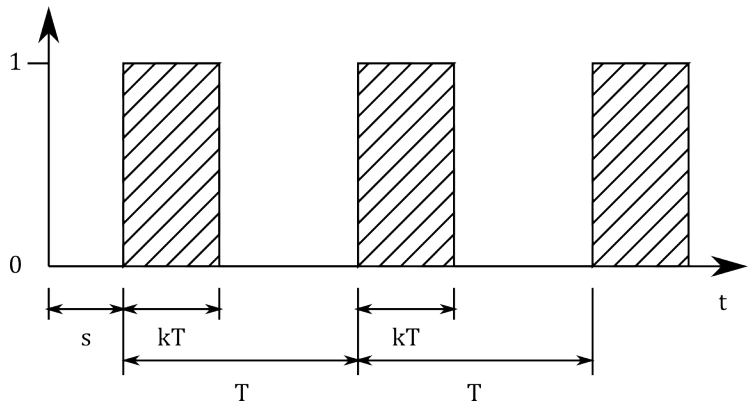

Figure 5: Operating profile of component i.

A naive implementation of the different operation profiles within the system calls for using local clocks with an extra state for each system element. The extra state would denote an element which is not failed, cannot fail (while in the state), and is not operating. The element would then move according to its operation profile from the operating state (during revenue service) to this *OFF* state. A new failure time is sampled anytime the element enters the *UP* state. The problem with such naive implementation is that it triggers an event twice a day for each element that does not operate continuously. On a replication reaching a final time of 30 years (typical life cycle length of a signaling system) triggering events twice a day considerably lengthens the run time of a single replication.

To overcome this issue a scaling function is introduced for the elements that do not operate continuously. During the simulation the pseudo-random number generator provides a variate which corresponds to the next event time. This event time is then scaled up according to the element's operating profile.

To define the scaling function consider a component/product/subsystem that operates for kT units every T units (with $k \in (0,1)$) and that its first operation cycle starts at time s (see Figure 5). Then, the relationship between the calendar time t and the element's operation time t_i is given by

$$f(t_i; s, k, T) = t = s + t_i + T(1-k)\left\lfloor \frac{t_i}{kT} \right\rfloor. \tag{1}$$

Note that (1) associates the points $t_i = nkT$, $n \in \mathbb{N}$ with the beginning of the operation period which for our purposes is a conservative assumption.

3.5 Replication Run Time Optimization

In the previous section we noted that to properly calculate the performance measures of the scenario under consideration it is imperative to keep note of whether a given instant of time belongs to a revenue service interval. It is equally relevant to keep track of the time remaining to enter/leave one such interval. The naive approach to do that would be to implement a subnet that acts as a local clock with two states denoting "revenue/not revenue". Including this clock increases the number of events that are triggered throughout the run of a replication significantly. For the signaling system under consideration a reasonable value for the expected number of functional failures is on the order of 10 [events per year]. On the other hand, the clock keeping track of the revenue service time would trigger twice a day which amounts to approximately 17520 [events per year]. Clearly any strategy that reduces the number of events triggered during the run of a replication will have an impact on the run time of the replication.

The approach we take follows the ideas on which the fault injection simulation method described in (Van Moorsel, Haverkort, and Niemegeers 1991) is based upon. In our case, we are interested in knowing where we are with respect to revenue service only on the restoration period which follows a failure. This is because during the periods of time where the system is operating and no failure has occurred we already know how to calculate the performance measures. Figure 6 shows this idea schematically.

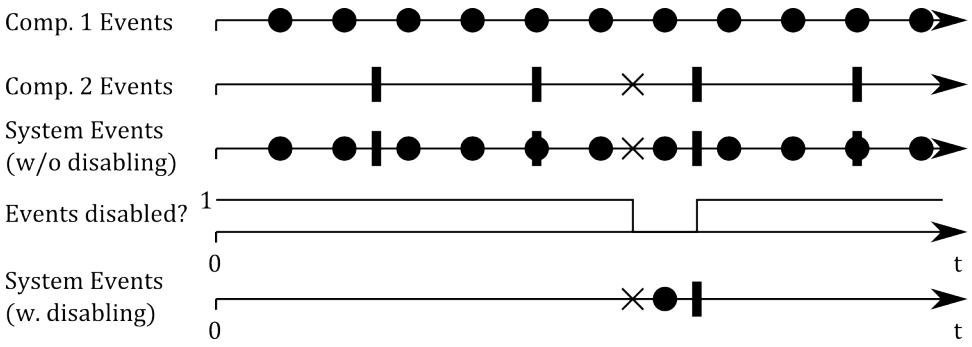

Figure 6: Selective Event Simulation. x denotes a failure event, bars and circles are inspections or preventive maintenance events for each component type.

Expanding on this idea we can use information that is known in the model (but unknown in reality) to further reduce the number of events triggered during the simulation of a replication. Inspections allow the detection of failures that were not detected by the on-board equipment. This can be for instance due to redundancy of the particular subsystem/component that has failed. In the simulation environment we always know when an undetected (by the system) failure has occurred. Hence, we can set up the model such that inspection transitions are only triggered if an undetected failure has occurred and the system has not been restored yet.

Both of the previous changes will reduce the number of events triggered during the lifespan of a replication, reducing the run-time it at the cost of increasing the complexity of output analysis. A simple example of the possible savings is shown next. Consider the SPN model shown in Figure 7. The model consists of three subnets. The two bottom subnets represent a simplified failure and restoration process of two independent components. The top subnet represents a clock that switches between revenue service interval and not revenue service daily (representing the system's operation profile). In an actual implementation another state for each component subnet would be added representing the component being in an off-service state (following the clock). Two cases are run. The first one (denoted "naive") corresponds to the one shown in Figure 7. The second one (denoted "selective event disabling") disables the clock (top subnet) while both components are operational. This can be done using either inhibitor arcs or a combination of guards and predicates. A Monte Carlo terminating simulation of the two cases was run. Each experiment ran 100000 replications and the terminating time was 30 years. The results are shown in Table 1.

Table 1: Selective event disabling run time comparisons.

Case	Run time [s]
Naive	179
Sel. event disabling	14

A 12.8X speed-up is obtained in this case. This provides more than enough incentive to further explore this modeling technique for the case of reliability measures in noncontinuous operating systems.

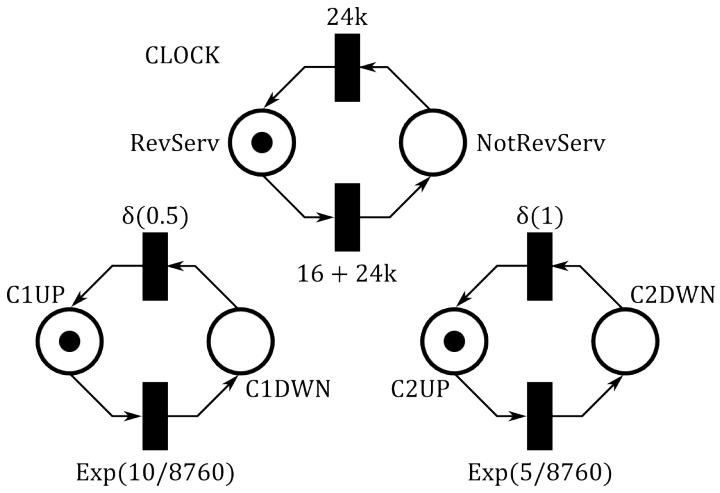

Figure 7: Selective Event Simulation Example.

4 CASE STUDY

In this section a case study is presented. It highlight the insight that can be obtained from including logistic support considerations in the modeling of maintenance process and at the same time show how the non-Markovian SPN framework allows capturing such activities and their effects with little extra effort. The analysis focuses on the effect of the logistic support elements (spare parts and workers) on the interval availability of the system.

4.1 System Description

The system is composed by five identical products (P1) located along the track (see Figure 8). The product is composed by two submodules in a 1oo2 architecture. Each submodule is composed by 12 elements in a series configuration. Of these elements one of them corresponds to two identical components in a 2oo3 architecture (see Figure 9). To perform the restoration operations both maintainers and spare parts need to be available. There is one warehouse in the system where both resources (workers and spares) are located.

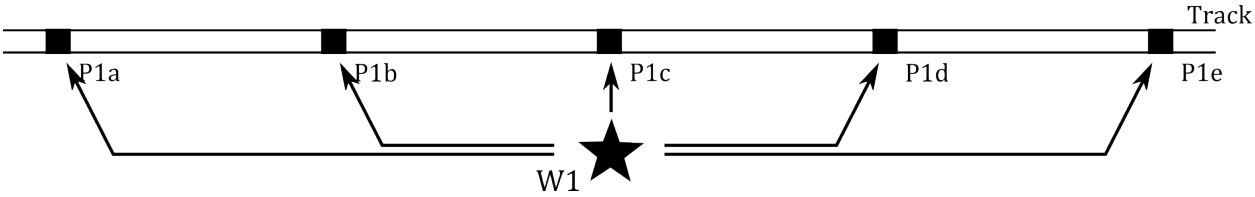

Figure 8: System Layout.

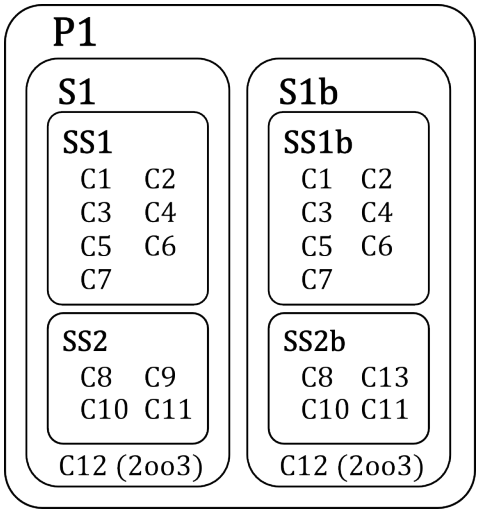

Figure 9: Product breakdown structure of P1.

4.2 Number of Workers Needed to Maintain a Given System

In this case we consider the top configuration shown in Figure 8 to be the system configuration (with respect to its logistic support). The question that needs to be answered is what is the number of workers needed to maintain the given system. The solution procedure is to perform a simulation experiment with different numbers of workers. The result from the simulation will be the interval availability of the system during the life cycle. A number of maintenance operators will be considered sufficient if it is the least among all the numbers such that the availability target is satisfied with a 95% probability. Three experiments using one, two and three maintenance operators are run.

The simulation experiment was performed three times with different seed values for the pseudo random number generator. An experiment consists of 1E5 replications of the life cycle of the system (terminating simulation at 30 years). The output from each experiment was the a point estimate and 95% confidence interval for the expected time spent by the system in the up state. The interval availability is calculated as the ratio of this value and the life cycle of the system. The failure data for each component is exponential, the travel times and the restoration times follow a uniform distribution. For demonstration purposes, the failure rates of each component are scaled up up with respect to their real values in order to increase the number of observed failures during the experiment. The travel times are proportional to the distance between the warehouse and the location of the equipment being restored.

The final model consists of approximately 2200 places, 2200 transitions and 100 tokens. The machine used to run the experiments has a dual core Pentium processor. The average run time for each experiment was 25 [min].

The results of the experiment are summarized in Table 2. For this case it seems that one maintenance operator is enough to satisfy an interval availability target of 0.98. Since the point estimator of the interval availability does not exhibit a significant difference between the three possible number of maintenance operators it can be concluded that for this system the number of maintenance operators does not have a major influence on the interval availability of the system. This was somewhat expected because the number of failures throughout the life of the system is still sparse enough.

5 CONCLUSION

Including logistics considerations into the model of the maintenance process allows the maintenance policy planner to answer questions related to the effect of shared resources on the dependability targets of the system.

Table 2: Simulation experiment results.

Number of Operators	Estimated Interval Availability	One-sided 95%CI
1	0.9805	$(0.9800, 1)$
2	0.9809	$(0.9804, 1)$
3	0.9809	$(0.9805, 1)$

In this work it has been shown how the non-Markovian stochastic Petri net framework is appropriate to handle in a straightforward way all of the relevant items present in the process being modeled.

Questions of simulation optimization, model reusability, . . . , etc. have not been discussed but are clearly issues that will require adequate consideration when the model of the full-scale system is created. From the example case it was seen that although the modeling methodology allows for a simple implementation of different maintenance policies of a given system the size of the resulting models could make its simulation impractical (a real signaling system is composed by hundreds of products like the ones modeled in the example case). Furthermore, the run time for the experiment, the width of the obtained confidence intervals and the typical interval reliability targets of the applications make the exploration of variance reduction techniques (such as importance sampling) a necessity.

REFERENCES

Asmussen, S., and P. W. Glynn. 2010. *Stochastic Simulation: Algorithms and Analysis*. Stochastic Modelling and Applied Probability. Springer.

Barlow, R. E., and F. Proschan. 1965. *Mathematical theory of reliability*. John Wiley, New York.

Bobbio, A., A. Puliafito, M. Telek, and K. Trivedi. 1998. "Recent developments in non-Markovian stochastic Petri nets". *Journal of Circuits Systems and Computers* 8 (1): 119–158.

Chang, K. P., D. Chang, and E. Zio. 2010. "Application of Monte Carlo Simulation for the Estimation of Production Availability in Offshore Installations". In *Simulation Methods for Reliability and Availability of Complex Systems*, Chapter 8, 233 – 252. Springer.

IEC 2005. "IEC 60300-3-3:2005 Dependability Management, Part 3-3: Application guide - Life cycle costing".

Kobbacy, K. A., and D. P. Murthy. (Eds.) 2008. *Complex System Maintenance Handbook*. Springer Series in Reliability Engineering. Springer.

Nicola, V. F., P. Shahabuddin, and M. K. Nakayama. 2001, September. "Techniques for Fast Simulation of Models of Highly Dependable Systems". *IEEE Transactions on Reliability* 50 (3): 246–264.

Shaikh, A., and A. Mettas. 2010. "Application of Reliability, Availability, and Maintainability Simulation to Process Industries: a Case Study". In *Simulation Methods for Reliability and Availability of Complex Systems*, Chapter 8, 173 – 197. Springer.

Tu, F., S. Ghoshal, J. Luo, G. Biswas, M. S., L. Jaw, and K. Navarra. 2007, March. "PHM Integration with Maintenance and Inventory Maagement Systems". In *Aerospace Conference, 2007 IEEE*.

Van Moorsel, A., B. Haverkort, and I. Niemegeers. 1991. "Fault Injection Simulation: A variance reduction technique for systems with rare events". In *First International Workshop on Performability modeling of Computer and Communication Systems*.

Volovoi, V. 2007. "System-Level Maintenance Policies via Stochastic Petri nets with Aging Tokens". In *Annual Reliability and Maintainability Symposium 2007, RAMS '07*.

Volovoi, V., and D. K. Peterson. 2011, December. "Coupling Reliability and Logistical Considerations for Complex System of Systems using Stochastic Petri Nets". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 1751 – 1762. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Zille, V., C. Brenguer, A. Grall, and A. Despujols. 2010. "Simulation of Maintained Multicomponent Systems for Dependability Assessment". In *Simulation Methods for Reliability and Availability of Complex Systems*, Chapter 12, 253 – 272. Springer.

AUTHOR BIOGRAPHIES

PIERRE DERSIN, born in Belgium in 1953, graduated from the Massachusetts Institute of Technology (MIT) in 1980 with a Ph.D. in Electrical Engineering after receiving a Master's degree in O.R. in 1976, also from MIT. He worked in the late seventies and early eighties on reliability of large electric power networks, as part of the Large Scale System Effectiveness Analysis Program sponsored by the US Department of Energy, from MIT and Systems Control,Inc. After some time with FABRICOM (Belgium and U.S), involved with fault diagnostic systems for factory automation, he joined ALSTOM Transport in 1990. With ALSTOM Transport, he has occupied several positions, mainly involved with RAMS and Maintenance, including R&D Manager of the Service business, and is now RAM Director in ALSTOM Transport's Information Solutions (i.e. railway signaling and communication) product line .He has contributed a number of communications and publications in conferences and journals (including IEEE, ESREL '92, the 1995 RAMS Symposium and French Lambda-Mu conferences),in the fields of RAMS, automatic control and electric power systems. His email address is pierre.dersin@transport.alstom.com.

RENÉ C. VALENZUELA is a Ph.D. candidate in the School of Aerospace Engineering at Georgia Institute of Technology. He holds Masters and Bachelors degrees in Mechanical Engineering from Universidad de Concepción, Chile. His research interests include simulation modeling and analysis of maintenance process and probabilistic risk assessment of aerospace systems. His email address is rene.valenzuela@gatech.edu