Proxel-Based Performability Analysis of Non-Markovian Phased-Mission Systems

Sanja Lazarova-Molnar
College of Information Technology
United Arab Emirates University
Al Ain, United Arab Emirates
sanja@uaeu.ac.ae

Abstract— **Phased-mission system is a system whose mission consists of a sequence of non-overlapping phases described by different models. The focus of this paper is to apply the proxel-based simulation method to performability analysis of phased-mission systems. Phased-mission systems have been researched for a long time and they encompass a large number of systems in industry. Most of the existing methods impose some limitations on the systems they can analyze or face loss of accuracy. The proxel-based method allows for a general class of phased-mission systems to be analyzed without imposing the typical limitations on the models or sacrificing accuracy. Lower level components of systems can be described using any formalism, i.e. Petri nets or state-transition diagrams. We provide modification of the original algorithm of the proxel-based method, adapted to be applied to phased-mission systems. In addition, we present results from experiments performed using the presented approach.**

Keywords-proxels; performability; phased-mission systems

I. INTRODUCTION

This paper proposes the use of the proxel-based simulation method [1, 2] for reliability analysis of phased-mission systems [3]. Phased-mission systems have been widely researched because they encompass a wide class of systems from the real world. A phased-mission system is a system observed as a series of sequential phases which are non-overlapping and in which the system functions differently; thus leading to different model descriptions for each phase. This includes failure rates of components, system configuration, as well as success criteria. In particular, autonomous systems are typically described and analyzed as phased-mission systems [4, 5], where each phase can be expressed as, e.g., a fault tree.

The method that we apply for performability analysis of phased-mission systems is a simulation method based on discrete-time Markov chains [6-8], consequently leading to highly accurate and reliable results. The proxel-based method can be observed as a hybrid method which is not as exact as analytical methods, nor does it employ random numbers as discrete-event simulation (otherwise known as Monte-Carlo simulation) does. The simulation process is conducted within one simulation run that follows all possible paths of one model along a discretized timeline. Result of a proxel-based simulation is approximation of the transient solution of the model. Furthermore, the method can handle general distribution functions, rather than only exponential.

As a result, our approach analyzes arbitrary non-Markovian phased-mission systems providing higher accuracy than discrete-event simulation. This includes the typical challenging classes of systems:

- systems with non-exponential activities,
- systems with random phase durations,
- systems with different repair policies (i.e. containing both non-repairable and repairable components),

adding to them the following special cases:

- systems with non-sequential phases, and
- systems with event-triggered phase transitions

aimed to support realistic modeling and hinder over-simplification of models.

To describe our approach, we provide an example model of a phased-mission system that encompasses all cases. The example model was chosen to be simple to support the comprehension of our approach.

In the following section we provide the essential background for explaining our approach.

II. PRELIMINARIES

In the following we briefly explain the preliminaries of our approach. We start by introducing the concept of performability, followed by the concept of a phased-mission system and finally we briefly describe the proxel-based method.

A. Performability

Performability is a concept that combines both reliability and perfomance of a system [9-11], thus evaluating the performance of one system with respect to its reliability, i.e. in spite of its failures.

Performance alone is a measure of how efficient one system is (can be measured as throughput, response time, etc.), whereas reliability is a measure of its ability to function correctly over a specific period of time i.e. how reliable it is. Reliability (or dependability) of a discrete and stochastic system S can be expressed mathematically in the following form:

$$R(t) = Pr(S \text{ operates correctly over a time interval } [0, t)).$$

If we now denote the lifetime of a system by L, and F is the distribution function of L, then the reliability of the system, R, at time t can be computed as

$$R(t) = Pr(L > t) = 1 - F(t).$$

Using common words, performability modeling tries to evaluate and answer the following question: "*How much work will be done (lost) in a given interval by a given system including the effects of its failures and repairs?*" and find the function that describes it. The accomplished work is then the accumulated performance over the specified time interval, given that the performance changes over time depending on the operativeness of the separate components of the system i.e. the performability.

Performance of a system is typically measured using a reward function $P(DS, \tau)$ which evaluates its efficiency in each of its discrete states (DS) and is also dependent on the time τ that the model has spent in it.

Typically, the goal of performability modeling is to obtain some the following measures, depending on their relevancy:

- *expected performance* of a system at certain point in time, taking in account effects of failures, repairs and other possible conditions of the system,
- *time-averaged performance* of a system over time interval (0, *t*),
- *amount of work accomplished* over time interval (0, *t*), etc.

We have successfully applied the proxel-based method to performability analysis of discrete stochastic models for obtaining the afore described measure [12]. In this paper we show how they can be obtained for the case of phased-mission systems. We use an example model to describe and test our approach.

B. Phased-mission systems

There is already a comprehensive theory on phased-mission systems as they have been actively investigated over the last three decades. Therefore, there are a number of classifications with regards to different aspects of phased-mission systems, summarized briefly in the following [13]:

- Static vs. dynamic; with regards to whether the failure of the mission depends only on the combination of its events, or also on the sequence of their occurrence.
- Repairable vs. non-repairable; with regards to whether the components in the system are repairable or not.
- Coherent vs. non-coherent; depending on whether the systems performance declines/increases with each components failing/repairing, or this relation nonexistent.
- Series vs. combinatorial; in a series system the whole system fails if a phase fails, whereas in a combinatorial one the failure of the system is expressed as a logical combination of the phase failures.
- Sequential vs. dynamic choice of mission phases; depending on the way the phases are traversed by the system to accomplish its goal.

The simplest analytical approaches for reliability analysis are the combinatorial approaches [3, 14]. However, they also impose many restrictions on the models, i.e. they exclude dynamic behavior, transient errors or imperfect recovery. Furthermore, combinatorial approaches assume that each phase of the mission affects the overall reliability at every point in time.

Another class of methods for reliability analysis of PMS are the Fault Tree based [15, 16] which are also related to the combinatorial approaches. The limitation of these approaches is that eventually they handle only memoryless events, i.e. events described using exponential distribution function.

Many of the other methods (non-analytic) assume Markov models for describing the system as well [17-19]. This again implies modeling only exponentially distributed activities.

Recently, there has been a Petri net-based approach that handles general non-exponentially distributed activities [20]. This approach is, however, based on discrete-event simulation and thus different to the one that we present, which is a deterministic one. In addition, almost all of the existing approaches focus on reliability exclusively, i.e. excluding evaluating performance of systems.

C. The Proxel-Based Method

The proxel-based method [1, 2] is a relatively novel simulation method, whose underlying stochastic process is a discrete-time Markov chain [8]and implements the method of supplementary variables [21]. The method, however, is not limited to Markovian models. On the opposite, it allows for a general class of stochastic models to be analyzed regardless of the involved probability distribution functions. In other words, the proxel-based method combines the accuracy of numerical methods with the modeling power of discrete-event simulation.

The proxel-based method is based on expanding the definition of a state by including additional parameters which trace the relevant quantities in one model through a previously chosen time step. Typically this includes, but is not limited to, age intensities of the relevant transitions. The expansion implies that all parameters pertinent for calculating probabilities for the future development of a model are identified and included in the state definition of the model.

Proxels (stands for probability elements), as basic computational units of the algorithm, follow dynamically all possible expansions of one model. The state-space of the model is built on-the-fly, as illustrated in Figure 1, by observing every possible transiting state and assigning a probability value to it (*Pr* in the figure stands for the probability value of the proxel). Basically, the state space is built by observing all possible options of what can happen at the next time step. The first option is for the model to transit to another discrete state in the next time step, according to the associated transitions. The second option is that the model stays in the same discrete state, which results in a new proxel too. Zero-probability states are not stored and, as a result, no further investigated. This implies that only the truly reachable (i.e. tangible) states of the model are stored and consequently expanded. At the end of a proxel-based simulation run, a transient solution is obtained which outlines the probability of every state at every point in time, as discretized through the chosen size of the time step. It is important to notice that one source of error of the proxel-based method comes from the assumption that the model makes at most one state change within one time step. This error is elaborated in [2].

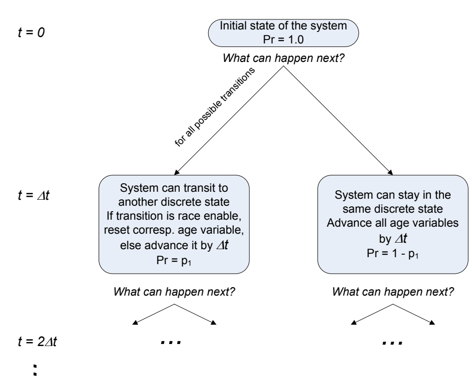

Figure 1. Illustration of the development of the proxel-based simulation algorithm

Each proxel carries the probability of the state that it describes. Probabilities are calculated using the instantaneous rate function (IRF), also known as hazard rate function. The IRF approximates the probability that an event will happen within a predetermined elementary time step, given that it has been pending for a certain amount of time τ (indicated as 'age intensity'). It is calculated from the probability density function (f) and the cumulative distribution function (F) using the following formula:

$$\mu(\tau) = \frac{f(\tau)}{1 - F(\tau)}$$

As all state-space based methods, this method also suffers from the state-space explosion problem [22], but it can be predicted and controlled by calculating the lifetimes of discrete states in the model. In addition, its efficiency and accuracy can be further improved by employing discrete phases and extrapolation of solutions [23]. More on the proxel-based method can be found in [2].

In the following we show how the proxel-based method can be adapted to handle phased-mission systems without imposing the typical limitations on the models.

III. The Proxel-Based Method For Phased-Mission Systems

The motivation behind the application of the proxel-based method to reliability analysis of phased-mission systems is the flexibility of the method that allows all relevant quantities to be easily traced and included in the state definition. We are confident that our approach will contribute to obtain more accurate performability measures for new classes of phased-mission systems. This approach may prove suitable for analyzing phased-mission systems where the aging of states needs to be recorded as well (i.e. not memoryless). The proxel-based method lifts the typical constraint of having only memoryless events in the model.

The modified format of a proxel that describes every possible state in a phased-mission system is the following:

(Phase Data, State Data, Route, Probability).

where

Phase Data = (Phase, Phase Age Intensity), and

State Data = (Discrete State, State Age Intensity Vector).

It is evident that the state definition needs to be supplemented with phase information which in turn tracks the age of the phase that the model is in.

The input to the algorithm specifies the transitions from one phase to another besides the descriptions for the models in each phase. This is not trivial when phase models contain different sets of states. In this paper, as an initial attempt to apply the proxel-based method to phased-mission systems, we focus on the case when the discrete state spaces of the phase models are equivalent.

A. The Algorithm

Inputs to the modified proxel-based algorithm are the models of each phase of the phased-mission system along with the basic simulation parameters (maximum simulation time and size of the time step). The algorithm works by checking the phase model $M(P)$ for the phase described in each proxel in the data structure *list*(*sw*). Depending on the model description for that phase, successor proxels are calculated and correspondingly stored in the data structure *list*(1-*sw*). This process is described in lines 6-10. Further it checks the phase transition model that describes the transitions among phases and generates proxels that signify transition to other phases, as described in lines 11-14. Finaly, the proxel that describes the model staying in the same discrete state and phase is generated and stored, a shown in line 15.

The variable *sw* is used for alternating between the two data structures. The number of time steps is t_{max} /Δt where t_{max} is the maximum simulation time and Δt is the size of the time step.

The statistics that are recorded, as shown in line 5, depend on the goal of the analysis. The standard result is a transient solution of each state separately, for each phase. Many other statistics can be deduced from this solution. There (line 5) is also the place where the performability statistics are collected and processed as described in the model. If performance rates are constant, then the calculation of performability measures is trivial and can be fully deduced from the transient solution of the system. Its calculation is more complex if that is not the case and thus has to be calculated explicitly (straightforward for the proxel-based method). In this paper we focus on the case where the performance rates are constant.

Input: Δt, t_{max}, *initial_proxel*
0. *current_proxel = initial_proxel*
1. *sw = 0;*
2. **for** $i = 0$ **to** $t_{max}/\Delta t$
3. *sw* = 1 – *sw;*
4. **foreach** *proxel* **in** *list(1 - sw)*
5. *record statistics;*
6. **foreach** *transition T(DS) in the model M(P)* **do**
7. *advance age variable associated with T(DS);*
8. *read the discrete state corresponding with T(DS);*
9. *calculate successor proxel and store it in list(sw);*
10. **end**
11. **foreach** *subsequent phase P(P) from the PTM*
12. *advance age variable associated with P(P);*
13. *calculate successor proxel and store it in list(sw);*
14. **end**
15. *generate the proxel that represents no changes in the discrete state and the phase and store in list(sw);*
16. **end**
17.**end**

The algorithm uses the following symbols:

- *DS* – discrete state that the model is in
- *T*(*DS*) – transition associated with discrete state DS
- *P* – phase that the model is in
- *M*(*P*) – model description in the phase *P*
- *PTM* – phase transition model (as described in Figure 3)

In the proceeding subsection along with an example, we explain how successor proxels are calculated. More details on this process and the corresponding algorithm can also be found in [2]. The major modification of the algorithm is that it traverses through the phases too and updates the phase age intensity as well.

B. Example

Our example model describes a three-phase system of two processors. Each processor can be in one of the two states: failed (down-D) or operating (up-U). This results into a state space with 4 discrete states: UU (both processors up), UD (first processor up and the second one down), DU (first processor down and the second one up), and DD (both processors down).

In each phase the system is described by a different model, as shown in Figure 2. In the first phase processors fail according to two distribution functions, whereas in the second phase the rates are described using another two distribution functions. In the third phase processors cannot be repaired at all. This implies that the number of transitions in the third phase drops to 4 from 8 in the preceding two phases. Across all phases, transitions that describe failures of processors are race enable. This is, however, not a limitation to the method, but an assumption made to suit the model description.

The set of discrete states across the three phases remains the same. This implies that the transitions of the states from phase to phase in the example phased-mission system are straightforward and, therefore, do not require special matching. The transitions from one phase to another are described by the model shown in Figure 2. From this figure it is evident that the phases in the example model are not sequential, i.e. the system from the second phase can transit either back to the first one or to the third phase. In addition to the diagram, a phase transition can also be event triggered, i.e. in our example the system can transit from *Phase 1* to *Phase 2* with the transition *Fail1.1* from the discrete state UD. This is the transition that describes the failure of the first processor when the second one has already failed.

The probability distribution functions that describe the activities are also shown in Figure 2. Each transition is denoted by the phase number and the processor number, e.g. Repair2.1 is the repair of the first processor in phase two. In the last phase there are only failure transitions because the processors in this phase are non-repairable.

The reward rates that describe the performance of the system in each of the discrete sates are denoted by the work units accomplished per time unit. Both processors have different performance rates. The first one has a rate of 10, whereas the second one has a rate of 1. Thus, the reward rates for the four states are the following:

- r(UU) = 11
- r(DU) = 1
- r(UD) = 10
- r(DD) = 0

We assume that performances do not differ across phases and that they are constant. Again, this assumption is not a limitation of the method, but one made for simplicity reasons only. We plan to further extend our approach to deal with variable performance rates that change across phases. This is supposed to be a very straightforward process due to the flexible nature of the proxel-based method.

C. *Details of the Experiment*

The proxel-based method departs from the initial proxel definition that describes the initial state of the phased-mission system, i.e.

((1, 0), (UU, 0, 0), 1.0).

The first component of the proxel, i.e. (1, 0) describes the phase data. In this case it describes that the model is in phase 1 and has been there for 0 time. The second element (UU, 0, 0) describes the state of the model within the phase, i.e. the model is in discrete state UU (both processors operating) and the clocks for the failure transitions for both of them are zeros. The last element, i.e. 1.0 describes the probability for the described configuration of the model.

For simplicity reasons we leave out the route parameter described in the original definition of the proxel because that one is implicitly calculated during the proxel-based simulation run. As shown in the initial proxel, one age variable is needed for tracking the time that the system spends in each phase, and two age variables for tracking the ages of the transitions associated with the two processors. Processors' age variables interchangeably track the ages of the fail/repair transitions, depending on the states that the processors are in. Thus, the initial proxel generates the following proxels:

a) change the phase - ((2, 0), (uu, Δt, Δt), p1),
b) one processor fails - ((1, Δt), (du, 0, Δt), p2),
c) the other processor fails - ((1, Δt), (ud, Δt, 0), p3), and
d) no changes - ((1, Δt), (uu, Δt, Δt), 1.0- p1- p2- p3),

describing four possible state changes of the model, along with the probabilities for their occurrence.

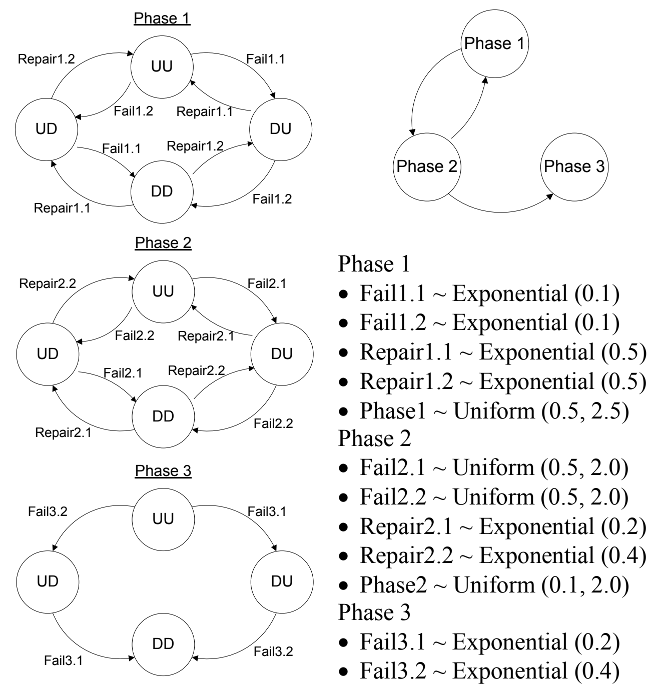

Figure 2. State-transition diagrams of the three phases of the sample phased-mission system and state-transition diagram of the phase transitions along with the input distribution functions

The phase change is described in case (a). For that reason the *phase data* element is set to (2, 0), i.e. the phase is changed to the next one and the zero indicates the age in the new phase which is obviously zero because the transition has just occurred. The *state data* element, (UU, Δt, Δt), contains the same discrete state, because the basic assumption of the proxel-based method is that only one transition can occur within one time step. However, both age intensities that describe the aging of both processors, have been advanced by one time step Δt. This means that the time during which transitions have been enabled in one phase is recorded and passed to the next phase.

Cases (b) and (c) describe failures of each of the two processors. Thus, the phase remains the same and the age variable is increased by Δt, resulting into a *phase data* element (1, Δt). The *state data* element for both cases contains the discrete state that the system has transition to due to the failure of the processor. The age variable that was tracking the aging of the failure for that processor now tracks its repair event. The other age variable is increased by Δt as the corresponding event is still pending.

The last case (d) describes the model residing in the same state; thus all age variables are advanced by a single time step and all other data remains the same. The probability in this proxel is the remainder from the parent proxel's probability, once all sibling proxels' probabilities are subtracted.

Output of this proxel-based simulation for the first three simulation steps is shown in Figure 3. Every line shows the output of the AddProxel function which displays the generated and stored proxels. States are enumerated in the following way: UU as 0, UD as 1, DU as 2, and DD as 3. In the first step we can observe the generation of the previously described proxels. However, one of them is missing and that is the one that describes the change of the phase. This proxel is not generated because the probability for this event is zero.

```
                         INITIAL PROXEL
AddProxel  ((1,  0dt), (0,  0dt,  0dt)), 1.00000e+000)

                             STEP 1
AddProxel  ((1,  1dt), (2,  0dt,  1dt)), 1.00000e-002)
AddProxel  ((1,  1dt), (1,  1dt,  0dt)), 1.00000e-002)
AddProxel  ((1,  1dt), (0,  1dt,  1dt)), 9.80000e-001)

                             STEP 2
AddProxel  ((1,  2dt), (2,  0dt,  2dt)), 9.80000e-003)
AddProxel  ((1,  2dt), (1,  2dt,  0dt)), 9.80000e-003)
AddProxel  ((1,  2dt), (0,  2dt,  2dt)), 9.60400e-001)
AddProxel  ((2,  2dt), (3,  0dt,  1dt)), 1.00000e-004)
AddProxel  ((1,  2dt), (1,  2dt,  1dt)), 9.90000e-003)
AddProxel  ((1,  2dt), (3,  1dt,  0dt)), 1.00000e-004)
AddProxel  ((1,  2dt), (2,  1dt,  2dt)), 9.90000e-003)

                             STEP 3
AddProxel  ((1,  3dt), (3,  2dt,  0dt)), 9.90000e-005)
AddProxel  ((2,  0dt), (2,  2dt,  3dt)), 4.30435e-004)
AddProxel  ((1,  3dt), (2,  2dt,  3dt)), 9.37057e-003)
AddProxel  ((2,  3dt), (3,  0dt,  2dt)), 9.90000e-005)
AddProxel  ((2,  0dt), (1,  3dt,  2dt)), 4.30435e-004)
AddProxel  ((1,  3dt), (1,  3dt,  2dt)), 9.37057e-003)
AddProxel  ((1,  3dt), (2,  0dt,  3dt)), 9.60400e-003)
AddProxel  ((1,  3dt), (1,  3dt,  0dt)), 9.60400e-003)
AddProxel  ((2,  0dt), (0,  3dt,  3dt)), 4.17565e-002)
AddProxel  ((1,  3dt), (0,  3dt,  3dt)), 8.99435e-001)
AddProxel  ((2,  3dt), (3,  0dt,  1dt)), 9.80000e-005)
AddProxel  ((2,  0dt), (1,  3dt,  1dt)), 4.26087e-004)
AddProxel  ((1,  3dt), (1,  3dt,  1dt)), 9.27591e-003)
AddProxel  ((2,  0dt), (3,  2dt,  1dt)), 4.34783e-006)
AddProxel  ((1,  3dt), (3,  2dt,  1dt)), 9.56522e-005)
AddProxel  ((1,  0dt), (3,  1dt,  2dt)), 2.65644e-012)
AddProxel  ((2,  3dt), (3,  1dt,  2dt)), 1.00000e-004)
AddProxel  ((1,  3dt), (3,  1dt,  0dt)), 9.80000e-005)
AddProxel  ((2,  0dt), (2,  1dt,  3dt)), 4.26087e-004)
AddProxel  ((1,  3dt), (2,  1dt,  3dt)), 9.27591e-003)
```

Figure 3. Output of the proxel-based simulation for the model from Figure 2

IV. EXPERIMENTS

A. *Experimental Environment*

The experiment was run on a standard workstation with an Intel Core2Duo Processor at 2.0 GHz and 1 GB RAM. The choice for Δt was 0.1 and the simulation was run up to time t = 6. This implies that the number of simulation steps was 60.

The computation time for this experiment was ca. 5 seconds. In the following we present the results, i.e. the statistics that were calculated during this simulation experiment.

B. *Results and Interpretation*

We start by showing the transient solutions of each of the four discrete states across the three phases. Therefore, in Figures 4 (a, b, c, and d), we show the transient solutions of the discrete states UU, UD, DU and DD, correspondingly. The abrupt changes in some of the functions are because of the use of uniform distribution functions which are known to have finite support. This implies that events can happen only within a certain time interval, outside of which the probability for their occurrence drops to zero.

In Figure 5 the performability of the system is charted as a performance rate of the complete system. From this figure we can conclude that the system degrades in performance with time. This behavior is expected because processors in the final phase are non-repairable. In Figures 6 and 7, the other two performability measures are charted, i.e. amount of work accomplished and time-averaged performance, correspondingly. All three measures are related and can be derived from each other.

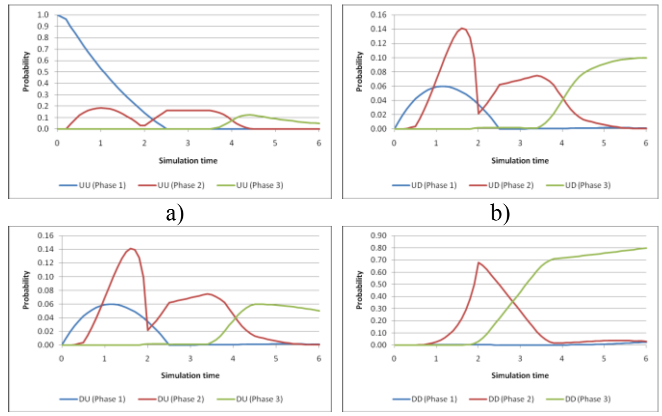

Figure 4. Transient solution of the states: UU (a), UD (b), DU (c) and DD (d) across the three phases

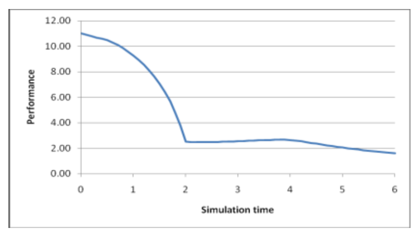

Figure 5. Performance of the sample phased-mission system.

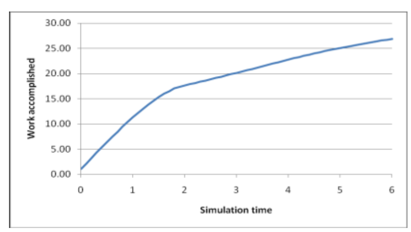

Figure 6. Amount of work accomplished of the sample phased-mission system.

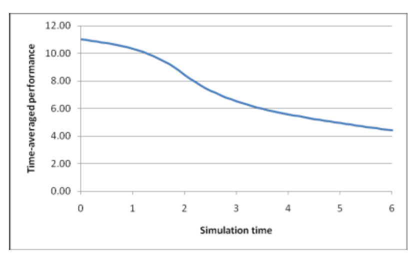

Figure 7. Time-averaged performance of the sample phased-mission.

All figures presented in this section show the power of the proxel-based method in terms of completeness of the results that it can deliver. However, this is only a fraction of the type of statistics the proxel-based method can record. The fact is that a high variety of statistics can be gathered, and all those from a single simulation run because the proxel-based method generates and observes all possible paths a discrete stochastic model can take.

V. DISCUSSION

The proxel-based method has shown to be very flexible and promising in analyzing phased-mission systems. In addition, it delivers complete results that fully depict the behavior and performability of one system. We have identified the following special cases of phased-mission systems that could be simulated and analyzed using the proxel-based method:

- Systems with phase transitions described by an arbitrary state transitions diagram (i.e. allowing for a dynamic choice of mission phases)
- Systems that involve phase transitions triggered by events
- Systems with discrete states associated with different performance rates (or functions) depending on the phase that the model is in
- Systems with phases described by different state spaces (also different numbers of discrete states)

These situations can be analyzed without limiting the activities to be exponentially distributed. This forms part of our future research. In addition to the flexible definition of the phased- mission systems that can be analyzed, the significance of the described approach is that the simulation is deterministic, i.e. the results that are obtained are more accurate and reliable compared to the ones obtained using discrete-event simulation.

VI. SUMMARY AND OUTLOOK

We have developed a new approach to performability analysis of phased-mission systems. The advantage of our approach is that it imposes the least number of constraints on the models. In particular it lifts the constraints of having only exponentially distributed activities, as well as deterministic durations of phases. In addition to this, the method delivers complete transient solution of the model, thus supporting the processes of decision making related to the system.

It is part of our future research to explore the full potential of the proxel-based method with regards to performability analysis of phased-mission systems. This will include analyzing phased-mission systems with complex phase transition models, as well as all special cases described in the previous section.

REFERENCES

[1] G. Horton, "A new paradigm for the numerical simulation of stochastic Petri nets with general firing times," *Proceedings of the European Simulation Symposium*, 2002.

[2] S. Lazarova-Molnar, "The Proxel-Based Method: Formalisation, Analysis and Applications," in *Faculty of Informatics*, vol. Ph.D. Magdeburg: University of Magdeburg, 2005.

[3] J. D. Esary and H. Ziehms, "Reliability analysis of phased missions," presented at Conference on Reliability and Fault Tree Analysis, 1975.

[4] R. Remenyte-Prescott, J. D. Andrews, and P. W. H. Chung, "An efficient phased mission reliability analysis for autonomous vehicles," *Reliability Engineering & System Safety*, 2009.

[5] D. R. Prescott, J. D. Andrews, and C. G. Downes, "Multiplatform phased mission reliability modelling for mission planning," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 223, pp. 27-39, 2009.

[6] E. Çinlar, *Introduction to stochastic processes*: Prentice-Hall, 1975.

[7] V. G. Kulkarni, *Modeling and Analysis of Stochastic Systems*: Chapman & Hall/CRC, 1995.

[8] W. J. Stewart, *Introduction to the Numerical Solution of Markov Chains.*: Princeton University Press, 1994.

[9] B. R. Iyer, L. Donatiello, and P. Heidelberger, "Analysis of performability for stochastic models of fault-tolerant systems," *IEEE Transactions on Computers*, vol. 100, pp. 902-907, 1986.

[10] R. M. Smith, K. S. Trivedi, and A. V. Ramesh, "Performability analysis: measures, an algorithm, and a case study," *IEEE Transactions on Computers*, vol. 37, pp. 406-417, 1988.

[11] B. R. Haverkort, *Performability modelling: techniques and tools*: Wiley, 2001.

[12] S. Lazarova-Molnar and G. Horton, "Description Framework for Proxel-Based Simulation of a General Class of Stochastic Models," *SCSC'05, Philadelphia, PA, USA*, 2005.

[13] K. B. Misra, *Handbook of Performability Engineering*: Springer, 2008.

[14] X. Zang, N. Sun, K. S. Trivedi, and N. C. Durham, "A BDD-based algorithm for reliability analysis of phased-missionsystems," *IEEE Transactions on Reliability*, vol. 48, pp. 50-60, 1999.

[15] J. K. Vaurio, "Fault tree analysis of phased mission systems with repairable and non-repairable components," *Reliability Engineering and System Safety*, vol. 74, pp. 169-180, 2001.

[16] R. A. La Band and J. D. Andrews, "Phased mission modelling using fault tree analysis," *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, vol. 218, pp. 83-91, 2004.

[17] M. Alam and U. M. Al-Saggaf, "Quantitative reliability evaluation of repairable phased-mission systems using Markov approach," *IEEE Transactions on Reliability*, vol. 35, pp. 498-503, 1986.

[18] J. B. Dugan, "Automated analysis of phased-mission reliability," *IEEE Transactions on Reliability*, vol. 40, pp. 45-52, 1991.

[19] A. K. Somani, J. A. Ritcey, and S. H. L. Au, "Computationally-efficient phased-mission reliability analysis forsystems with variable configurations," *IEEE Transactions on Reliability*, vol. 41, pp. 504-511, 1992.

[20] Y. Mo, D. Siewiorek, and X. Yang, "Mission reliability analysis of fault-tolerant multiple-phased systems," *Reliability Engineering and System Safety*, vol. 93, pp. 1036-1046, 2008.

[21] D. R. Cox, "The analysis of non-Markovian stochastic processes by the inclusion of supplementary variables," *Proceedings of the Cambridge Philosophical Society*, vol. 51, pp. 433-441, 1955.

[22] F. J. Lin, P. M. Chu, and M. T. Liu, "Protocol verification using reachability analysis: the state space explosion problem and relief strategies," *ACM SIGCOMM Computer Communication Review*, vol. 17, pp. 126-135, 1987.

[23] C. Isensee and G. Horton, "Approximation of Discrete Phase-Type Distributions," *Proceedings of the 38th annual Symposium on Simulation*, pp. 99-106, 2005.