International Journal of Performability Engineering, 10(2), pp. 211-225, March 2014.

A Direct Method for Determining Design and Support Parameters to Meet an Availability Requirement

T. JAZOULI, P. SANDBORN and A. KASHANI-POUR

Center for Advanced Life Cycle Engineering (CALCE), University of Maryland, College Park, MD 20742, USA

Abstract: Customers of systems with high availability requirements are in some cases interested in migrating from buying the actual system to buying the availability of the system through "availability-based" contracts. Availability contracts are a subset of outcome-based contracting mechanisms that include: Power-by-the-Hour and Performance-Based Logistics (PBL).

Discrete event simulation is usually a preferred approach to model and predict the life-cycle characteristics (cost and availability) of large populations of complex real systems managed over long periods of time with significant uncertainties. However, while using discrete event simulation to predict the availability of a system or a population of systems based on known or predicted system design and support parameters is relatively straightforward; determining the design and support parameters that result in a desired availability is generally performed using search-based methods that can become impractical for designing systems with more than a few variables and when significant uncertainties are present. This paper presents a direct method that is not search based and uses an availability requirement to predict the required logistics, design (including reliability) and operation parameters using discrete event simulation in a time forward direction. This paper also addresses managing availability requirements for systems that include prognostics and health management (PHM) strategies. PHM methods are incorporated into systems to avoid unanticipated failures that can potentially impact system safety and operation, result in additional life-cycle cost, and/or adversely affect the system availability.

Key Words: *availability, availability-based contracts, performance-based logistics (PBL), Prognostics and Health Management (PHM).*

1. Introduction

Availability is the fraction of time that a service or a system is functional when it is requested for use or operation. Availability accounts for both the frequency of the failure (reliability) and the ability to restore the service or system to operation after a failure (maintainability). The maintenance ramifications, generally translate into how quickly the system can be repaired upon failure, are usually driven by the logistics management and are directly influenced by the maintenance policy used. The frequency of failure is related to the reliability of the system, i.e., how long it will be operational (i.e., "up") before it fails.

Many types of availability can be measured (e.g., inherent, achieved, operational, etc.) for parts, subsystems and systems. This paper focuses on the operational availability (A_O) since it is commonly used in availability contracts. A_O accounts for all types of maintenance and logistics downtimes. A_O is the ratio of

the accumulated uptime and the sum of the accumulated uptime and downtime:

$$A_O(t_o) = \frac{UT(t_o)}{UT(t_o) + DT(t_o)} \tag{1}$$

where uptime (*UT*) is the total accumulated operational time during which the system is up and running and able to perform the tasks that are expected from it. Downtime (*DT*) is generated when the system is not operating when requested due to repair, replacement, waiting for spares or any other logistics delay time. The summation of the accumulated uptimes and downtimes represents the total operation time for the system. Note that the availability in (1) is not steady-state, it depends on when (during the life of the system) the observation is made, t_o.

Customers of mission, safety and infrastructure critical systems such as avionics, large scale production lines, servers, and infrastructure service providers with high availability requirements are increasingly interested in buying the availability of a system via "availability-based contracting," instead of actually buying the system itself. Availability-based contracts are a subset of outcome-based contracts [1]; where the customer pays for the delivered outcome, instead of paying for specific logistics activities, system reliability management or other tasks. Well known availability and outcome-based contracts and contract mechanisms include the Availability Transformation: Tornado Aircraft Contract–ATTAC [2], PBL – Performance-Based Logistics [3] and Power by the Hour [4].

1.1 Availability Optimization

Availability requirements are usually used as the constraint or an objective function within optimization problems, e.g., [5-9], where the decision parameters can be reliability, preventive maintenance scheduling, system configuration, inventory policies and the objective functions are a derivate of total life-cycle cost (e.g., downtime cost, maintenance cost, holding cost, or a combination of these).

Examples of existing work associated with maximizing a cost-benefit function that combines the accumulated life-cycle costs associated with a specific system's management (e.g., logistics, maintenance, reliability, etc.) and the availability achieved include [6,10-13]. Multi-objective methods can also use discrete event simulation to assess the future costs, availability or redundancy level as their objective function [14]. However, in the majority of optimization works the goal is to maximize the reliability or availability at specific points in time (or minimizing downtime costs). Often these works consider availability as a function with fixed parameters and total life-cycle cost is calculated using simulation based methods (Monte-Carlo, heuristic algorithms) at each decision point. The need for a direct method arises from the fact that, when the objective function is a stochastic function related to the total life cycle, it is not practical to calculate it for an uncertain future at any time step.

The majority of the existing work represents the availability using analytical models based on different fixed-value parameters describing inventory, reliability and maintenance (e.g., [5,8,15]). Some recent contributions have addressed the stochastic nature of different parameters and look at availability as a state for the whole system or its part's and model's transitions to other states via Markov Chains [13,16,17],

state machines [18] and Petri nets [19]. In all of these works there is minimal attention paid to variability of each member of the fleet and the stochastic nature of each parameter (or lack of knowledge of the distributional properties of parameters) as well as shrinking of uncertainty as time progresses. The existing body of work proves the feasibility of availability optimization for simplified life-cycle scenarios and system characteristics. However, when availability needs to be evaluated for large populations of complex systems, over long term support, and when many uncertain design and support parameters are considered, direct use of discrete event simulation is the most flexible approach for modeling and analysis of availability, and performance analysis of different simulation scenarios [20]. The direct approach simulates the reality of operational decision making by focusing on existing data accumulated during operation. This flexibility allows discrete event simulation to be also used for the calculation of the stochastic objective function as an essential step of search-based optimizations [14].

Discrete event simulation is capable of simulating a population of systems through each system instance's unique life cycle. Discrete event simulations do not necessarily require analytical equations to determine the unknown system parameters (or any other quantity of interest) as a function of other known parameters, since the sequence and outcome of the events are generated and accumulated in simulation time. These capabilities become crucial when uncertainties are included, where each sample of the same quantity, i.e., system parameter, could result in a different event outcome, producing a different sequence of events (*sample path*) and results for every system instance.

Availability requirements can be satisfied by running discrete event simulators in the forward direction (forward in time) for many permutations of the system parameters and then selecting the system parameter values that generate the required availability output, e.g., [13,14,21]. There have also been attempts to perform reverse simulation (run discrete event simulators backwards in time) in software and project management applications where task planning is the focus rather than degradation of material [22,23].

The computational burden associated with search-based approaches makes them expensive for real-world problems (particularly for real-time problems with what-if scenarios). These methods also lack accommodating an availability requirement that is represented as a probability distribution. Designing products (and their support logistics) for availability contracts, includes availability optimization but with different assumptions than most existing works. The existing availability optimization approaches described previously in this section all seek to determine system parameters that maximize availability at a specific point in time rather than guaranteeing that a minimum availability requirement is met at all times during the support life of the fleet of systems.

This paper presents a non-search based, non-iterative, discrete event simulator based direct methodology that uses an availability requirement expressed as a probability distribution applied at all times during a system's support life as an input to the process of determining the management of a system (as opposed to an availability output that is a consequence of system management and logistics inputs). The next section describes the problem we are solving and our solution approach. Section 3 provides case study examples that demonstrate the approach and its application to systems that include prognostics and health

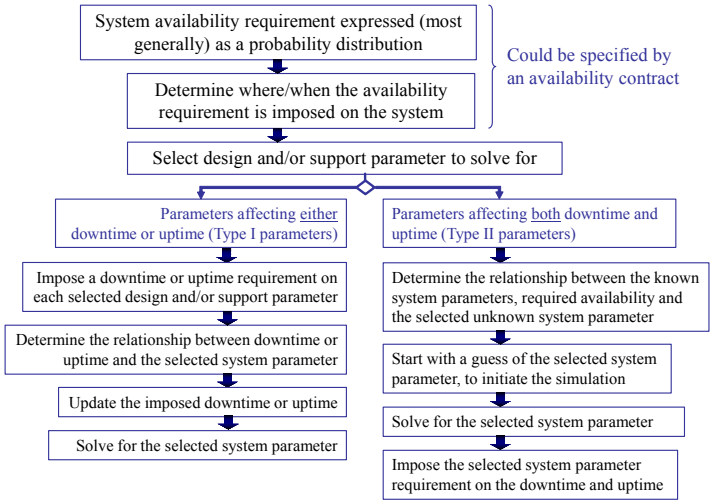


Fig. 1: Solution Approach (note, this paper only addresses Type I parameters, see [24] for a treatment of Type II parameters)

management (PHM). Section 4 discusses shortcomings and possible extensions of the methodology.

2. Solution Approach

Using conventional discrete event simulator models, availability is computed based on known or estimated system reliability, operation and logistics parameters, etc. The goal of the methodology proposed in this paper is to reverse the problem. In other words, to create a methodology that determines the necessary system reliability, operational parameters, and/or logistics management parameters to meet a specific availability requirement. The methodology described in this section is a direct method, i.e., it does not search the parameter space for design parameters that satisfy the availability requirement or iterate the value of parameters in order to satisfy the availability requirement, it solves for the parameters directly.

The methodology introduced uses the availability requirement to compute and impose the necessary uptimes and downtimes throughout the system's life using a discrete event simulator. Then, it uses the imposed uptime and downtime values to solve for the unknown system parameter. Figure 1 shows the steps to formulate and execute the proposed solution. Note there is no feedback mechanism in shown in Figure 1 because this is a direct solution method.

2.1 Interpreting the Availability Requirement and Where it is Imposed

A realistic availability requirement is generally expressed as a probability distribution. Since, even when a contract specifies the availability requirement as a single value, the interpretation of this single value either leads to considering the average availability of a population of systems, i.e., the average of a distribution; or the single value is the minimum availability of all system instances within the population. These interpretations are consistent with the fact that the reliability of the product or system is represented as a probability distribution (or, more accurately a set of probability distributions each corresponding to a different relevant failure mechanism); thus using a logistics management plan that is common across the

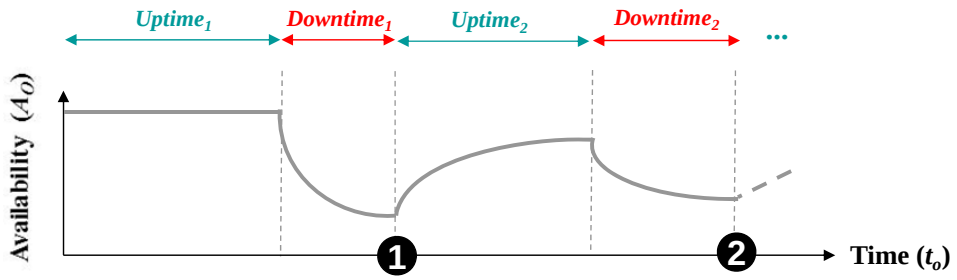


Fig. 2: Availability Variations as a Function of Observation Time

population, each system instance will have a different availability depending on the failure dates of the subsystem instances that comprise it and operational profile variations.[1]

To generalize the model, a conservative approach is adopted by fulfilling an availability requirement at all times during the entire support life of the system. For the remainder of this discussion it is assumed that the availability requirement implies that the operational availability (A_O) should not drop below the availability requirement value at any time during the entire support life (assuming that the system's operation does not start in the down state).

As shown in Figure 2, A_O always decreases during downtimes and increases during uptimes. In other words, A_O reaches its local minimum values at the end of every downtime (e.g., points 1 and 2 in Figure 2). Thus, if the availability requirement is satisfied at the end of every downtime (i.e., minimum A_O values satisfy the requirement), it will be satisfied at all times during the support life of the system. Therefore the approach is to impose the availability requirement at the end of every downtime.

2.2 Determine the Type of System Parameter

In the context of design for availability, two distinct types of system parameters exist: parameters affecting either uptime or downtime (but not both) – Type I parameters, and parameters concurrently affecting both uptime and downtime – Type II parameters. Only Type I parameters are addressed in this paper. See [24] for a treatment of Type II parameters.

For Type I parameters, one of the two quantities (e.g., uptime) is known and can be determined from the known system parameters, while the other quantity is unknown (e.g., downtime) and dependent on the selected unknown system parameter.

For Type II parameters, a change in the value of the parameter could produce a change in both uptime and downtime. Both quantities (uptime and downtime) are dependent on this type of parameter. When one of these system parameters is unknown, then both uptime and downtime are unknown. This means, a relationship between the known system parameters, required availability and the selected unknown system parameter needs to be defined, to solve for this type of parameter.

[1] Another possible interpretation of an availability requirement represented as a probability distribution is that it represents a histogram of the requirements on all subsystems that make up a system. Critical subsystems have higher required availability, while less critical subsystems may have a lower required availability.

2.3 Type I Parameter Solution

The A_O is a function of accumulated uptimes and downtimes. Therefore, imposing an availability requirement means either imposing an uptime requirement while the downtime is generated by the system parameters; or vice versa. Note, for Type I parameters either an uptime or a downtime requirement is imposed, not both. The imposed downtimes or uptimes are computed at defined times or events as a function of the required A_O.

For the remainder of the discussion in this section we will assume that downtimes are imposed to satisfy an availability requirement (all uptimes are ignored). In this case, by generating the appropriate downtimes, only the system parameters that are responsible for generating the downtimes can be computed as outputs; other system parameters that do not influence downtimes should be fixed, i.e., they are used only as inputs. The imposed downtimes are given by,

$$DT_1 = \frac{(UT_1)(1-A_O)}{A_O} \tag{2}$$

$$DT_k = \frac{(1-A_O)\sum_{j=1}^{k} UT_j}{A_O} - \sum_{j=1}^{k-1} DT_j \tag{3}$$

Equations (2) and (3), which are derived from (1), express examples of the relationships that are used to impose the first downtime (DT_1) and a k^{th} downtime (DT_k), respectively. Where A_O represents the specified availability requirement, and UT_j and DT_j correspond to the j^{th} uptime and downtime respectively.

The criterion of imposing either an uptime or downtime requirement is based on the unknown system parameter. For example, if the uptime remains constant while varying a unknown system parameter, the downtime is imposed to meet the availability requirement; and vice versa.

Assuming that the unknown system parameter that we are interested in computing to meet the availability requirement are explicitly related to the downtimes, we need to establish a relationship between the unknown system parameter and the imposed downtimes. Based on this relationship, the unknown system parameter can be computed and updated as soon as the downtime values are determined.

Basically, we have two unknown quantities: the unknown system parameter, and the downtime. Therefore, we need to establish two relationships. Since availability is by definition a function of uptime and downtime; therefore we can solve for downtime as a function of the required availability and uptime, which generates the first relationship; which is used to solve for the downtime. Also, since the types of system parameters addressed here explicitly affect the downtime, then the unknown system parameter can be expressed as a function of downtime; hence the second relationship. To summarize, the downtime is imposed through the availability requirement, then the imposed downtime quantity is used to solve for the unknown system parameter. A similar approach could be adopted if the selected system parameter affects the uptime (instead of downtime).

For demonstration, we assume that the unknown system parameter is explicitly dependent on the required downtime, and the rest of the known system parameters are responsible for generating the uptime.

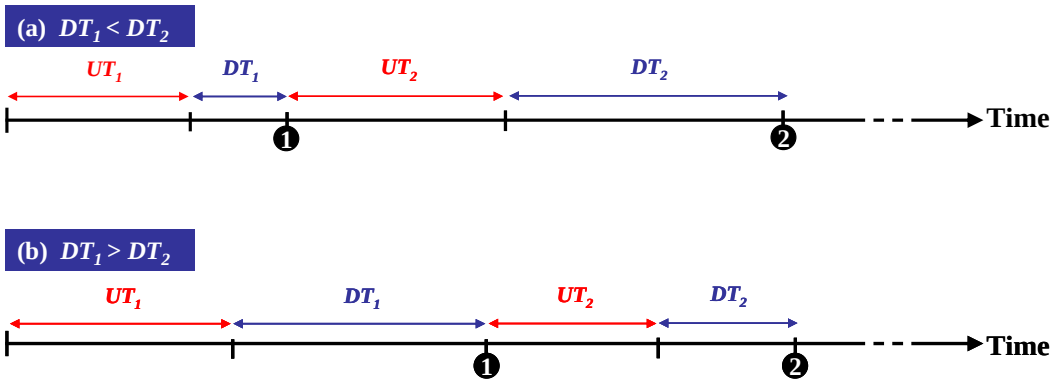


Fig. 3: Downtime Requirement Scenarios

Although the procedure of defining the relationship between system parameter and required downtimes has been discussed, the challenge here is that availability is not determined by a single downtime value, but rather a sequence of downtime values that are not necessarily identical; each resulting in different computed values for the system parameter. As a result, by the end of the simulation (i.e., the end of the system's contracted support period) we could generate multiple values for the same system parameter with no way to determine which value to use to fulfill the availability requirement. Our objective is to select a single downtime value that is the maximum allowable downtime to meet a specific availability requirement, and then use this quantity as a constant downtime value that will fulfill the availability requirement at every point throughout the entire support life. To derive the maximum allowable downtime value, we have explored two scenarios.

The first scenario is illustrated in Figure 3a, where the first imposed downtime (DT_1) duration is shorter than the second imposed downtime (DT_2), where both downtimes have been imposed based on the same availability requirement. In this case, averaging the downtimes would generate an average downtime ($DT_{Average}$) that is larger than DT_1, thus the availability requirement will not be fulfilled at the end of DT_1,

$$\frac{UT_1}{UT_1 + DT_{Average}} < \frac{UT_1}{UT_1 + DT_1} \tag{4}$$

In this situation the maximum allowable downtime duration that the system can accommodate, without failing to satisfy the availability requirement, is constrained by the value of DT_1. Therefore, the DT_2 value should be substituted for DT_1.

The second scenario is illustrated in Figure 3b, where DT_1 is larger than DT_2. In this case, averaging the two downtimes would generate a $DT_{Average}$, which is smaller than DT_1, thus the availability requirement will be fulfilled at the end of DT_1,

$$\frac{UT_1}{UT_1 + DT_{Average}} > \frac{UT_1}{UT_1 + DT_1} \tag{5}$$

Notice that the availability requirement at any specific time includes all accumulated previous downtimes. Thus, when using the average value at the end of DT_2, the availability requirement will still be satisfied since the accumulated averages are just the accumulated downtimes, i.e., summation of DT_1 and DT_2.

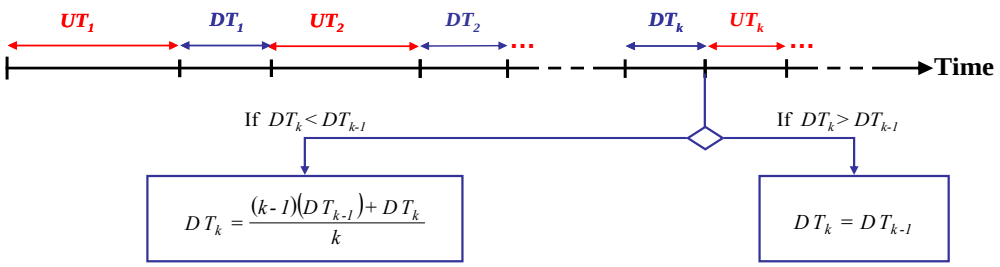


Fig. 4: General Case of Updating Downtime Requirement

Therefore, in this situation the maximum allowable downtime duration that the system can accommodate without violating the availability requirement is the average downtime given by,

$$\frac{UT_1 + UT_2}{UT_1 + UT_2 + DT_{Average} + DT_{Average}} = \frac{UT_1 + UT_2}{UT_1 + UT_2 + DT_1 + DT_2} \tag{6}$$

Figure 4 summarizes both scenarios in one general case. The model imposes the required downtime to meet the availability requirement, and then it evaluates the current downtime with respect to the previous one. If the currently imposed downtime is larger than the previous one, then the model substitutes the current downtime value for the previous one. But if the currently imposed downtime is shorter than the previous one, then the model averages the current downtime value with all previous ones. The goal of this procedure is to generate one unique value of the maximum allowable downtime that meets the availability requirement. Note, during this procedure, the unknown system parameter gets updated as soon as the downtime values are updated.[2]

If the unknown system parameter is explicitly generating the uptime (instead of the downtime); then, by analogy, a similar procedure is used to impose and update the uptimes to derive one unique value of the system parameter. Once the final value of the downtime (or uptime) is imposed and updated, the unknown system parameter is solved for.

3. Application of the Methodology

The methodology described in Section 2 has been implemented within an existing discrete event simulation model. The model used [25,26] is a discrete event simulation that follows a population of sockets[3] through their lifetime from first line replaceable unit (LRU) installation in the socket to the retirement of the socket. It is a stochastic simulation (implemented as a Monte Carlo model) of a timeline where specific events are added to the timeline and the resulting event order and timing can be used to determine the resulting number of failures avoided, life-cycle cost and availability (as an output).

[2] This is similar to a "greedy algorithm" that follows the problem solving heuristic of making the locally optimal choice at the end of each downtime with the hope of finding a global optimum.

[3] A socket is the place in a system where an LRU resides. A socket may be occupied by one or more LRUs during the support life of the system depending on the failure rate of the LRU. The imposed availability requirement is effectively on the socket (not the LRU), since an LRU could fail, be repaired and be placed into a different socket after repair.

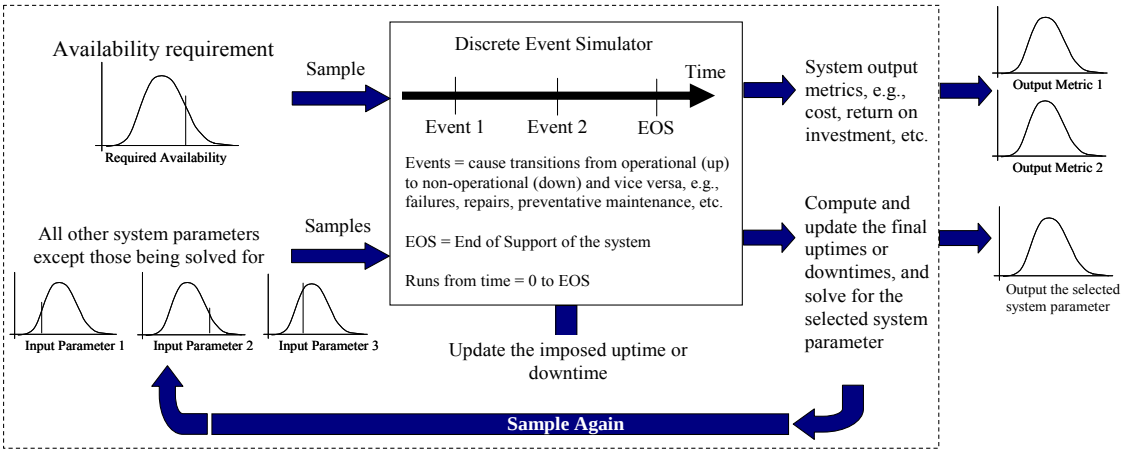


Fig. 5: Solution Process

In the discrete event simulator, failures of the LRUs in the sockets are modeled by sampling time-to-failure (*TTF*) distributions corresponding to the failure mechanisms that are relevant. When a failure occurs the LRU is removed from the socket and replaced with a spare (if one is available), the failed LRU may then be repaired and returned to the spares pool or discarded. The length of time to perform the maintenance activity depends on when in the operational cycle of the system the failure occurs, whether the maintenance event was unscheduled or scheduled, and the availability of spares.

The discrete event simulator used also models the impacts of including prognostics and health management (PHM) [27] within a system. PHM methods (similar to condition-based maintenance (CBM)) are incorporated into systems to avoid unanticipated failures that can potentially impact system safety and operation, and allows the conversion of traditionally unscheduled maintenance (or fixed interval maintenance) to scheduled maintenance performed based on the condition of the item. To determine if a maintenance event is scheduled or unscheduled, a probability distribution associated with the effectiveness of the PHM approach is constructed using a combination of both the time-to-failure (*TTF*) distribution associated with the LRU and the failure distribution associated with the particular PHM approach. The construction is defined differently for each PHM sustainment approach (e.g., data-driven, model-based,), see [25] for a detailed discussion of these models.

Figure 5 shows the process used to generate a distribution of system parameter values using a discrete-event simulator. The Monte Carlo implementation of the model samples the required availability distribution and other quantities that may be described using probability distributions, and then uses the quantities to solve for a value of the system parameter using the methodology described in Section 2. This process is repeated for each socket in the population. The resulting system parameter distribution (the output of the model) is the required quantity to meet the availability requirement (the input to the model).

3.1 Determining Type I Parameters

The objective in this case study example is to determine the spares replenishment lead time, i.e., inventory

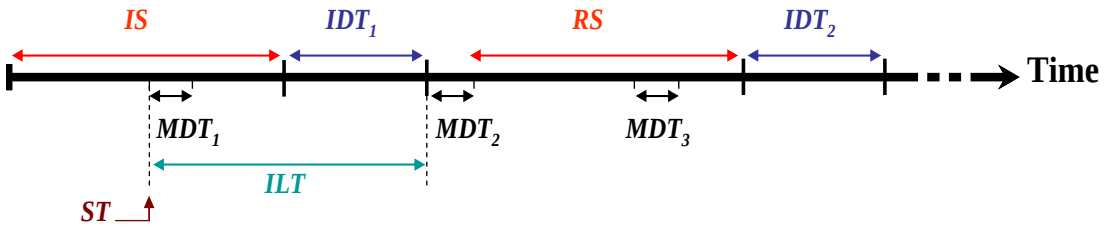


Fig. 6: Implication of the Inventory Model Parameters on the Timeline

lead time (*ILT*), to fulfill a specific availability requirement.[4] The case study inputs are provided in the Appendix, including the LRU description, inventory management, and operational profile.

In this example, the *ILT* is the unknown system parameter, where *ILT* is the amount of time it takes to receive replenishment spares when additional spares are ordered at the inventory spares threshold (*ST*) value (i.e., minimum quantity of held spares); this example case assumes that the inventory downtime (when the inventory runs out of spares, and the system is down waiting for replenishment spares) is larger than any maintenance downtime. We also assume that once the spares are received, the spare can be immediately installed in the system. Figure 6 shows the relationship between the various inventory parameters in this example where *MDT* is the maintenance downtime.

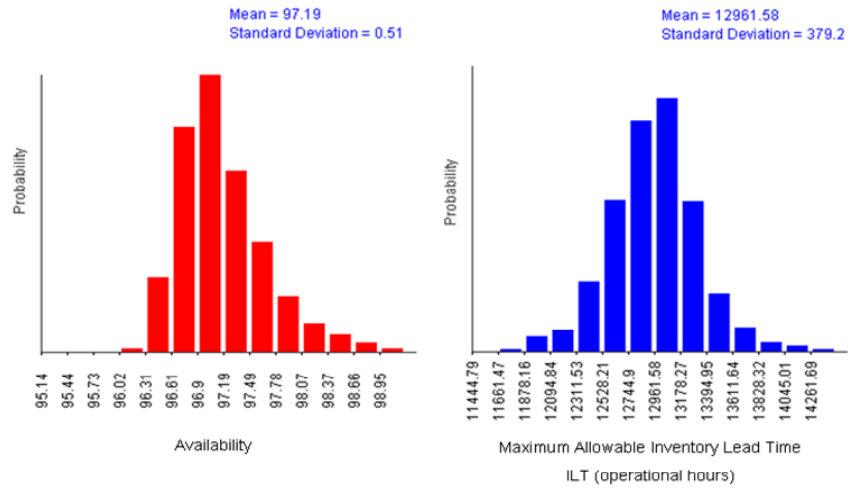
The *ILT* requires imposing a downtime requirement since varying the *ILT* only affects the downtime values, i.e., how long the system will be down waiting for spares to be replenished. Varying the *ILT* generates different *IDT* values; however the uptime values remain constant since they are primarily a function of the duration corresponding to using inventory *ST*; where the duration corresponding to using inventory held spares is a function of the duration corresponding to using the initial spares (*IS*) and duration corresponding to using the replenishment spares (*RS*). The *ILT* only defines the start of the next uptime, but it does not define the uptime duration.

Because, the *IDT* is purely a function of *ILT* and the duration to use the *ST*, where *STT* is the corresponding period of time to use all remaining spares; the *IDT* only depends on how low the inventory level is allowed to drop before ordering additional spares and how long it will take to receive those spares,

$$ILT = STT + IDT_1 \tag{7}$$

For this example we assume that the *MDT* are given and cannot be modified, i.e., the maintenance lead time, replacement time and repair time are specified as inputs. To fulfill the availability requirement at the end of the first *IDT*; IDT_1 should satisfy the A_o requirement (as defined in (1); where $IS\text{-}MDT_1$ corresponds to the accumulated uptime and $IS+IDT_1$ corresponds to the sum of the accumulated uptime and downtime), thus satisfy the following equation:

[4] A verification of the methodology was performed using this example in [28], using the following steps: 1) using the availability distribution requirement as an input to the method, determine the required *ILT* distribution (output). 2) Use the generated *ILT* distribution as an input to the existing discrete event simulation (described in the introduction to this section) to predict an availability distribution as an output. 3) Compare the availability distribution input requirement to the availability distribution determined as an output. The first step is sufficient to determine the *ILT* distribution. The second and third steps were performed to qualitatively verify the methodology.



(a) (b)

Fig. 7: (a) Required Availability Distribution (input to the model). (b) Computed Maximum Allowable Inventory Lead Time (*ILT*) in Operational Hours, for an Unscheduled Maintenance Policy

$$IDT_1 = \frac{IS - MDT_1}{A_O} - IS \tag{8}$$

Once IDT_1 is determined, the *ILT* can be computed by satisfying (7). Finally, the *ILT* is updated as the downtime requirement gets updated throughout the entire support life.

The required availability distribution used as an input for this example case is shown in Figure 7a. Considering an availability requirement that is expressed as a probability distribution makes the process of determining the necessary system parameters to meet the availability requirement challenging, since every system instance could have a different availability based on the sampled value from the probability distribution (other interpretations of the availability requirement are possible, see Section 2.1). This process is illustrated in Figure 5 and discussed in the introduction of this section. Figure 7b shows the computed *ILT* that is required to meet the availability requirement.

Determining the maximum allowable *ILT* for a specific availability requirement could be used to improve logistics management and potentially reduce life-cycle cost. If the availability drops below a specified threshold value, a cost penalty could be assessed; determining upfront the appropriate *ILT* could avoid these potential cost penalties. Also, knowing the maximum allowable *ILT* information, system supporters could require their suppliers to deliver within a specific lead time.

3.1.1. Unscheduled maintenance vs. data-driven PHM

In this section we want to determine the appropriate spares replenishment lead time, i.e., inventory lead time (*ILT*), to fulfill the availability requirement specified in Figure 7a for an unscheduled maintenance approach and a data-driven PHM approach applied to the same system.

Simulating with the imposed availability (input) requirement shown in Figure 7a, the *ILT* (output)

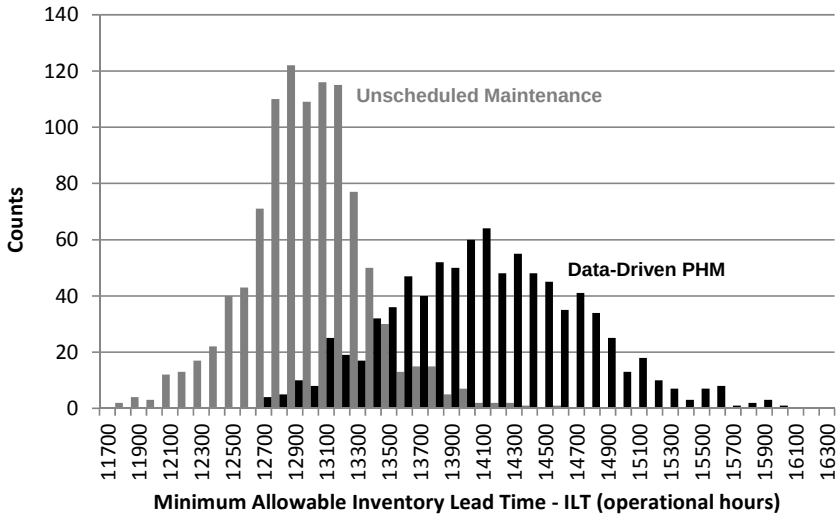


Fig. 8: Computed Maximum Allowable Inventory Lead Time (*ILT*) for an Unscheduled Maintenance Policy, and for a Data-Driven PHM Approach. Both Maintenance Approaches Satisfy the Same Availability Requirement

satisfying this requirement was determined for the unscheduled maintenance policy, Figure 8, the light gray histogram bars (this distribution is the same as Figure 7b). The methodology has also been applied to the same system managed using a data-driven PHM approach.[5]

The input data (provided in the Appendix) was used for the data-driven PHM approach; a prognostic distance of 600 operational hours was used and a symmetric triangular distribution with a width of 500 hours was assumed to represent the effectiveness of the data-driven PHM approach (see [27] for details of modeling the PHM uncertainty). Simulating with the imposed availability requirement shown in Figure 7a, the maximum allowable *ILT* satisfying the contract requirement was determined, and the generated *ILT* probability distribution is shown in Figure 8 (black histogram bars).

In this example, the data-driven PHM approach allows for a larger *ILT* (mean = 13,936 operational hours), compared to the unscheduled maintenance case (mean = 12,961 operational hours).[6] In other words, using a data-driven PHM approach allows a given availability requirement to be met if *ILTs* are longer, or alternatively stated, the use of PHM would allow a supply chain with longer *ILTs* to be used. The use of a data-driven PHM approach has shifted the maximum allowable *ILT* distribution by approximately 1000 hours to the right. This result is due the fact that data-driven PHM has provided early warning of failures; therefore creating the opportunity to transition maintenance actions from unscheduled to scheduled events reducing the accumulated operational downtime. For a fixed *ILT*, this would result in an improved operational availability of the system. However, since in this example the same availability requirement

[5] Data-driven PHM means that one or more attributes of the system are directly observed and those observations are used to determine the health of the system (e.g., monitoring for precursors to failure, use of canaries, anomaly detection). See [27] for the details of data-driven PHM.

[6] From Figure 8, there is an 88% probability that a sample from the *ILT* distribution using a data-driven PHM (black bars) will be greater than a sample from the *ILT* distribution using an unscheduled maintenance (light grey bars).

was imposed on both cases (unscheduled maintenance and data-driven PHM), thus the accumulated operational downtime was used as a fixed quantity (imposed by the contract availability requirement); then the avoided unscheduled maintenance downtime was added to the *IDT*, resulting in a larger allowed *ILT*.

In this case study, both approaches, unscheduled maintenance and data-driven PHM, we were able to determine the unknown system parameter (the maximum allowable *ILT* in this case) satisfying the availability requirement, but data-driven PHM allows larger *ILTs* compared to the unscheduled maintenance policy case.

4 Summary and Discussion

This paper presented a new direct method based on discrete event simulation for using an availability requirement to determine unknown system design and support parameters; the approach is general and could be applied to any type of system, even when uncertainties are included and the availability requirement is represented as a probability distribution.

A demonstration of the determination of the parameters that affect either uptime or downtime (inventory lead time (*ILT*) was used as an example) was provided. The demonstration shows how the necessary logistics management and reliability information for a system could be determined to meet a specific availability requirement. The application-specific example results show that the model predicted a larger allowable *ILT* for the data-driven PHM case relative to unscheduled maintenance when both maintenance approaches are constrained to meet the same availability requirement. Also, the example demonstrates that the use of PHM in cases where availability requirements are imposed can provide value beyond the commonly articulated failure avoidance and minimization of lost remaining useful life. An ability to determine this type of information could be crucial to availability contracts and to any system with high availability requirement.

The results presented in this paper are focused on deriving single system parameters that are necessary to meet a specific availability requirement. However, the methodology could be extended to address the concurrent determination of multiple system parameters; which would involve two cases. First, if system parameters are dependent (i.e., if a relationship between the unknown system parameters could be determined), then the methodology could be used to solve for the unknown dependent system parameters. Second, if the system parameters are independent, then the application of an optimization approach would be required since the established relationships that are used to solve for the unknown system parameters may have more unknowns than the actual number of derived equations. However, even in this case, the methodology is still efficient in terms of reducing the large and complex optimization "search-based" iterative problem, where every generated set of system parameters may or may not satisfy the availability requirement; to a basic "non-search-based" problem where the unknown system parameters have to be optimized to satisfy one single predefined relationship.

The analysis presented in this paper makes several simplifying assumptions that could be addressed in future work. Most notably this work assumes that there is no redundancy in the system. To account for

redundancy, the model must analyze cases where a unit's failure does not necessarily generate an operational downtime of the system, since other redundant units could substitute for the failed unit and maintain the operation of the system while the failed unit is being repaired or replaced. An extension of the discrete-event simulation model to handle redundancy is possible and could be performed by concurrently modeling all redundant units and only accounting for downtime when all redundant units are non-operational. This paper provides a method of guaranteeing that an availability requirement is met at all times, however, the system provider or manufacturer's target may be to minimize life-cycle cost, so there may be situations (depending on the size and form of incentives and penalties specified in the contract) in which the system's availability is allowed to drop below the required availability.

Acknowledgements

The authors acknowledge the National Science Foundation, Division of Design and Manufacturing Innovation (Grant Number: CMMI-1129697), for their support. The authors would also like to thank the companies and organizations that support research activities at the Center for Advanced Life Cycle Engineering (CALCE) at the University of Maryland.

Appendix – Case Study Data Inputs

This appendix provides model inputs and assumptions that are used for the case study examples presented in this paper. The LRU used in this example is an avionics multifunction display (MFD). The operational profile is summarized in Table A.1 [26], and a 20 years support life was chosen based on [29].

Table A.1: Operational Profile

Factor	Multiplier	Total
Support life: 20 years	2,429 flights per year	48,580 flights over support life
7 flights per day	125 minutes per flight	875 minutes in flight per day
45 minutes turnaround between flights	6 preparation periods per day (between flights)	270 minutes between flights per day

The assumed reliability for this case example, i.e., time-to-failure (*TTF*), of the LRU is modeled as a Weibull distribution with β=1.1, η=200 operational hours and γ=9000 operational hours. Table A.2 summarizes the spares inventory (per socket) assumptions.

Table A.2: Spares Inventory

Factor	Quantity
Initial spares purchased for each socket	5
Threshold for spare replenishment	≤ 1 spares in the inventory per socket
Number of spares to purchase per socket at replenishment	4
Billing due date when ordering additional spares	2 years from purchase date

References

[1] Ng I. C. L., R. Maull, and N. Yip, *Outcome-Based Contracts as a driver for Systems Thinking and Service-Dominant Logic in Service Science: Evidence from the Defence Industry*, European Management Journal, vol. 27, no. 6, pp. 377-387, December 2009.

[2] BAE 61972 BAE Annual Report 2008.

[3] Beanum R., *Performance-Based Logistics and Contractor Support Methods*, Proceedings of the IEEE Systems Readiness Technology Conference (AUTOTESTCON), September 2006.

[4] Knowledge@Wharton, '*Power by the Hour': Can Paying Only for Performance Redefine How Products are Sold and Serviced?* http://knowledge.wharton.upenn.edu/article.cfm?articleid=1665, February 21, 2007.

[5] Barlow, R. E., F. Proschan, and L. C. Hunter, *Mathematical Theory of Reliability*, Society for Industrial and Applied Mathematics, 1996.

[6] P. Alfredsson, *Optimization of Multi-Echelon Repairable Item Inventory Systems with Simultaneous Location of Repair Facilities*, European Journal of Operational Research, vol. 99, no. 3, pp. 584–595, Jun. 1997.

[7] P. Hokstad, H. Langseth, and B. H. L. and J. Vatn, *Failure Modeling and Maintenance Optimization*, International Journal of Performability Engineering, vol. 1, no. 1, pp. 51-64, Jul. 2005.

[8] Dekker, R., *Applications of Maintenance Optimization Models: A Review and Analysis*, Reliability Engineering & System Safety, vol. 51, no. 3, pp. 229–240, 1996.

[9] Immonen, A., and E. Niemelä, *Survey of Reliability and Availability Prediction Methods from the Viewpoint of Software Architecture*, Software and Systems Modeling, vol. 7, no. 1, pp. 49–65, 2008.

[10] Frangopol, D. M., and K. Maute, *Life-Cycle Reliability-Based Optimization of Civil and Aerospace Structures*, Computers & Structures, vol. 81, no. 7, pp. 397–410, Apr. 2003.

[11] Canfield, R. V., *Cost Optimization of Periodic Preventative Maintenance*, IEEE Transactions on Reliability, vol. R-35, no. 1, pp. 78–81, 1986.

[12] Taboada, H. A., J. F. Espiritu, and D. W. Coit, *MOMS-GA: A Multi-Objective Multi-State Genetic Algorithm for System Reliability Optimization Design Problems*, IEEE Transactions on Reliability, vol. 57, no. 1, pp. 182–191, 2008.

[13] Kajal, S., P. C. Tewari, and P. Saini, *Availability Optimization for Coal Handling System Using Genetic Algorithm*, International J. of Performability Engineering, vol. 9, no. 1, pp.109-116, 2013.

[14] Lins, I. D., and E. L. Droguett, *Redundancy Allocation Problems Considering Systems with Imperfect Repairs Using Multi-Objective Genetic Algorithms and Discrete Event Simulation*, Simulation Modelling Practice and Theory, vo. 19, pp. 362-381, 2011.

[15] Sherif, Y. S., and M. L. Smith, *Optimal Maintenance Models for Systems Subject to Failure–A Review*, Naval Research Logistics Quarterly, vol. 28, no. 1, pp. 47–74, 2006.

[16] Albright, S. C., and A. Soni, *Markovian Multiechelon Repairable Inventory System*, *Naval Research Logistics*, vol. 35, no. 1, pp. 49–61, 1988.

[17] Sato, N., and K. S. Trivedi, *Accurate and Efficient Stochastic Reliability Analysis of Composite Services Using Their Compact Markov Reward Model Representations*, Proceedings of the IEEE International Conference on Services Computing, SCC 2007, pp. 114 –121, 2007.

[18] Hershey, P. C., *High Availability Data Processing System and Method Using Finite State Machines*, U.S. Patent 554407706-Aug-1996.

[19] Volovoi, V., *Modeling of System Reliability Petri Nets with Aging Tokens*, Reliability Engineering & System Safety, vol. 84, no. 2, pp. 149–161, 2004.

[20] Caro, J. J., J. Möller, and D. Getsios, *Discrete Event Simulation: The Preferred Technique for Health Economic Evaluations?* Value Health, vol. 13, no. 8, pp. 1056–1060, Dec. 2010.

[21] Janakiraman G., J. Santos, and Y. Turner, *Automated System Design for Availability*, Proceedings of the International Conference on Dependable Systems and Networks (DSN'04), 2004.

[22] Raffo D, and S. Setamanit, *Supporting Software Process Decisions Using Bi-directional Simulation*, International Journal of Software Engineering and Knowledge Engineering (IJSEKE), vol. 13, no. 5, pp. 513-530, 2003.

[23] Reynolds A., and G. McKeown, *Construction of Factory Schedules Using Reverse Simulation*, European Journal of Operational Research, vol. 179, no. 3, pp. 656-676, June 2007.

[24] Jazouli T., and P. Sandborn, *Using PHM to Meet Availability-Based Contracting Requirements*, Proceedings of the IEEE International Conf. on Prognostics and Health Management, June 2011.

[25] Sandborn P., and C. Wilkinson, *A Maintenance Planning and Business Case Development Model for the Application of Prognostics and Health management (PHM) to Electronic Systems*, Microelectronics Reliability, vol. 47, no. 12, pp. 1889-1901, December 2007.

[26] Feldman K., T. Jazouli, and P. A. Sandborn, *A Methodology for Determining the Return on Investment Associated with Prognostics and Health Management*, IEEE Trans. on Reliability, vol. 58, no. 2, pp. 305-316, June 2009.

[27] Pecht M., *Prognostics and Health Management of Electronics*, Wiley, 2008.

[28] Jazouli T., and P. Sandborn, *A Design for Availability Approach for Use with PHM*, Proceedings of the International Conference on Prognostics and Health Management, October 2010.

[29] Federal Aviation Administration, *Investment Analysis Benefit Guidelines: Quantifying Flight Efficiency Benefits. Version 3.0*, Investment Analysis and Operations Research Group, June 2001.