PROMPT, A decision support system for opportunity-based preventive maintenance

Rommert Dekker and Cyp van Rijn

Erasmus University Rotterdam

June 11, 2003

Abstract

In this paper we describe an operational decision support system, called PROMPT, for systematic optimisation of maintenance activities and executing them at opportunities. Hereby, an opportunity is any event at which a unit can be maintained preventively without incurring cost penalties for the shutdown of the unit. The d.s.s. was developed to take care of the random occurrence of opportunities and restricted duration. Moreover, it is able to handle a multitude of different maintenance activities. Finally, we describe experience gained in a fieldtest with PROMPT.

1 Introduction

Maintenance management has been described as the last frontier of scientific management. Whereas in many other fields, such as production, logistics, personal, finance, administration, management science and industrial engineering have been active for a long time and their results have had its impact, the maintenance manager has long been missing tools to improve his decision making. The work of a maintenance man has been dominated by unexpected events, making management a difficult task. The last decade has seen however, an increasing attention for maintenance management. Underlying reasons are twofold. First of all, the amount of equipment in production plants has increased in time and despite improvements in maintainability, large percentages of personnel and operational expenditure are in the maintenance area. The second reason is that trends in production technology and concepts (like Just-In-Time) have stressed a high and a timely and almost continuous production, thereby making downtime costlier and maintenance more important.

A main improvement in the last decade for maintenance management has been the introduction of maintenance management information systems, providing the maintenance manager with up to date information. The area of decision support however, runs behind, and although maintenance management information systems contain a lot of data, these will be worthless, if they do not improve decision making.

Within the area of operations research a lot of effort has been spent on developing and analysing maintenance optimisation models. It has been such a fruitful area, that review papers mention hundreds of articles (Smith and Sherif [1981] list 524 papers) and many more have appeared since. The impact on actual operations of all these papers, has been marginal, and in few other fields there seems to be such a discrepancy between theory and practice. This gap is being narrowed by the improvements and cost reductions in computer technology, making computers and software also available for the maintenance function, and thereby allowing the use of sophisticated models.

In this paper we describe a decision support system, called PROMPT, which uses operations research models to assist the maintenance manager in optimising preventive maintenance and to support him in executing preventive maintenance at the right time. It is an attempt to bridge the gap between theory and practice, yet most of the theory needed was only developed during the construction of the d.s.s. PROMPT addresses that preventive maintenance that is carried out to reduce downtime or to secure safety. It offers both planning and scheduling tools to the user and is especially developed to make use of maintenance opportunities, thereby avoiding scheduled downtime.

In this paper we will first give an overview of the problem characteristics for which PROMPT was developed. Thereafter we give an overview of PROMPT, its models, and what it is doing. Furthermore, we state our experiences with a field test of the PROMPT system, which considered both initialisation of the system as well as the effect of its advice. Finally we give an evaluation of the system.

The PROMPT system which is described in this paper is the successor of an earlier prototype which is described in Van Aken et al. [1984]. Similar to the present PROMPT system the prototype was directed at giving advice for opportunity based preventive maintenance. Although this system was considered to be successful, it had two major shortcomings. First of all its objective was to increase reliability, whereas in a later stage not all failures were considered to be of equal performance. Secondly, it could not indicate how much preventive maintenance is cost effective, as the models assumed that more preventive maintenance always implied more reliability. The present PROMPT system, as described here, is a completely new system, in which we took advantage of the experiences obtained with the earlier prototype.

There are no comparable systems to PROMPT. In Dekker[1992] an overview is given of maintenance decision support systems. Most of them are tactical tools, which address a single unit or component and allow to optimize a single action on that. Some maintenance management information systems contain a reliability module, but hardly ever an optimisation module and certainly not for opportunity maintenance.

2 Problem description

If preventive maintenance is applied to a unit, there is a preference to carry it out only at those moments in time when the unit is not required for production. In some cases, where units are used continuously (e.g. in the process industry) this may cause problems. Execution of preventive maintenance is then restricted to costly annual shutdowns. In some systems however, short lasting interruptions of production occur by times for a variety of reasons, e.g. breakdowns of or maintenance on essential units. During these interruptions some other units are not required and can be maintained preventively, in which case we speak of maintenance opportunities. Unfortunately, these opportunities can mostly not be predicted in advance. Because of the random occurrence of opportunities and of their limited duration, traditional maintenance planning fails to make effective use of them.

The objective of PROMPT is to give decision support for opportunity-based preventive maintenance. For PROMPT an opportunity is defined as any moment in time at which preventive maintenance can be carried out without adverse effects of a unit shutdown being incurred. The user of PROMPT has to identify the opportunities and to report them to the system to get advise. PROMPT assumes that although opportunities occur randomly, they do occur repeatedly and provide 'enough' time to use them for preventive maintenance. PROMPT primarily focuses at routine preventive maintenance as first line maintenance (greasing, etc.) can be executed during normal operations and major overhauls are too large for opportunities and have to be planned in advance.

An opportunity-based policy is of importance for continuously used equipment, for which downtime costs are high. Examples of such equipment are gas turbine driven power generators at offshore production platforms. A typical aspect of offshore platforms is that in a limited amount of space all the equipment has to be installed, and that therefore in the design phase as few equipment has been installed as possible, making downtime costs high. Another aspect is that the production of the platform has a high economic value. Although usually production is not lost but rather deferred, there is a strong incentive to recover the large investments as soon as possible and therefore even deferred production has a high cost value.

To make effective use of opportunities, preventive maintenance has to be split up into packages which can be fully carried out at an opportunity. Both mechanical, instrument as well as electrical maintenance is included and different age indicators, like runhours, starts and stops are allowed.

The tasks PROMPT has to carry out are threefold. First of all it should indicate how much, if at all, preventive maintenance, is cost-effective and for each maintenance package it should determine an optimal policy. Secondly, PROMPT should schedule the cost-effective packages in such a way that as much as possible is adhered to the optimal interval. For safety related maintenance PROMPT assumes that a maximal interval can be specified by the user. In that case PROMPT tries to execute the safety-related packages at the last possible opportunity within the required interval. Finally, PROMPT should administrate failures and preventive maintenance results so that in course of time a better insight into failures can be obtained.

3 An outline of PROMPT

3.1 Introduction

In order to reach the objectives set, a variety of problems have to be tackled. Here we describe some of the ideas behind PROMPT and its main optimisation methods.

3.2 Hierarchy of units

Any real system can be decomposed into units, parts and components. Several hierarchies may exist, but we use the one applied by maintenance. Such a decomposition is important since it imposes a lot of rules for maintenance and for each level different information may be available, which has to be translated to other levels. In agreement with the operating company for which PROMPT was developed the following hierarchy was assumed. In PROMPT a system is defined as a whole of units performing a specified task with a measurable output on which a lost or deferred product value can be set. PROMPT assumes that a system is built up of sub-systems in a series configuration. A subsystem consists of one or multiple units in parallel, fulfilling a certain task. A unit can be a gas turbine, compressor, pump or any other physical entity. The unit is the highest level at which PROMPT gives advice. PROMPT assumes that if for a unit an opportunity occurs, it is for all parts of the unit. The planned maintenance routines are subdivided into maintenance packages each consisting of one or more maintenance activities, which are not overlapping. Apart from this maintenance hierarchy, PROMPT also considers a physical hierarchy, in which the unit is subdivided into elements each having one or more failure modes. The user is free to define the elements which do not have correspond with one maintenance activity.

3.3 Balancing maintenance costs and benefits

For time-based preventive maintenance which is carried out to reduce the number of failures and to prevent unscheduled downtime, PROMPT should be able to determine the best frequency. This requires a balancing of the costs and benefits of that maintenance. The costs are easily calculated, as they consist of manhours and materials. Benefits are more difficult to quantify. One can determine the number of failures prevented, but not all failures are equally important. An alternative is to determine the amount of unscheduled downtime prevented (i.e. that downtime that would be caused by failures), which makes more sense. This leaves two ways to determine how much maintenance should be carried out. First of all, one can set a target for the intrinsic availability (i.e. the availability excluding standby hours), which comes down to setting a target value for the unscheduled downtime, and determine how much planned maintenance is required to reach that target. A second option is to set a cost penalty to unit downtime and determine for each maintenance package whether its execution pays off against the savings in reduced downtime. We have chosen for the last option for a number of reasons. It may seem equally difficult, setting either a target for downtime, or a cost penalty for downtime and special models may be required for each. One should realise however, that a target availability does not take the value of production into account, which can be done for a cost penalty. Furthermore, a cost penalty allows a balancing for each maintenance package, whereas a target availability requires a simultaneous balancing of all maintenance packages, which is far more difficult. The latter could be simplified, by making some artificial choices, but that would be arbitrary. A final argument is that a target availability is more difficult to handle in case of unrevealed failures causing no direct downtime.

As said before, PROMPT is based on a cost balancing, and a special model has been set up to support in setting such a cost penalty.

3.4 Unit downtime cost penalty

In general it is a difficult task to set a cost value to unit downtime as systems may consist of many units, each performing specific tasks. The same was true for the systems we considered appropriate for application of PROMPT. No downtime cost values were available for the development team. Complicating aspects consisted of the presence of (non-identical) standby units, variations in demand and the fact that for utility units output has an indirect value (e.g. what is the value of 1 MWh?). Another problem on e.g. production platforms is that downtime may cause for deferred rather than for lost production. Hence there should be a cost value for deferred production, which may depend on many factors (e.g. tax regimes). Luckily, this aspect had been tackled by the company's economic department. It is in fact partly a subjective problem, as a cost value for deferred production is in fact a statement of management on how much they are prepared to pay for prevention of deferred production. In fact the same holds for the unit downtime penalty: it is a statement of how much management is prepared to pay to prevent unit downtime.

Given a cost value for deferred production at system level, we developed a special economic model to deal with the other aspects. The model assumes that the unit in question forms with other (not necessarily identical) parallel units a subsystem. The effect of loss of the unit is considered at subsystem level only and unavailability of other subsystems is neglected in this respect. Basically the cost penalty for a given unit is calculated as follows. First an enumeration of the states (either working or failed) of all other units in the subsystem is performed. Next for each combined state of other units the costs caused by losing the unit in question are determined. The cost penalty then follows by taking a weighted summation of the costs per state multiplied with the probability of occurrence of that state. Notice that this is a marginal cost value, i.e. the costs incurred by one hour of extra downtime of a unit. It is not the allocation of the actual downtime costs over the units. The model is an extension of the so-called k-out-of-n availability models to nonidentical machines and varying demand. It is not clear whether this cost allocation is the best one. There is almost no literature in this respect, although the problem arises in most systems with parallel units. Almost all papers assume that either the cost penalty is given for a particular unit, or that the unit has only one failure mode, which is an unrealistic assumption. For utility units we considered the production systems sustained by it. Costs of downtime of a utility unit then follow from the loss or deferment of production of those production systems which have be shutdown because of the loss of the utility unit. In this assessment we take into account the availability of other utility units which are capable to take the duty over of the unit considered.

The unit downtime cost penalty was explicity stored in the database of the d.s.s. with the idea that it might change in time, because of e.g. depletion of the field from which the platform was producing.

3.5 Failure models

After having established a cost penalty for unit downtime, we will in this section consider the positive effects of each preventive maintenance activity in detail. For reasons of language simplicity we will regard in this section all elements addressed by one activity as being one component (although in practice this is not necessary the case). In PROMPT a failure is defined as 'any event after which a component stops functioning in a prescribed way'. In general two types of failures should be distinguished, viz. revealed and unrevealed failures and a separate failure model should be used for each. A failure model describes the relationships between failure and its consequences and contains a quantitative prediction mechanism of failures. The latter occurs through probability distributions, which may be in any type of condition indicator (e.g. calendar time, runhours, etc.), as long as the indicator is predictable in time. PROMPT's failure models have been set up in such a way that they are consistent with the findings of inspections.

3.5.1 The revealed failure model

The revealed failure model assumes that a failure is directly noticed and that an appropriate action is undertaken. Consequences of the failure are assumed to occur directly after the failure. As a result of the failure the unit may breakdown with a certain probability, p_{ud} (to be specified per component). Costs of failure are split up into indirect cost due to unit downtime (in case it breaks down) and direct costs due to repair of the component. If the expected downtime amounts to d hours, the unit downtime cost penalty to c_{ud} and the repair costs to c_r, then the total expected cost of failure c_f is given by $c_f = c_r + p_{ud} d c_{ud}$. Time to failure is modelled through a two parameter Weibull distribution with shape parameter λ and scale parameter β. Other data the user had to specify included average downtime in case of a unit breakdown, average time needed for repair, average number of men required for repair and additional material costs (normal material costs were incorporated as a surcharge on the manpower costs).

3.6 The unrevealed failure model

The unrevealed failure model assumes that a failure of the component may remain hidden, until either an inspection or some severe consequences occur. As not the failure event itself is important, but the time being in the failed state, the model assumes a cost rate for being in a failed condition (like in the Barlow and Hunter [1960] model. This cost rate is obtained by assuming that the time between the component failure and consequences is quite long and may be approximated by an exponential distribution. Let c_c denote the cost value of these consequences can be given. From specifying the probability p_{ud} on these consequences in a certain interval of length T, given a component failure halfway during that interval one can then calculate the cost rate c_{fr} from $c_{fr} = p_{ud}c_c/T$. Cost of inspection and subsequent action were assumed to be independent of the resulting action and had to be specified by the user. Although we first assumed that the time to failure was exponentially distributed, we later changed it to a two-parameter Weibull distribution.

3.7 Condition indicators

Apart from calendar time PROMPT also allows other cumulative indicators, such as runhours and number of starts and stops. The only requirement for a cumulative indicator was that it can be predicted in time. To this end we used both an historical estimate for the time conversion factor on the long run and an exponential smoothing prediction mechanism to predict the conversion factor in the short run. We hoped to include state condition indicators derived from condition monitoring as well. However, none of the state condition indicators we are aware of, allows a time-related quantitative prediction of component failure (in terms of probabilities). On the contrary, they merely indicate whether some (often not which) failue is imminent or not and are therefore not suited to plan maintenance at opportunities.

3.8 Preventive maintenance packages

Opportunity-based maintenance requires that preventive maintenance is split up into packages. The larger packages are, the more difficult it may be to exectue it at an opportunity of limited duration (some one or two days). On the other hand, it is usually not economic neither convenient from an administrative point of view to execute all maintenance activities separately. Hence activities were grouped into packages. We therefore assumed that the user would be able to define maintenance packages. It further appeared practical to advise only full maintenance packages, even if one of its activities was already carried out because of a failure. Furthermore, failures provided usually no time for preventive maintenance as the failed component had to be repaired as soon as possible and no time was left over. The user had to be given the freedom to report either a renewal or a repair of the component to its state before the failure.

3.9 Optimisation models

3.9.1 General approach

The optimisation problem faced by PROMPT can be summarised as: "plan and schedule a number of maintenance packages, each consisting of one or more maintenance activities, at randomly occurring opportunities of a restricted duration". The planning would include determining the long term optimal policy, whereas the scheduling should indicate at a given opportunity which packages should be carried out with what priority given their long term optimal policy. Realistic number of maintenance packages lie between 50 and 100.

Although PROMPT was set up with one specific application in mind, it was the idea that it should be as general as possible, and thus not focus on one unit specifically. This resulted in the following detailed problem characteristics and assumptions.

(i) We assumed the occurrence of opportunities could be described by a renewal process and that a user was able to specify both a mean and a variance of the time between opportunities, valid on the long run.

(ii) A user should have the freedom, on the other hand, to overrule the long-term distribution of the interval to the next opportunity if he has more information available.

(iii) Decisions concerning executing of maintenance packages only need to be taken at opportunities.

(iv) The opportunity duration is not known beforehand, and even during the actual opportunity, it may change. Therefore no exact duration can be used in the scheduling.

(v) Interactions between maintenance packages of any kind may be neglected.

A literature search revealed some opportunity models (see e.g. Jorgenson et al. [1967], Woodman [1967], Duncan and Scholnick [1973], Sethi [1977], Vergin and Scriabin [1977] and Bäckert and Rippin [1985]), but none of them was capable of dealing fully with our problem. Some papers applied Markov decision models in which the number of components determined the dimension of the state space. This approach however, is computationally not tractable in case of more than three components. Jorgenson et al. [1967] presented a model for multiple components with exponential time between opportunities, but does not specify how to optimise. None of the models was able to deal with a restricted opportunity duration, neither with different failure models.

We therefore developed novel models to deal with this complex problem. In our case opportunities are created by causes outside the unit and upon failure of one of its components the unit is repaired as soon as possible and no time for further preventive maintenance is available. Our first conclusion was therefore that the only correlation between the packages consisted of competing for the restricted time at an opportunity. Accordingly we reduced the original problem to the following: *"determine for each maintenance package separately an optimum policy, which indicates when it should be carried out at an opportunity, independently of all other packages. Furthermore, determine from the outcomes of these models which maintenance packages should be executed at a given opportunity.* To this end we introduced the so-called one-opportunity-look-ahead policies, which can be considered as a generalisation of marginal cost approach (originally introduced by Berg [1980]). At each opportunity these policies compare for each package the costs of deferring the execution to the next opportunity with the minimum long term costs. In the next sections the approach will be discussed in more detail.

3.9.2 Maintenance activity optimisation models

Consider a maintenance activity addressing a revealed failure of a specific component. Basically the maintenance optimisation can be tackled by the age or block replacement model with the extra restriction that preventive maintenance is restricted to opportunities. We took the block replacement model since that can be extended to multiple activities in a package and non-exponential times between opportunities can be handled (for age replacement only exponentially distributed times between opportunities can be handled; non-exponential times become very difficult, see Dekker and Dijkstra [1992]). We did modify the block policy to avoid replacing new components, but that will be explained later. In the block replacement model a component is replaced preventively every t time units against costs c_p and upon failure at costs $c_f (> c_p)$. Let $F(t)$ be the c.d.f. of the time to failure and let $M(t)$ be the associated renewal function, indicating the expected number of failures in $[0, t]$. The long-term average costs $g(t)$ follow easily from renewal theory and amount to

$$g(t) = \frac{c_p + M(t)}{t}, \tag{1}$$

For unrevealed failures we used Barlow and Hunter's [1960] model, which goes as follows. A component is inspected every t time units and repaired without extra costs upon failure. For every time unit the component is failed a cost rate c_{fr} is incurred. Let $F(t), f(t)$ be the c.d.f, p.d.f. of the time to failure respectively. The long-term average costs $g(t)$ then equal

$$g(t) = \frac{c_p + \int_0^t c_{fr}(t-x)f(x)dx}{t} = \frac{c_p + \int_0^t c_{fr}(1-F(x))dx}{t} \tag{2}$$

Next consider the case that preventive maintenance or replacement can only be done at opportunities. Suppose (as was the case in our problem) that opportunities are generated independently from the component failure processes and that their occurrence can be modelled by a renewal process. The block policies are then extended to control limit policies of the type: "*maintain a component at the first opportunity if more than t time units have passed since the previous preventive maintenance*". Let the random variable Z_t denote the forward recurrence time to the next opportunity if t time units have passed since the last preventive maintenance at an opportunity. Notice that executions of the maintenance activity at an opportunity can be considered as total renewals. Hence the renewal cycle has length $t + Z_t$. In case of block replacement the expected number of failures is given by $E(M(t + Z_t))$, where the expectation is with respect to Z_t. This leads to the following formula for the expected average costs $g_Y(t)$ of executing a maintenance activity with control limit t

$$g_Y(t) = \frac{c_p + \int_0^\infty M(t+z)dP(Z_t \le z)}{t + EZ_t}. \tag{3}$$

Dekker and Smeitink [1990] show that the same conditions are needed for existence of a unique minimum t^* to $g_Y(t)$ as for the standard block replacement model. Furthermore, that t^* is the unique solution to the following optimality equation.

$$c_f E[M(t+Y) - M(t)] - g_Y^* EY \begin{cases} < 0 & \text{for } 0 < t < t^* \\ = 0 & \text{for } t = t^* \\ > 0 & \text{for } t > t^* \end{cases} \tag{4}$$

where g_Y^* denotes the minimum average costs. Notice that $c_f E[M(t+Y) - M(t)]$ can be interpreted as the expected costs of deferring execution of the activity from the present opportunity at time t to the next one, Y time units ahead.

The analysis of the opportunity block replacement model does not make use of the interpretation of the cost over an interval. In fact any other cost function may be used as well (as is also remarked in Dekker [1995]). Accordingly the analysis is easily set over to the unrevealed failure model with $M(t)$ replaced by $\int_0^t (1 - F(x))dx$.

To calculate the integrals in eq. (3) we approximated Z_t in first instance by a three point distribution with reasonably chosen values and probabilities. Later, in Dekker and Smeitink [1990] it appeared that Z_t can be approximated by the forward recurrence time of a Coxian-2 distribution in case the coefficient of variation is larger than 0.5 and by the stationary excess distribution in the other case. For the renewal function a simple but effective approximation was developed (see Smeitink and Dekker [1989]).

Notice that $g_Y(t)$ is a function of one variable, implying that optimisation is not too difficult. We applied a fixed step size search combined with a bisection procedure to determine the first minimum of $g_Y(t)$. Theoretically, this is not necessarily the overall minimum as differences in lifetimes may cause for multiple minima. However, as most distributions had such a large variance which flattens out peaks, we did not encounter any problems.

3.9.3 Maintenance package optimisation models

Notice that both the block replacement model and the inspection model are easily extended to a package containing multiple activities. Suppose that the execution of package costs c_p and that n_r activities address revealed failures (with failure time distributions $F_i(t)$ and failure costs $c_{f,i}, i = 1, ..., n_r$) and n_u unrevealed ones (with failure time distributions $F_j(t)$ and cost rates $c_{fr,j}, j = 1, ..., n_u$). The total long-term average costs $g_Y(t)$ then amount to

$$g_Y(t) = \frac{c^p + E[\sum_{i=1}^{n_r} c_{f,i} M_i(t + Z_t) + \sum_{j=1}^{n_u} \int_0^{t+Z_t} c_{fr,j}(1 - (F_j(x))dx]}{t + EZ_t} \tag{5}$$

The analysis is again similar to the one component case. In principle one could encounter multiple minima in the optimization, but in all cases considered we encountered no problems.

3.9.4 Ranking criterion for multiple maintenance packages

Apart from indicating the optimal control-limit and hence an optimal long-term frequency with which an package was to be executed, we also need to set priorities in case too many packages had to be carried out at an opportunity.

Notice therefore that eq. (4) provides a means to set priorities. Below we extend it to the package case. Let $RC(t)$ be defined by

$$RC(t) = \sum_{i=1}^{n_r} c_{f,i} E[M_i(t + Y) - M(t)] + \sum_{j=1}^{n_u} E[\int_t^{t+Z_t} c_{fr,j}(1 - F_j(x))dx] - g_Y^* EY \tag{6}$$

with g_Y^*, the minimum average costs of the total package. We can interpret $RC(t)$ as the expected costs of deferring the execution of the package to the next opportunity, Y time units ahead, minus the long-term average costs over that time. Hence it is an ideal candidate to rank packages on. Notice that at an opportunity we only have to calculate the first part of $RC(t)$; as g_Y^* can be stored in the database we only need to calculate it upon initialisation of the d.s.s.

The idea is now to execute those maintenance packages with the highest ranking value until the opportunity is fully used. Notice that the ranking criterion is myopic: a package may be delayed multiple times at an opportunity. Including that effect, however, was considered to be too complex. The procedure was tested in Dekker and Smeitink [1994] and performed quite well.

Next, we did modify the block policy to take recent failure replacements into account. If for some revealed failure components actual ages were known we replaced the renewal function in eq. (6) by the expected number of failures given the present age(s), using the c.d.f. and its convolutions. This idea was elaborated in Roelvink and Dekker [1995] and appeared to be cover most of the difference between age and block replacement, even in the multi-component case.

Finally, we did want to allow the user to enter a specific interval (either as point value or as three point distriubtion) to the next opportunity, which could differ from the long-term distribution of the time between opportunities. In that case we replace the r.v. Y in eq. (5) by the interval specified.

3.10 Type of advice

Once we have calculated for each maintenance package a criterion indicating its importance for being carried out, we are still left with the problem of which maintenance packages to carry out, as each of them may require a different man effort. In principle we considered two problem approaches:

(i) Support the user with a ranked list of maintenance packages, from which he makes the final selection, taking all kinds of extra information into account.

(ii) Provide the user with an interactive knapsack scheduling program which determines an optimal selection given the time constraints.

Approach (ii) is to be preferred from a theoretical point of view, as that best guarantees optimality. The disadvantage is however, that it is far more complex, it requires a program on the spot and the ability of the user to run it, and furthermore, to specify the problem exactly. The latter was not trivial, as execution times of a maintenance packages can vary greatly, and besides, the opportunity duration may not be known exactly.

So the main question became to determine the extra value of a knapsack approach above a simple list heuristic: select the packages from the list and carry them out out, one by one until the opportunity has fully been used. Unfortunately, results in this respect can not be found in literature. We therefor carried out a quick investigation which indicated that the maximal relative improvement of a knapsack optimisation above the straightforward list procedure is small in realistic cases (usually less than 5%).

Furthermore, the knapsack procedure has the disadvantage of being sensitive to the constraint formed by the actual opportunity duration. As the list from which a selection has to be made will be short in practice it is no that difficult for a maintenance supervisor to determine the best selection. Even the more, he may be very pleased with having the freedom to take that decision rather than being degraded by a system telling him what to do. Besides, he also has to check whether the required spare parts are available. Therefore we decided to give the ranked list of maintenance packages as advice. An example is given in Appendix I.

3.11 Software

Although the company for which later a field test would occur had an extensive maintenance management information system in use, we did decide to develop PROMPT separately from it, with the intention to make connecting links once PROMPT had demonstrated its value. One of the reasons behind was that PROMPT needs more detailed information than what is in the maintenance management information system.

The main part of the PROMPT software consists of a database which has been written in a 4th generation database language, in order to secure easy reporting facilities. As language we chose FOCUS, in order to provide compatibility between a mainframe and a PC version. The optimisation occurs through Fortran subroutines.

Total code consists of some 20,000 lines. Although originally PROMPT was set up for a personal computer (PC), we later switched to a mainframe, as the complexity was too large to be handled by the then existing PC's (IBM PC-AT) and the PC FOCUS version.

4 Field test of PROMPT on major gas turbines

In this section we briefly describe a field test of PROMPT on three Rolls Royce Avon gas turbines, one for main power generation and two which served as oil pumps.

4.1 Defining maintenance activities and set up of maintenance packages

This was in fact a major task. Before PROMPT, routine maintenance was lumped together in large packages of say, 150 hours which were executed during the yearly platform shutdown. Each task had to be written down in detail, with exact specifications of the equipment addressed. Thereafter all activities had to be combined into maintenance packages. Although there are optimisation aspects involved, this was purely done by engineering judgement, grouping those activities which could easily be executed together. The type of the maintenance activities could be either mechanical, instrument or electrical. Furthermore, for each package one had to determine the best condition indicator, being either runhours, calendar time or number of starts and stops.

4.2 Experiences with the economic model for unit downtime penalites

Although the model developed to assess cost penalties for unit downtime was considered to be quite general, the field test revealed that practice has many unexpected aspects. For example, when assessing the consequences of loss of power for the power generating system it appeared that not every MW output was of equal value. In case of power loss the production systems are shutdown in order of importance. Another special feature was encountered with pumps. The model assumed that the throughput of units in parallel was the sum of the individual throughputs, which is not valid for pumps in a serial configuration (the pressure build-up is non-linear in the capacity). Using the model philosophy, however, it was not difficult to extend the model with these new aspects and to arrive at reasonable cost penalties for unit downtime. It does show, however, that it is difficult to build generally applicable models and that in each case specific unmodelled factors may dominate, which require a good economist with reliability knowledge. Moreover, hard coding models in software appeared to be dangerous in case no alternative ways of determination (e.g. hand calculation) are allowed. The experiences did learn us that all these problems can be overcome and that at the end realistic cost penalties for unit downtime can be obtained.

4.3 Initialisation at component level

The initialisation at component level was in fact the bulk of the work. Actually, it was a learning process, since we first did an initialisation for one unit, then changed the procedure and redid it for the other two units. Data had to be provided for maintenance packages as well as for maintenance activities. For each maintenance package one had to assess the man effort required to execute it, the type of condition indicator, and as option, special materials costs. Although execution times may vary widely in practice, it was not a too difficult job to give reasonable estimates. With respect to maintenance activities the following data were required by PROMPT. First of all the type of the dominant failure mode, being either revealed or unrevealed. Next to that the consequences of a failure in terms of costs and potential downtime and finally the time to failure distribution.

Severe problems were encountered in obtaining the component time to failure distributions. The data collected so far in the maintenance management information system was lumped over many failure modes and not registered using the PROMPT hierarchy. Furthermore, as the maintenance packages created for PROMPT were reasonably detailed, the amount of data per component was low. For even a third of the components no data was available over a period of two years in which we pooled over four machines. Therefore we decided to use expert judgement for initial estimates and to update it with later originating data. A full description of the procedure can be found in Dekker [1989]. Below we will give a short review.

As experts we used maintenance technicians having several years of experience with the unit in question. As they were difficult to access - they were working in weekly shifts at the offshore platform - we choose to send a questionnaire to obtain the data. As we had to model wear out, we needed at least two characteristics of the time to failure distribution. Furthermore, we decided to ask control question in order to investigate the value of the answers. Although the questions were formulated with care, it appeared from the analysis that there were considerable inconsistencies within the answers. We decided therefore to send a second questionnaire asking additional information. The response was not enthusiastic - people had to answer with probability statements about twenty questions - and the experts had to be pressed to give their answers. From the analysis we learned that some questions are difficult to answer when components are regularly maintained: e.g. the mean time to failure is difficult to estimate if most components are maintained before that age. Questions which concerned the failure fraction in the historical maintenance interval and in twice that interval were considered to give most reliable information on the lifetime distribution.

For the other two units in the field test, for which only a limited extra data was needed, we used another method. Based on the last questions we developed a data collection program which was able to analyse the questions directly and to give the experts direct feedback - in terms of mean time to failure and the optimum maintenance interval (resulting from a simplified optimisation). An analyst from the local head office used the program for elicitation. This turned out to be a success and removed all of the inconsistency between experts' answers. Two problem aspects remained however, viz. the problem of combining different experts opinion and the problem of updating the experts opinion with later originating data. To that end a separate study was initiated with Cooke from Delft University which resulted after a year in a special method (see Van Noortwijk et al. [1989] and Van Dorp [1989]).

4.4 Operational experiences

Operational experiences with the PROMPT system were good, although some shortcomings were pinpointed. In the field test PROMPT was running on mainframe computers and system operation occurred onshore by supporting staff. The advantage of this was that software errors could faster be solved, as it still concerned a field test. The PROMPT system did require some effort to operate. Data on usage of the units were easy to input, or to change, if erroneous values had been inserted. Producing the ranking provided no problem either. The main problem was however in the reporting of the corrective maintenance. Not enough facilities had been provided to secure a reporting which was consistent with the PROMPT database. Remember that a new structuring of equipment had been made in order to set up PROMPT, in terms of failure modes and elements. The existing maintenance management information system did not use these concepts, as it contained only much larger entities, like a whole subunit. The onshore personnel had then to find out which failure mode actually occurred. On the longer term this is considered to be too time consuming.

4.5 Experiences with software

Experiences with the software were positive. Although we used the term decision support system for PROMPT, it is better described as a structured decision system, as the support it provides is always of the same form. As the main advice is at an opportunity and as opportunities occur repeatedly, there is much to say for structured advice. Developing such large computer systems does put a different light on mathematical optimisation. The larger software gets, the more difficult it is handled and checked beforehand. Software errors can produce completely wrong results and thereby destroy all value of optimisation.

A major problem encountered concerned database integrity. In order to secure this, all kinds of protection mechanism were built into the system, next to the already existing protection mechanism provided by FOCUS. This made it very time consuming for the user to change data which were inputted erroneously. The user did want to have the flexibility of changing data like in a spreadsheet, but that is not what database languages provide. Especially the so-called key variables, from which the database is structured, are extremely difficult to change. Users do not always have beforehand the right description of their database elements, implying that later on difficult changes have to be made or that a user is left with a difficult to understand database. The latter may be a cause for future errors.

4.6 Experiences with the advice

Although the prototype software was running on a mainframe, and advice had to be sent by telex, its acceptance was excellent. Every two weeks a ranking list was made and sent offshore, so that if an opportunity occurred it was directly available. As the number of maintenance packages advised for execution was usually small and the priorities differed largely, there was no problem in making the actual schedule. Users found that there were far more opportunities than expected, and that they were well equipped to make effective use of them.

4.7 Evaluation

It is always difficult to evaluate a decision support system as decisions are taken by people, using information from various sources, and the decision support system only has a supporting role. Furthermore, as operations and circumstances change with time and are different from what was envisaged at the start of a project, it is often not possible to make a proper comparison between the situation before and after the introduction of the decision support system. Finally, many advantages or disadvantages are difficult to measure, let alone to quantify. Nevertheless, some evaluation always has to be done, and here we will give some results of the evaluation of PROMPT. We will consider three ways of the evaluation, viz. theoretical comparisons, actual performance comparisons, and finally, management and user acceptance. Benefits of PROMPT were classified into four aspects, being execution of preventive maintenance at opportunities rather than at forced shutdowns, optimisation of the preventive maintenance frequencies, value as a management tool and finally, administrative facilities to learn about the effects of maintenance.

In the theoretical evaluation we assume that the reality is as the PROMPT models assume. In fact, one of the advantages of a decision support system is that one can make this kind of evaluations. Inserting a historical maintenance interval (or an estimation of it) into PROMPT makes it possible to apply the PROMPT programmes to evaluate the average costs under the historical interval and the savings obtained by optimising the maintenance frequency. Results indicate that relative savings of 20% to 30% were obtainable. The absolute savings however, were not that large, as the amount of maintenance suited for execution at opportunities was limited. A calculation of the value of executing preventive maintenance at opportunities is only possible if one is able to describe the alternative precisely. This also requires to set a cost penalty to unit downtime during the annual shutdown. Depending on that value the outcome of the savings varies widely, but it can be substantially. The last two aspects are difficult to quantify. It is a fact that preventive maintenance is always overshadowed by corrective maintenance and that there is a large backlog of activities. One of the problems of maintenance management is to control this backlog. As PROMPT keeps track of what has been done and what still has to be done, it provides management at any time advice on what is most important to be done.

A practical evaluation of PROMPT consists of comparing the actual behaviour, i.e. the actual availability, of the units for which it gives advice with that of other units. This is however very difficult to realise. First of all the actual availability of a unit is a realisation of many random processes and one has to include many units and use a long time scale to make a statistically sound comparison. Furthermore, only a part of the failure modes of a unit were addressed by PROMPT. What aggrevates this problem even more, is that reporting of availability is often lousy. For example, if a unit is being repaired and if it is thereafter not directly needed, then the repair may take far more time and the restoring of its availability may be postponed to the moment it is again needed for service. These events can have a substantial effect on the reported availabilities. As from raw data it is very difficult to find out whether that has occurred, the evaluation is difficult to make. Given the data available there was no evidence that the availability during the PROMPT field test was substantially different from that before.

Let us now turn to the final part of the evaluation, being management and user acceptance. The actual platform maintenance supervisors were very enthusiastic about the PROMPT advice, and did not want to stop the fieldtest. The actual PROMPT advice was in fact very flexible, and provided exactly what they needed. The time needed to do the failure mode analysis and assessment of failure time distribution was considerable (as it often is). There were some complaints on the complexity and difficulties in managing the database. A final version of PROMPT has to be simplified and to require far less data input, certainly when comparing the costs of initialising PROMPT with the amount of money going on in the part of maintenance suitable for execution at opportunities of the units in question. Besides, other problems may overshadow PROMPT temporarily, thereby destroying the discipline needed to maintain it (on the platform in question there was a lengthy shutdown caused by other reasons).

5 Conclusions

PROMPT can be considered as a major step forward in applying scientific methods to maintenance management. It has its pro's and cons. Its pro's are undoubtedly the structured approach leading to an optimisation of preventive maintenance. Its con however, mainly consists of being a complex system, and the long time effort required to initialise it. Future work will be directed at reducing the initialisation effort and simplifying the system while keeping the benefits of the structured approach.

6 Acknowledgements

The authors like to thank Mrss. van Oorschot, Cooper and Hartley from Shell Expro Aberdeen for their cooperation on the PROMPT project. The actual development of PROMPT was done by Ernest Montagne, Joop van Aken, Dick Turpin and the authors.

7 References

Barlow, R.E. and Hunter, L.C. [1960], Optimum preventive maintenance policies, Oper. Res. 8, p. 90-100.

Barlow, R.E. and Proschan, F. [1965], Mathematical Theory of Reliability. New York: John Wiley.

Bäckert, W. and Rippin, D.W.T.[1985], The Determination of Maintenance Strategies for Plants Subject to Breakdown. Computers Chemical Engineering 9 p. 113-126.

Berg, M.B., [1980], A Marginal Cost Analyses for Preventive Maintenance Policies. European Journal of Operational Research. 4 p. 136-142.

Cho, D.I. and Parlar, M. [1991], A survey of maintenance models for multi-unit systems, Eur. Journ. Oper. Res., 51, p. 1-23.

Dekker, R. and Smeitink, E. [1991],"Opportunity-based block replacement: the single component case", Eur. Journ. of Oper. Res., 53, p. 46-63.

Dekker, R. and Smeitink, E. [1994], Preventive maintenance at opportunities of restricted duration, Naval Research Logistics, 41, p. 335-353.

Dekker, R. and Dijkstra, M.C. [1992], Opportunity-based age replacement: exponentially distributed times between opportunities, Naval Research Logistics, 39, p.175-190.

Dekker, R. [1992], Applications of maintenance optimisation models: a review and analysis, report Econometric Institute 9228/A, Erasmus University Rotterdam, The Netherlands, to appear in the Reliability Engineering and System Safety.

Dekker, R. [1995], Integrating optimisation, priority setting, planning and combining of maintenance activities, Eur.Journ. Oper. Res., 82, p. 225-240.

Dekker, R. and Smeitink, E. [1994], Preventive maintenance at opportunities of restricted duration, Naval. Res. Log., 41, p. 335-353.

Dekker, R. [1989], Use of expert judgement for maintenance optimization. First Report of the ESRRDA Project group on expert judgement.

Duncan, J. and Scholnick, L.S. [1973], Interrupt and Opportunistic Replacement Strategies for Systems of Deteriorating Components. Operational Research Quarterly 24: 271-283.

Hanscom, M.A. and Cleroux, R. [1975], The Block Replacement Problem. Journal of Statistical Computations and Simulations 3: 233-248.

Jorgenson, D.W., McCall, J.J. and Radner, R. [1967], Optimal replacement policy, North-Holland, Amsterdam.

Roelvink, I.F.K. and Dekker, R. [1995], Marginal cost criteria for group replacement, Eur. Journ. Oper. Res. 84/2, p.xx.

Sethi, D.P.S. [1977], Opportunistic Replacement Policies. In I.N. Shimi and C.P. Tsokos [eds.], The Theory and Applications of Reliability, vol. 1, pp. 433-447. Academic Press New York.

Sherif, Y.S. and Smith, M.L. [1981], Optimal Maintenance Models for Systems Subject to Failure - A Review. Naval Research Logistics Quarterly 28: 47-74.

Smeitink, E. and R. Dekker [1990], "A simple approximation to the renewal function", IEEE Trans. on Reliab., p.71-75.

Van Aken, J.A., Schmidt, A.C.G., Wolters, W.K. and Vet, R.P. van der [1984], Reliability-based method for the exploitation of maintenance opportunities, Proc. 8th. Adv. in Rel. Techn. Symp. p. B3/1/1-B3/1/8.

Van Dorp, R. [1989], Expert opinion and Maintenance Data to Determine lifetime distributions, Master thesis Delft University of Technology.

Van Noortwijk, J.M., Dekker, R., Mazzuchi, T.A. and Cooke, R.M. [1992], "Expert judgement in maintenance optimization", IEEE Trans. on Reliab., 41, p.427- 432.

Vergin, R.C. and Scriabin, M. [1977], Maintenance scheduling for multicomponent equipment. AIIE Transactions 9: 297-305.

Woodman, R.C. [1967]. Replacement policies for components that deteriorate. Operational Research Quarterly 18: 267-280.

Appendix I - example of advice