MODELLING DISCRETE EVENTS IN SYSTEM DYNAMICS MODELS A CASE STUDY

Dr. E.F. Wolstenholme B.Sc., Ph.D., M.Tech., C.Eng., University of Bradford
Dr. R.G. Coyle B.Sc.,(Eng), Ph.D., University of Bradford

Dr. Wolstenholme graduated in Mining Engineering from Nottingham University in 1965 and, after completing a doctoral study in Mining and a Master's Degree in Operational Research, joined the staff of the National Coal Board, initially as an Operational Research Manager and later a Purchasing Manager. In 1976 he joined the staff of Bradford University Management Centre as a Senior Research Fellow in System Dynamics and currently is a lecturer in Management Science and Assistant Director of the System Dynamics Research Group. His research interests lie in the use of System Dynamics in Mining and in information system design. He is a Chartered Engineer, a Member of the Institute of Mining Engineers and a Member of the Fellowship for Operational Research.

Dr. Coyle graduated in Mining Engineering from Imperial College, London, after which he was a trainee Mining Engineer with the National Coal Board and then a Work Study Engineer at ICI. He has been a Lecturer at Huddersfield Polytechnic and Leicester Polytechnic and before his present position as Director of the System Dynamics Research Group, he was a lecturer in OR at the London Business School. He was Visiting Fellow at Harvard University and the MIT Business School in 1967, has been Visiting Professor at the University of Mons in Belgium and the Polytechnic Institute of Wroclaw, Poland, and is a consultant to many industrial companies throughout the world.

ABSTRACT

It is sometimes argued that discrete and random events are very hard to model in the DYNAMO and DYSMAP syntax, and that the logical facilities in these languages are inferior to those of FORTRAN. This paper demonstrates, by examining a rather complicated modelling problem, that neither of those propositions is necessarily correct. It is shown that, once one understands the underlying problem, DYSMAP is usually far more flexible, efficient, and friendly to the user than is FORTRAN.

A subsidiary aim of the paper is to demonstrate how equations can be built up, piece by piece, to produce required dynamic behaviour, rather than simply emerging somehow from the mind of the modeller.

Background

In underground coal mines, the coal is dug by a mining machine, and carried to the surface by a series of conveyors, via intermediate storage bunkers. In a moderately large colliery there would be several such machines working simultaneously in different parts of the mine and, if a storage bunker becomes too nearly full, or more machines will have to be stopped. If, on the other hand, bunkers become empty, it is more difficult to maintain an adequate flow of coal to the preparation plant on the surface, or perhaps, to an adjacent electrical power station.

Bunker control is, therefore, an important aspect of routine colliery management, but is made more difficult by conditions at the coal face. A mining machine may easily be capable of producing 500 tons/hour on average, but this could vary from 300 to 700 tph, depending on the hardness of the coal, the roof and floor conditions, the skill of the operator, and many other factors. Apart from that variation, the machine may itself be producing or stopped. The machine may stop because it has itself broken down, because the conveyor which it feeds has broken down, or because locally adverse geological conditions, such as a weak roof, stop the machine until they can be corrected. Collectively we shall refer to all such stoppages as 'Breakdowns', and we shall suppose that the machine's conveyor is capable of handling the maximum rate of output from it, so that machine stoppages are not caused by inability to handle production peaks, i.e. stoppages are random exogenous events, and not state-determined.

We require, therefore, to handle no less than three random processes in the model. The first is the random variation in the output rate, which may be regarded as white noise. The second is the discrete change of state from production to breakdown and back again. The third is the length of the production run or the breakdown which may most realistically be regarded as Normal processes, each with its own mean and variance. To handle such complexity in an event-based simulation language such as CSL, would be fairly challenging, but to do so in DYSMAP would often be regarded as beyond the capacity of the language. We shall show that such is not the case.

Apart from these processes, we have to incorporate the fact that the machines only operate during certain shifts, totalling 12 hours out of the 24 hours of each day. For the sake of brevity, we shall not allow for the effect of weekend breaks, though the extension of the method will be fairly obvious.

We require, therefore, to model random events on a coal face, to produce a coal face sector for a model of bunker control policies.

Naturally, if the bunker control policy is inadequate, or the bunkers are just too small, the coal-face machinery will have to be stopped until the surplus coal can be got away. Such stoppages are, however, state-determined and are not included in our model of randomised production, since our purpose here is to explain some equation-writing techniques, and the analysis of bunker control policies is dealt with in Wolstenholme (1980).

Although we have set the example in a coal-mining context, similar phenomena occur in many other production situations, especially in other types of mine.

The need for randomisation of production is clearly established in this particular case, but should be carefully justified in others. One should not include random production in a model merely because it looks impressive, or adds flavour to a model.

Details

The details assumed in this paper are:

1) The mine operates with two production shifts in each 24 hours. The first is from 6 a.m. to midday, the 'day' shift, and the second from 2 p.m. to 8 p.m., more reasonably called the 'afternoon' shift. (i.e. 0600 – 1200 and 1400 – 2000). If the machine is broken down when a shift ends, we assume that the maintenance crews can repair it before the start of the next.

2) The length of unbroken production periods is normally distributed with a mean of 1 hour, and standard deviation 0.1 hours, and the length of breakdowns is similarly distributed with mean 0.5 hours and standard deviation 0.05 hours.

   Actual mining machinery has a far better performance record than that, but we wish to produce a sufficient number of events in a shift to test the equations thoroughly.

Methodological Comment

DYNAMO and DYSMAP are nearly always regarded as continuous simulation languages, and are usually firmly contrasted with the event-based languages, such as CSL. In this example we show how, with a little ingenuity, one can produce events within continuous languages. We do, however, suggest that it is important to use the tool for the job and not to force DYNAMO into doing something it is not designed for, when events have to be treated on a large scale and as the prime focus of attention. In this example, we are considering the problem of control in a system which has both continuous and event-based processes in it, and, for that kind of problem DYSMAP is ideal for the control and continuity aspects, and [illegible] the discontinuous events, as we shall show.

Timing of Main Events

We first calculate the time within the working cycle, because we are dealing with a cyle which may repeat many times in a given LENGTH.

A TDIFF.K=TIME.K–CEND.K (1)

L CEND.K=CEND.J+DT*PULSE(CDUR/DT,CDUR,CDUR) (2)

N CENDD=O

TDIFF = (HRS) Time within working cycle (the cycle is usually a day)

CEND = (HRS) Time at which current cycle started

CDUR = (HRS) Cycle Duration. Usually, CDUR = 24.

Next, we define the Face State to be 1 when a shift is present, and O otherwise.

A FST.K=CLIP(1,0,TDIFF.K,DSST)-CLIP(1,0,TDIFF.K,DSE)
+CLIP(1,0,TDIFF.K,ASST)-CLIP
(1,0,TDIFF.K,ASE) (4)

FST = (1) Zero-One Variable for Face State

DSST = (HRS) Time at which day shift starts

DSE = (HRS) Time at which day shift ends

ASST = (HRS) Time at which afternoon shift starts

ASE = (HRS) Time at which afternoon shift ends.

The four parameters DSST, DSE, ASST and ASE, are expressed in terms of a 24 hours clock!

Some mines produce coal on five days per week, feeding the product direct to a power station, which works on seven days. If this is significant, equation 4 can be modified by multiplying the whole thing by another zero-one variable, WST, for Week State. This can be provided by creating two variables, TWEEK and WEND, exactly corresponding to TDIFF and CEND in equations 1 to 3, and then defining WST, by a CLIP function.

Notice that defining variables such as FST by using suitably timed PULSE functions requires care. The PULSE requires 1 DT to take effect so that the change of state must start 1 DT before the required time, or timing will be that much in error.

Equations for Machine and Face Output

The machine's randomised output is easily defined in terms of its nominal output of 500 tph, and sampled NOISE.

A OUT.K=NOMOUT*(1+FAC*SAMPLE(NOISE(3),
NPERD,O)) (5)

OUT = (T/HRS) Randomised Output of Machine

NOMOUT = (T/HRS) Nominal Output of Machine

FAC = (1) Scaling factor for random variation

NPERD = (HRS) Noise Period

The reason for the SAMPLE function is that we do not want the production rate to change each DT. The value of DT is chosen to be a parameter of the simulation calculation, not of the process being simulated. In this case, the soft coal which might produce the higher output rate could extend for 30 or 40 metres and could take half an hour to cut. In such a case NPERD would be set to 0.5 (hours), whereas DT might have to be very much smaller to obtain satisfactory dynamics from the delays (see Coyle, 1977). The parameter FAC is to force the NOISE to be in the correct range. NOISE samples between -0.5 and +0.5 so that, since NOMOUT = 500, the minimum and maximum outputs would be 250 and 750 tons/hour respectively. We require 300 and 700 which means that FAC has to be 0.8. For a fuller discussion see Coyle, 1979.

The Face Output can then be treated as the randomised output produced from equation 5, multiplied by the Face State, so that the machine does not produce when the face is not working, and by the machine state, to prevent production when the machine is broken down.

A FOUT.K=OUT.K*MST.K*FST.K (6)

Now, the purpose of the NOISE in equation 5 was to represent random encounters with hard coal etc. which produce output variations. Equation 5 continues to produce such stochastic effects, even though the result might be suppressed in equation 6 because no shift is working, or because the machine is broken down. One might, therefore, face the daunting prospect of altering the noise period in equation 5 to allow the sampling of coal variation to start again at the point where it left off when the machine stopped operating. Although that could be done, it is unnecessary, because NPERD will generally be rather small, so the Law of Large Numbers (popularly miscalled the Law of Averages), ensures that we shall get an adequate sample of the random production rates, while the machine *is* working, and we shall not bias the result beyond the inaccuracy of the data by allowing the sampling to continue even when the machine is not, in strict accuracy, working.

It is *always important* to judge whether or not additional fine detail is really going to do more than add a purely illusory respectability to a model.

Duration of Machine States

It is easy to calculate the length of production runs and breakdowns

A LBD.K=DT*INT(NORMRN(0.5,0.05)/DT) (7)

A LPR.K=DT*INT(NORMRN(1.0,0.1)D/T) (8)

LBD = (HRS) Length of a Breakdown

LPR = (HRS) Length of a Production Run

In both equations, a random sample is taken from the normal distribution with appropriate parameters. This is divided by DT to convert the time into time steps, and the integer part function INT ensures that the event will last an exact number of DTs. The multiplication by DT converts back into time units. The point of this procedure, which very slightly distorts the normality of the distribution, is to ensure that, when we use the SAMPLE function in a later equation, the sampling does not occur in the middle of a DT which will make the checking of the model easier. The purist who does not wish to adulterate his normal distribution can reduce the distortion by adding 0.5*DT into the parentheses of the INT function.

At this point, we have to be very careful and exact in our approach, as there are several possibilities. We start by realising that, within a shift, the machine can only alternate between states 1 and 0, because a production run is followed by a breakdown, not by another production run. If, therefore, we had a variable called PERD, which would alternate between the production duration and the breakdown length, as randomly sampled by equations 7 and 8, we could write

A MST.K=SAMPLE(1-OMST.K,PERD.K,O) (9)

L OMST.K=OMST.J+(DT/DT) (MST.J-OMST.J) (10)

N OMST=O (11)

MST = (1) Present Machine State

OMST = (1) Machine State at end of Previous DT

PERD = (HRS) Interval Between Changes of State.

Equation 9 ensures that the state will change back and forth between O and 1, but it is only valid for the conditions within a production shift.

We shall deal, in a moment, with an equation for PERD: first we must revise equation 9 to cope with the start of a shift.

When a shift commences, the machine has to be in state 1, producing. Either it was working at the end of the last shift, in which case it will still be capable of working as nothing has happened to cause it to breakdown, (we ignore the prospect of a roof fall during the inter-shift break), or it was broken down at the end of the shift, in which case the maintenance team have stayed behind to fix it. Strictly speaking, if the machine was working at the end of a shift we ought to allow for the unexpired portion of its random production period to be carried over to the next shift, rather than starting a new run. In practice, the data on run lengths are just not good enough to justify the complication.

If we define an equation for the Old Face State, OFST, by analogy with equations 10 and 11 for the Old Machine State

L OFST.K=OFST.J+(DT/DT)(FST.J-OFST.J) (12)

N OFST=FST (13)

then, at the start of a new shift, and *only* then, FST = 1 and OFST = O, so FST*(1-OFST)=1. We can therefore replace equation 9 by

A MST.K=SAMPLE(MAX(1-OMST.K,FST.K*
(1-OFST.K),PERD.K,O) (14)

The max function will work as follows. If, at the end of a shift, MST=O and OMST=O, then when PERD signals the shift end, MST will change to 1 and, 1 DT later, OMST will also change to 1. When PERD, as described below, generates the start of a new shift, with the old equation 9, 1 - OMST would have been zero, and the machine would have been broken down. However, FST*(1-OFST) will be 1, the MAX function will pick this out, and MST will be 1.

The reader may well find this rather hard to follow from the text description and he should, in any case, run the program listed at the end of this paper and carefully scrutinise the output.

Equations for the Event Period

Finally, we deal with the equation for PERD, the sampling interval in equation 14, and, operationally, the interval between changes of machine state.

The day is divided into five periods: the time to the start of the day shift, the day shift itself, the interval between shifts, the afternoon shift, and the balance of the 24 hour day. The first period is DSST hours long and ends when the shift starts. This would be given by

```
A   PERD.K=DSST*CLIP(1,0,DSST-DT,TDIFF.K)        (15)
```

The DSST-DT is because CLIP takes the first argument, 1 in this case, when the third is greater than *or equal* to the fourth. If we had used DSST alone, the CLIP would still be adding DSST to whatever should happen during the shift.

The inter-shift duration, ASST-DSE, and the time after the end of the afternoon shift are easily seen to be given by more CLIP functions added to equation 15, to get

```
A   PERD.K=DSST*(CLIP(1,0,DSST-DT,TDIFF.K)
X   +(ASST-DSE)* CLIP(1,0,TDIFF.K,DSE)
                         *CLIP(1,O,ASST-DT,TDIFF.K)
X   +(CDUR-ASE)*CLIP(1,O,TDIFF.K,ASE)            (16)
```

Following the arguments used in connection with equation 14, the shifts must each start with a production run, so at the time of shift starts, PERD has to pick up whatever random value equation 8 is generating for LPR. This adds another continuation line to equation 16.

```
X   +LPR.K*FST.K*(1-OFST.K)
```

Finally, during the shifts themselves, we have two possibilities: either the machine was working and OMST=1, in which case the next PERD has to be LBD, or it was stopped, OMST was O, and PERD must become LPR. This can, however, only happen when the face is working and both FST and OFST are one. It will be convenient to interrupt our consideration of the development of the equation for PERD, and introduce a new variable called SPERD, which is the length of a machine run or breakdown according to the randomisation, SPERD will be

```
A   SPERD.K=(LPR.K*(1-OMST.K)+LBD.K+OMST.K)
                             *FST.K*OFST.K      (17)
```

Equation 17 will give the correct contribution of a production run or a breakdown to PERD for the greater part of a production shift. It will, however, give fallacious results when SPERD would cause the time of the next event to come after the end of a shift because the small overlap from SPERD will be added to the value derived from the part of the equation for PERD presented as equation 16, which will mean that the first event of the next shift will not coincide with the commencement of that shift. The effect would scarcely matter with the values quoted above for the distributions of LPR and LBD but, with more realistic values, an appreciable error would creep in. The correct contribution from the machine to PERD must, therefore, take account of the shift-working pattern, and is

```
MIN(SPERD.K,(DSE-TDIFF.K)*TV.K+(ASE-TDIFF.K)
                                    *(1-TV.K))
```

where TV is a zero/one variable recording whether the shift being considered, if there is one, is the day or the afternoon shift, and

```
A   TV.K=CLIP(1,0,DSE,TDIFF.K)                   (18)
```

The complete equation for PERD is therefore

```
A   PERD.K=DSST*CLIP(1,0,DSST-DT,TDIFF.K)
X   +(ASST-DSE)*CLIP(1,0,TDIFF.K,DSE)
                         *CLIP(1,0,ASST-DT,TDIFF.K)
X   +(CDUR-ASE)*CLIP(1,0,TDIFF.K,ASE)
X   +LPR.K*FST.K*(1-OFST.K)
X   +MIN(SPERD.K,(DSE-TDIFF.K)*TV.K+(1-TV.K)
                  *(ASE-TDIFF.K))                (19)
```

It is an important aspect of this paper to examine, not only how the equations are put together, but also just what it is about the DYSMAP language that enables it to work.

The variable TDIFF is continuous in the sense that it acquires a new value with each DT of the simulated LENGTH. This means that PERD, as calculated by equation 19, will at times stay constant for fairly long periods under the influence of the first three lines of that equation, but, during shifts, it will take on a new value each DT. How then, does this square with equation 14?

The answer is that DYSMAP takes a sample at the indicated time and stores the new value of PERD, which happens to be calculated then, and holds it as the fixed interval which must pass before the next sample is to be taken. All values of PERD calculated during the intervening time are ignored. This can be confirmed by producing a very detailed printout and including an extra variable.

```
A   NEXT.K=SAMPLE(TIME.K+PERD.K,PERD.K,
                                   PERD.K)      (20)
```

It will be seen that the values of MST change at exactly the times indicated by NEXT.

Second methodological comment

Equations 14, 17, 18 and 19 are the real heart of this method. Explaining them has taken some space, and understanding the problem in the first place called for some mental effort. That would have been necessary in any case, regardless of the programming language one was using. We contend, however, that, once that comprehension has been established, the actual coding and debugging was *far* easier in DYSMAP than it would have been in FORTRAN, and took up far fewer lines of model. We therefore suggest that, for dynamic problems, the logical facilities in DYSMAP and DYNAMO are by no means inferior to those of FORTRAN; quite the opposite, in fact, once one has mastered the trick of using them.

Performance Indicators for the System

One of the attributes of colliery engineering is the amount of coal lost due to breakdowns, and we now discuss two equations for calculating the cumulative lost production during any given LENGTH. We shall show that neither equation is satisfactory.

When the machine breaks down it may be being simulated as producing at, say, 650 tons/hour because it is in soft coal. In the real world, it would still be in the same patch of coal when it was repaired and would, therefore, resume production at the same rate for the duration of the soft patch. In the model, however, equation 5 will continue to sample away, producing random production rates, for the duration of the breakdown. For example, in one run of the model, the machine broke down at a production rate of 620 tph, but resumed at 560 tph. Altering the model to correct for this would be rather tedious, and may not matter very much if one is to do a fairly large number of runs. There are, however, two possible equations for performance indicators.

The first measures the integral of the difference between the value of OUT produced by equation 5, and FOUT from equation 6, the difference being that the latter goes to zero when production stops, either because the machine has stopped, or because there is no shift present.

```
L   LOUT.K=LOUT.J+DT*(OUT.J-FOUT.J) *FST.J        (21)
N   LOUT=O
```

If FST=O because there is no shift, then no lost output is accummulated.

The alternative is to use

```
L   LNOUT.K=LNOUT.J+DT*NOMOUT*FST.J*
                              (1-MST.J)           (22)
```

which integrates the loss of nominal output for the duration of machine stoppages.

In one run with a 48 hour LENGTH the values happened to be LOUT = 4156 tons, and LNOUT = 3812 tons. Neither value is the 'true' loss of output, which is a concept which is far from easy to define.

Conclusion

We have tried in this paper to show how a rather complicated modelling problem can be handled in DYSMAP, a problem which we at first expected to be far more difficult than eventually proved to be the case. The results demonstrated that the DYNAMO/DYSMAP syntax had far greater modelling power than had previously been asserted (particularly with DYSMAP's extensive diagnostics and its very useful dimensional analysis software – see Ratnatunga, 1979).

We have, however, also sought to show something of the process of *constructing* an equation set to produce previously stipulated behaviour. We feel that in too many cases the equations in a model are presented to the reader as though they somehow emerged fully grown from the mind of the modeller. This has, perhaps, contributed to some degree to the criticism of some system dynamic models. It seems to us that modelling requires that equations be constructed to meet given requirements, and that they must be shown to do so, rather than merely being presented.

We do not suggest that equations of this degree of complexity have any validity or justification outside of this particular project. We would certainly discourage people from using them in a model merely because they look complicated or, even worse, 'scientific'.

```
MAP
* PROGRAM TO ILLUSTRATE RANDOMISED
                              PRODUCTION
NOTE
NOTE FILE NAMED SHIFT-MOD
NOTE
NOTE COALFACE MODEL
NOTE
NOTE FACE STATE 1 MEANS A SHIFT IS PRESENT
NOTE
A   FST.K=CLIP(1,0,TDIFF.K,DSST)-CLIP
                         (1,0,TDIFF.K,DSE)
X   +CLIP(1,0,TDIFF.K,ASST)-CLIP(1,0,TDIFF.K,ASE)
L   OFST.K=OFST.J+(DT/DT) (FST.J-OFST.J)
N   OFST=FST
C   DSST=6
C   DSE=12
C   ASST=14
C   ASE=20
NOTE
NOTE EFFECTS OF RANDOMNESS ON MACHINE
                        POTENTIAL OUTPUT
NOTE
A   OUT.K=NOMOUT*(1+FAC*SAMPLE(NOISE(3),
                              NPERD,0) )
C   NOMOUT=500
C   FAC=0.8
C   NPERD=.5
NOTE
NOTE ACTUAL PRODUCTION WITH EFFECTS OF
                               SHIFTS AND
NOTE RANDOM BREAKDOWNS
NOTE
A   FOUT.K=OUT.K*MST.K*FST.K
NOTE
NOTE CYCLE WITHIN THE WORKING DAY
NOTE
A   TDIFF.K=TIME.K-CEND.K
L   CEND.K=CEND.J+(DT)*PULSE
                    (CDUR/DT,CDUR,CDUR)
N   CEND=O
C   CDUR=24
NOTE
NOTE SAMPLING OF DURATION OF BREAKDOWNS
                                AND RUNS
```

```
NOTE
A   LBD.K=DT*INT(NORMRN(.5,.05)/DT)
A   LPR.K=DT*INT (NORMRN(1,.1) /DT)
NOTE
NOTE CHANGES OF MACHINE STATE
NOTE
NOTE MACHINE STATE=1 MEANS MACHINE IS RUNNING
NOTE
L   OMST.K=OMST.J+DT* (MST.J-OMST.J) /DT
N   OMST=O
A   MST.K=SAMPLE(MAX(1-OMST.K,FST.K* (1-OFST.K) ),PERD.K,O)
X   *FST.K
A   SPERD.K=(LPR.K*(1-OMST.K)+LBD.K*OMST.K) *FST.K*OFST.K
A   PERD.K=DSST*CLIP(1,0,DSST-DT,TDIFF.K)
X   +(ASST-DSE)*CLIP(1,O,TDIFF.K,DSE)*CLIP (1,0,ASST-DT,TDIFF.K)
X   +(CDUR-ASE)*CLIP(1,O,TDIFF.K,ASE)
X   +LPR.K*FST.K*(1-OFST.K)
X   +MIN(SPERD.K, (DSE-TDIFF.K)*TV.K+(1-TV.K)* (ASE-TDIFF.K) )
A   TV.K=CLIP(1,O,DSE,TDIFF.K)
A   NEXT.K=SAMPLE(TIME.K+PERD.K,PERD.K, PERD.K)
NOTE
NOTE PERFORMANCE INDICATORS
NOTE
L   LOUT.K=LOUT.J+DT(OUT.J-FOUT.J) *FST.J
N   LOUT=O
L   LNOUT.K=LNOUT.J+DT*NOMOUT*FST.J* (1-MST.J)
N   LNOUT=O
NOTE
NOTE CONTROL STATEMENTS
NOTE
C   DT=.125
A PRTPER.K=6-STEP(6-DT,6)+STEP (2-DT,12+DT/2) -STEP(2-DT, 14)
X   +STEP(10-DT,20)-STEP(10-DT,30)+STEP(2-DT,36) -STEP(2-DT,38)
X   +STEP(4-DT,44)
C   PLTPER=.0625
C   LENGTH=48
PRINT 1) TDIFF,CEND
PRINT 2) FST,OFST
PRINT 3) MST,OMST
PRINT 4) PERD,SPERD
PRINT 5) LPR,LBD
PRINT 6) OUT,FOUT
PRINT 7) NEXT
PRINT 8) LOUT,LNOUT
PLOT  MST=M,FST=F(0,2)/FOUT=0 (-700,700)
PLOT  PERD=P,LBD=8,LPR=R(0,6)
NOTE
NOTE VARIABLE DEFINITIONS
NOTE
D  ASE=(HRS)  TIME OF END OF AFTERNOON SHIFT
D  ASST=(HRS)  TIME OF START OF AFTERNOON SHIFT
D  CDUR=(HRS)  LENGTH OF WORK CYCLE, USUALLY A DAY
D  CEND=(HRS)  TIME OF END OF WORK CYCLE
D  DSE=(HRS)  TIME OF END OF DAY SHIFT
D  DSST=(HRS)  TIME OF START OF DAY SHIFT
D  DT=(HRS)  SOLUTION INTERVAL
D  FAC=(1)  SCALING FACTOR IN NOISE ON PRODUCTION RATE
D  FOUT=(T/HRS)  ACTUAL MACHINE OUTPUT WHEN WORKING
D  FST=(1)  STATE OF FACE:STATE=1 WHEN A SHIFT IS PRESENT
D  LBD=(HRS)  RANDOMISED LENGTH OF A BREAKDOWN
D  LENGTH=(HRS)  SIMULATED DURATION
D  LNOUT=(T)  LOST OUTPUT ON NOMINAL BASIS
D  LOUT=(T)  LOST OUTPUT ON 'ACTUAL' BASIS
D  LPR=(HRS) RANDOMISED LENGTH OF A PRODUCTION RUN
D  MST=(1) STATE OF MACHINE:STATE=1 WHEN MACHINE IS WORKING
D  NEXT=(HRS) TIME AT WHICH NEXT CHANGE OF MACHINE STATE IS EXPECTED
X                TO OCCUR, ACCORDING TO THE RANDOM SAMPLING OF BREAKDOWNS
X                AND RUN LENGTHS
D  NPERD=(HRS) PERIOD OF NOISE ON PRODUCTION RATE
```

107 D NOMOUT=(T/HRS) NOMINAL MACHINE OUTPUT

108 D OFST=(1) OLD FACE STATE

109 D OMST=(1) STATE OF MACHINE AT END OF PREVIOUS DT

110 D OUT=(T/HRS) RANDOMISED MACHINE OUTPUT

111 D PERD=(HRS) TIME TO NEXT EVENT

112 D PLTPER=(HRS) PLOTTING INTERVAL

113 D PRTPER=(HRS) PRINTING INTERVAL

114 D SPERD=(HRS) DURATION OF EVENT FROM SAMPLING

115 D TDIFF=(HRS) TIME WITHIN WORKING DAY

116 D TIME=(HRS) SIMULATED TIME

117 D TV=(1) TIME VARIABLE TO DENOTE WHICH PART OF DAY IS

118 X HAPPENING

119 RUN DYNAMICS OF RANDOM BREAKDOWNS

120 +

References

Coyle, R.G., Management System Dynamics, John Wiley & Sons, 1977.

Coyle, R.G., Equations for Systems, University of Bradford Printer 1979. (see particularly, examples 20 & 30)

Ratnatunga, A.K., DYSMAP Users Manual, University of Bradford Printer, 1979.

Wolstenholme, E.F., Application of System Dynamics to the Design of Control Policies for Collery Coal Clearance Systems. Paper AFCET Conference, November 1980.