Proceedings of the 2002 Winter Simulation Conference
E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.

PROJECT MANAGEMENT – CRITICAL PATH METHOD (CPM) AND PERT SIMULATED WITH PROCESSMODEL

LeRoy F. Simmons

Information Systems and Operations Management
Loyola College
4501 N. Charles St.
Baltimore, MD 21210, U.S.A.

ABSTRACT

ProcessModel simulation allows the activity times of a project to be represented by a variety of distributions and further the resulting project time may also be represented by a variety of distributions. This is a significant improvement over the traditional methods of CPM and PERT. Program Evaluation and Review Technique (PERT) takes the CPM network and adds distributions to represent the activity times of the project. CPM assumes the activity times to be constant, which is not likely in the real world. PERT assumes the activity times of the project to be distributed as Beta distributions and the resulting project time to be a Normal distribution. This is better than assuming them to be constant, but these assumptions are needlessly restrictive. This paper demonstrates how simulation with ProcessModel can remove these needless restrictions.

1 EXAMPLE FOR CPM

Chase et al.(2001 on page 68) present a project for designing a computer. The project requires the seven activities in Table 1 be completed.

Table 1: CPM Example

Activity	Designation	Node #	Immediate Predecessors	Time (weeks)
Design	A	1	-	21
Build prototype	B	2	A	5
Evaluate equipment	C	3	A	7
Test prototype	D	4	B	2
Write equipment report	E	5	C,D	5
Write methods report	F	6	C,D	8
Write final report	G	7	E,F	2

The network in Figure 1 was developed using the Note #'s and the Immediate Predecessors.

Simulating this network with ProcessModel provided Figure 2.

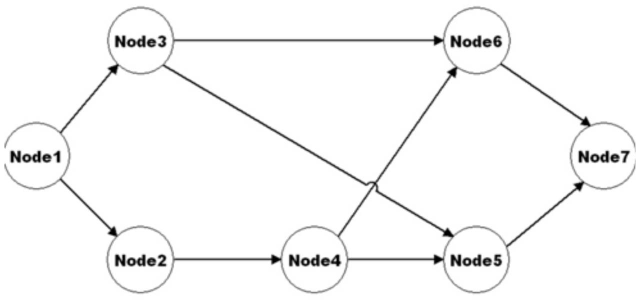


Figure 1: Network of Project

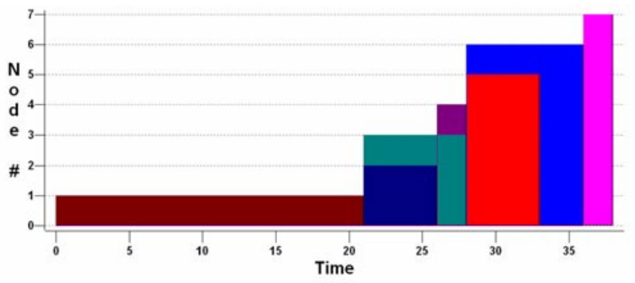


Figure 2: Results from ProcessModel

The "Design" activity (Node1) starts the project and finishes after 21 weeks. At that point the "Build Prototype" (Node2) and "Evaluate equipment"(Node3) activities begin. The Node2 activity ends after week 26 and at that point the "Test prototype" (Node4) activity begins. …

The project is completed in 38 weeks. Since the "Write equipment report" (Node5) activity ends without another activity beginning, this activity is said to have "slack" time. The "Write equipment report" activity could take another 3 weeks and the project would not be delayed. All the other activities are said to be "critical", because they have no slack time. If a critical activity takes longer to complete than expected, the project will also take longer to complete.

Once the above network is modeled with ProcessModel it is easily taken to PERT and beyond by simply adding distributions to the model.

2 EXAMPLE FOR PERT

Chase et al. (2001 on page 71) suggested using the data in Table 2 to perform a PERT analysis.

Table 2: PERT Example

			Time		
Activity	Designation	Node #	optimistic	most-likely	pessimistic
Design	A	1	10	22	28
Build prototype	B	2	4	4	10
Evaluate equipment	C	3	4	6	14
Test prototype	D	4	1	2	3
Write equipment report	E	5	1	5	9
Write methods report	F	6	7	8	9
Write final report	G	7	2	2	2

The following graph (Figure 3) was generated in Excel from the output from a ProcessModel simulation. The graph shows the distribution of the project completion time.

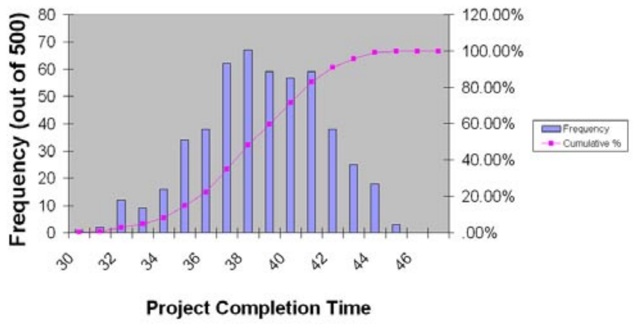
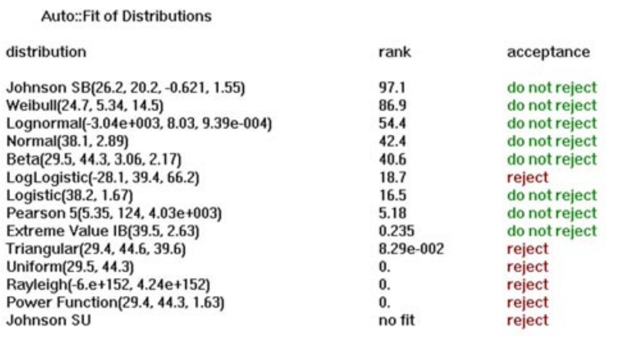


Figure 3: Completion-Time Graph

Chase et al. (2001) show how the traditional approach to PERT analysis would calculate the probability of the project being completed in 35 weeks. Their result was that there is a 19% chance the project will be completed in 35 weeks. The results from the ProcessModel simulation suggest that 15% would be a better estimate. The traditional approach estimated the project-time variance to be 11.90 while ProcessModel's estimate was 8.35. The traditional approach has overestimated the variability of the distribution. Further, the following output (Table 3) from ProcessModel suggests that the project completion time is better estimated with distributions other than the Normal distribution.

<u>How does one get ProcessModel to perform a CPM/PERT-like analysis?</u>

Table 3: Output from ProcessModel



Auto::Fit of Distributions

distribution	rank	acceptance
Johnson SB(26.2, 20.2, -0.621, 1.55)	97.1	do not reject
Weibull(24.7, 5.34, 14.5)	86.9	do not reject
Lognormal(-3.04e+003, 8.03, 9.39e-004)	54.4	do not reject
Normal(38.1, 2.89)	42.4	do not reject
Beta(29.5, 44.3, 3.06, 2.17)	40.6	do not reject
LogLogistic(-28.1, 39.4, 66.2)	18.7	reject
Logistic(38.2, 1.67)	16.5	do not reject
Pearson 5(5.35, 124, 4.03e+003)	5.18	do not reject
Extreme Value IB(39.5, 2.63)	0.235	do not reject
Triangular(29.4, 44.6, 39.6)	8.29e-002	reject
Uniform(29.5, 44.3)	0.	reject
Rayleigh(-6.e+152, 4.24e+152)	0.	reject
Power Function(29.4, 44.3, 1.63)	0.	reject
Johnson SU	no fit	reject

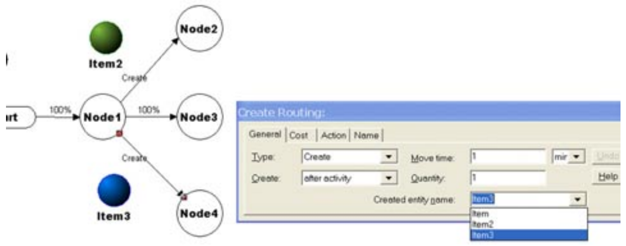
3 CPM AND PERT USING PROCESS MODEL

The modeling trick that enables CPM/PERT networks to be modeled in ProcessModel involves the following of two rules.

- **<u>Rule 1</u>** – When route(s) [arc(s)] exit a Node, there must be one "100%" route. If there is more than one route, the others are made "Create" routes.

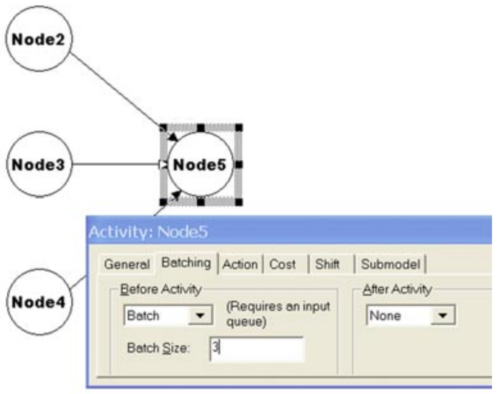
  In Figure 4 when Node1 is finished Nodes 2-4 may begin.
- **<u>Rule 2</u>** – When a Node has more that one route coming into it, "Batching" is used at that Node, where the batch size is the number of routes coming into the Node.

  In Figure 5 Node5 will not begin until all of it immediate predecessors (Node2, Node3, and Node4) are finished.



Type: Create
Move time: 1 min
Create: after activity
Quantity: 1
Created entity name: item3

Figure 4: Implementation of Rule 1



Before Activity: Batch
Batch Size: 3
After Activity: None

Figure 5: Implementation of Rule 2

The following template will get you started. Free download of the template is available at `(http://leroysimmons.com)`.

The template (Figure 6) has all the ProcessModel structure that is needed to perform a CPM/PERT for up to twenty activities (Nodes). If more activities are required additional Nodes can be added.

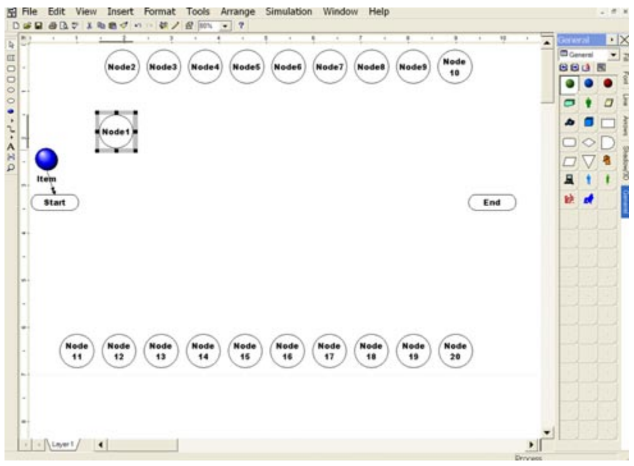
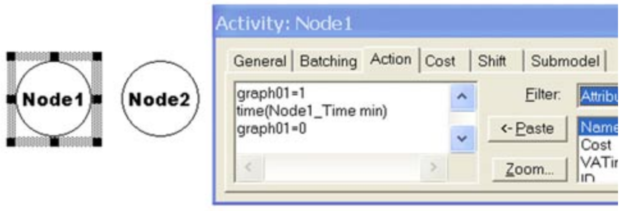


Figure 6: ProcessModel Template

Each Node is preprogrammed as in Figure 7. The variable graph01 is set to 1 when Node1 activity begins and then it is return to 0 when it ends. When this variable is later graphed it will represent the occurrence of the Node1 activity. Node2 is preprogrammed with graph02 =2, Node3 is preprogrammed with graph03=3, and so on. The variable Node1_Time for Node1 allows the activity time to be entered elsewhere. Node2 has been preprogrammed Node2_Time, Node3 has been preprogrammed Node3_Time, and so on.



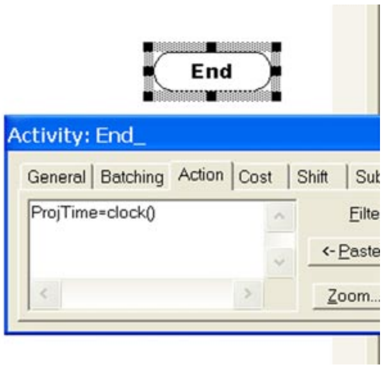
```
graph01=1
time(Node1_Time min)
graph01=0
```

Figure 7: Preprogrammed Nodes

The end activity (Figure 8) is a dummy activity that ends the project thus allowing the project time (variable – ProjTime) to be recorded.

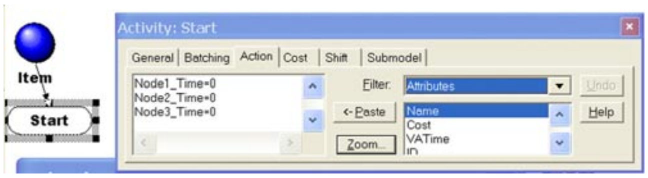
Stored in the Start activity (Figure 9) are all the activity times. As seen below the CPM/PERT template has all the activity times by default set to zero. By replacing the zeros with constants you will have the ProcessModel version of CPM. By replacing the zeros with distributions you will have the ProcessModel version of PERT.

Figure 10 shows what the Chase example would look like using the CPM/PERT template.



```
ProjTime=clock()
```

Figure 8: End Node



```
Node1_Time=0
Node2_Time=0
Node3_Time=0
```

Figure 9: Start Node

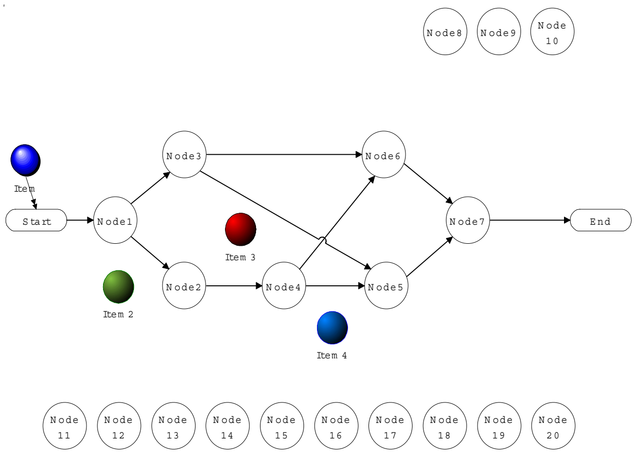


Figure 10: Chase Example Using Template

Free download of this completed model is available at `(http://leroysimmons.com)`.

REFERENCE

Chase, Richard, Nicholas Aquilano and Robert Jacobs, 2001, *Operations Management for Competitive Advantage*, ninth edition, McGraw-Hill.

AUTHOR BIOGRAPHY

LEROY F. SIMMONS is a Professor of Information Systems and Operations Management at Loyola College of Maryland. He received his Ph.D. from the University of Tennessee in 1973. His research interests are in business forecasting and the simulation of business' processes.