# A Simulation-Based Time Reduction Approach for Resource Constrained Design Structure Matrix

Hisham M. Abdelsalam, Mohamed H. Rasmy, and Hayam G. Mohamed

***Abstract*—Project scheduling is an important research and application area in engineering management. Recent research in this area addresses resource constrained as well as stochastic duration. This paper demonstrates a simulation-based model for solving resource-constrained research and development projects (R&D) scheduling problems and calculates the total time of DSM using discrete event simulation. This model used Design Structure Matrix (DSM) to represent the information exchange among various tasks of a project, instead of a simple binary precedence relationship. DSM is capable of quantifying the extent of interactions as well. In particular, these interactions are characterized by rework probabilities, rework impact and learning. Results obtained showed an improved solution compared to earlier studies.**

***Index Terms*—DSM (Design Structure Matrix), PSO (Particle Swarm Optimization), RCPSP (Resource Constrained Project Scheduling Problems).**

## I. Introduction

Many research development (R&D) projects are inherently complex, thereby making effective management of the tasks, resources, and teams necessary to bring new products to market problematic. Frequently, managers of such R&D projects are overwhelmed with more complicated factors such as stochastic task times, ill-defined specifications, complex interrelationships between tasks, and information dependencies. Several researchers have recently developed an alternative project management tool called the Design Structure Matrix (DSM) that explicitly bears in mind the iterative nature of research development projects [1].

The purpose of this paper is to construct a simulation model using Discrete Event Simulation to calculate the total time of resource constrained design structure matrix. DSM works as a system analysis tool, which provides a compact and clear representation of a complex system. It captures the interactions/interdependencies/interfaces between system elements. It can also be considered a project management tool which provides a project representation that allows for feedback and cyclic activity dependencies.

This paper is organized as follow firstly, gives a brief introduction on design structure matrix (DSM), secondly discusses the related work of resource constrained DSM simulation, thirdly describes the discrete event simulation model, and finally discusses results obtained, conclusions and future work.

Manuscript received June 25, 2013; revised August 27, 2013.

The authors are with the Faculty of Computers and Information, Cairo University, 5 Ahmed Zewail st., Orman, Giza, 12311, Egypt (e-mail: h.abdelsalam@ fci-cu.edu.eg, hayam@ fci-cu.edu.eg).

## II. Design Structure Matrix

The basic DSM is a simple binary (A cell can hold one of only two values (0, 1), or (an "x" mark, an empty cell)) *n* - square matrix (where *n* is the number of system elements), with *m* non-empty elements (where *m* is the number of couplings among different system elements). A typical DSM is shown in Fig 1. Activity names are placed on the left hand side of the matrix as row headings and across the top row as column headings in the same order (order of their execution); a main DSM assumption is that activities are undertaken in the order listed from top to bottom. An off-diagonal mark (**x**) represents a coupling (an information flow, or a dependency) between two activities. If an activity *i* depends on (receives information from) activity *j*, then the matrix element (rowi , columnj ) *i j* contains an off diagonal mark (**x**) otherwise the cell is empty.

Marks below the diagonal (sub-diagonal marks) are indicative of feed-forward couplings (i.e. from upstream activities to downstream activities), while those above the diagonal (super-diagonal) represent feedback couplings (i.e. from downstream activities to upstream activities). As they imply iterations, the latter type of couplings should be eliminated if possible or reduced to the maximum extent. If certain feedback couplings cannot be eliminated, the activities can be grouped into iterative sub-cycles (or circuits). For example, in Fig. 1, activities (1, 2, 3) and activities (6, 7, 8, 9, 10) are grouped into two iterative sub-cycles. A primary goal in basic DSM analysis is to minimize the number of feedbacks and their scope by restructuring or re-architecting the process [1]. In other words, by re-sequencing the execution of the activities to get the DSM into as lower-triangular form as possible. To achieve this goal, Steward [2], [3] proposed a two phase approach: partitioning and tearing. In addition to Steward's partitioning heuristic, several methods are found in literature: the Path Searching method [4], the Reach ability Matrix method [5], the Triangularization Algorithm [6] and the Powers of the Adjacency Matrix Method [7].

## III. Theoretical Background of Resource Constrained DSM Simulation

Browning and Eppinger [8] used simulation to analyze iterative processes based on a DSM model as well as to account for normal variance of development task durations. However, this first DSM simulation method is based on rather restrictive assumptions regarding activity concurrency and rework, Roemer *et al*., 1999 discussed time-cost trade-offs in multiple overlapped tasks. However, this model is limited to a single path, assuming that sequential iterations take place

among sub-tasks within a task.

Abdelsalam and Bao [9] presented a simulation-based optimization framework that determines the optimal sequence of activities execution within a product development project that minimizes project total iterative time given stochastic activity durations. Browning [10] used a DSM-based model of a process to quantify a process configuration's expected cost and duration and their variances. Cost, duration (or schedule), and variances in both are largely a function of the number of iterations required in the process execution and the scope or impact of those iterations. Since iterations may or may not occur (depending on a variety of variables), this model treats iterations stochastically, with a probability of occurrence depending on the particular package of information triggering rework.

Zhang [11] presented a simulation –based optimization model for solving resource-constrained product development project scheduling problems .the model uses the design structure matrix (DSM) to represents the information exchange among various tasks of the project, modelling based on DSM allows iteration to take place .adding resource factors to DSM simulation. He not only model the constraints posed by resource requirements, but also explore the effect of allocating different amount of resources on iterations.

In this research, we develop discrete event simulation model to calculate the total time of resource constrained DSM. Adding to the previous work the resource-constrained assignment, different type of distribution allowed for each activity duration, and concurrency assumption in achieving the activity.

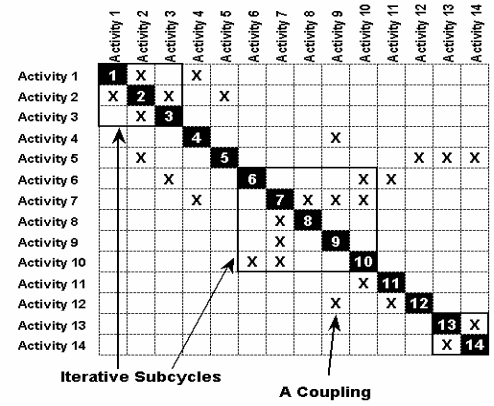


Fig. 1. Typical DSM.

## IV. Methodology

In this Section we discuss the methodology used to calculate the total time of the resource constrained DSM, we will discuss the model in details, characteristics, variables and model steps.

### A. Discrete Event Simulation

Discrete event simulation is a class of simulations that rely on repeated random sampling to compute results. They are commonly used when it is infeasible or impossible to compute an exact result with a deterministic algorithm. It is an extension of Browning's work on DSM simulation [8], [10].

### B. Model Characteristics

The model characteristics are as follow,

- Time of the activity is stochastic and follow any distribution according to user preferences like (e.g. Triangular, uniform)
- Concurrency is allowed in performing the activities.
- Amount resources available are limited.
- Several types of resources are allowed
- Total amount of resource available is more than any single activity requirements; at any time, at least one activity can be executed.
- SOF criterion is used to resolve resource conflict.
- Learning curve (LC) is a measure of the activity when it is repeated, when activity does work for more than one time the duration of that activity is represented by learning curve.

TABLE I: List of Simulation Variables

| Variable | Description |
|---|---|
| *n* | Number of activities |
| *m* | Number types of resources |
| Act_ID | vector of length n which contain the activities name |
| Act_dur | Actual duration vector, vector of length n which containing the duration value of each activity that will calculated once by sampling the duration of the activity in the initialization step |
| Sim_Clock | A variable giving the current value of simulated time |
| LC | Learning Curve which is a measure of the characteristic of the activity when it repeats. When activity does work for more than once the duration of that activity is represented by Learning Curve |
| New_dur | New duration vector, a vector of length n containing the last duration value for each activity |
| F_dur | Finished duration vector, a vector of length n containing a duration value for each activity after calculating it by the equation:<br>F_dur = New_dur + Sim_Clock |
| Iter_num | number of iterations, indicates the number of times the activity worked during the simulation |
| $EM_{n2}$ | Event matrix of the activities, put in it Act_ID and F_dur of the activities which will work in a specific time step |
| MayWN | May Work Now vector, a vector of length *n* with [0, 1] entry for each activity, indicates that the activity in the May work or not depending on resource availability. If no problem in resources, so WN=MayWN, and put in EM. |
| WN | Work Now vector, a vector of length *n* with [0, 1] entry for each activity, indicates that the activity in the EM or not. WN vector will be 1 if the activity already in the EM, 0 otherwise |
| FV | Finished vector: a vector of length *n* with [0, 1] entry of each activity, indicates if the activity finished or not. 1 if the activity finished, 0 otherwise |
| RA | vector of length m represents resource availability for each type |
| Res | matrix of size n*m represent amount requested for each activity from each type of resources |
| $RP_{ji}$ | *P* (activity *i* causes rework for activity *j*), the rework probabilities of one activity causing the rework of another |
| $RI_{ji}$ | Rework impact, % of activity *j* reworked because of input change from activity *i* |

### C. Simulation Model Variables

The simulation model variables are as shown in Table I.

### D. Algorithm

The following Table provides the detailed steps for the Simulation model.

TABLE II: THE PROPOSED ALGORITHM

**Inputs**: input variables of the project.

**Outputs**: the expected total time of the project.

**1:** Initialize these vectors, New_dur vector, Iter_num, WN, FV, and Sim_Clock with zero.

**2:** Sample the time of the activities, according to any distribution.

**3:** Initialize the $EM_{n2}$ by identifying activities that have no predecessors to start immediately to be in the May WN vector.

**4:** check on resource availability, if there is no problem in resources (i.e. amount available of resources >= amount required for a activity) put MayWN in WN vector and put activity in EM, then this activity can start; otherwise go to step 5.

**5:** assign resources to activity according any selected rule like SOF, MOF, FCFS, MRF in our model; we use the SOF rule.

**6:** Calculate F_dur ,F_dur = Act_dur + Sim_Clock, then Put 1 in the WN cell that corresponds to this activity.

**7:** Arrange EM, Arrange the durations of the activities in ascending order, Update the Sim_Clock with the least duration, FV = 1, and modify the duration of this activity by the following equation: New_dur = New_dur * LC.

**8:** Find all activities that have Feed-Forward relation and check the resources availability. Next, add them in the EM.

**9:** Find all activities that have Feed-back relation and add them in the EM, modify WN of this activity to be zero, remove this activity from the EM, arrange the EM ascending, and, at last, update the Sim_Clock again by the first activity.

**10:** Find Feed-Forward relation for the finished activity: search for the activities which the finished activity input to them. If those depend only on the finished activity, OR depends on another activities, but they were finished and they are not put in the WN and FV. After check the resources availability, put them in the EM, calculate F_dur (F_dur = New_dur + Sim_Clock), and then set WN of those activities = 1.

**11:** Find Feed-back relation for the finished activity: search for the activities which have a feed-back relation, check if this activity will start or not by generating a random number. If Random Number < PR and WN of this activity = 0 and finished =1, check the resource availability then put this activity in the EM, modify its time by the following equation: New_dur = New_dur * LC *RI.

**12:** For the activities which will rework, search for the activities that have a feed-forward relationship, then check if those activities will put in the EM by generating random number as before, but taking into consideration that FV of them = 1 and WN of them = 0.

**13:** repeat these steps until all activities are executed.

## V. RESULTS

We present the results obtained using our simulation model in solving numerical example, this example contain a set of activities with stochastic duration follow a triangular distribution, DSM define the relation feed-back and feed–forward between activities, Rework probability matrix, Rework Impact Matrix [8], resource availability, resource requested for each activities, number of simulation runs =200,We applying our simulation model to several cases without concurrency and with concurrency assumption and figure the results obtained.

Base Case: solving the original Example without resource constrained like [8], we find this output, Fig. 2, shows probability mass functions (PMFs) and cumulative distribution functions (CDFs) for Duration outcome the mean duration, E [Duration], is 141.64 days with a standard deviation, $\sigma_S$, of 23.20 days. Median duration is 135.06 days, even though the Duration distribution is skewed to right (skewness = 0.74). We note from this results that this model give results better than Tyson R. Browning simulation model.

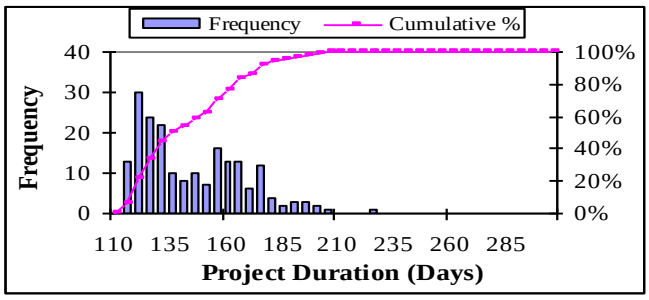


Fig. 2. Duration PMFs, CDFs output in base case.

Resource Constrained Case: using the original example [10] and adding to this resource constrained feature. Number type of resources=2: Type 1=10 units, Type 2=12 units. Each activity has resources requested from each type. After we ran the model, we reached this output: Fig. 3 shows *probability mass functions* (PMFs) and *cumulative distribution functions* (CDFs) for duration outcome; the mean duration, E [*Duration*], is 231 days with a standard deviation, $\sigma_S$, of 20.13 days. Median duration is 230.40 days, even though the duration distribution is skewed to left (skewness = 0.04).

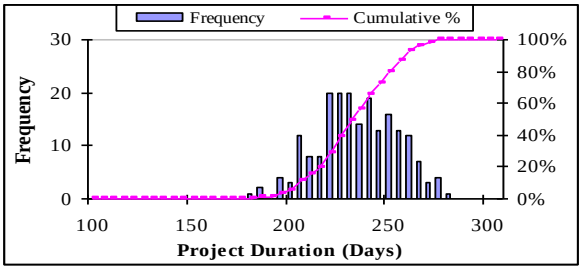


Fig. 3. Duration PMFs, CDFs output in resource constrained case.

### A. Model Validation

In this section, we introduce different scenarios to prove that the results of our model are logical.

Case 1: If we increase the total amount of available resources, type 1 of resource = 50, type 2 of resource=50, using the example data given in [8], we found out the following output: Fig 4 demonstrates *probability mass functions* (PMFs) and *cumulative distribution functions* (CDFs) for duration outcome; the mean duration, E [*Duration*], is 171 days with a standard deviation, $\sigma_S$, of 21.73 days. Median duration is 167.89 days, even though the duration distribution is skewed to right (skewness = 0.29). We can infer from these results that the expected project total time will decrease (to be 171 days), which proves the inverse relationship between the total amount of resources and the total time of the project.

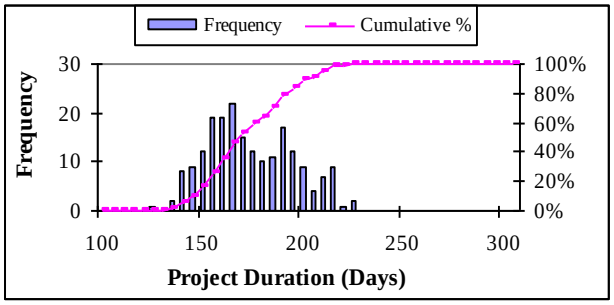


Fig. 4. Duration PMFs, CDFs output in case 1.

Case 2: if we decrease the total amount of available resources type 1 = 5 units, type 2 = 6 units, using the example data given , we obtained this output: Fig. 5 shows *probability mass functions* (PMFs) and *cumulative distribution functions* (CDFs) for duration outcome; the mean duration, E [*Duration*], is 239 days with a standard deviation, $\sigma_S$, 21.55 days. Median duration is 239.40 days, even though the duration distribution is skewed to Left (skewness = 0.17). We can conclude from the previous results that the expected project total time will increase, which proves the inverse relationship between the total amount of resources and the total time of the project.

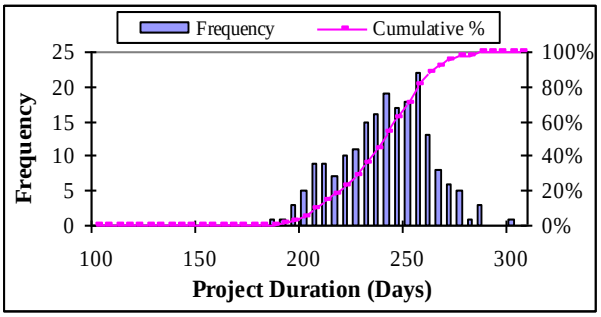


Fig. 5. Duration PMFs, CDFs output in case 2.

Case 3: If we increase one type of resources and control the second to become constant:

Case 3a: type one increases to be = 50 units, and type two remains constant=12 units. Using the example data given in [8] we found the output the mean duration, E [*Duration*], is 229 days with a standard deviation, $\sigma_S$, 21.81 days. Median duration is 231.40 days, even though the duration distribution is skewed to right (skewness = 0.31).

Case 3b: Type one =10 units remaining constant, and type two increases to be = 50 units. By using the example data in [8], we reached this output, E [*Duration*], is 167 days with a standard deviation, $\sigma_S$, of 19.62 days. Median duration is 165.27 days, even though the duration distribution is skewed to right (skewness = 0.48). The expected total time decreased so much from the original since most of the activities depended on the second type of resources which did not constrain the activities achieved.

Case 4: in our model we have the flexibility to make each activity follow different type of distribution; we have the following distributions to deal with. Gamma, uniform, triangular... Let assume the activities in [8] all follow different distribution we obtain this results. Fig. 6 shows *probability mass functions* (PMFs) and *cumulative distribution functions* (CDFs) for Duration outcome the mean duration, E [*Duration*], is 378.76 days with a standard deviation, $\sigma_S$, of 43.06 days. Median duration is 379.51 days, even though the Duration distribution is skewed to right (skewness = 0.39).

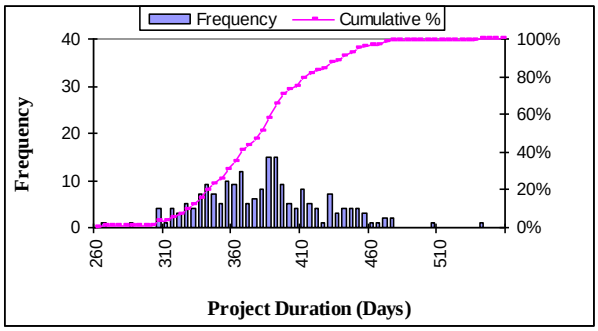


Fig. 6. Duration PMFs, CDFs output in case 4.

### B. Concurrency Assumption

In our model, we apply the concurrency assumption with concurrency factor = 0.5, which does not mean that an activity starts when another activity is finished (finish-to-start relationships); the activity will not wait till all activity duration is finished, but it can start when 0.5 of the duration is finished only. This concurrency factor can change according to user preferences. When we ran the simulation model with the concurrency assumption, we obtained the results shown in Table III. We are interested only in the expected total time of the project. When we compared between with-concurrency and without-concurrency assumptions, we noted that the expected total time decreased in some cases and increased in others depending on the interdependencies between activities.

TABLE III: RESULTS OBTAIN USING CONCURRENCY ASSUMPTION IN SEVERAL CASES

| Case | Expected total time of the project |
|---|---|
| Base | 135 days |
| Resource constrained | 227 days |
| 1 | 165 days |
| 2 | 238 days |
| 3 | Case a=225 days |
| | Case b=164 days |
| 4 | 376 days |

## VI. CONCLUSION AND FUTURE WORK

In this paper we have presented a simulation-based model for solving RCPSPs. we use DSM as our primary data repository and representation tool. It provides a clear visualization, eases modifications to RCPSPs, and allows us to use many mathematical analytic methods to analyze a problem. More importantly, rework, which commonly occurs in process development projects, can now be easily modeled through the use of rework probabilities and rework impacts using DSM. In our simulation–based approach allows us to generate according to rework probabilities and rework impacts stochastically ,instead of a single value of total project durations ,we examine a distribution of total project

durations, as result of many discrete event simulation runs. We can see clearly how rework evolves and contributes to the total project durations. In the future we will try to use PSO to find the optimal sequence within the resource constrained DSM, and find the minimum time using RCPSPs simulation model, and find the maximum utilization of the available resources.

## REFERENCES


[1] T. R. Browning, "Applying the design structure matrix to system decomposition and integration problems: A review and new directions," *IEEE T Eng Manage*, vol. 48, no. 3, pp. 292-306, 2001.

[2] D. V. Steward, *Systems Analysis and Management: Structure, Strategy, and Design*, New York, PBI, 1981.

[3] D. V. Steward, "The Design Structure System: A Method for Managing the Design of Complex Systems," *IEEE T Eng. Manage*, vol. 28, no. 3, pp. 71-74, 1981.

[4] D. A. Gebala and S. D. Eppinger, "Methods for Analyzing Design Procedures," in *Proc. the ASME Third International Conference on Design Theory and Methodology*, pp. 227-233, 1991.

[5] J. N. Warfield, "Binary matrices in system modeling," *IEEE T Syst Man Cyb*, vol. 3, no. 5, pp. 441-449, 1973.

[6] N. L. Kusiak and J. Wang, "Reengineering of Design and Manufacturing Processes," *Comput. Ind. Eng.*, vol. 26, no. 3, pp. 521-536, 1994.

[7] W. P. Ledet and D. M. Himmelblau, "Decomposition Procedures for the Solving of Large Scale Systems," *Adv Chem Eng*, vol. 8, no. 3, pp. 185- 254, 1970

[8] T. R. Browning and S. D. Eppinger, "Modelling impacts of process architecture on cost and schedule risk in product development," *IEEE T Eng Manage*, vol. 49, no. 4, pp. 428-442, 2002.

[9] H. M. Abdelsalam and H. Bao, "A Simulation-based Optimization Framework for Product Development Cycle Time Reduction," *IEEE T Eng Manage*, vol. 53, no. 1, pp. 69-85, 2006.

[10] T. R. Browning, "Modelling and Analyzing Cost, Schedule, and Performance in Complex System Product Development," Ph.D. dissertation, MIT, Cambridge, MA, 1998.

[11] Y. Zhang, "A simulation-based resource optimization and time reduction model using design structure matrix," Thesis, Massachusetts Institute of Technology, 2008.



**Hisham M. Abdelsalam** is an associate professor of operations research and decision support and the director of the decision support and future studies center in the Faculty of Computers and Information, Cairo University. Dr. Abdelsalam holds a master of science and a Ph.D. in mechanical engineering from Old Dominion University, Norfolk, Virginia, USA. He obtained his bachelor degree with honors in mechanical engineering from Cairo University (Cairo, Egypt). Professionally, Dr. Abdelsalam has acted as the director of research and Development in the National Management Institute (NMI) and, also, as the manager of the Future Studies Center, Information and Decision Support Center (IDSC), the Egyptian Cabinet, Cairo, Egypt. Dr. Abdelsalam has led several funded research projects, and his research interests extend to cover the areas of decision support, modeling and simulation, new product development projects, supply chain management, and e-government.



**Hayam G. Mohamed** received her B.Sc. degree in operations research and decision support from Cairo University, Cairo, Egypt, in 2005, and M.Sc. in operations research and decision support, Cairo university, in 2011. Her work experience is primarily related to academic. Mrs. Gamal works at the Operation Research and Decision Support Department as a teaching assistant.