Res Eng Design (2016) 27:55–71
DOI 10.1007/s00163-015-0205-0

ORIGINAL PAPER

An analytical approach to estimate the expected duration and variance for iterative product development projects

Walid Nasr[1] · Ali Yassine[1] · Omar Abou Kasm[1]

Received: 16 May 2015 / Revised: 6 October 2015 / Accepted: 9 October 2015 / Published online: 26 October 2015

Abstract The paper presents an analytical method for finding the expected duration and variance of a product development (PD) project network. A PD project network is a stochastic activity network (such as a PERT network) which allows for probabilistic repetition of activities (i.e., activity rework). When rework is allowed, estimating the process duration and variance becomes difficult. Most existing literature refers to the use of simulation in such scenarios; however, few analytical methods exist to solve this problem. One such method is called the reward Markov chain (RMC) which only considers sequential activity networks and which we use as a starting point in our proposed method. In this paper, we extend the RMC method to solve mixed networks (i.e., a combination of parallel and sequential activities) and more complicated practical issues that may arise in PD environments, specifically coupled activities and parallel rework.

Keywords Project networks · Product development · Activity rework · Iteration · Design structure matrix (DSM)

1 Introduction

Product development (PD) is the process of developing new products and services in order to meet the demand and expectations of evolving customer needs (Ulrich and Eppinger 2008). An essential element for the success of PD projects is to meet customer needs on timely basis (Takeuchi and Nonaka 1986; Hum and Sim 1996; Majava et al. 2013). When a target launch date is set for a product, the company must ensure meeting such deadline or else it risks losing substantial market share (Hendricks and Singhal 1997; Herm 2013). Setting a suitable deadline requires calculating the expected completion time (and variance) for the PD project. The problem arises not only because some of these activities have stochastic durations, but also some activities may be repeated several times due to the existence of feedback dependencies with the potential to generate rework; i.e., the repetition of already finished or completed activities (Browning and Ramasesh 2007; Unger and Eppinger 2009). Feedbacks are a typical characteristic of any complex design and development project and a potential source of design iterations, which can account for one-third to two-thirds of the project duration and cost (Osborne 1993; Meier et al. 2007). This fact makes the study of project management in the presence of iteration, as suggested in this paper, a central issue for the engineering design community.

When feedback does not exist, well-known techniques such as CPM, critical path method and PERT, program evaluation and review technique, are normally used to determine sufficiently accurate distributions for project duration (Mantel et al. 2007; Pinto 2012). In the presence of feedback, simulation techniques have been used for finding the accurate distribution of a PD project duration (Browning and Eppinger 2002; Cho and Eppinger 2005; Abdelsalam and Bao 2006). However, few analytical techniques have also been proposed such as the reward Markov chain (Smith and Eppinger 1997) and signal flow graph (Eppinger et al. 1997) to estimate PD project duration, but can only tackle limited activity network structures, specifically sequential networks. The purpose of this paper

✉ Ali Yassine
ay11@aub.edu.lb

[1] Engineering Management Program, Faculty of Engineering and Architecture, American University of Beirut, P.O. Box 11-0236, Beirut 1107 2020, Lebanon

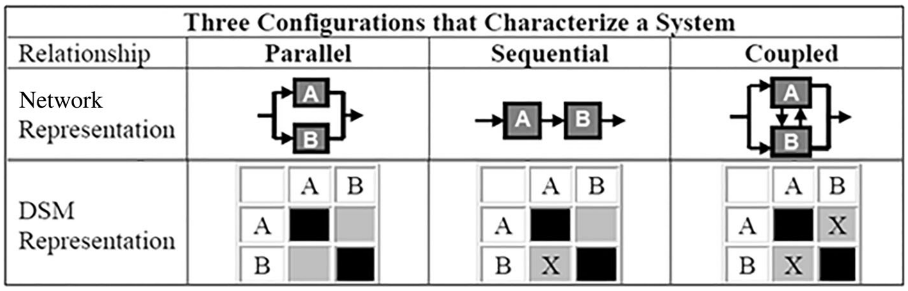


Fig. 1 Configurations that characterize a system (adapted from Yassine 2004)

is to find the expected duration and variance of any PD project network using analytical methods.

We present an approach to efficiently and accurately calculate the *m*th moment of the duration of a stochastic network with feedback. The sources of randomness are the durations of the individual activities and the probability of feedback after completing an activity. A design structure matrix (DSM) representing the dependency or information flow between activities provides valuable insight when determining the sequence of activity execution and provides managerial insights on which activities/teams need to be grouped or supervised collectively. The objective of this work is to investigate the duration of a network given the information flow and rework proportions via a DSM. This work bridges the literature on DSM and project management.

The method presented in this paper transforms a project network with feedback into a traditional project network (i.e., without feedback) by fitting an approximate probability distribution to the rework resulting from the initial completion of every activity. We refer to this approach by *cycle elimination* (CE) which results in an acyclical network for which classical project scheduling techniques such as PERT/CPM can be utilized. The complexity involved in simulating the resulting simplified network decreases significantly when compared to the original network or DSM.

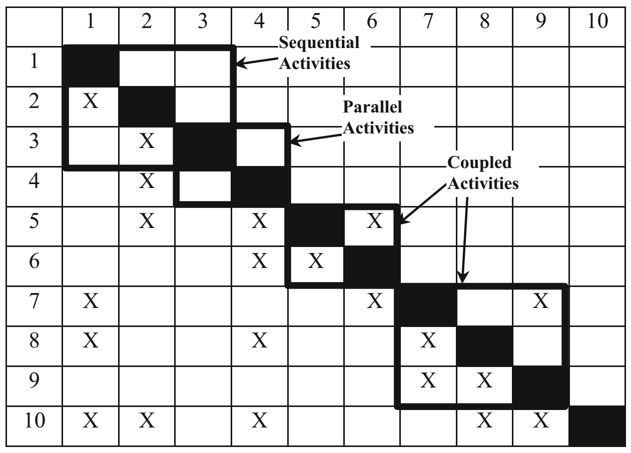
The rest of the paper is organized as follows. In the next section, a detailed literature review is presented. The proposed base methodology, which is called CE approach, is discussed in Sect. 3. We investigate a fully sequential network, where a linear set of equations is presented to solve for the moments of the network duration. In Sect. 4, we extend our methodology to allow for mixed networks, which include a mixture of sequential and parallel activities. In Sect. 5, we introduce another feature to the method which considers coupled activities in the network. Section 6 relaxes the assumption of sequential rework and proposes an extension to the base methodology which allows for parallel rework. A case study is presented in Sect. 7 to illustrate the proposed methodology. Finally, a summary discussion, future recommendations and conclusions are presented in Sect. 8.

2 Literature review

As mentioned earlier, product development (PD) project networks differ from traditional project networks mainly due to the existence of feedback dependencies with the potential to generate rework (Browning and Ramasesh 2007). Incorporating rework potential into the project network makes the task of determining the project duration difficult (Karniel and Reich 2009). Few analytical and simulation techniques exist in the literature dealing with the analysis of iterative projects; however, these techniques suffer from serious limitations. In this section, we discuss these approaches and their limitations, but first we start by introducing the design structure matrix (DSM) as an alternative approach to compactly represent project networks.

The design structure matrix (DSM) was developed to represent networks with feedback (Steward 1981; Browning 2001; Yassine and Braha 2003). Three types of relationships are represented in a DSM, parallel, sequential and coupled (Eppinger et al. 1994; Yassine 2004). Figure 1 shows the network representations versus the DSM representations for these three configurations. The "X" mark in the DSM indicates the presence of a predecessor relationship (arrow in the graph representation) between two different activities. This representation can be extended to an $(n \times n)$ matrix, i.e., network with n activities. A DSM with 10 activities is considered in Fig. 2. An "X" at the intersection of Column 2 and Row 4 infers a connection between Activity 2 and Activity 4; specifically, Activity 4 requires input from Activity 2 or Activity 2 is the predecessor of Activity 4. If the activities are executed in the order they appear in the DSM, then all elements above the diagonal indicate the possibility of feedback after completing an activity for the first time. As such, Fig. 2 shows Activities 1, 2 and 3 as sequential activities, Activities 3 and 4 as parallel activities, Activities 5 and 6 are coupled, and Activities 7, 8 and 9 as also coupled.

To quantify the possibility of feedback, a rework probability DSM substitutes an "X" with the probability of rework. Similarly, a rework proportion DSM substitutes an "X" with the rework impact or the proportion of rework to



	1	2	3	4	5	6	7	8	9	10
1	■									
2	X	■								
3		X	■							
4		X		■						
5		X		X	■	X				
6				X	X	■				
7	X					X	■		X	
8	X			X			X	■		
9							X	X	■	
10	X	X		X				X	X	■

Fig. 2 10 × 10 DSM

be performed (Browning and Eppinger 2002). The duration for each activity is usually entered in the diagonal elements.

Smith and Eppinger (1997) present an approach which considers a Markovian representation for a sequential PD network with feedback and solves a modified form of Gaussian elimination to find the expected duration. Our proposed approach, in this paper, builds on the sequential and iterative model of Smith and Eppinger (1997), where we utilize a similar definition of a stage. A stage in Smith and Eppinger (1997) corresponds to the nominal-completion of an activity along with the resulting rework. A reward Markov chain (RMC) approach is utilized to determine the expected duration of the network, where neither parallel work/rework, nor stochastic activity durations are considered.

Smith and Eppinger (1997) approach works as follows. Let R_I be the remaining duration of the network given that we just started activity I for the first time, for $i = 1, \ldots, n$. The approach begins with the final node N and calculates the expected time to finish the project given that the project is currently at this node, $E[r_n]$. The next step excludes the final node n and calculates $E[r_{n-1}]$. Similarly, $E[r_i]$, is calculated for $i = 1, \ldots, n$ and the expected duration of the entire network is set to $\sum_{i=1}^{n} E[r_i]$. This approach assumes that activity execution is sequential and parallel work is not allowed. That is, the completion of an activity cannot result in the immediate rework of two or more activities in parallel (i.e., only one activity at a time is reworked), and the event of reworking a task is mutually exclusive from the event of reworking any other task. The completion of an activity for the first time can result in a stream of upstream activities to be reworked sequentially.

Along similar lines, Eppinger et al. (1997) consider a deterministic network where signal flow graph (SFG) analysis is utilized to compute the mean and variance of the network duration. The base model is represented by a signal flow network where activities are represented by nodes. In such a network, the logical relationship at the exit from a node is an OR relationship. As a result, it is not possible to model an activity having more than one immediate predecessor. Extensions which include random activity durations or parallel rework are also addressed. Stochastic durations are achieved by allowing an activity to take on discrete durations with associated probabilities. Every possible discrete duration, which an activity can assume, results in an additional network state. Extensions also allow for parallel rework where combinations of activities being reworked simultaneously are denoted by additional states. A path is defined by a sequence of activities to be executed sequentially where the same activity can appear more than once. Parallel rework of activities is also considered in terms of paths where every path is denoted by a state and assigned a deterministic duration and a probability.

The signal flow graph (SFG) approach and the reward Markov chain (RMC) of Smith and Eppinger (1997) are equivalent and calculate the same expected network duration (Pinkett 1998). More recent approaches consider merging the DSM approach with the Graphical Evaluation Review Technique (GERT), Martinez Leon et al. (2013), where a framework is also presented to quantify the impact of looping.

Browning and Eppinger (2002), Cho and Eppinger (2005) and Abdelsalam and Bao (2006) develop a simulation algorithm to determine the expected duration and cost for a PD project. Only minor differences exist between these various DSM simulation models. The core difference lies in two aspects of the simulation: the sampling of the activity durations from known distributions and the modeling of the dynamic progress of the project.

Browning and Eppinger (2002) simulation approach utilizes two inputs: the rework probabilities and proportions (represented by two DSMs). Every iteration within the simulation identifies the set of active activities as being the most upstream and not requiring information from any unfinished upstream activities. The active activity with the shortest duration is reworked, and the residual work of the remaining activities is shortened accordingly. The resulting rework is evaluated (which might trigger rework on completed upstream activities), and the set of active activities is updated as well as the proportion of completed work. The simulation approach assumes that the set of active activities are adjacent within the DSM sequence. The approach also requires that all activities which are upstream relative to the active set of adjacent activities and input information to the active set have been completed. A drawback for such an approach is that coupled activities that share information cannot both be simultaneously part of the set of active activities. One of the coupled activities has to wait for the other to be completed.

The simulation approach of Browning and Eppinger (2002) accounts for the entire state of the network at every iteration by defining a work vector which identifies the work remaining for every activity. An advantage of such an approach is that it accounts for the dependencies between activities across different stages. Our approach assumes that the duration of a stage is independent of other stages. This allows for network decomposition where each stage is analyzed independently within a network structure. This significantly reduces the computational complexity of analyzing the entire network which is mainly a result of the large number of possible states which identify the state of the network at any point in time.

Existing approaches to analyzing the resulting stochastic networks with no feedback include the classical CPM/PERT and Monte-Carlo simulation approaches (Pinto 2012). In the case study, in this paper, we selected a Monte-Carlo simulation approach which is relatively straightforward in a no feedback activity network (Mantel et al. 2007). Simulating a stochastic network with feedback as is the case in Browning and Eppinger (2002) is much more computationally complex where at any point in time the network is defined by the set of active activities as well as the residual times of the activities.

Utilizing our proposed approach maintains the precedence relationships within the network which allows a manager to view the project schedule as an activity network with no feedback. Consequently, this preserves the advantages of using an activity network in project management which include easily identifiable milestones, facilitates the communication of time estimates relating to start and end dates of stages, among other well-established advantages of activity network techniques.

3 The proposed cycle elimination (CE) method

This paper investigates the duration of a stochastic and iterative project network and assumes the following inputs are provided: (1) the feedback probabilities and rework proportions via two DSMs; (2) the deterministic activity duration or the probability distributions of activity durations; and (3) the sequence of activity execution. Notice that the sequence of executing activities can either be suggested from the DSM structure (Yassine and Braha 2003; Yassine 2004), or from a classical network representation such as an activity-on-node (AON) representation (Pinto 2012).[1]

[1] In many DSM cases, several alternative sequences might be possible and reasonable, depending on the feedback structure in the network.

An iterative PD network allows an activity to be reworked (i.e., repeated) multiple times. We refer to the process of completing an activity for the first time by the *nominal-execution* of the activity. Denote the time to complete Activity i for the first time (*nominal-time* of Activity i) by T_I. We make the following assumptions: (1) If Activity i is an immediate predecessor for Activity j, then we cannot begin working on Activity j unless the nominal-execution of Activity i is completed along with the rework resulting from the nominal-execution of Activity i; and (2) the rework resulting from the nominal-execution of an activity can result in a succession of upstream activities to be reworked where rework is sequential. Notice that although rework is assumed to be sequential, the sequence in which the nominal-execution of the activities occurs is not necessarily sequential. Assumptions i and ii constitute the basis of our approach where the nominal-execution of an activity along with the feedback resulting from the nominal-execution is augmented to form a stage. An important characteristic of a stage is that its completion does not result in feedback to other activities or stages. In this section, we present the steps of the approach in detail on a network where the nominal-execution of activities is sequential. We extend our approach, in Sect. 4, to the general case of mixed networks, where the nominal-execution of activities includes activities that can be performed in parallel.

3.1 Sequential networks

In this paper, we refer to a network as sequential if the nominal-execution of the activities is performed sequentially and the order in which activities appear in a DSM determines the sequence of the nominal-execution of the activities. Consider a sequential network with n activities where the nominal-time of Activity I is T_I for $i = 1, \ldots, n$, and the nominal-execution of Activity i can result in rework in upstream activities. In this paper, we represent the feedback probabilities and rework proportions by two $(n \times n)$ DSMs, **DSM1** and **DSM2**, respectively. We utilize the commonly used DSM notation where the entries of the ith column of **DSM1** represent the rework probabilities resulting from the nominal-completion of Activity i. If P_{ij} is the probability of reworking Activity j after completing Activity i, then $P_{ij} = \mathbf{DSM1}(j, i)$. Similarly, W_{ij} is the proportion of rework that is required on Activity j after performing Activity i, $W_{ij} = \mathbf{DSM2}(j, i)$. In this work, we set $P_{II} = 0$ and $W_{II} = 0$ for $i = 1, \ldots n$, and we assume any immediate feedback (i.e., self-iteration) is accounted for by the probability distribution of the activity. The mean and standard deviation of the nominal-completion times for each activity are presented in the diagonal entries of the

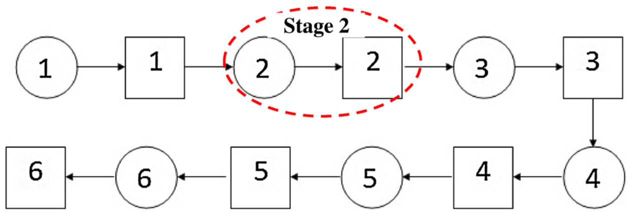


Fig. 3 Stage network

DSM1 and **DSM2** matrices, respectively; i.e., **DSM1**(i, i) = $E[t_i]$ and **DSM2**(i,i) = Stdev(t_i) for i = 1,…, n.

Define Stage k as the combination of performing the nominal-execution of Activity k and all the rework resulting from the nominal-execution of Activity k before proceeding to Activity $K + 1$, for $k = 1, \ldots n$. If T_k is the time required to complete Stage k, then:

$$T_K = T_K + R_K, \text{ for } \quad k = 1, \ldots, n. \tag{1}$$

where R_k is the time required to complete the rework resulting from Activity k before proceeding to Activity $k + 1$. A six-stage network is represented in Fig. 3 where a circle node represents the nominal-execution of an activity with duration, T_K, and a square node represents rework with duration R_K, for $k = 1, \ldots, 6$. Stage k requires a square rework node if there exists at least one nonzero entry above the diagonal in Column k of the probability DSM. Such a representation accounts for rework within a stage, and the moments of the duration of each stage,$E\left[T_k^m\right]$ for $m \geq 1$, are calculated independently.

Therefore, the mth moment of the duration to complete Stage k is:

$$E\left[T_k^m\right] = E\left[T_k^m|\text{no rework}\right]\text{Prob}(\text{no rework}) + \sum_{i=1}^{k} E[T_k^m|\text{rework}i]\text{Prob}(\text{rework}i) \tag{2}$$

where $E\left[T_k^m|\text{no rework}\right]$ and $E[T_k^m|\text{rework}i]$ are the mth moments of the duration of Stage k conditioned on reworking Activity i. Equation 2 assumes that one activity is reworked at a time (parallel rework is not allowed) and the event of reworking an upstream activity is mutually exclusive from the event of working any other upstream activity. Consequently, the sum of the non-diagonal entries in any column of the probability DSM cannot exceed 1.

Let $R_{ij;k}$ be the remaining time required to complete Stage k given that Activity i has just been reworked and resulted in further rework in Activity j. At Stage k, the mth moment of $R_{ij;k}^m$ is calculated by conditioning on the probability of having no rework after Activity j is reworked or conditioning on reworking Activity u after j has been reworked for $u = 1, \ldots, k$. The mth moment of $R_{ij;k}$ (for k = 1,…,n; and i, j = 1,…,k) is:

$$\begin{aligned} E\left[R_{ij;k}^m\right] &= E\left[\left(R_{ij;k}^m\right)^m|\text{no rework}\right]\text{Prob}(\text{no rework}) \\ &\quad + \sum_{u=1}^{k} E\left[\left(R_{ij;k}^m\right)^m|\text{rework}u\right]\text{Prob}(\text{rework}u) \\ &= E\left[(W_{ij}t_j)^m\right]\text{Prob}(\text{no rework}) \\ &\quad + \sum_{u=1}^{k} E\left[(W_{ij}t_j + R_{ju;k})^m\right]\text{Prob}(\text{rework}u). \end{aligned}$$

$$\begin{aligned} \Rightarrow E\left[R_{ij;k}^m\right] &= E\left[(W_{ij}t_j)^m\right]\left(1 - \sum_{u=1}^{k} P_{ju}\right) \\ &\quad + \sum_{u=1}^{k} E\left[(W_{ij}t_j + R_{ju;k})^m\right]P_{ju}. \end{aligned} \tag{3}$$

Similarly Eq. (2) is expressed as,

$$E[T_k^m] = E[t_k^m]\text{Prob}(\text{no rework}) + \sum_{u=1}^{k} E\left[(t_k + R_{ku;k})^m\right]\text{Prob}(\text{rework}u).$$

$$\begin{aligned} \Rightarrow E[T_k^m] &= E[t_k^m]\left(1 - \sum_{u=1}^{k} P_{ku}\right) \\ &\quad + \sum_{u=1}^{k} \text{E}\left[(t_k + R_{ku;k})^m\right]P_{ku}. \end{aligned} \tag{4}$$

The random variable $R_{ku,k}$ represents the duration of the rework generated by completing Activity k given that Activity u is the first upstream activity to be reworked. The duration $R_{ku,k}$ is incurred with probability P_{ku} where $\sum_{u=1}^{k} P_{ku} \leq 1$, and $\left(1 - \sum_{u=1}^{k} P_{ku}\right)$ is the probability of no rework after completing Activity k for the first time.

At Stage k, for $k = 1, \ldots, n$, the starting ($k \times k$) partition only, of matrices **DSM1** and **DSM2**, is utilized. Although the entries of the probability and rework proportion DSMs are not a function of the stage, it is useful to note that it is possible to dynamically adjust **DSM1** and **DSM2** at each stage. This can be accounted for in Eqs. (3) and (4) by simply replacing P_{IJ} and W_{IJ} by $P_{IJ;K}$ and $W_{IJ;K}$, respectively, for all $k = 1, \ldots, n$ and $i,j = 1, \ldots, k$. Dynamically changing the matrices **DSM1** and **DSM2** might be necessary if the nominal-completion of an activity indirectly alters the rework probabilities and proportions of upstream activities.

We summarize the implementation of the CE approach for a sequential network as follows:

Step 1: If column i of the probability DSM has at least one nonzero entry above the diagonal, then a square node is inserted after Activity i in the AON network or the DSM for i = 1,…,n.

Step 2: Feedback originating from Activity i is eliminated, and Stage i is now composed of a circle node representing the nominal-execution of Activity i and a square node representing the rework resulting from the nominal-execution of Activity i before proceeding to subsequent activities.

Step 3: The first two moments of the stage durations, $E[T_i]$ and $E[T_i^2]$, for $i = 1,\ldots,n$, are calculated by solving the system of linear equations in (3) and (4) for $m = 1$, 2. Note that the mean and variance of the rework nodes (square nodes) are calculated as $E[R_i] = E[T_i] - E[t_i]$ and $\text{Var}(R_i) = \text{Var}(T_i) - \text{Var}(t_i)$, respectively, for $i = 1,\ldots,n$.

Step 4: The mean and variance of the entire project network duration are calculated by summing the expected duration and variance, $\sum_{i=1}^{n} E[T_i]$ and $\sum_{i=1}^{n} \text{Var}[T_i]$, respectively, over all stages.

Next, we present the system of linear equations for the first two moments of the network duration. An algorithm to efficiently generate a matrix representation of the system of linear equations to compute the first two moments is presented in "Appendix."

3.1.1 Calculating the expected duration

The expected duration at Stage K can be calculated using Eqs. (3) and (4) for $m = 1$. Equation (3) requires solving a set of k^2 linear equations. For $m = 1$ and for $k = 1,\ldots,n$, $i,j = 1,\ldots,k$, Eqs. (3) and (4), respectively, are reduced to,

$$E[R_{ij;k}] = E[W_{ij}t_j]\left(1 - \sum_{u=1}^{k} P_{ju}\right) + \sum_{u=1}^{k} E[W_{ij}t_j + R_{ju;k}]P_{ju} = E[W_{ij}t_j] + \sum_{u=1}^{k} E\left[R_{ju;k}\right]P_{ju} \tag{5}$$

$$E[T_k] = E[t_k]\left(1 - \sum_{u=1}^{k} P_{ku}\right) + \sum_{u=1}^{k} E[t_k + R_{ku;k}]P_{ku} = E[t_k] + \sum_{u=1}^{k} E[R_{ku;k}]P_{ku} \tag{6}$$

At each Stage K, the system of linear Equations in (5) and (6) is represented by the following compact matrix representation,

$$\mathbf{AX}_1 = \mathbf{b}_1 \tag{7}$$

where $\mathbf{A}$ is a $(k^2 + 1) \times (k^2 + 1)$ square matrix, and $\mathbf{X_1}$ and $\mathbf{b_1}$ are $(k^2 + 1) \times (1)$. The first k^2 entries of $\mathbf{X_1}$ correspond to the expected rework as denoted by the right-hand-side expression of Eq. (6). The $K^2 + 1$ entry of $\mathbf{X_1}$ corresponds to $E[T_k]$. We present two algorithms in "Appendix" to efficiently populate the entries of matrices $\mathbf{A}$ and $\mathbf{b_1}$, respectively. The expected duration of the entire sequential network is sum of the expected durations of all the stages.

The expected network duration is obtained by calculating $\sum_{i=1}^{n} E[T_i]$.

3.1.2 Calculating the variance

The variance of the time to complete Stage k is,

$$\text{Var}(T_k) = E\left[T_k^2\right] - E^2[T_k]. \tag{8}$$

Here we make the assumption that the stage durations are independent. The variance of the entire network is:

$$\text{Var(Project)} = \sum \text{Var}(T_k). \tag{9}$$

The second moment, $E\left[T_k^2\right]$, at Stage K, is calculated using Eq. (4) for $M = 2$ and solving Eq. (3) requires solving the set of K^2 linear equations. For $m = 2$, Eqs. (3) and (4), respectively, are,

$$E[R_{ij;k}^2] = E\left[(W_{ij}t_j)^2\right]\left(1 - \sum_{u=1}^{k} P_{ju}\right) + \sum_{u=1}^{k} E\left[(W_{ij}t_j + R_{ju;k})^2\right]P_{ju} = E[(W_{ij}t_j)^2] + \sum_{u=1}^{k} E\left[2W_{ij}t_jR_{ju;k} + R_{ju;k}^2\right]P_{ju} \tag{10}$$

$$E[T_k^2] = E[t_k^2]\left(1 - \sum_{u=1}^{k} P_{ku}\right) + \sum_{u=1}^{k} E\left[(t_k + R_{ku,k})^2\right]P_{ku} = E[t_k^2] + \sum_{u=1}^{k} E\left[2t_kR_{ku,k} + R_{ku,k}^2\right]P_{ku} \tag{11}$$

The following system of linear equations is solved to obtain the second moment of the duration of Stage k, $E[T_k^2]$:

$$\mathbf{AX}_2 = \mathbf{b}_2, \tag{12}$$

where $\mathbf{A}$ is a $(k^2 + 1) \times (k^2 + 1)$ square matrix, and $\mathbf{X_2}$ and $\mathbf{b_2}$ are $(k^2 + 1) \times (1)$. Notice that the matrix $\mathbf{A}$ is obtained when solving for the expected duration of the Stage k. The algorithm to obtain $\mathbf{b_2}$ is presented in "Appendix." The $k^2 + 1$ entry of $\mathbf{X}$ corresponds to $E[T_k^2]$. Note that the activity duration, t_i, is a random number where the first two moments $E[t_i]$ and $E[t_i^2]$ are assumed to be known.

The $k^2 + 1$ entry of $\mathbf{X_2}$ corresponds to $E[T_k^2]$. The variance of the duration of the entire network is calculated as the sum of the variances of all stages. Obviously, the assumption made here is that the duration of Stage i is independent of the duration of Stage J, for $i,j = 1,\ldots n$ and $i \neq j$. The variance of the entire network duration is obtained by calculating $\sum_{i=1}^{n} \text{Var}[T_i]$. Next, we present an example to illustrate the stage computations for a sequential network with rework.

Fig. 4 DSM—Example 1

Rework Probabilities (**DSM1**)

	1	2	3	4	5	6
1	**74**					0.40
2	0.27	**20**		0.31		
3	0.25	0.32	**72**		0.50	
4		0.55	0.23	**41**		
5			0.29	0.17	**36**	
6					0.77	**59**

Rework Proportions (**DSM2**)

	1	2	3	4	5	6
1	**37**					0.35
2	0.75	**10**		0.30		
3	0.24	0.27	**36**		0.48	
4		0.86	0.18	**20.5**		
5			0.72	0.15	**18**	
6					0.61	**29.5**

3.2 Example 1: Sequential network

Consider a sequential network with rework probabilities and stochastic activity durations. The DSM for the corresponding rework probabilities and proportions is presented in Fig. 4. The mean and standard deviation for each activity are presented in the diagonal entries of the rework probabilities and proportions matrices, respectively, Fig. 4.

Feedback is eliminated by calculating the mean and variance of the time to complete each of the six stages, $E[T_k]$ and $\mathrm{Var}(T_k)$ for $k = 1, 2,\ldots, 6$. As an illustration on how the algorithm operates, we present the calculations of the mean of the first four stages: $E[T_1]$, $E[T_2]$ $E[T_3]$ and $E[T_4]$. The time to complete Stage 1 is simply T_1, since the nominal-completion of Activity 1 does not result in rework. This results in $E[T_1]= E[t_1] = 74$. The completion of Stage 2 considers the following DSM partitions, Fig. 5.

Notice that the nominal-completion of Activity 2 does not result in feedback to Activity 1 which is the only upstream activity at this stage. The time to complete Stage 2 is simply T_2, and $E[T_2]= E[t_2] = 20$. The expected time to begin work on Activity 3 for the first time (i.e., nominal-execution) is $E[T_1]+E[T_2] = 94$. Similarly, solving for the completion time of Stage 3 considers the following DSM partitions, Fig. 6. The expected time to complete the first three stages is $E[T_1]+E[T_2] + E[T_3] = 166$.

At Stage 4, the following DSM is considered, Fig. 7, where completing Activity 4 for the first time can result in rework in Activity 2 which in turn can result in rework in Activities 3 and 4, Fig. 8.

The nominal-completion of Activity 4 can result in rework in Activity 2 with probability $P_{42} = 0.31$. This is represented by the following equation,

$$E[T_4] = E[t_4] + P_{42}E[R_{42;4}] = 41 + 0.31 \times E[R_{42;4}], \quad (13)$$

where $R_{42;4}$ represents the amount of rework required to complete Stage 4 given that Activity 4 has just been completed and requested rework from Activity 2. Reworking Activity 2 can result in further rework in either Activity 3 or 4, but not both (as depicted by P_{23} and P_{24}). Notice that this is a result of the sequential rework assumption.[2] Note that although $P_{35} = 0.29$, reworking

(**DSM1**)

	1	2
1	**74**	
2	0.27	**20**

(**DSM2**)

	1	2
1	**37**	
2	0.75	**10**

Fig. 5 DSM at Stage 2—Example 1

(**DSM1**)

	1	2	3
1	**74**		
2	0.27	**20**	
3	0.25	0.32	**72**

(**DSM2**)

	1	2	3
1	**37**		
2	0.75	**10**	
3	0.24	0.27	**36**

Fig. 6 DSM at Stage 3—Example 1

(**DSM1**)

	1	2	3	4
1	**74**			
2	0.27	**20**		0.31
3	0.25	0.32	**72**	
4		0.55	0.23	**41**

(**DSM2**)

	1	2	3	4
1	**37**			
2	0.75	**10**		0.30
3	0.24	0.27	**36**	
4		0.86	0.18	**20.5**

Fig. 7 DSM at Stage 4—Example 1

Activity 5 is obviously not considered at Stage 4 since the nominal-execution of Activity 5 has not been initiated. Reworking Activity 5 is only considered at Stages 5 and 6. Solving for the expected value of $R_{42;4}$ requires solving the following system of 4 equations and 4 unknowns:

$$\begin{aligned} E[R_{42;4}] &= E[W_{42}t_2] + P_{23}E[R_{23;4}] + P_{24}E[R_{24;4}] \\ &= 0.30 \times E[t_2] + 0.32 \times E[R_{23;4}] + 0.55 \times E[R_{24;4}] \end{aligned} \quad (14)$$

$$\begin{aligned} E[R_{23;4}] &= E[W_{23}t_3 + P_{34}E[R_{34;4} \\ &= 0.27 \times E[t_3] + 0.23 \times E[R_{34;4}] \end{aligned} \quad (15)$$

$$\begin{aligned} E[R_{24;4}] &= E[W_{24}t_4] + P_{42}E[R_{42;4}] \\ &= 0.86 \times E[t_4] + 0.31 \times E[R_{42;4}] \end{aligned} \quad (16)$$

$$\begin{aligned} E[R_{34;4}] &= E[W_{34}t_4] + P_{42}E[R_{42;4}] \\ &= 0.18 \times E[t_4] + 0.31 \times E[R_{42;4}] \end{aligned} \quad (17)$$

This results in a value of $E[R_{42;4}] = 39.86$. The expected duration of Stage 4 becomes, $E[T_4] = 41 + 0.31\times 39.86 = 53.36$. The expected time to complete Stage 4 and all the resulting rework before proceeding to Activity 5 for

[2] Setting $P_{24} = 0$ (i.e., a purely sequential network where rework is also sequential), there is no need to make this assumption.

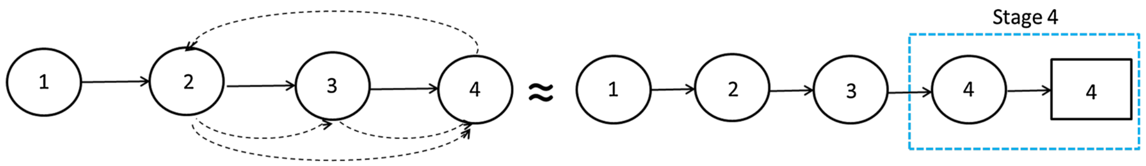


Fig. 8 Cycle elimination at Stage 4

the first time is $E[T_1]+E[T_2]+E[T_3]+E[T_4] = 74+20+72+53.36 = 219.36$.

Similarly, the second moment of the duration of Stage 4 can be calculated by the following equation,

$$\begin{aligned} E\left[T_4^2\right] &= E\left[t_4^2\right] + P_{42}E\left[R_{42;4}^2\right] + P_{42}E[2t_4R_{42;4}] \\ &= 2101.25 + 0.31 \times E\left[R_{42;4}^2\right] + 0.31(2\times 41\times 39.86) \end{aligned}, \quad (18)$$

Solving for the second moment of $R_{42;4}$ requires solving the following system of 4 equations:

$$\begin{aligned} E[R_{42;4}^2] &= E[(W_{42}t_2)^2] + P_{23}E[2W_{42}t_2R_{23;4}] + P_{23}E\left[R_{23;4}^2\right] \\ &\quad + P_{24}E[2W_{42}t_2R_{24;4}] + P_{24}E[R_{24;4}^2] \end{aligned} \quad (19)$$

$$\begin{aligned} E[R_{23;4}^2] &= E[(W_{23}t_3)^2] + P_{34}E[2W_{23}t_3R_{34;4}] \\ &\quad + P_{34}E[R_{34;4}^2] \end{aligned} \quad (20)$$

$$\begin{aligned} E[R_{24;4}^2] &= E[(W_{24}t_4)^2] + P_{42}E[2W_{24}t_4R_{42;4}] \\ &\quad + P_{42}E[R_{24;4}^2] \end{aligned} \quad (21)$$

$$\begin{aligned} E[R_{34;4}^2] &= E[(W_{34}t_4)^2] + P_{42}E[2W_{24}t_4R_{42;4}] \\ &\quad + P_{42}E[R_{24;4}^2] \end{aligned} \quad (22)$$

This results in $E\left[R_{42;4}^2\right] = 2493.8$ and $\mathrm{Var}(T_4) = E\left[T_4^2\right] - E[T_4]^2 = 1040.5$. The variance of the time to complete Stage 4 and all the resulting rework before proceeding to Activity 5 for the first time is $\mathrm{Var}(T_1) + \mathrm{Var}E(T_2) + \mathrm{Var}(T_3) + \mathrm{Var}(T_4) = 1369.0 + 100.0 + 1296.0 + 1040.5 = 3805.5$.

Similarly, we solve for the first two moments of Stages 5 and 6 and we obtain, $E[T_5] = 64.48$, $E[T_6] = 83.20$, $\mathrm{Var}(T_5) = 1980.0$, and $\mathrm{Var}(T_6) = 3022.9$. Adding the mean and variance at each stage, we obtain the mean and standard deviation of the entire network as 367.04 and 93.85, respectively. Table 1 summarizes the results of this example.

For a system where the sequential rework assumption holds (i.e., activities cannot be reworked in parallel) and the nominal-execution of activities is performed sequentially, our proposed method provides numerically exact results for the mean and variance of the network completion time.

Table 1 Stage durations (Example 1)

Stage i	Nominal-duration		Rework duration		Stage duration	
	$E[t_i]$	$\mathrm{Var}[t_i]$	$E[R_i]$	$\mathrm{Var}[R_i]$	$E[T_i]$	$\mathrm{Var}[T_i]$
1	74	1369	0	0	74	1369
2	20	100	0	0	20	100
3	72	1296	0	0	72	1296
4	41	420.25	12.36	620.3	53.36	1040.5
5	36	324	28.48	1655.5	64.48	1979.5
6	59	870.25	24.20	2152.7	83.20	3022.9

Recent research by Braha and Bar-Yam (2007) has shown that the underlying network structure of a PD network can provide useful information about its performance. Additionally, they have also shown that real-world PD networks indeed deviate from sequential structures; many of these networks exhibit the "small world" property and follow a "scale-free" distribution (for the nodal degrees). Although the sequential work and rework assumption, made in Sect. 3, is useful to simplify the calculations of the various moments for project duration, this assumption must be relaxed to be able to properly capture the underlying network structure of a PD projects. Accordingly, Sect. 4 considers networks where the nominal-completion of activities is not necessarily sequential. Section 6 addresses the complexity involved in relaxing the sequential rework assumption, and we present approximations to analyze networks with parallel rework.

4 Solving mixed activity networks

In this section, we consider mixed networks where nominal-execution of activities is not necessarily sequential. However, the finish to start relationship between activities can be inferred from the DSM (as illustrated in Fig. 2) and easily represented by the commonly used AON representation. If there is no feedback and the activity durations are deterministic, then the project's duration can be calculated by the critical path method (CPM). In the case of no feedback but the duration of the activities are stochastic, PERT can be used or the AON network can be simulated efficiently to obtain accurate point estimates (Mantel et al. 2007; Pinto 2012). Eliminating feedback from a mixed

Fig. 9 DSM—Example 2

(DSM1)

	1	2	3	4	5	6
1	**74**					0.40
2	0.27	**20**		0.31		
3	0.25		**72**		0.50	
4		0.55		**41**		
5			0.29	0.17	**36**	
6					0.77	**59**

(DSM2)

	1	2	3	4	5	6
1	**37**					0.35
2	0.75	**10**		0.30		
3	0.24		**36**		0.48	
4		0.86		**20.5**		
5			0.72	0.15	**18**	
6					0.61	**29.5**

network significantly reduces the complexity involved in analyzing the network. This serves as a motivation behind applying the CE approach on a mixed network to obtain an approximate acyclic AON network (i.e., with no feedback).

Implementing the CE approach on a mixed network is comparable to the sequential case presented in Sect. 3 where a stage denotes the nominal-execution of an activity as well as the resulting rework that is performed on upstream activities before proceeding to subsequent stages. Again, the duration of Stage k is the time to complete Activity k for the first time, t_i, in addition to the time to complete the resulting upstream rework, R_k. Also, the rework resulting from the nominal-execution of an activity is represented by a square node.

However, in the mixed network case, the completion of a stage can initiate the nominal-execution of more than one subsequent stage in accordance to the AON/DSM precedence relationships. The CE approach preserves the sequence of the nominal activity execution and eliminates feedback by inserting a square node following an activity which can result in reworking upstream activities. The rework associated with the nominal-execution of an activity is also assumed to be performed sequentially. This assumption allows us to utilize Eqs. (5) and (6) to calculate the first moments of the stage durations. Similarly, due to the sequential rework assumption, Eqs. (10) and (11) are solved to obtain the second moments of the stage durations.

The CE approach for mixed networks is similar to the sequential approach, but excludes Step 4 where the mean and variance of the network duration are obviously not obtained by simply summing the mean and variance at each stage. As stated earlier, the motivation behind implementing the CE approach for mixed networks is to transform a network with feedback into a no feedback network. This is achieved by fitting a distribution to the first two moments of the rework at each stage. So Step 4 is replaced by:

Step 4: An approximate probability distribution is fitted to the first two moments of the rework duration R_I, for $i = 1,\ldots,n$.

Notice that Step 4 is required if the network is to be analyzed via Monte-Carlo simulation; otherwise, the CE approach results in an approximate stochastic network with no feedback with known mean and variance at every node.[3] Obviously, more accurate approximate distributions can be obtained by fitting higher-order moments which can be calculated by solving the system of linear equations of (3) and (4) for $m > 2$.

4.1 Example 2: Mixed networks

We consider the probability and rework DSMs of Example 1, but assume that reworking Activity 2 does not result in rework in Activity 3 and reworking Activity 3 does not result in rework in Activity 4 (i.e., $P_{23} = W_{23} = 0$ and $P_{34} = W_{34} = 0$). The resulting probability and rework proportion DSMs are presented in Fig. 9.

According to the DSM of Fig. 9, Activity 1 is the predecessor for Activities 2 and 3 and Activity 2 is the predecessor for Activity 4. However, Activity 3 does not require information from Activities 2 and 4. Therefore, the nominal-execution of Activity 3 can be performed in parallel with the nominal-execution of Activity 2 or 4. Furthermore, Activities 3 and 4 are both the predecessors of Activity 5, and finally Activity 5 is the predecessor of Activity 6. Figure 10 depicts an AON network which illustrates the sequence in which the nominal-execution of the activities is performed according to the DSM model.

According to the DSM of Fig. 9, Activities 4, 5 and 6 have at least one nonzero entry above the diagonal. Accordingly, the nominal-execution of Activities 4, 5 and 6 can result in feedback. Feedback is eliminated by implementing Step 1 of the CE method where a square node is inserted after Activities 4, 5 and 6, Fig. 11.

The network is composed of six stages where Stages 4, 5 and 6 have a rework component in addition to the nominal-execution time. The expected duration of Stages 1, 2 and 3 are 74, 20 and 72, respectively (Fig. 9). At Stage 4, the DSM of Fig. 9 is partitioned to include the first four activities resulting in the DSM partition shown in Fig. 12. Figure 13 illustrates the sequence of feedback that might

[3] In the case study described in Sect. 7, the resultant stochastic project network is analyzed via Monte-Carlo simulation (Sea 2001; Davis 2008), where a Lognormal distribution is fitted for the stage duration with mean and variance $E[T_k]$ and $\mathrm{Var}(T_k)[R_k^2]$ for $k = 1,\ldots,n$.

occur when the nominal-execution of Activity 4 is completed.

At Stage 4, Activity 4 can result in the first-order rework in Activity 2 which in turn can result in the second-order rework in Activity 4 and so on. The expected duration for Stage 4 is calculated using the following equation.

$$E[T_4] = E[t_4] + P_{42}E[R_{42;4}] = 41 + 0.31 \times E[R_{42;4}], \tag{23}$$

Solving for the expected value of $R_{42;4}$ requires solving the following system of linear equations.

$$\begin{aligned} E[R_{42;4}] &= E[W_{42}t_2] + P_{24}E[R_{24;4}] \\ &= 0.30 \times E[t_2] + 0.55 \times E[R_{24;4}] \end{aligned} \tag{24}$$

$$\begin{aligned} E[R_{24;4}] &= E[W_{24}t_4] + P_{42}E[R_{42;4}] \\ &= 0.86 \times E[t_4] + 0.31 \times E[R_{42;4}] \end{aligned} \tag{25}$$

Implementing Step 3 of the CE algorithm, the mean and variance of the rework resulting from completing Stage 4 are 9.49 and 494, respectively. Similarly at Stage 5, the DSM is partitioned to include the first five activities only. Implementing Step 3 of the CE algorithm, the mean and variance of the rework resulting from completing Stage 5 are 24.61 and 1,219, respectively.

The complete DSM is considered when evaluating the rework resulting from the nominal-execution of Activity 6. The sequence of feedback as a result of completing Activity 6 is illustrated in Fig. 14. It is worth noting that the nominal-completion of Activity 6 can result in rework in Activity 1 which in turn can result in rework in Activities 2 or 3, but not both. Although the nominal-completion of Activities 2 and 3 can be performed in parallel, the rework in Activities 2 and 3 (if triggered due to feedback) cannot be performed in parallel. This is based on the sequential rework assumption. Implementing Step 3 of the CE algorithm, the mean and variance of the rework resulting from completing Stage 6 are 20.28 and 1,417, respectively.

At Step 3 of the CE algorithm, the mean and variance of the rework at each stage is added to the corresponding mean and variance of the nominal-duration. Figure 11 represents the resulting no feedback network with mean and variance at each stage given in Table 2.

(DSM1)

	1	2	3	4
1	**74**			
2	0.27	**20**		0.31
3	0.25		**72**	
4		0.55		**41**

(DSM2)

	1	2	3	4
1	**37**			
2	0.75	**10**		0.30
3	0.24		**36**	
4		0.86		**20.5**

Fig. 12 DSM at Stage 4—Example 2

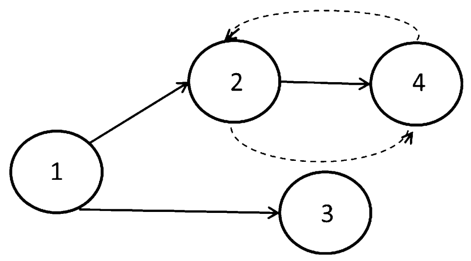


Fig. 13 Rework at Stage 4 (Example 2). *Dotted arrows* represent probabilistic feedback/rework

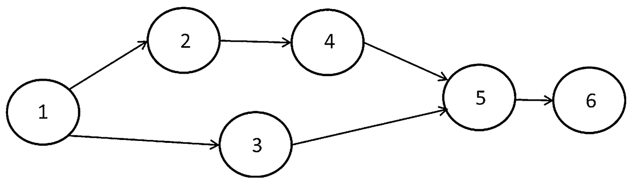


Fig. 10 Sequence of nominal-execution via an AON representation

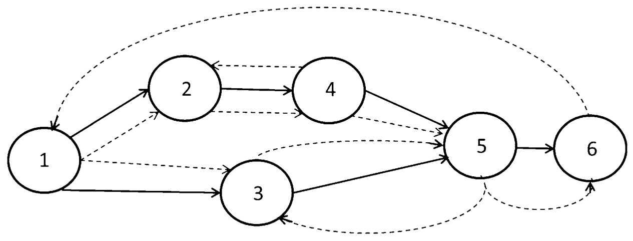


Fig. 14 Rework at Stage 6 (Example 2)

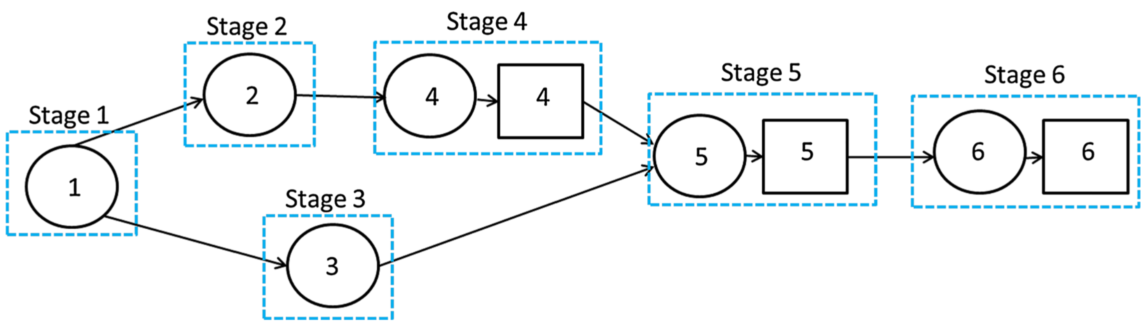


Fig. 11 Network with rework nodes (Example 2)

The CE approach results in an approximate stochastic network with no feedback with known mean and variance at every node. Implementing Step 4 of the CE algorithm is relatively simple since simulating the network in Fig. 11 is relatively straightforward where the network has two paths, 1–2–4–5–6 and 1–3–5–6, with no feedback.

5 Solving coupled-activity networks

We define coupled activities as a set of activities that are the predecessors of each other; however, they can be started together (i.e., there is no obvious execution order). Note that if the set of activities are nominally sequential as in Figs. 12 and 13, Activities 2 and 4, then we do not consider them coupled, but sequential with feedback.

Consider two coupled activities, *i* and *j*, where Activity *j* appears after Activity *i* in the DSM. The rework at Stage *j* of the CE approach accounts for the rework in Activity *i* due to the nominal-completion of Activity *j*, but the CE approach does not consider the rework in Activity *j* due to the nominal-completion of Activity *i*. In the case where the nominal-completion of Activities *i* and *j* are performed in parallel, there is a need to account for the rework in Activity *j* due to the nominal-completion of Activity *i*. To account for the rework in Activity *j*, an artificial activity, i', is inserted after Activity *j* in the DSM where completing Activity i' results in rework in Activity *i* with probability of 1. As a result, the rework resulting from Activity *i'* captures the rework generated by completing Activity *i* and accounts for reworking Activity *j*. So the artificial activity, *i'*, is inserted after Activity *j* in the probability and rework DSMs, and we set $P_{i'i} = W_{i'i} = 1$. The final step is to replace the rework at Stage *i* with the rework resulting from completing the artificial activity, i'.

The purpose behind inserting the dummy activity, i', is to capture the rework invoked by reworking Activity *i* which also accounts for reworking Activity *j*. This would not be possible otherwise since Activity *i* is considered upstream relative to Activity *j* according to the DSM sequence. The dummy activity achieves this purpose by (1) appearing after Activity *j* in the DSM and (2) reworking 100 % of Activity *i* with certainty, $P_{i'i} = W_{i'i} = 1$. Evaluating the duration of the dummy stage, i', is equivalent to revaluating the duration of Stage *i* where the rework of all the activities appearing before i' in the DSM is accounted for (which now include Activity *j*).

In the case where *n* activities are coupled, i.e., the nominal-completion of the *n* activities is performed in parallel and feedback can occur between the *n* activities, a similar approach can be implemented where $N - 1$ artificial activities are required. Note that the artificial activities are inserted in the DSM after the coupled block.

5.1 Example 3: Coupled activities in a network

Here we reconsider the probability and rework proportion DSMs of Example 2 as well as the nominal-execution sequence, with the added modification of making Activities 2 and 3 coupled. This is implemented by providing P_{23}, W_{23}, P_{32} and W_{32} with positive values. The resulting probability and rework proportion DSMs are presented in Fig. 15. Since Activities 2 and 3 are coupled, we insert an artificial Activity $2'$ after Activity 3. This is illustrated in the DSM of Fig. 16.

The rework resulting from Activity $2'$ captures the rework generated from the nominal-completion of Activity 2 which also accounts for reworking Activity 3. As a result, the rework at Stage 3 accounts for the rework in Activity 2 due to completing Activity 3. Also at Stage $2'$, which follows Stage 3, the rework in Activity 3 due to the nominal-completion of Activity 2 is now accounted for. Implementing the CE approach on the DSM of Fig. 16 results in the stage durations shown in Table 3. The rework resulting

Table 2 Stage durations (Example 2)

Stage *i*	Nominal-duration		Rework duration		Stage duration	
	$E[t_i]$	$\mathrm{Var}[t_i]$	$E[R_i]$	$\mathrm{Var}[R_i]$	$E[T_i]$	$\mathrm{Var}[T_i]$
1	74	1369	0	0	74	1369
2	20	100	0	0	20	100
3	72	1296	0	0	72	1296
4	41	420	9.49	494	50.49	914
5	36	324	24.61	1219	60.61	1543
6	59	870	20.28	1417	79.28	2287

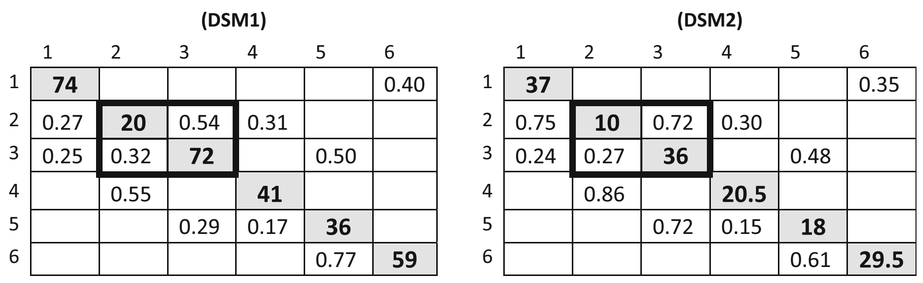


Fig. 15 DSM—Example 3

(DSM1)

	1	2	3	4	5	6
1	**74**					0.40
2	0.27	**20**	0.54	0.31		
3	0.25	0.32	**72**		0.50	
4		0.55		**41**		
5			0.29	0.17	**36**	
6					0.77	**59**

(DSM2)

	1	2	3	4	5	6
1	**37**					0.35
2	0.75	**10**	0.72	0.30		
3	0.24	0.27	**36**		0.48	
4		0.86		**20.5**		
5			0.72	0.15	**18**	
6					0.61	**29.5**

Fig. 16 DSM with artificial Activity 2′—Example 3

(DSM1)

	1	2	3	2′	4	5	6
1	**74**						0.40
2	0.27	**20**	0.54	1	0.31		
3	0.25	0.32	**72**			0.50	
2′				**0**			
4		0.55			**41**		
5			0.29		0.17	**36**	
6						0.77	**59**

(DSM2)

	1	2	3	2′	4	5	6
1	**37**						0.35
2	0.75	**10**	0.72	1	0.30		
3	0.24	0.27	**36**			0.48	
2′				**0**			
4		0.86			**20.5**		
5			0.72		0.15	**18**	
6						0.61	**29.5**

Table 3 Stage durations (Example 3)

Stage i	Nominal-duration		Rework duration		Stage duration	
	$E[t_i]$	$\mathrm{Var}[t_i]$	$E[R_i]$	$\mathrm{Var}[R_i]$	$E[T_i]$	$\mathrm{Var}[T_i]$
1	74	1369	0	0	74	1369
2	20	100	30.53[a]	492[a]	50.53	592
3	72	1296	13.46	394	85.46	1690
4	41	420	16.44	1163	57.44	1583
5	36	324	56.66	6012	92.66	6336
6	59	870	35.48	5334	94.48	6204

[a] Rework duration of 2′

from completing the artificial Activity 2′ replaces the rework for the nominal-completion of Activity 2.

6 Accounting for parallel rework

The model considered up to this point assumes that rework generated from the completion of an activity is performed sequentially. The sequential rework assumption is comparable to the reward Markov chain approach presented by Smith and Eppinger (1997), where the sum of the off-diagonal probabilities in each column of the DSM must be ≤ 1.

For clarity, let $\mathbf{P}^{(p)}$ denote the probability rework matrix which allows for parallel rework (i.e., the sum of the off-diagonal probabilities in a column can exceed 1). For example, consider the case where Activity k can result in reworking upstream activities i and j independently. If $P_{ki}^{(p)} = 0.3$ and $P_{kj}^{(p)} = 0.8$, then the probability of reworking both activities in parallel after completing Activity k is $P_{ki}^{(p)}P_{kj}^{(p)} = 0.24$. In such a case, the event of reworking Activity i and the event of reworking Activity j are not mutually exclusive events.

Notice that artificial activities can be created to account for situations where activities are allowed to be executed in parallel. Adding artificial activities to account for all combinations of scenarios where activities could be worked in parallel would significantly increase the computational complexity involved in the system-state representation, making the analysis computationally prohibitive. This is due to the very large number of states or artificial activities that would have to be accounted for. For example, in a situation where eight activities are to be reworked in parallel, the resulting system–state representation has $2^8 - 1 = 255$ states.

We present an approximation approach to reduce the computational complexity due to parallel rework where we introduce a priority vector at Stage k, v_k, of length k, where $1 \leq v_k(j) \leq k$. Let $v_k(j)$ denote the activity with the jth priority at Stage k for $k = 1,\ldots,n$ and $j = 1,\ldots,k$. The activity with the highest priority is $v_k(1)$, the activity with the second highest priority is $v_k(2)$ and so forth. The assumption here is that the activity with the highest priority, out of the pool of activities being called for parallel rework, determines the additional rework.

The nominal-completion of an activity can result in the parallel execution of different rework paths, where a rework path is defined by a sequence of activities being reworked in series. Notice that the number of possible rework paths can be infinite, but a finite subset of paths is activated for rework [see Eppinger et al. (1997)]. The probability that a rework path is initiated can be calculated from the rework probabilities of the activities comprising the path. Our approach accounts for the random selection of the set of paths to be reworked in parallel and considers the duration of the path with the highest priority. This is achieved as follows. Whenever multiple activities are called for parallel rework, the approximation only accounts for the activity with the highest priority. The completion of the activity with highest priority in turn might activate a random set of activities for rework where the priority vector is utilized again to select the next activity to be reworked. In a similar fashion, the next activity to be reworked is selected until no further activities are called for rework. Accordingly, the parallel rework probabilities along with the priority vector would result in an approximate sequential probability vector.

Utilizing the prioritization vector, we present a transformation from the parallel rework probability DSM,$\mathbf{P}^{(p)}$, to the sequential rework probabilities. At Stage k, for $i = 1,\ldots,k$, $u = 1\ldots k$ and $i \neq v_k(u)$,

Fig. 17 Rework probabilities and proportions

DSM1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	■													
2	0.4	■							0.2					
3		0.5	■	0.4										
4	0.3		0.5	■										
5	0.4		0.5		■	0.1		0.1				0.3	0.1	
6	0.1				0.4	■								
7	0.4					0.4	■							
8						0.5		■				0.5		
9	0.4		0.5	0.5				0.5	■					
10				0.1		0.5	0.2	0.1		■	0.4			
11						0.5	0.5	0.5		0.5	■			
12	0.4					0.4	0.5			0.5	0.4	■		
13	0.5				0.5							0.4	■	
14	0.3	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	■

DSM2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	IC
1	■														35%
2	0.5	■							0.1						20%
3		0.3	■	0.5											60%
4	0.4		0.8	■											33%
5	0.1		0.1		■	0.1		0				0.3	0.1		40%
6	0.1				0.3	■									100%
7	0.5					0.8	■								35%
8						0.5		■				0.5			100%
9	0.3		0.3	0.3				0.3	■						25%
10				0.1		0.5	0.4	0.3		■	0.3				50%
11						0.5	0.5	0.3		0.3	■				75%
12	0.5					0.3	0.5			0.5	0.5	■			30%
13	0.9				0.9							0.3	■		28%
14	0.5	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	■	70%

$$P_{i,v_k(u)} = P^{(p)}_{i,v_k(u)} \prod_{z=1,v_k(z)\neq j}^{u-1} \left(1 - P^{(p)}_{j,v_k(z)}\right) \quad (26)$$

Equations (3) and (4) only consider sequential rework where the sum of the off-diagonal probabilities in a DSM column is <1. The transformations in (26) utilize prioritization in order to utilize Eqs. (3) and (4) for the case where the sum of the off-diagonal probabilities in a DSM column is allowed to be >1.

7 Case study

A simulation algorithm is developed in Browning and Eppinger (2002) to approximate the duration and cost of a PD network. The inputs to the simulation algorithm include the rework probability and proportion DSMs, information on the probability distribution of the activity durations, and improvement curves (IC) to account for learning factors. They simulate an actual industrial case of an uninhabited combat aerial vehicle (UCAV). We utilize inputs presented in Browning and Eppinger (2002) to illustrate the CE approach and compare our results to their simulation algorithm results. The rework probabilities and proportions as provided by Browning and Eppinger (2002) are presented in Fig. 17 along with the improvement curves. The first two moments of the nominal-completion of the 14 activities is provided in Table 4.

The first step is to transform the probability DSM of Fig. 17 to a sequential probability DSM using Eqs. (26). The activity prioritization vector,v_k, is determined by the sequence of activity execution as presented by the DSM where the more upstream the activity, the higher the priority (i.e., lower numbered activities have a higher priority). The resulting sequential DSM is presented in Fig. 17. Consider Column 11 in Fig. 17 to illustrate the implementation of the probability transformation of Eq. (26). Activity 11 can cause rework to activities 10, 12 or 14

Table 4 Activity durations

Activity i	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$E[t_i]$	2.3	6.17	3.22	10.5	18.533	10	8.4	6.17	20	12.33	18.53	15.77	32.83	5.25
$E[t_i^2]$	5.35	38.87	10.49	110.79	351.05	100.17	70.91	38.87	400.67	155.46	351.05	249.83	1079.54	27.70

Fig. 18 Probability DSM after prioritization

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	■													
2	0.4	■							0.2					
3		0.5	■	0.4										
4	0.18		0.5	■										
5	0.17		0.25		■	0.1		0.1				0.3	0.1	
6	0.025				0.4	■								
7	0.09					0.36	■							
8						0.27		■				0.35		
9	0.05		0.125	0.3				0.45	■					
10				0.03		0.135	0.2	0.045		■	0.4			
11						0.067	0.4	0.202		0.5	■			
12	0.03					0.027	0.2			0.25	0.24	■		
13	0.02				0.3							0.14	■	
14	0.01	0.2	0.05	0.11	0.12	0.016	0.08	0.08	0.32	0.1	0.144	0.084	0.36	■

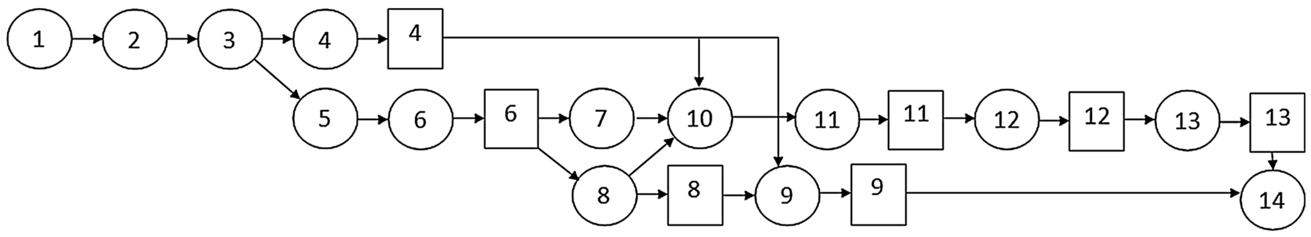


Fig. 19 Network with rework nodes (UCAV case study)

depending on the current stage. Activity 10 has the highest priority, and as a result $P_{11,10} = P_{11,10}^{(p)} = 0.4$. In the case where Activity 11 is recalled at a Stage K, for $K = 12$ or $K = 13$, then activities 10 or 12 can be recalled. As a result, the probability of recalling Activity 12 is $P_{11,12} = P_{11,12}^{(p)}\left(1 - P_{11,10}^{(p)}\right) = 0.24$. Similarly, if Activity 11 is reworked at Stage 14, the next resulting activity to be reworked can be 10, 12, or 14. Since Activity 14 has the lowest probability, it is reworked with probability $P_{11,14} = P_{11,14}^{(p)}\left(1 - P_{11,10}^{(p)}\right)\left(1 - P_{11,12}^{(p)}\right) = 0.144$.

Notice that according to the UCAV DSM, completing Activity 14 does not result in any rework. The probability $P_{11,14}$ is calculated for completeness in Fig. 18 although Activity 11 is recalled for rework at Stage 14 with probability 0. The learning proportions are accounted for by multiplying the IC with the associated rework proportion. The activity precedence relationships and AON are inferred from the DSM, Fig. 19. The CE approach is implemented, and the resulting mean and variance at each stage are given in Table 5. Traditional project management techniques can be used to analyze the obtained network. We choose spreadsheet Monte-Carlo simulation for our analysis (Seal 2001; Davis 2008). A relatively straightforward approach is to simulate the duration of the five possible paths in the network of Fig. 19 where the probability distribution of each activity is approximated by a lognormal distribution. At each iteration of the Monte-Carlo simulation, the network duration is the maximum duration of the paths. We perform 10,000 iterations and present the simulated point estimate of the network duration and half-width in Table 6. The resulting point estimate obtained by simulating the no feedback network is compared to the simulation results of Browning and Eppinger (2002) where the percentage difference is 1.99 %.

8 Summary and concluding remarks

The approach presented in this paper merges the literature on DSM with classical project management techniques. The CE approach is introduced to reduce a PD project stochastic network (i.e., with feedback) to a traditional

Table 5 Stage durations

Stage i	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$E[T_i]$	2.30	6.17	3.22	11.68	18.53	10.20	8.40	6.37	20.44	12.33	20.50	20.09	33.66	5.25
$\mathrm{Var}[T_i]$	0.06	0.83	0.12	5.10	7.57	0.81	0.35	1.90	3.25	3.36	19.67	35.41	12.79	0.14

Table 6 UCAV case study results

	Expected duration (days)	Half-width
CE network simulation	140.87	0.18
Browning and Eppinger's simulation	138	
% Mean Error	1.99 %	

project management network (i.e., with no feedback). The no feedback network is obtained analytically without resorting to simulation. The proposed method requires the user to input the rework probability and proportion DSMs as well as the calculated first two moments of the duration of each activity. The first two moments at every stage can be calculated using the Algorithm presented in the Appendix.[4] An AON with stochastic durations and no feedback is obtained where every stage represents a node. Achieving the stochastic and no feedback network does not require simulation, and the moments of the stage durations can be automatically computed based on the DSM rework probabilities and proportions. Existing approaches to analyzing the resulting stochastic networks with no feedback include the classical CPM/PERT approaches and Monte-Carlo simulation, which is relatively straightforward in no feedback project networks.

The main source of computational complexity involved in obtaining the expected duration is to solve a system of linear equations where the number of equations does not exceed $(k^2 + 1)$. The complexity to obtain a higher moment requires an additional system of linear equations to be solved which does not exceed $(k^2 + 1)$ as well. An algorithmic approach is presented in the Appendix to generate the system of linear equations for the first two moments for any probability and proportion rework DSMs. It is worth noting again that obtaining the stochastic no feedback network does not require Monte-Carlo simulation.

The computations are efficient for networks with <30 activities where the solution can be obtained within seconds. The computational effort increases exponentially as the number of activities increases. A fully connected 50 activity DSM requires 3.75 min to solve and around 15 min to solve a 100 activity DSM on a core i5 CPU with 6 GB RAM. This paper considers sequential and mixed networks where every activity and the rework resulting from completing the activity for the first time denote a stage. At the stage level, modeling parallel rework can be computationally extensive. The computational complexity of reworking activities in parallel is accounted for by prioritizing the activities that need to be reworked simultaneously. Prioritization schemes can be as straightforward as giving priority to the more upstream activities or can involve a thorough investigation of the looping generated by each activity. Future work can incorporate more complex looping prioritizations schemes to account for parallel rework. The authors in Martinez Leon et al. (2013) for example present a loop criticality index. Future work can also address dynamically changing PD networks where the DSM entries are dependent on the stage or on the rework order.

A main approximation of the CE approach is that the durations of the stages are independent. The independence approximation would not work well if the rework at a certain stage is highly correlated with the rework involved in different stages. For example, Activities i and j are reworked at Stage z if and only if they are reworked at Stage x. An approach to account for high dependency between stages can take place during the analysis of the stochastic no feedback network. For example, if simulation is used to obtain point estimates on the duration of the resulting stochastic network, then correlation between stages can be accounted for when generating the associated random variates.

Another source of error is a consequence of the priority approximations presented in Sect. 6 to account for parallel rework. The approximations could provide inaccurate results in the case where the rework within a stage constitutes several paths to be reworked in parallel where the durations of the paths are comparable in expected value and exhibit high variability. In such a case, the stage will

[4] Although the approach presented only calculates the mean and variance of the stage duration, the paper starts by presenting the set of linear equations to calculate the mth moment. Matching higher moments can lead to more accurate approximating distributions but can significantly increase the complexity in selecting the approximating distributions. A PD manager can chose to accept the added complexity of considering higher moments in return for the added accuracy.

have to be investigated independently where an approximate distribution of the rework duration should be provided.

Appendix

See Figs. 20, 21 and 22.

```
Step 1: Set $\mathbf{A} = \text{zero}\left(\left(k^2+1\right)\text{x}\left(k^2+1\right)\right)$

Step 2: For $i = 1,\ldots,k$
            For $j = 1,\ldots,k$
                  For $u = 1,\ldots,k$
                        Set $row = (i-1)k + j$
                        Set $col = (j-1)k + u$
                        Set $\mathbf{A}(row, col) = \mathrm{P}_{ju}$
      Set $row = k^2 + 1$
      For $u = 1,\ldots,k$
            Set $col = (k-1)k + u$
            Set $\mathbf{A}(row, col) = \mathrm{P}_{ku}$

Step 3: Set $\mathbf{A}$    $\mathbf{I} - \mathbf{A}$
```

Fig. 20 Algorithm to populate the entries of A

```
For $i = 1,\ldots,k$
      For $j = 1,\ldots,k$
            Set $v = (i-1)k + j$
            Set $\mathrm{b}_{\mathbf{1}}(v) = \mathrm{W}_{ij}\,\mathrm{E}[t_j]$
Set $\mathbf{b_1}(k^2+1) = \mathrm{E}[t_k]$
```

Fig. 21 Algorithm to populate the entries of b_1

```
For $i = 1,\ldots,k$
      For $j = 1,\ldots,k$
            Set $v = (i-1)k + j$
            Set $\mathrm{b}_{\mathbf{2}}(v) = \mathrm{W}_{ij}^2\,\mathrm{E}\left[t_j^2\right] + 2\mathrm{W}_{ij}\,\mathrm{E}[t_j]\left(\sum_{u=1}^{k}\mathrm{E}[R_{ju;k}]P_{ju}\right)$
Set $\mathrm{b}_{\mathbf{2}}(k^2+1) = \mathrm{E}\left[t_k^2\right] + 2\mathrm{E}[t_k]\left(\sum_{u=1}^{k}\mathrm{E}[R_{ku;k}]P_{ku}\right)$
```

Fig. 22 Algorithm to calculate the elements of b_2

References

Abdelsalam H, Bao H (2006) A simulation-based optimization framework for product development cycle time reduction. IEEE Trans Eng Manag 53(1):69–85

Braha D, Bar-Yam Y (2007) The statistical mechanics of complex product development: empirical and analytical results. Manag Sci 53(7):1127–1145

Browning TR (2001) Applying the design structure matrix to system decomposition and integration problems: a review and new directions. IEEE Trans Eng Manag 48(3):292–306

Browning TR, Eppinger SD (2002) Modeling impacts of process architecture on cost and schedule risk in product development. IEEE Trans Eng Manag 49(4):428–442

Browning TR, Ramasesh RV (2007) A survey of activity network-based process models for managing product development projects. Prod Oper Manag 16(2):217

Cho S, Eppinger S (2005) A simulation-based process model for managing complex design projects. IEEE Trans Eng Manag 52(3):316–328

Davis R (2008) Teaching note-teaching project simulation in excel using PERT-beta distributions. INFORMS Trans Educ 8(3):139–148

Eppinger S, Whitney D, Smith R, Gebala D (1994) A model-based method for organizing tasks in product development. Res Eng Design 6(1):1–13

Eppinger S, Nukala M, Whitney D (1997) Generalized models of design iteration using signal flow graphs. Res Eng Design 9(2):112–123

Hendricks KB, Singhal VR (1997) Delays in new product introductions and the market value of the firm: the consequences of being late to the market. Manag Sci 43(4):422–436

Herm S (2013) When things go wrong, don't rely on committed consumers: effects of delayed product launches on brand trust. J Prod Innov Manag 30(1):70–81

Hum SH, Sim HH (1996) Time-based competition: literature review and implications for modelling. Int J Oper Prod Manag 16(1):75–90

Karniel A, Reich Y (2009) From DSM-based planning to design process simulation: a review of process scheme logic verification issues. IEEE Trans Eng Manag 56(4):636–649

Majava J, Haapasalo H, Belt P, Mottonen M (2013) Product development drivers in literature and practice. Int J Prod Dev 18(6):512–530

Mantel S, Meredith J, Shafer S, Sutton M (2007) Project management in practice. Wiley, New York

Martinez Leon HC, Farris JA, Letens G, Hernandez A (2013) An analytical management framework for new product development processes featuring uncertain iterations. J Eng Technol Manag 30:45–71

Meier C, Yassine AA, Browning TR (2007) Design process sequencing with competent genetic algorithms. ASME J Mech Des 129:566–585

Osborne RP (1993) Product development cycle time characterization through modeling of process iteration. Master Thesis, MIT

Pinkett R (1998) Product development process modeling and analysis of digital wireless telephones. Master Thesis, MIT

Pinto JK (2012) Project management: achieving competitive advantage, 3rd edn. Prentice Hall, Upper Saddle River

Seal KC (2001) A generalized PERT/CPM implementation in a spreadsheet. INFORMS Trans Educ 2(1):16–26

Smith R, Eppinger S (1997) A predictive model of sequential iteration in engineering design. Manag Sci 43(8):1104–1120

Steward D (1981) The design structure system: a method for managing the design of complex systems. IEEE Trans Eng Manag 28(3):428–442

Takeuchi H, Nonaka I (1986) The new product development game. Harv Bus Rev 64(1):137–146

Ulrich K, Eppinger SD (2008) Product design and development. McGraw-Hill/Irwin, New York, NY

Unger DW, Eppinger SD (2009) Comparing product development processes and managing risk. Int J Prod Dev 8(4):382–402

Yassine A (2004) An introduction to modeling and analyzing complex product development processes using the design structure matrix (DSM) method. Quadermidi Manag (Italian Management Review) 51(9):72–88

Yassine A, Braha D (2003) Complex concurrent engineering and the design structure matrix method. Concurr Eng 11(3):165–176