Proceedings of DETC'98
1998 ASME Design Engineering Technical Conferences
September 13-16, 1998, Atlanta, Georgia, USA

DETC98/DAC-6013

USING THE DESIGN STRUCTURE MATRIX TO ESTIMATE PRODUCT DEVELOPMENT TIME

Maria Carrascosa
Department of Mechanical Engineering
Massachusetts Institute of Technology
Room 3-436, 77 Massachusetts Avenue
Cambridge, MA 02139
Email: carras@mit.edu

Steven D. Eppinger
Sloan School of Management
Massachusetts Institute of Technology
Room E53-347, 77 Massachusetts Avenue
Cambridge, MA 02139
Email: eppinger@mit.edu

Daniel E. Whitney
Center for Technology, Policy and Industrial Development
Massachusetts Institute of Technology
Room E40-243, 77 Massachusetts Avenue
Cambridge, MA 02139
Email: dwhitney@mit.edu

ABSTRACT

This model estimates the probability of completing a product development process over time. The Design Structure Matrix (DSM) framework is used to capture the information dependencies between tasks using the concepts of Probability of Change and Impact. The model incorporates a stochastic element that represents the likelihood of changes resulting in task iterations.

The model captures the dynamic behavior of a product development process formed by a combination of parallel, serial and coupled tasks. The model relaxes the assumption that coupled tasks take place in a complete parallel or serial iteration. It can be used to compare the development time of the project for different task sequences and overlapping degrees. This tool allows for identification of the leverage points in the system, providing information about the most effective way to reduce development time.

This project was a joint effort with a Hewlett-Packard division, and the observations and practical application presented are based on this field experience.

INTRODUCTION

Product development process is the new ground for competitive advantage in manufacturing firms. Companies are facing increasing pressure to reduce the time to market of their development efforts. This need is forcing companies to focus in the development process itself and to efficiently organize people, resources and processes to dramatically improve their performance. Accordingly, much has been written about product development performance and improvement (Takeuchi and Nonaka (1986), Clark and Fujimoto (1987), Stalk (1988), Whitney (1990), Wheelwright and Clark (1992)).

Alexander (1964) introduced the information processing view to analyze a development process. From this perspective, a product development effort is the process of transforming input information about customer needs into output information corresponding to manufacturable designs and tooling for production. Each individual development activity is an information processing unit that receives information from previous activities and transforms it into information suitable to be used by subsequent tasks. Information needs create dependencies between tasks that determine the product development structure and the most appropriate task sequencing. According to the information dependencies between them (Eppinger *et al* (1994)) the tasks can be classified (Figure 1) as: *Parallel* if there is no information exchange and can thus be performed simultaneously, *serial* if there is an uni-directional information flow and *coupled* tasks if they are mutually dependent and the information flows both ways.

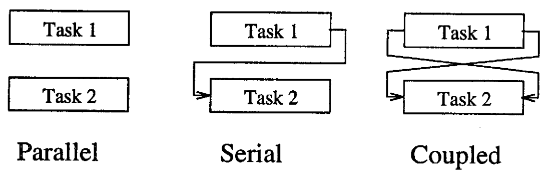


Figure 1. Parallel, serial and coupled tasks

Steward (1981) developed the Design Structure Matrix (DSM) as a tool to identify the dependencies between tasks and to sequence the development process. In this matrix, a task is assigned to a row and a corresponding column. Reading down a column reveals which tasks receive information from the task corresponding to the column. Reading across a row reveals all the tasks whose information is required to perform the task corresponding to the row. Figure 2 shows an example DSM.

	A	B	C	D	E	F	G	H	I
A	╳								
B	X	╳							
C	X	X	╳						
D			X	╳					
E	X	X	X		╳				
F			X	X	X	╳			
G	X	X	X			X	╳	X	X
H	X	X				X	X	╳	X
I							X	X	╳

Figure 2. Design Structure Matrix

Coupled tasks present a challenge in the product development process due to the complexity of the information exchange. These tasks are quite common (Kline (1985)) and need information from each other to perform their problem-solving activity. Reflecting industry practice, most of the literature focuses on an iterative approach to managing coupled tasks. Modeling this situation requires an analysis of the information exchange during task execution (Eastman (1980)). There are some exceptions to this approach: Zettelmeyer (1996) analyzed the information exchanges between two coupled tasks executed serially and without possibility of iteration. Christian (1995) simulated in detail the communication exchange in a project with multiple tasks, assuming all the information transfer is certain and does not cause iteration. Ward *et al* (1997) used a design convergence approach and focused on reducing the sets of designs of each task to those that enabled overall feasible solutions.

Several authors have addressed the issue of improving a development process with multiple tasks when iteration is involved. There are two extreme ways of executing iterative design.

- In *serial* iteration tasks are performed one after each other with the understanding that the results are tentative and that each task may be repeated several times until a solution is reached. Smith (1997b) studied the serial execution of coupled tasks to obtain the task sequence that reduces total development time. Eppinger (1997) obtained the development time when task sequencing is given. Ahmadi (1994) analyzed the development time of serial coupled tasks including the learning effects gained in the iteration process. Kusiak and Wand (1993) worked on serially sequencing tasks based on a matricial representation different from DSM.
- In *parallel* iteration the tasks are completed simultaneously with frequent exchanges of information. Smith (1997a) explored parallel iteration to identify the tasks that are the main generators of iteration in the development process. AitSahlia (1995) compared the serial and parallel approach in terms of the time and cost of the development process. Hoedemaker (1995) explored the limits to parallel development processes due to the increase in communication needs between tasks.

Other authors have focused in understanding in greater detail the information exchange between 2 tasks. Krishnan *et al* (1997) describes the information exchanged using two characteristics: upstream information *evolution* and downstream iteration *sensitivity.* These concepts are used to determine the most appropriate overlapping strategy between serial tasks. Ha and Porteus (1995) explore the appropriate communication frequency between two overlapping tasks. Terwiesch and Loch (1996) present a framework for exchanging preliminary information.

These two-task models do not extende these concepts to multiple tasks. On the other hand, the multiple task models have sacrificed the detailed analysis of the information flwo for the greater scope, and have been constrained to the analysis of pure parallel or pure serial iteration. In this paper the task-to-task interactions are modeled in detail for multiple tasks projects. This approach relaxes the assumption that coupled tasks take place in a pure serial or pure parallel way. It allows for the analysis of a product development situation where both coupled and serial tasks may be partially overlapped. The DSM framework is used to describe the information exchanged.

MODEL DESCRIPTION

This model evaluates the progress of the product development process over time, and in particular, the probability of the product development process being completed over time. In all the cases tried, the probability of being completed had an S-shaped function (Figure 3). The model can be used to analyze different scenarios of the product development process by modifying the start times, the tasks duration and the characteristics of the information flow.

The model requires information about the different

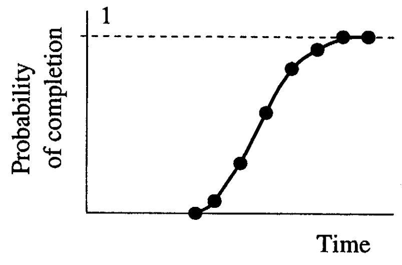


Figure 3. Probability of completion over time

tasks that integrate the development process. The input data are task durations, task start times and the information flow between each pair of tasks described by the concepts of Probability of Change and Impact describe the information. Section introduces these concepts.

Information flow characterization

The information exchanged between activities takes various forms such as part dimensions, prototypes, etc. In this paper, we follow Krishnan *et el* (1997) approach; we focus on exchanged information that can be described as a collection of parameters. A parameter is defined as a scalar or combination of scalar pieces of information that are defined simultaneously and represent the information exchange between two tasks. Examples of parametric information include part dimensions and customer specifications.

The concepts Probability of Change and Impact are used to characterize a parameter. These concepts are an extension of the concepts of *evolution* and *sensitivity* developed by Krishnan *et al.* Task evolution represents how close the unfinalized output design parameter is to its final value. It is a function of the progress of the task. Downstream sensitivity represents the relationship between the change in the upstream information from the last communication and the duration of the resulting iteration downstream.

In Krishnan's model the evolution and the sensitivity describe the uncertainty of the information and its effect based on the *size* of a change of a parameter during task execution. In our model, the probability of change and the impact of change depend on the *timing* of the changes (regardless of their size), and are a function of the state of completion of a task. The other important difference is that Krishnan's model is deterministic, and we introduce an stochastic element.

It is important at this point to define the state of completion of a task and the accumulated development time. The state of completion of task i (τ_i) measures the progress of task i in time units. It is the time passed since the start of the task minus the rework time required by all the changes affecting that task, and assumes a linear progress of the task. If a development project started 10 months ago and a specific task i began 3 months ago, then $\tau_i = 3$ months. If task i has just been affected by a change that requires 1 month of rework, its state of completion is $\tau_i = 2$ months. The accumulated development time keeps track of the time passed from the start of the project. In this example, the accumulated development time is $T = 10$ months.

The accumulated development time (T) is the 'global' time measure, while the states of completion (τ_i) keep track of the 'local' effective time spent in each task i.

Probability of change Most parameters have a default value, which represents the initial best guess or the default solution. The probability of change of a parameter represents the likelihood of the default value changing over time. It is a way to measure the uncertainty in its definition.

The probability of change of a parameter is a function of the state of completion of the task defining it. Though any shape of this curve is theoretically possible, based on our field experience, two shapes (concave-up and convex-up) represent most parameters. As more of the task is completed the parameter is more certain and its probability of change decreases. A convex-up shape means the probability of change is still high close to the task completion and the parameter is likely to have changes until late (Slow). On the other hand, a concave-up shape means the probability of change is fairly low close to the task completion and the parameter is likely to change only in the early stages (Fast). Figure 4 shows these shapes.

The probability of change of the parameter defined by task i and communicated to task j is defined as

$$\rho_{j,i}(\tau_i), 0 \leq \tau_i \leq L_i \qquad (1)$$

$$\int_0^{L_i} \rho_{j,i}(s)ds < 1 \qquad (2)$$

where τ_i is the state of completion of task i and L_i is the duration of task i if no rework is necessary. The condition of the integral of the probability of change being less than 1 assures the convergence of the development process. In reality, projects can generate changes and rework forever.

Impact The impact quantifies the effect of a change on the tasks receiving the information. The impact of change is a function of the progress of the task affected and the size of the change. Based on our field experience and due to the complexity of using both factors to characterize the impact, we made the simplifying assumption that the impact of change is only a function of the progress of the task.

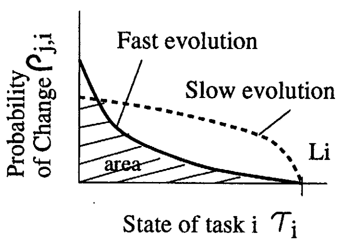


Figure 4. Probability of change: information defined by task j for task i

The impact is measured in time units. The shapes for of the impact of change identified in our field work are constant and proportional. Figure 5 shows both shapes.

The impact caused to task i due to changes from task j is defined as

$$\sigma_{i,j}(\tau_i), 0 \leq \tau_i \leq L_i \tag{3}$$

$$\sigma_{i,j}(\tau_i) \leq \tau_i \tag{4}$$

where τ_i is the state of completion of task i and L_i is the duration of task i if no rework is necessary.

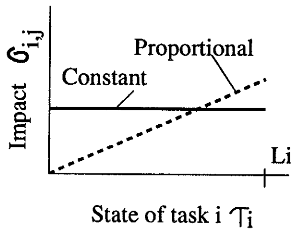
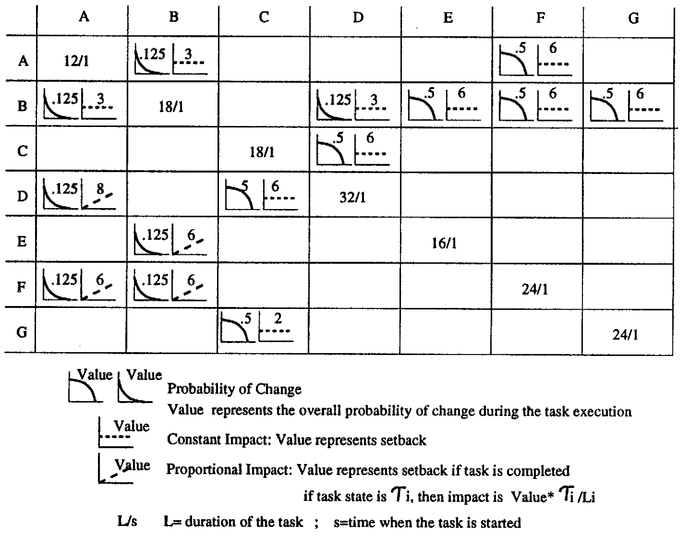


Figure 5. Impact caused by task j on task i

DSM representation The DSM framework captures these interactions between tasks. Figure 6 shows the DSM of a practical application at Hewlett-Packard where these concepts were implemented [1].

The off-diagonal cells store the probability of change and impact of the corresponding parameter. For example, cell row E, column B. The probability of change of the information coming from task B and need by task E is 0.125. The shape of the curve is concave-up, so the parameter's evolution is fast. The impact of change of task E is proportional to the state of task E: a change from task B causes



	A	B	C	D	E	F	G
A	12/1	.125 .3				.5 6	
B	.125 .3	18/1		.125 .3	.5 6	.5 6	.5 6
C			18/1	.5 6			
D	.125 8		.5 6	32/1			
E		.125 6			16/1		
F	.125 6	.125 6				24/1	
G			.5 2				24/1

Value Value Probability of Change
Value represents the overall probability of change during the task execution
Value Constant Impact: Value represents setback
Value Proportional Impact: Value represents setback if task is completed
if task state is τ_i, then impact is Value* τ_i /Li
L/s L= duration of the task ; s=time when the task is started

Figure 6. Information flow characterization using DSM

task E a setback of 6 time units if task E is completed. If task E is only x% completed, the setback is $x\% \cdot 6$ time units. If the impact of change were constant, the setback would be the same regardless of the degree of completion of the task.

The diagonal cells store two pieces of information: the first one (L) is the duration of the task if it is not affected by any changes. The second one (s) is the time when the task is started; s=1 means the task starts at the beginning of the development process. In the case studied, all the tasks where started simultaneously.

This DSM matrix is not symmetric. Tasks (A-B, C-D) are coupled in a symmetric way : the probability of a change (shape and value) and the impact are the same for both information flows. Tasks B-E, A-F are coupled, but not symmetric: the probability of change and the impact are not the same for each information flow. Also, some tasks A-D, B-G have a unidirectional information flow, where only one of the tasks can significantly affect the other.

Effect of changes in the product development process

There are many types of changes which take place in product development, i.e. specification changes, parameter changes, personnel, supplier, etc. While changes are a frequent ocurrence in the development of a new product, their impact in the development time is usually underestimated. We have observed that changes which were expected to cause a short delay in a project often take much longer to be fully absorbed by the organization. Our explanation is that changes in a task not only add work to other tasks, but

[1]Data has been disguised preserve confidentiality.

also affect the certainty of the parameters defined. The affected tasks, challenged to cope with the new situation, may not be able to work within the constraints, triggering new changes in areas not directly related to the original change. This could explain why and how changes spread through the organization and usually have a far more significant impact than anticipated.

The concepts of probability of change and impact of change reflect this behavior. To explain it, let us review the notation used so far:

T	is the accumulated development time
$\tau_i(T)$	is the state of completion of task i at time T
L_i	is the duration of task i if no rework needed
$\rho_{i,k}(\tau_k(T))$	is the probability of a change in task k affecting task i, function of $\tau_k(T)$
$\sigma_{i,k}(\tau_i(T))$	is the impact of a change caused by task k on task i, function of $\tau_i(T)$

Figure 7 graphically represents the impact of a change in a task. For the purpose of the explanation, let us assume that changes in task k at times T_1 and T_2 affect task i. We assume these changes happen with certainty, but in reality, changes have a probability of happening at any point in time. Task i begins T_0 times units after the development project begins. At time T_1 the state of task i would be $\tau_i(T_1) = T_1 - T_0$ if there are no changes. If a change takes place at time T_1, it has an impact of change $\sigma_{i,k}(\tau_i(T_1))$, so task i changes from state $\tau_i(T_1) = T_1 - T_0$ to state $\tau_i'(T_1) = T_1 - T_0 - \sigma_{i,k}(\tau_i(T_1))$. The same dynamics take place at T_2.

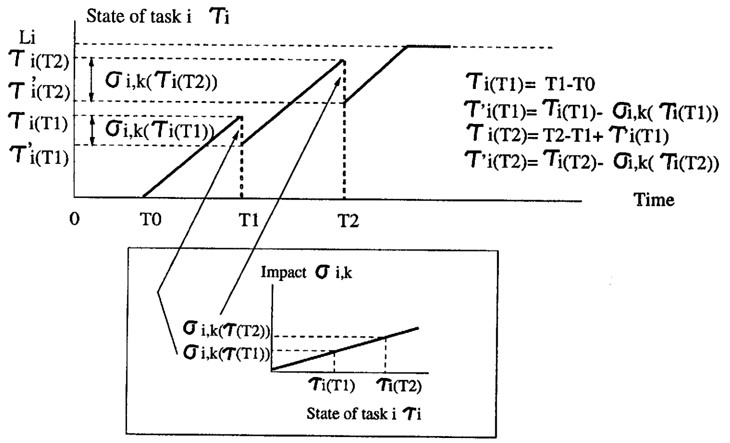


Figure 7. Progress of a task

The characteristics of task i change accordingly (Figure 8). The impact of a change from task $j, \forall j \neq i$ at time T_1 goes from $\sigma_{i,j}(\tau_i(T_1))$ to $\sigma_{i,j}(\tau_i'(T_1))$ and the probability of change from $\rho_{j,i}(\tau_i(T_1))$ to $\rho_{j,i}(\tau_i'(T_1))$.

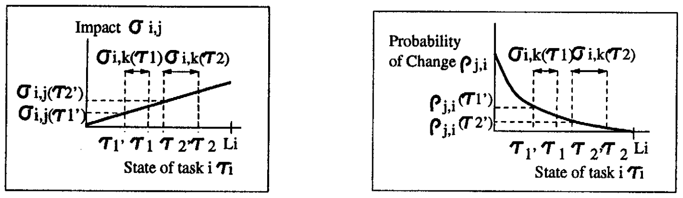


Figure 8. Effect of changes in task i

Summary of assumptions

- The interaction between each pair of tasks is defined at most by two parameters, each of them determined by one task and communicated to the other.
- A task duration is fixed unless affected by another task.
- A task i can be set back to a prior state of completion due to changes in parameters defined by the rest of the tasks. The probability of change of these parameters determine the likelihood of a setback. The impact of change of the task receiving the information defines the magnitude of the set back. When a task is set back the probability of change and the impact of all its parameters change accordingly.
- If several parameters affecting a task change simultaneously, the highest impact of change determines the set back for that task.
- All the probability of change are independent of each other.

MATHEMATICAL DERIVATION OF THE MODEL

Discrete stochastic processes

The mathematical representation chosen is a discrete stochastic model. A task is modeled as a Markov chain where the transition probabilities between states are a function of the state of all the tasks in the product development process. This representation has the following advantages: First, a Markov chain can represents the possible setback of a task by allowing transitions to previous states. Second, this mathematical representation keeps track of the probability of being in a particular state over time.

The concepts of task state of completion, the probability of change and the impact are a continuous function of time. However a Markov chain is a discrete representation of the system and a time interval Δ is chosen to make the parameters discrete. Δ is fixed through the development process and the same for all the tasks. Δ defines the sensitivity of the model, i.e: if Δ is a month, the model will not be able to discriminate between effects that are less than a month different. A good role of thumb for choosing Δ is to make it less or equal to half the relevant time measure of the problem. In some projects development the time measure is weeks, in others up to quarters or semesters. The discrete representation of the concepts presented are:

1. **Accumulated Development Time** The time index $t, (t \in [1, 2, 3, ...])$ is the discrete representation of the accumulated development time T. The relationship between t and T is

$$T = \Delta \cdot t \tag{5}$$

2. **State of completion of a task** $k_i (k_i \in [1, 2, 3...D_i)$ represents the state of completion of task i (Figure 9). D_i is the number of states of completion of task i.

$$D_i = \text{round}\frac{L_i}{\Delta} + 1 \tag{6}$$

where L_i is the duration of a task if no rework is needed. The last state D_i is an artificial state added to represent that task i is already completed and cannot cause changes in other tasks. The relationship between k_i and τ_i is

$$\tau_i = \Delta \cdot k_i \tag{7}$$

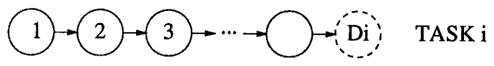


Figure 9. Markov representation of a task

3. **Probability of change** $p_{j,i}(k_i)$ represents the probability of task i having a change while in states $((k_i - 1) \cdot \Delta, k_i \cdot \Delta]$ that affects task $j, \forall j \neq i$, and is a function of the discrete state of completion k_i.
The relationship between $p_{j,i}(k_i)$ and $\rho_{j,i}(\tau_i)$ is

$$p_{j,i}(k_i) = \int_{(k_i-1)\Delta}^{k_i \cdot \Delta} \rho_{j,i}(\frac{s \cdot L_i}{\Delta \cdot (D_i - 1)})ds, \; k_i \in [1, ..., D_i - 1] \tag{8}$$

As mentioned before, state D_i represents that task i is completed. A task completed does not cause any changes that may affect other tasks. Therefore

$$p_{j,i}(D_i) = 0 \tag{9}$$

4. **Impact of change** $s_{i,j}(k_i)$ represents the impact on task i due to changes in task j while task i is in state $((k_j - 1) \cdot \Delta, k_j \cdot \Delta]$. The setback is always a multiple of the time interval Δ. $s_{i,j}(k_i) = m$ means that task i goes from state of completion k_i to state of completion $k_i - m + 1$. $s_{i,j}(k_i) = 1$ means that the current state k_i needs to be repeated. $s_{i,j}(k_i) = 0$ means that the setback is negligeable.
The relationship between $s_{i,j}(k_i)$ and $\sigma_{i,j}(\tau_i)$ is

$$s_{i,j}(k_i) = \text{round}\,\frac{\sigma_{i,j}(k_i \cdot \Delta)}{\Delta}, \; k_i \in [1, ..., D_i - 1] \tag{10}$$

A setback when the state of the task is D_i sends the task to the same state task as a setback if the state is $D_i - 1$. Therefore

$$s_{i,j}(D_i) = s_{i,j}(D_i - 1) + 1 \tag{11}$$

States and transitions in the product development process

State of the product development process The state of a product development process with N tasks at time t is

$$K = [k_1, k_2, ..., k_N] \tag{12}$$

where k_i is the state of task $i \in [1, ..., N]$ at time t. The combination of all the possible states of the system is the state space Ω.

$$\Omega = \{X = [x_1, x_2, ..., x_N], \; \forall i \in [1, N], x_i \in [1, ..., D_i]\} \tag{13}$$

Task transitions The transition between states takes place at the end of each time unit. The "default transition" is to move to the next state of completion but a task may be set back to an earlier state due to changes.

Let's define

$q_m^{k_i}$	mth state that can be accessed from state k_i
$P(q_m^{k_i}/k_i)$	Probability of the transition from k_i to $q_m^{k_i}$

The model is clock driven, the state of the product development process is evaluated at intervals of length Δ, but the transitions between states are a function of the state of the system. To explain in detail $q_m^{k_i}$, let's assume that task i can only be set back by changes in task j. In this case, $p_{i,j}(k_j)$ represents the probability of a change affecting task i and $s_{i,j}(k_i)$ the magnitude of the setback. There are two possible transitions from k_i(Figure 10):

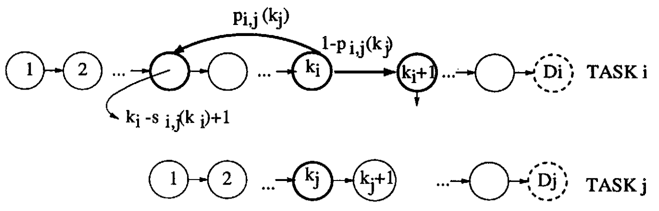


Figure 10. Transitions from state k_i

1. If there are changes in task j task i suffers a setback. The probability of a change is $p_{i,j}(k_j)$ and the magnitude of the setback is by $s_{i,j}(k_i)$.The new state is then $k_i - s_{i,j}(k_i) + 1$. At most, the setback can send a task to its starting state. To enforce this constraint the final state is calculated $\max(k_i - s_{i,j}(k_i) + 1, 1)$.
2. If there are no changes in task j, then task i moves forward to $k_i + 1$. At most, the task state is D_i. To enforce this constraint the transition state is $\min(k_i + 1, D_i)$. The probability of this transition is $1 - p_{i,j}(k_j)$.

Transitions of the development process The transitions of each of the tasks are independent of each other, so the fact that there is a transition in a task does not make it more or less likely to have a transition in another.

Matrix representation of the model

$$\Pi(t+1) = \mathbf{P}\Pi(t) \tag{14}$$

represents the progress of the product development process over time where

$\Pi(t)$	is the vector of $\pi_K(t) \;\; \forall K \in \Omega$
	$[\pi_{[1,1,...1]}(t), ..., \pi_{[k_1,k_2...k_N]}(t), ..., \pi_{[D_1,D_2,...D_N]}(t)]'$
$\pi_K(t)$	is the probability of the system being in state $K = [k_1, k_2, ...k_N]$ at time t
$\mathbf{P}$	is the transition matrix
P_{LK}	prob(L/K) is the probability that the state of the system at time $t+1$ is vector L given that the system at time t was in state K

It is important to notice that the subindex used for both $\mathbf{P}$ and π are vectors. The component of the vector/matrix with subindex K refers to state K, but does not indicate the position in the vector/matrix[2].

$\mathbf{P}$ is a constant matrix because the state of the system at time $t+1$ depends only on the state of the state at time t and of no former history. We use this property to determine the state of the product development process at time $t+1$:

$$\Pi(t+1) = \mathbf{P}\Pi(t) = \mathbf{P}\mathbf{P}\Pi(t-1) = \mathbf{P}^t\Pi(1) \tag{15}$$

where $t = 1$ is the initial time, $\Pi(1) = \pi_K(1) = 0, \forall K \neq A, \pi_A(1) = 1$ where $A = [a_1, a_2, ...a_N]$ is the inital state of the development process.

In a general case, the product development process started at $c \geq 0$ time units ago. The accumulated development time is calculated from the current time. For the tasks that have already started, the initial state is their current state. If task i comprising D_i states was scheduled to start $m > c$ time units after the beginning of the development process, then a number of states $b_i = \text{round}\frac{m-c}{\Delta}$ are added at the beginning of the task i such that now $k_i \in [1, ..., b_i, 1 + b_i, ..., D_i + b_i]$. The characteristics of the new states are $\forall j \neq i, s_{i,j}(k_i) = 0, p_{j,i}(k_i) = 0$ if

[2] Any relationship that identifies a state with a position in the vector/matrix can be used. The equivalence we used was $K = [k_1, k_2, ...k_N]$ is located in position $\kappa = 1 + \sum_{i=1}^{N}(k_i - 1)\prod_{l=i+1}^{N} D_l$.

$k_i \in [1, ..., b_i]$ and the rest of the states keep their Probability of Change and Impact of Change characteristics. The initial state of task i is $a_i = 1$

$\pi_D(t)$ is component representing the probability of the product development process being completed by time t. $D = [D_1, D_2, ...D_N]$ is the state representing all tasks are in their completion states.

Vector TTM represents the probability of the product development process being completed over time. Θ is the time frame in which the product development process will be completed with certainty $1 - \epsilon,\ \epsilon \to 0$.

$$TTM = [\pi_D(1), \pi_D(2), ...\pi_D(\Theta)]. \tag{16}$$

Matrix P

P is constant over time. Each column K of **P** represents the transitions that are possible from state $K = [k_1, k_2, ..., k_N]$ in one time unit. There is a two step process to obtain each column K:

1. Obtain the transitions for each task i $\forall i \in [1, N]$. The state of the system is evaluated every time unit. Task i is in state k_i and can be set back by any task from which it receives information. A change coming from task $j \neq i$ determines the setback of task i if task j is the task with the highest impact of change of those affecting i. Without losing generality, we assume for the purposes of the explanation that $[s_{i,1}(k_i) > s_{i,2}(k_i) > ... > s_{i,i-1}(k_i) > s_{i,i+1}(k_i), ..., > s_{i,N}(k_i)]$. Therefore, the probability of task i being set back by task 1 is the probability $(p_{1,l}(k_1))$ of task 1 having a change. For any other task $j \neq i, j > 1$, the probability of task i being set back by task j is the probability of task j having a change and there not being a change in any task with a higher impact of change on task i $(\Pi_{l=1}^{j-1}(1 - p_{i,l}(k_1))$. Using this reasoning and the results from Section :

task i affected by task	$q_m^{k_i} =$ states that can be accessed from k_i	$P(q_m^{k_i}/k_i) =$ $\text{prob}(x_i(t+1) = q_m^{k_i}/x_i(t) = k_i\ )$
1	$\max(k_i - s_{i,1}(k_i) + 1{,}1)$	$p_{i,1}(k_1)$
j	$\max(k_i - s_{i,j}(k_i) + 1{,}1)$	$p_{i,j}(k_j)\prod_{l=1}^{j}(1 - p_{i,l}(k_l))(j \neq i)$
N	$\max(k_i - s_{i,N}(k_i) + 1{,}1)$	$p_{i,N}(k_N)\prod_{l=1}^{N-1}(1 - p_{i,l}(k_l))(j \neq i)$
none	$\min(k_i + 1, D_i)$	$\prod_{l=1}^{N}(1 - p_{i,l}(k_l))(j \neq i)$

As task i cannot affect itself, the maximum number of different transitions for task i is N ($N - 1$ due to changes caused by a task plus the case of no changes). If impact of task i to changes in several tasks is the same, some of the resulting states $q_m^{k_i}$ are identical and can be combined. The probability of the resulting state is the sum of the probabilities. In general, there are $C_i \leq N$ different transitions that are possible from k_i. These states are $[q_1^{k_i}, ..., q_{C_i}^{k_i}]$.

2. Obtain the transitions for the state of the development process K. So far we have characterized how the state of a task changes over time. The transitions from states happen simultaneously in all the tasks at the end of each time unit, and we are interested in knowing how the whole product developing process is evolving over time.
The transitions of each possible state are independent of each other, and so the number of possible transitions is the combination of the transitions of each individual tasks $\prod_{i=1}^{N} C_i$

$$\begin{aligned} \mathbf{P}_{LK} &= \text{prob}(L/K) \\ &= \prod_{i=1}^{N} P(l_i/k_i) \quad l_i \in [1, ..., D_i]\ \forall i \\ P(l_i/k_i) &= 0 \text{ unless } l_i \in [q_1^{k_i}, ..., q_{C_i}^{k_i}]\ \forall i \end{aligned} \tag{17}$$

Following these 2 steps $\forall K$ matrix P is obtained

PRACTICAL APPLICATION

This application illustrates the use of the model on a completed pilot project at Hewlett Packard. The product development process consisted of seven different tasks, mainly in R&D and manufacturing, and involved two different divisions. The data have been disguised to preserve confidentiality.

Data Gathering

The first part of the data gathering consisted on defining the different tasks, and identifying the most significant parameters exchanged between them. This information was obtained by interviewing 2-3 senior engineers in the project. The second step was characterizing the probability of change and impact of change of these parameters. These interviews also identified the different types of probability of change and impact of change presented. About 15 engineers involved in the development process were interviewed in this second stage.

Results

The model predicts that the probability of being completed by the completion of the project ($t = 6$) is 80% (Figure 11). The model tends to give conservative estimates of the Probability of Completion, so the results seem reasonable.

Sensitivity Analysis

The effect of inaccurate estimates was evaluated in Figure 11. The probability of change of all the parameters was increased and decreased by 20 % to quantify the effect in the development time. It is worth noting that in this case the penalty for increasing the probability of change is much smaller than the gain from decreasing them.

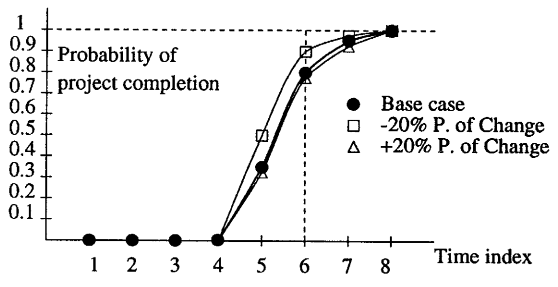


Figure 11. Base case and sensitivity analysis

Further analysis determined the leverage points of the system (which small changes would yield the biggest reduction in the development time). The changes that would have the most significant effect on the development time are:

- Case 1: Reducing the longest task duration. Figure 12 shows the effect of reducing by 25% the duration of the longest task.
- Case 2: Reducing the probability of change of the parameters that affect the longest task. Figure 12 shows the effect of reducing by a factor of 4 the probability of changes impacting the longest task.
- Case 3: Reducing the probability of change of the tasks that can affect the greatest number of tasks. Figure 12 shows the effect of reducing by 50% the probability of change of this parameter.

CONCLUSIONS

In this model, a richer understanding of the coupling between product development tasks is gained. The framework characterizes the uncertainty in the definition of the information and the impact of change of changes in the development process for bi-directional information flow. It analyzes a general product development situation where the coupled tasks take place in a mixture of serial and parallel iterations and where the serial tasks are partially overlapped.

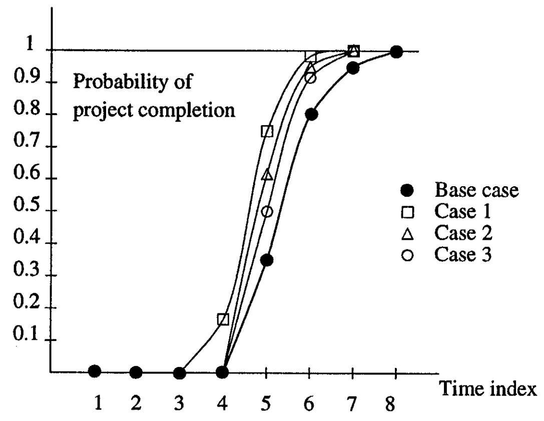


Figure 12. Opportunities for improvement

The model provides an explanation of how changes ripple through the organization and usually have a more significant effect than anticipated: when a change affects a task, this task is challenged to cope with the new situation and may not be able to work within the new constraints, triggering new changes in areas not directly related with the original change.

The model estimates the probability of the product development project being completed over time. It can be used to identify the leverage points in the system and to provide information about the most effective ways to reduce development time. The effect of different task sequencing, degrees of concurrency and start times can be evaluated.

There are several limitations to the model: first, the model does not include learning effects. If a task is setback and goes to a prior stage of completion, the experience gained in the first pass of the task may speed up the process, or reduce the probability of further changes; this effect is not taken into account in the model. Second, there is no control in the evolution of the product development process. Usually in the development process some measures can be taken to paliate the effect of a change (increase resources in that task, reduce product features....), however these dynamics are not considered. Third, the number of tasks that it can handle is limited to a few. This is due to computational limitations caused by the fact that the number of states of the system increases very rapidly with the number of tasks.

REFERENCES

AitSahlia, F., Johnson, E and Will, P. "Is Concurrent Engineering Always a Sensible Proposition?." IEEE transactions on Engineering Management, Vol. 42, No.2, 1995.

Alexander, C. *"Notes on the Synthesis of Form."* Harvard University Press, Cambridge, 1964.

Ahmadi, R., and Hongbo W. "Rationalizing Product Design Development Process." Working Paper, Anderson Graduate School of Management, UCLA, 1994.

Christian, A.D. and Seering, W.P. "A Model of Information Exchange in the Design Process." 1995 Design Engineering Technical Conference DE-Vol.83.

Clark, K.B. and Fujimoto, T. "Overlapping Problem Solving in Product Development." Harvard Business School Working Paper No. 87-048, Cambridge MA, 1987.

Eastman, R. "Engineering Information Release Prior to Final Design Freeze." IEEE transactions on Engineering Management Vol. EM-27, No.2, 1980, 37-41.

Eppinger,S., Whitney,D.,Smith,R. and Gebala, D. "A Model-Based Method for Organizing Tasks in Product Development." *Research in Engineering Design,* Vol.6, 1-13, 1994.

Eppinger, S. Nukala, M. and Whitney, D. "Generalized Models of Design Iteration Using Signal Flow Graphs." *Research in Engineering Design,* Vol. 9, No. 2, 1997.

Kline, S. "Innovation is not a Linear Process." *Res. Management*, 28, 4 (1985), 36-45.

Krishnan, V. , Eppinger, S. and Whitney, D. "A Model-Based Framework to Overlap Product Development Activities." *Management Science* Vol. 43, No. 4, April 1997.

Kusiak, A. and Wang, J. "Efficient Organizing of Design Activities." *International Journal of Production Research* Vol. 31, No. 4, 1993.

Ha, A. and Porteus, E. "Optimal Timing of Reviews in the Concurrent Design for Manufacturability." *Management Science* Vol. 41, No. 9, September 1995.

Hoedemaker, G., Blakburn, J. and Wassenhove, L. "Limits to Concurrency." Vanderbilt University Owen School of Management. Working paper 1995.

Smith, R., Eppinger, S. "A Predictive Model of Sequential Iteration in Engineering Design." *Management Science* Vol. 43, No.8, August 1997.

Smith, R., Eppinger, S. "Identifying Controlling Features of Engineering Design Iteration." *Management Science* Vol. 43, No.3, March 1997.

Stalk, G. "Time-The Next Source of Competitive Advantage." *Harvard Business Review* July-August 1988, 41-51.

Steward, D. "The Design Structure Matrix: A Method for Managing the Design of Complex Systems." IEEE Transactions on Engineering Management, Vol EM-28, No. 3, August 1981.

Takeuchi, H. and Nonaka, I. "The New New Product Development Game" *Harvard Business Review* January-February 1986, 137-146.

Terwiesch, C and Loch, C. "Management of Overlapping Development Activities: a Framework for Exchanging Preliminary Information" INFORMS October 97 presentation.

Terwiesch, C and Loch, C. "Communication and Uncertainty in Concurrent Engineering." INSEAD Working Paper 96/68/TM, 1996.

Ward, A and Finch,W. "A Set-Based Aystem for Eliminating Infeasible Designs in Engineering Problems Dominated by Uncertainty." Proceedings of DETC'97, ASME Design Engineering Technical Conference. DETC97/DTM-3886.

Wheelwright, S and Clark, K. *"Revolutionizing Product Development: Quantum Leaps in Speed, Efficiency and Quality."* Free Press, New York, 1992.

Whitney, Daniel E. "Designing the Design Process." *Res. Engineering Design*, 2 (1990), 3-13.

Zettelmeyer, F. "On the Optimality of Market Orientation." University of Rochester Working Paper, June 1996.