MIT Sloan School of Management

Working Paper 4362-02
April 2002

A COORDINATION-THEORY APPROACH TO EXPLORING PROCESS ALTERNATIVES FOR DESIGNING DIFFERENTIATED PRODUCTS

Naoki Hayashi, George Herman

A coordination-theory approach to exploring process alternatives for designing differentiated products

Naoki Hayashi George Herman
Center for Coordination Science, MIT Sloan School of Management
nao@mit.edu gherman@mit.edu

Abstract

This paper describes a new systematic method for exploring and evaluating alternatives of a product design process for differentiated products - those that share some elements but also have differentiating features. Based on coordination theory, the method clarifies the opportunities and risks of process alternatives. The method consists of three steps:

1) finding applicable differentiation approaches,
2) finding applicable patterns of process coordination, and
3) evaluating total costs of the process alternatives.

We categorized the differentiation approaches as a taxonomy of design processes; the taxonomy includes approaches of adding or removing differentiating elements or sorting results. We also categorize how these are limited by type of interim resource in a design process. We outline three patterns of process coordination and how this interacts with the choice of product differentiation approaches. We show how the process alternatives vary in the success rate of the coordination and how this probability affects total cost of executing a design process. It raises an awareness of the importance of managing dependencies between activities, which many process analyses don't focus on.

We also show how to calculate the success rate associated with varying the coordination cost or how to calculate coordination cost associated with a desired success rate. These calculated values indicate "break-even points" for the cost of the process.

Keywords

Coordination theory, Product design, Process analysis

Introduction

The speed of commerce continues to increase [Fine 1998]. Also, the global nature of commerce increases the span of who are competitors [Christensen 1997; Porter 1998]. Leisurely product line introduction no longer keeps market share as product lifecycles shrink. To improve and to sustain competence of product lines, rapid launches of varied products are required [Clark & Fujimoto 1991; Robertson & Ulrich 1998]. One stumbling block to quick product introduction is the product design process. Reducing the time (and therefore cost) to design varied products is key in many industries [Fujimoto et al 1999; Fujimoto et al 2001; Iansiti 1997; Ulrich 1995]. One approach taken is to adopt a product differentiation approach based on sharing some elements among varied products [Meyer & Lehnerd 1997; Nobeoka & Cusumano 1995; Ulrich & Eppinger 2000; Wheelwright & Clark 1992].

To implement a product differentiation approach, it is required that the approach is mapped to an executable process correctly. Many would agree upon a process for designing a product under a specific context (e.g., organizational capability, market positions of itself and competitors, government's rules for competition). At the same time, they would agree that there is no always-the-best structure [Drucker 1999]. Thus, in general, one would have to explore a design space where there are many process alternatives.

Lee and Tang have proposed a production process model in which one serialized chain of operations is divided into two at a specific point, and have explained differentiation approaches and their effects by shifting the chain-dividing point in that production process [Lee & Tang 1997]. However, they have not addressed product design processes; their model does not describe process alternatives of a design process. There is also excellent research about how to create process structures [Crowston 1997; Davenport 1993; Grover et al 1995; Kettinger et al 1997; Malone et al 1999; Nadler & Tushman 1997; Pentland et al 1999; Sterman 2000; von Hippel 1990] and also how to differentiate products [Kotler 1999; Nobeoka 1996; Porter 1990; Ulrich & Eppinger 2000]. However, the research has not given clear, theoretical explanation associating a type of differentiated product and the structure of a product design process. Thus, one cannot systematically explore design process alternatives that meet a specific differentiation approach. Not having an easy way to choose the correct product design process can lead to delay (and therefore cost) or impact the quality of the product design by choosing an inferior product design process.

In this paper, we address one question: How can one systematically find a product design process that fits a specific differentiation approach? Based on coordination theory, we describe a new systematic method for exploring and evaluating alternatives of a product design process for differentiated products.

The method

Background: Coordination theory

From the view of coordination theory, a process consists of three types of elements: resources, activities, and dependencies. A ***resource*** is produced and/or consumed during a process. For example, material used in a production process is a resource. Equivalently, specification documents, drawings, and mock-ups of a product are resources in a product design process. An ***activity*** is a partitioned action that produces and/or consumes resource(s); for example, "assembling material" is an activity. Activities are themselves processes and we use the two terms interchangeably. A ***dependency*** is a relation among activities mediated by producing or consuming resource(s); for example, there is a dependency between "procuring material" and "assembling material." There are three basic types of dependencies: flow, sharing, and fit (Fig.1) [Malone et al 1999].

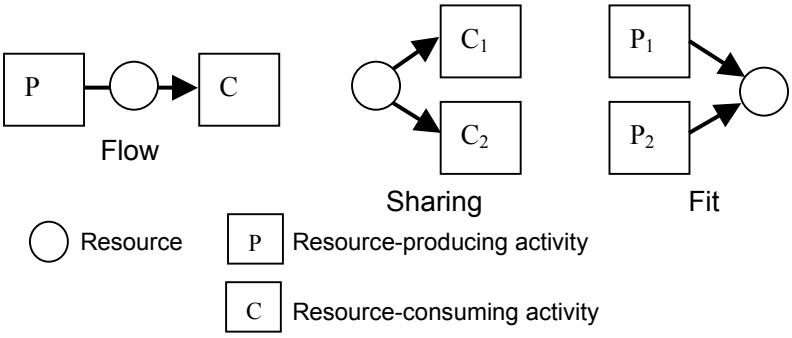


Fig.1: 3 basic types of dependencies

A ***flow*** dependency occurs when one activity produces a resource that is used by another activity. A ***sharing*** dependency occurs when multiple activities all use (or could use) the same resource. A ***fit*** dependency occurs when multiple activities jointly produce a single resource. Using these three basic types, any process can be decomposed into a structure of activities and dependencies.

In coordination theory, coordination is defined as "managing dependencies among activities." [Malone & Crowston 1994]. This raises an awareness of the importance of managing dependencies. The managers of a process often focus on how to manage activities. However, if a critical dependency of a process is not managed well, the process's efficiency and effectiveness become low even if all activities in the process perform well. Thus, how to manage dependencies in a process has an impact on the efficiency and effectiveness of the process.

Key: See the design process as a coordinated process

By looking at the design process as a coordinated process, we can clarify opportunities and risks in the process.

Opportunities are recognized as choices for how to coordinate the design process. Assessing different choices for coordinating dependencies allows for a systematic exploration of the process alternatives. For example, changing the types of dependencies among the component activities is one way of creating different processes made up of the same component activities. One can compare and contrast which dependency will be more manageable for ones organization. Having more manageable dependencies may allow for a more manageable overall process.

Risks are derived from potential coordination failure in a process. Managing a dependency between design activities sometimes fails; for example, if two design teams working on different parts of a product misunderstand the interface of the parts, the parts will not fit. If a failure of coordination occurs, design activities within the process must be redone or reworked; this increases the cost (and time) of executing the process. Knowing where the risks exist may allow for avoiding this extra cost and time.

The 3 steps

The method we propose consists of three steps:

1) finding applicable differentiation approaches,

2) finding applicable patterns of process coordination, and

3) evaluating total costs of the process alternatives.

Hereafter, we describe details of the steps.

1) Finding applicable differentiation approaches

In step 1, differentiation approaches are categorized as a taxonomy of product design processes. Like a class hierarchy in an object-oriented software system, the taxonomy consists of an “is-a” relation and a “has-a” relation of design activities. We show the root part of the taxonomy in Fig. 2 as a variant of UML class diagrams [Fowler & Scott 2000]. [♦]

At the root of the taxonomy is the most generic description of a differentiation approach. The generic description of a product design process for differentiated products has the component activities “design an interim

♦ In our variant of UML, a box represents an activity (class), not an object class and a link with black circles represents a dependency, not an association.

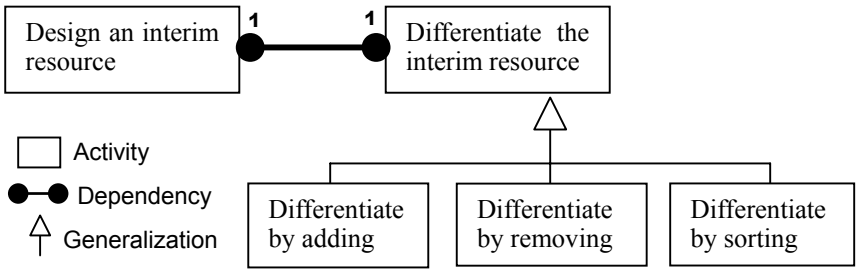


Fig.2: Taxonomy of differentiation approaches

resource" and "differentiate the interim resource", and a dependency between the activities. According to varied differentiation approaches, the component activity "differentiate the interim resource" is specialized as adding differentiating elements to the interim resource, removing differentiating elements from the interim resource, or sorting the interim resource.

In the "adding" approach, designers make differentiating elements that can attach to the interim resource. In the "removing" approach, designers identify removable elements/parts of the interim resource. In the "sorting" approach, designers specify a sorter of the interim resource. In Fig.3, we show relationships between a design process of each differentiation approach and a production process.

One example of the "adding" approach is a car design process. Designing a car chassis is the component activity "design an interim resource" and designing a car body is the component activity "differentiate by adding." The same interim resource may be used in many final products by having different bodies added to it. This approach has been adopted by not only assembled product industries but also in non-assembled product industries such as chemical products. As another example, the additional tracking service of the USPS is an "adding" approach of service differentiation.

One example of the "removing" approach is shown in the design of Intel's Pentium III and Celeron processors with Coppermine core [Schmid 2000]. At the circuit design, the Pentium III and Celeron are the same except the size of the 2nd cache. In the Celeron processors, there is the same size of 2nd cache as the Pentium III processors but half of the cache is "killed." The design of the entire processor is "design an interim resource". The design to remove half of the second cache is "differentiate by removing". As another example, cutting glass is a "removing" approach on a non-assembled product.

One example of the "sorting" approach is frequency-based selection of CPUs. In this approach, the design of the CPU is the same, but the results are sorted based on actual frequencies. The sorting process is the "differentiate by sorting" activity. Another example is hand-made china. China has varied quality even if made from the same clay, by the same artisan. To maintain ones reputation, the designers break all china except excellent ones during the production; a kind of sorting. Sorting paintings for an exhibition and selecting which publication medium is appropriate for a particular article are other examples of the "sorting" approach.

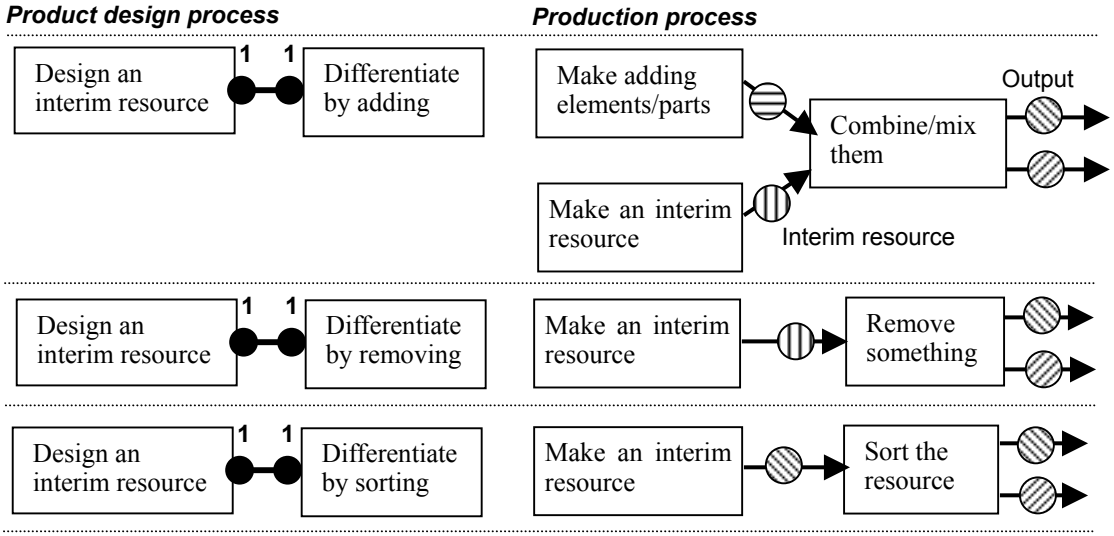


Fig.3: Product design process and production process

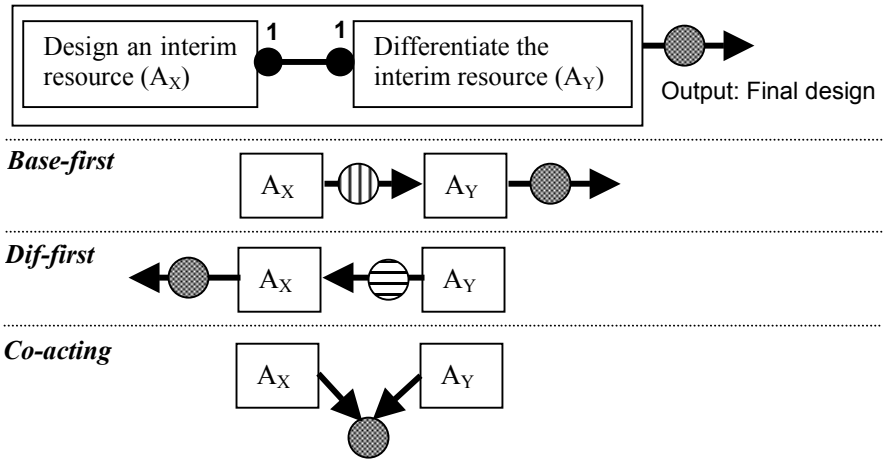


Fig. 4: Applicable patterns of process coordination

The type of interim resource limits applicable specializations. If interim resources in a production process are modifiable, one can choose any approach from among the three approaches. One can modify the interim resource by adding or deleting elements or not modify it and simply sort the results. If the interim resource is not modifiable, then one can only choose the "sorting" approach.

2) Finding applicable patterns of process coordination

In step 2, applicable patterns of process coordination, i.e., applicable dependencies among design activities are explored.

From a coordination perspective, three kinds of dependency patterns are possible between the component activity "design an interim resource" and the component activity "differentiate the interim resource." These are a flow from one activity to the other activity, the reverse of that flow, and a fit of the two activities (Fig. 4). Hereafter, we call the flow from the activity "design an interim resource" as the pattern "base-first," the flow from the activity "differentiate the interim resource" as the pattern "dif-first," and the fit as the pattern "co-acting."

The difference between the pattern "base-first" and the pattern "dif-first" is which activity uses the result of the other activity. For example, in the pattern "base-first", the component activity "differentiate the interim resource (A_Y)" can't finish before the other activity has finished because the activity A_Y consumes the design of the interim resource.[†]

[†] In the pattern "base-first", the activity A_Y can begin before the activity A_X has finished; in this case, the activities run concurrently. The same thing can happen in the pattern "dif-first."

The difference between the pattern "co-acting" and the other flow patterns is the activities' contribution to the final design. In the pattern "co-acting", both activities have direct contribution to the final design. In the flow patterns, the first activity in a flow has indirect contribution to the final design; in other words, if the second activity in the flow isn't able to use the result of the first activity in the final design, the whole process fails.

As written above, there are three possible patterns; however, applicable patterns are limited by the differentiation choice in step 1. For example, when the specialized activity "removing" is chosen, the pattern "base-first" is applicable but the pattern "dif-first" is not because the one can't design the removal of an element that has not yet been designed. In Fig. 5, we show the relation among differentiation approaches, interim resource types, and applicable coordination patterns.

Approach	***What types of interim resources are applicable for?***	***Which coordination patterns are applicable for?***
Adding	Modifiable	Base-first Dif-first Co-acting
Removing	Modifiable	Base-first Co-acting
Sorting	Modifiable Not-modifiable	Base-first Co-acting

Fig.5: Differentiation approaches, interim resource types, and applicable coordination patterns

One can choose different methods from among these possible applicable patterns of process coordination during the execution of the process. As an example, we will use a product manufacturer that we call "Firm Z" [personal communication 2001].

In Firm Z, before beginning design activities for a new machine, the leading developers discuss and decide the spatial allocation of functional modules in the machine and interface specifications among the modules. Then, the modular design activities are executed in parallel according to this decision. This managerial method is modeled as the pattern "co-acting" and managing fit dependency before activities. This is one of the acceptable choices for a modifiable interim resource where the

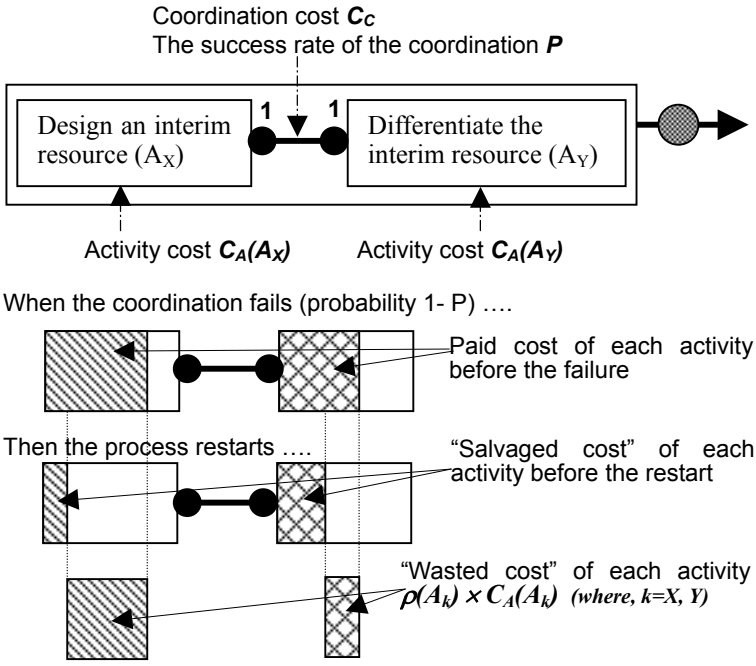


Fig.6: Parameters of E equation

differentiating elements are added.

However, the method doesn't always succeed because some problems are not recognized before the modules are connected; for example, a vibration of the machine may affect the quality of another part of the machine, but its impact is hard to estimate before a prototype of the machine is actually made. If the initial coordination fails, a section of Firm Z coordinates the recovery activities by using a different method. The coordinator picks one activity up and asks developers to redo the activity; if problems rise again, he or she repeats the coordination. This managerial method is modeled as the pattern "base-first" or "dif-first". Thus, in a design process of Firm Z's product, the managers of the process switch dependency management to one of the two other patterns for coordinating "Differentiate by adding".

The Firm Z case shows an example of exceptions and exception handling in a product design process [Klein 1999]. In this case, an exception is the coordination failure and its handling is the switching of the coordination pattern.

3) Evaluating total costs of the alternative processes

In step 3, the risks of the various process alternatives are evaluated by comparing the costs of the processes. If the coordination of a product design process fails, design activities within the process must be redone or reworked; thus, the coordination failure increases the overall cost of executing the process. We introduce the expected value of a process cost as an index; the expected value E is uniquely calculated by the coordination cost C_C; activities' costs $C_A(A_X)$, $C_A(A_Y)$; rates of redoing activities $\rho(A_X)$, $\rho(A_Y)$; and the success rate of the coordination P. Here, $0 \le \rho(A_X) \le 1$, $0 \le \rho(A_Y \le 1$, and $0 \le P \le 1$. The meanings of the parameters are explained in Fig.6.

(i) The simplest case: Using one method of dependency management

First, we show the simplest equation of the expected value E. In this case, process coordination and how to manage the dependency are not changed during the execution of the process. Also, for this initial research, we assume that these parameters are constant during the process to simplify the analysis (e.g. no learning between iterations of a process). Further research can be done where there is 'learning' or other causes of variability in the various parameters.

Using the parameters shown in Fig.6, the expected value E_{SP} is calculated as shown as below.

$$
\begin{aligned}
E_{SP} &= P \times (C_C + C_A(A_X) + C_A(A_Y)) \\
&\quad + (1 - P) \times P \times \{ 2\, C_C + (1 + \rho(A_X)) \times C_A(A_X) + (1 + \rho(A_Y)) \times C_A(A_Y) \} \\
&\quad + (1 - P)^2 \times P \times \{ 3\, C_C + (1 + 2\,\rho(A_X)) \times C_A(A_X) + (1 + 2\,\rho(A_Y)) \times C_A(A_Y) \} \\
&\quad + \ldots\ldots \\
&= P \times (C_C + C_A(A_X) + C_A(A_Y)) \\
&\quad + \textstyle\sum_i \{ (1 - P)^i \times P \times ((1 + i) \times C_C + (1 + i \times \rho(A_X)) \times C_A(A_X) + (1 + i \times \rho(A_Y)) \times C_A(A_Y)) \} \\
&\qquad \textit{[here, } 1 \le i \textit{]} \\
&= \mathbf{(1 / P) \times \{ C_C + (P + \rho(A_X) - P \times \rho(A_X)) \times C_A(A_X) + (P + \rho(A_Y) - P \times \rho(A_Y)) \times C_A(A_Y) \}}
\end{aligned}
$$

[see Appendix for how to transform the equation]

$$E_{SW} = P_1 \times (C_{C1} + C_A(A_X) + C_A(A_Y))$$
$$+ (1 - P_1) \times P_2 \times \{ C_{C1} + C_{C2} + (1 + \rho(A_X)_1) \times C_A(A_X) + (1 + \rho(A_Y)_1) \times C_A(A_Y) \}$$
$$+ (1 - P_1) \times (1 - P_2) \times P_2 \times \{ C_{C1} + 2C_{C2} + (1 + \rho(A_X)_1 + \rho(A_X)_2) \times C_A(A_X) + (1 + \rho(A_Y)_1 + \rho(A_Y)_2) \times C_A(A_Y) \}$$
$$+ (1 - P_1) \times (1 - P_2)^2 \times P_2 \times \{ C_{C1} + 3C_{C2} + (1 + \rho(A_X)_1 + 2\rho(A_X)_2) \times C_A(A_X) + (1 + \rho(A_Y)_1 + 2\rho(A_Y)_2) \times C_A(A_Y) \}$$
$$+ \ldots$$
$$= P_1 \times (C_C + C_A(A_X) + C_A(A_Y)) + (1 - P_1) \times P_2 \times \{ C_{C1} + C_{C2} + (1 + \rho(A_X)_1) \times C_A(A_X) + (1 + \rho(A_Y)_1) \times C_A(A_Y) \}$$
$$+ (1 - P_1) \times P_2 \times \sum_i \{ (1 - P_2)^i \times (C_{C1} + C_{C2} + i \times C_{C2} + (1 + \rho(A_X)_1 + i \times \rho(A_X)_2) \times C_A(A_X) + (1 + \rho(A_Y)_1 + i \times \rho(A_Y)_2) \times C_A(A_Y)) \}$$
[here, $1 \leq i$]
$$= \mathbf{(1 / P_2) \times \{ P_2 \times C_{C1} + (1 - P_1) \times C_{C2} + (P_2 + P_2(1 - P_1)(1 - P_2) \times \rho(A_X)_1 + (1 - P_1)(1 - P_2) \times \rho(A_X)_2) \times C_A(A_X)}$$
$$\mathbf{+ (P_2 + P_2(1 - P_1)(1 - P_2) \times \rho(A_Y)_1 + (1 - P_1)(1 - P_2) \times \rho(A_Y)_2) \times C_A(A_Y) \}}$$

(ii) An alternative: Switching methods of dependency management

Next, we consider a case like Firm Z. In such a situation, the expected value E is modified from the equation in case (i) above. We assume the following.

a) The switch in coordination occurs only once, after the first coordination failure. (Further research would allow for changes in coordination after each iteration.)
b) Coordination costs, rates of redoing activities, and success rates may be different in iterating the two methods.
c) Activities' costs are the same.
d) Above parameters are constant.

Under the assumption, the expected value E_{SW} is calculated as above. Here, the parameters that have subscript '1', e.g., P_1, are for the process pattern before the switching; the other parameters are for the pattern after the switching.

Note that the success rate P_2 is the conditional probability that occurs only if the first coordination failure has occurred. It is possible that applying a certain coordination pattern becomes easier after the other pattern fails; in other words, the success rate of a pattern may increase when it applies after the other pattern fails. In the case of Firm Z, if applying the pattern "base-first" becomes easier after the fit fails by making a tentative final design, the success rate of the pattern "base-first" is better in the second iteration than the value of the success rate in the initial iteration.

By using the expected value E as an index, one can evaluate different management styles in the same axis. As an example, we contrast typical serial engineering (typical SE), typical concurrent engineering (typical CE), and set-based concurrent engineering (set-based CE) like at Toyota [Sobek et al 1999]. Typical SE is a waterfall-style design process and only one activity is executed at a time during the process. Typical CE is also a flow-style design process; however, the latter activity in the flow order can overlap and run during the former activity's execution and the information from the latter activity will be used in the former activity. Set-based CE includes parallel, well-integrated activities. By using frequent communication between the activities, a process of set-based CE begins from various design alternatives and reaches a final design gradually.

In our taxonomy, the typical SE coordination pattern is "base-first" or "dif-first" and the latter activity in the process does not run until the former is finished. The rate of redoing the latter activity ($\rho(A_k)$, k=X or Y) is 0 and the coordination cost (Cc) is very low; however, it is also expected that the success rate of coordination (P) is low.

The typical CE coordination pattern is "base-first" or "dif-first" and that the latter activity in the process begins before the former is finished. The rate of redoing the latter activity is not 0 and the coordination cost is medium; it is expected that the success rate of coordination is middle or high.

The set-based CE is modeled as the coordination pattern "co-acting" and the coordination cost is very high. However, it is expected that the success rate of coordination is also very high.

By determining the other parameters (e.g., activities' costs) in the E equation, it is able to calculate which of the three management styles is superior according to the situation. As the example shows, the different kinds of process management can be evaluated by using the expected value. Allowing for all three approaches to design to be evaluated against each other is one of the differentiating items of our approach.

Related research

Eppinger et al have proposed an evaluation method based on techniques for signal flow graphs [Eppinger et al 1997].

In the method, a product design process is mapped to a signal flow graph: a node represents an activity and an edge represents a possible flow from one activity to another activity. Introducing an activity's lead time as a value of a node and a probability of redoing an activity as a value of edge, the expected value of the process lead time is calculated by using the total value of graph transmission between the start and the finish nodes of the signal flow graph.

Their method and our method are similar in using probabilities in a process and the expected value as an evaluation of the process. However, the modeling viewpoints of the probabilities are different.

In their method, a probability represents the rates of possible states that occur after the end of one activity; the state transition never happens if the activity has not finished. By using this kind of probability, it is hard to describe a parallel execution of activities. To describe that multiple activities run concurrently (not reciprocally) as in set-based CE or the first iteration of Firm Z's design process, requires a highly complicated signal flow graph. One technique to simplify the graph is that such activities are treated as a node; however, this changes the way of partitioning activities in their method.

In our method, serial execution, partial concurrent execution, and parallel execution of activities can be evaluated by using the same equation. One does not have to think about the state transition of activities to evaluate the process. One can focus simply on the cost of and the success rate of coordination between activities.

As another method of evaluating a process, Smith and Eppinger have proposed the work transformation matrix (WTM) [Smith & Eppinger 1997]. The WTM represents activities' lead-times and the relations among coupled activities within a product design process. The diagonal elements of the WTM represent an activity's lead-time. The off-diagonal elements of the WTM represent the rate of the redoing an activity that is triggered by the end of another activity; for example, if element $m_{12} = 0.1$, it means 10 % of activity 1 must be redone if activity 2 is finished. By using the WTM and techniques of matrix operations, one can calculate the evaluation value of the coupled activities as an accumulation of the lead-time.

In this method, an activity's result must be redone based on the results of the other coupled activities. As the model of a process, the relations among activities are deterministic. Thus, their method is not able to represent the Firm Z case since the success of the first attempt by using "co-acting" is not deterministic but probabilistic.

Implications of the expected value of total cost of a process

Typical examples of E's curve

Here, we contrast typical curves of expected values of the simplest case where the X axis is success rate of the coordination process and the Y axis is the expected value of the cost (the "P vs E" plane). In each graph in Fig.7, 4 processes are contrasted; they all are the same in proportion of "wasted cost" (ρ) but differ in coordination cost (C_C) and activities' cost ($C_A(A_X)$ and $C_A(A_Y)$). By using several graphs, we can contrast processes that have different ρ.

Analyzing these graphs, we claim that
1) coordination cost (C_C) drives the overall cost as the probability of success (P) drops, especially if C_C is much higher than $C_A(A_X)$ and $C_A(A_Y)$, and
2) the proportion of "wasted cost" (ρ) has an impact on expected value (E).

Thus, looking toward a "Productivity Frontier [Porter 1998]", one must achieve the following:

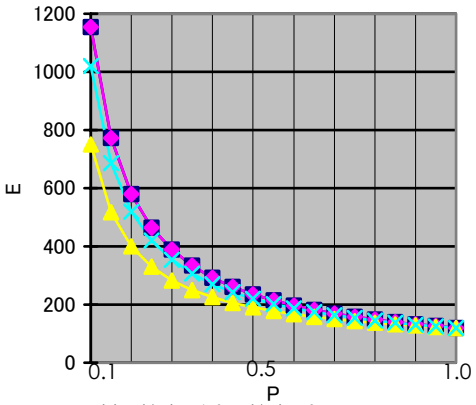
- Achieve high probability of satisfying usability (P) AND low coordination cost (C_C).
- Achieve high P AND low rate of "wasted cost" (ρ). Especially, reduce rework (ρ) of the expensive activity's cost ($C_A(A_k)$).
- If $C_A(A_k)$ and $\rho(A_k)$ are much higher than the respective values of the other activity, to reduce the risk of failure, find a process that performs A_k later than the other activity.
- Reduce C_C AND $C_A(A_k)$.

"Break-even point" for additional coordination cost

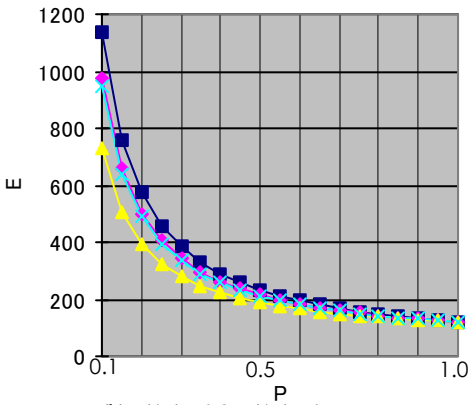
Assume that by adding some effort to coordination we can raise the probability of successful coordination. We show this by adding coordination cost α to the current coordination cost C_C in the simplest case. We can calculate a "break-even point" between the extra cost of coordination and the reduced "waste cost" due to an increased probability of success (P_{BE}) as below.

$$P_{BE} = \{ C_C + \alpha + \rho(A_X) * C_A(A_X) + \rho(A_X) * C_A(A_Y) \} / \{ E - C_A(A_X) + \rho(A_X) * C_A(A_X) - C_A(A_Y) + \rho(A_Y) * C_A(A_Y) \}$$

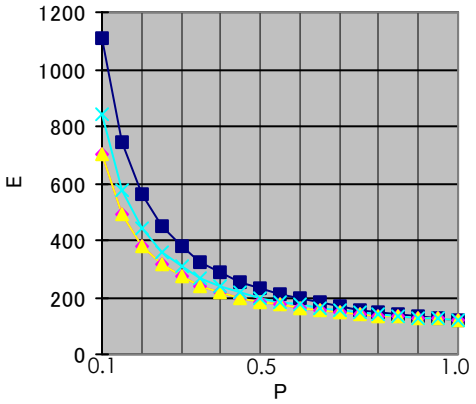
here, $E = (1/P) * \{ C_C + (P + \rho(A_X) - P * \rho(A_X)\} * C_A(A_X) + (P + \rho(A_Y) - P * \rho(A_Y)\} * C_A(A_Y) \}$



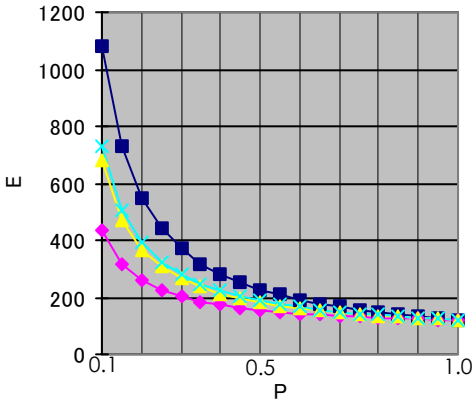
(a) $\rho(A_X) = 1.0$, $\rho(A_Y) = 0.5$



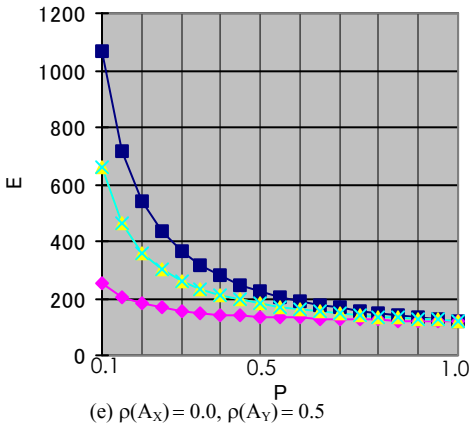
(b) $\rho(A_X) = 0.8$, $\rho(A_Y) = 0.5$



(c) $\rho(A_X) = 0.5$, $\rho(A_Y) = 0.5$



(d) $\rho(A_X) = 0.2$, $\rho(A_Y) = 0.5$



(e) $\rho(A_X) = 0.0$, $\rho(A_Y) = 0.5$

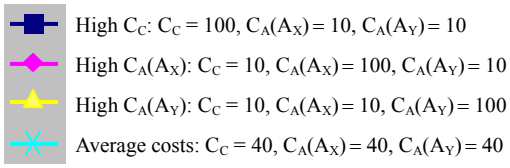


Fig.7: Curves of E_{SP}

If the probability is raised to P' by adding the cost α and $P_{BE} \leq P'$, the additional cost α is paid off. The corollary is also true. For a specified increase in probability of success, we can calculate the additional coordination cost that can be afforded. If we assume that one raises probability of satisfying usability from P to P', we have a "break-even point" of additional coordination cost (α_{BE}) as below.

$$\alpha_{BE} = E * P' - C_C - (P' + \rho(A_X) - P' * \rho(A_X)) * C_A(A_X) - (P' + \rho(A_Y) - P' * \rho(A_Y)) * C_A(A_Y)$$

here, $E = (1/P) * \{ C_C + (P + \rho(A_X) - P * \rho(A_X)\} * C_A(A_X) + (P + \rho(A_Y) - P * \rho(A_Y)\} * C_A(A_Y) \}$

If the additional coordination cost α is used to raise the probability and $\alpha \leq \alpha_{BE}$, the added cost α is paid off.

We illustrate an example of the relation between break-even points on the Cc-P plain in Fig. 8. In the example, $C_C = C_A(A_X) = C_A(A_Y) = 40$, $\rho(A_X) = 0.8$, $\rho(A_Y)_2 = 0.5$; as an initial P (Pi), we assume 0.7, 0.5, and 0.3. If Pi = 0.7 and the additional cost $\alpha = 10$, $P_{BE} = 0.776$. If Pi = 0.5 and the target probability P' = 0.6, $\alpha_{BE} = 18.4$.

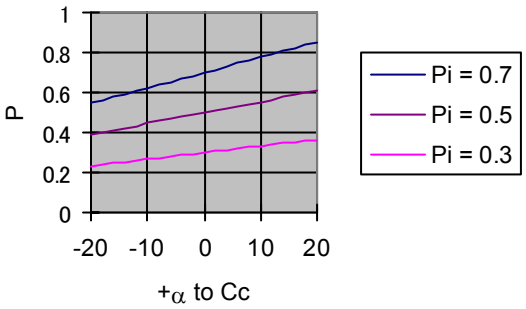


Fig 8: Break-even points

Implications of the E equation of switching dependency management

Here, we consider what conditions make switching dependency management between iterations affordable by using the expected values E_{SP} and E_{SW}. We assume the first iteration of the simplest case and the switching case are the same; then, for the parameters in E_{SP}, assume $P = P_1$, $C = C_{C1}$, $C_A(A_X) = C_A(A_X)_1$, $C_A(A_Y) = C_A(A_Y)_1$, $\rho(A_X) = \rho(A_X)_1$, and $\rho(A_Y) = \rho(A_Y)_1$. Under the condition, if $E_{SW} < E_{SP}$, the switching is paid off. Obviously, if $P_1 < P_2$, $C_{C2} \leq C_{C1}$, $\rho(A_X)_1 \leq \rho(A_X)_2$, and $\rho(A_Y)_1 \leq \rho(A_Y)_2$, the switching is paid off. Even if $P_1 > P_2$, paid-off cases exist. For example, assume $P_1 = 0.8$, $P_2 = 0.5$, $C_{C1} = 100$, $C_A(A_X) = C_A(A_Y) = 10$, $\rho(A_X)_1 = \rho(A_X)_2 = 1.0$, and $\rho(A_Y)_1 = \rho(A_Y)_2 = 0.5$. In this case, $E_{SP} = 148.75$, $E_{SW} = 0.4 * C_{C2} + 124.5$. Then, if $C_{C2} < 60.625$, the switching is paid off.

Tools for calculating E values and "break-even points"

To calculate the expected values and the "break-even points", we implemented calculators. An image of one of the calculator's window is shown in Fig.9. We think this kind of tools helps a manager to understand the impact of coordination failure on the total cost of a process.

Conclusion

Summary of the method

The 3 steps

We propose a 3-step method for exploring and evaluating alternatives of a product design process for differentiated products. In step 1, by using the taxonomy and identifying the type of an interim resource, one can systematically find applicable differentiation approaches. In step 2, by using patterns of process coordination and limitations from the approach chosen in step 1, one can find applicable process patterns as process alternatives. In step 3, by using the expected value of a process cost as an index, one can quantitatively evaluate process alternatives chosen in step 2. Thus, by using the method, one can systematically explore and evaluate design process alternatives.

Implications of the expected value of total cost

Based on the method, this paper shows implications for managing business processes. We show how coordination cost and the success rate of the coordination impact the overall cost of a process. It raises an awareness of the importance of managing dependencies between activities, which many process analyses don't focus on. We also show how to calculate the success rate associated with varying the coordination cost or how to calculate coordination cost associated with a desired success rate. These calculated values indicate "break-even points" for the cost of the process.

Implications for product design

A manager of a product design effort often focuses on the utility of the product design, but may not pay attention to

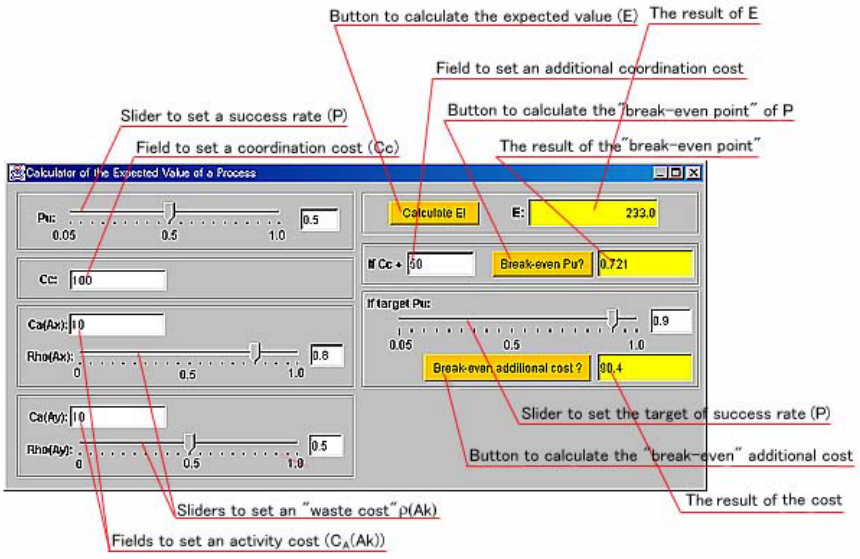


Fig.9: Calculator

the design process itself. Focusing on the dependencies among different components of the overall design may increase the probability of a successful effort, reducing the time to market of a new product. By making explicit the costs associated with having to rework portions of the design, a manager may have a better understanding of the overall cost of failed coordination and may alter the choice of the design process to minimize the total cost.

Future work

We plan to extend the method above in the following four directions.

- Extending the method to be applicable for a multi-value cost: For example, a time cost and a financial cost in a design process may be in trade-off situation; by introducing a matrix representation to the expected value, the method may handle such a situation.
- Extending the method to be applicable for a multi-activity process: In this paper, we only considered a process that has two activities; we would like to extend the method to apply to a process that has more than two activities.
- Extending the analysis to non-design processes: While we only explicitly consider coordination of a design process in this paper, the technique should be applicable to any process that coordinates a fit dependency.
- Extending the method to allow for variable costs. By introducing probability distribution on the parameters of the E equation, 'learning' or other causes of variability will be described.

Acknowledgement

We would like to thank Prof. Thomas Malone of the MIT Center for Coordination Science for discussing the method. We are grateful for helpful comments from Dr. Mark Klein of the MIT Center for Coordination Science and Mr. Masahito Watanabe of Department of Economics, Georgetown University. This research has been funded by Fuji Xerox Co., Ltd.

References

- Christensen, C.M. (1997). The innovator's dilemma, Harvard Business School Press.
- Clark, K.B. and T. Fujimoto (1991). Product development performance, Harvard Business School Press.
- Crowston, K. (1997). A coordination theory approach to organizational process design, Organization Science 8(2): 157-175.
- Davenport, T.H. (1993). Process innovation: Reengineering work through information technology, Harvard Business School Press.
- Drucker, P.F. (1999). Management challenges for the 21st century, Harper Business.
- Eppinger, S.D., M.V. Nukala, and D.E. Whitney (1997). Generalized models of design iteration using signal flow graphs, Research in Engineering Design, 9(2): 112-123.
- Fine, C.H. (1998). Clockspeed: Winning industry control in the age of temporary advantage, Perseus Books.
- Fowler, M. and K. Scott (2000). UML distilled: Applying the standard object modeling language, 2nd edition, Addison Wesley Longman.
- Fujimoto, T. and M. Yasumoto (ed.) (1999). Seikou-suru seihin kaihatsu: Sangyou-kan hikaku no shiten, Yuhikaku.
- Fujimoto, T. and A. Takeishi, Y. Aoshima (ed.) (2001). Business architecture: Strategic design of products, organizations, and processes, Yuhikaku.
- Grover, V., J.T.C. Teng, et al. (1995). Technological and organizational enablers of business process reengineering. in Business process change. V. Grover and W. J. Kettinger (ed.), Idea Group Publishing.
- Henderson, R.M. and K.B. Clark (1990). Architectural innovation: The reconfiguration of existing product technologies and the failure of established firms, Administrative Science Quarterly 35(1): 9-30.
- Iansiti, M. (1997). Technology integration: Making critical choices in a dynamic world, Harvard Business School Press.
- Kettinger, W.J. and J.T.C. Teng, S. Guha (1997). Business process change: A study of methodologies, techniques, and tools, MIS Quarterly 21(1): 55-80.
- Klein, M. (1999). Towards a systematic repository of knowledge about managing collaborative design conflicts, MIT Sloan School of Management Working Paper, 4096-99.
- Kotler, P. (1999). Marketing management, the millennium edition, Prentice-Hall.
- Lee, H.L. and C.S. Tang (1997). Modeling the costs and benefits of delayed product differentiation, Management Science 43(1): 40-53.
- Malone, T.W. and K. Crowston (1994). The interdisciplinary study of coordination, ACM Computing Surveys 26(1): 87-120.
- Malone, T.W., K. Crowston, et al. (1999). Tools for inventing organizations: Toward a handbook of organizational processes, Management Science 45(3): 425-443.
- Meyer, M.H. and A.P. Lehnerd (1997). The power of product platforms: Building value and cost leadership, The Free Press.
- Nadler D.A. and M.L. Tushman (1997). Competing by design: The power of organizational architecture, Oxford University Press.
- Nobeoka, K. and M.A. Cusumano (1995). Multiproject

strategy, design transfer, and project performance: A survey of automobile development projects in the US and Japan, IEEE Transactions on Engineering Management 42(4): 397-409.

- Nobeoka, K. (1996). Multiproject senryaku, Yuhikaku.
- Pentland, B.T., C.S. Osborne, et al. (1999). Useful descriptions of organizational processes: Collecting data for the Process Handbook, MIT Sloan School of Management Working Paper 4082-99.
- Porter, M.E. (1980) Competitive strategy, The Free Press.
- Porter, M.E. (1998) On competition, Harvard Business School Press.
- Robertson, D. and K.T. Ulrich (1998). Planning for product platforms, Sloan Management Review 39(4): 19-31.
- Schmid, P (2000). Intel Celeron performance guide, Tom's Hardware Guide (www. tomshardware.com), July 20, 2000.
- Smith, R.P. and S.D. Eppinger (1997). Identifying controlling feature of engineering design iteration, Management Science, 43(3): 276-93.
- Sobek, D.K.II, A.C. Ward and J.K. Liker (1999). Toyota's principles of set-based concurrent engineering, Sloan Management Review 40(2): 67-83.
- Sterman, J.D. (2000). Business dynamics, Irwin McGraw-Hill.
- Ulrich, K.T. (1995). The role of product architecture in the manufacturing firm, Research Policy 24(3): 419-440.
- Ulrich, K.T. and S.D. Eppinger (2000). Product design and development, 2nd ed., McGraw-Hill.
- von Hippel, E. (1990). Task partitioning: An innovation process variable, Research Policy 19(5): 407-418.
- Wheelwright, S.C. and K.B. Clark (1992). Creating project plans to focus product development, Harvard Business Review 70(2): 70-82.
- Personal communication with a developer of a machine manufacture (2001).

Appendix

$$E_{SP} = P \times (C_C + C_A(A_X) + C_A(A_Y))$$

$$+ \sum_i \{(1-P)^i \times P \times ((1+i) \times C_C + (1 + i \times \rho(A_X)) \times C_A(A_X) + (1 + i \times \rho(A_Y)) \times C_A(A_Y))\} \quad \textit{[here, } 1 \le i\textit{]}$$

$$= P \times (C_C + C_A(A_X) + C_A(A_Y))$$

$$+ P \times (C_C + C_A(A_X) + C_A(A_Y)) \times \sum_i \{(1-P)^i\}$$

$$+ P \times (C_C + \rho(A_X) \times C_A(A_X) + \rho(A_Y) \times C_A(A_Y)) \times \sum_i \{ i \times (1-P)^i \}$$

$$= P \times (C_C + C_A(A_X) + C_A(A_Y)) + P \times (C_C + C_A(A_X) + C_A(A_Y)) \times (1-P) / P$$

$$+ P \times (C_C + \rho(A_X) \times C_A(A_X) + \rho(A_Y) \times C_A(A_Y)) \times (1-P) / P^2$$

$$= C_C + C_A(A_X) + C_A(A_Y) + (C_C + \rho(A_X) \times C_A(A_X) + \rho(A_Y) \times C_A(A_Y)) \times (1-P) / P$$

$$= (1/P) \times \{ C_C + (P + \rho(A_X) - P \times \rho(A_X)\} \times C_A(A_X) + (P + \rho(A_Y) - P \times \rho(A_Y)) \times C_A(A_Y)\}$$