PRODUCT DEVELOPMENT AND CONCURRENT ENGINEERING

Christoph Loch
INSEAD, Fontainebleau, France.

Christian Terwiesch
University of California, San Diego, USA.

Description and Historical Perspective

Concurrent Engineering (CE) represents a key trend in product development over the last decade, which has changed academic and industrial approaches of looking at the product development process. It can be defined as "integrating the new product development process to allow participants making upstream decisions to consider downstream and external requirements". Central characteristics of a concurrent development process are activity overlapping, information transfer in small batches, and the use of cross-functional teams (Gerwin and Susman, 1996).

Many elements of CE have been recognized since the first half of this century (Smith, 1997). They include the formation of cross-functional teams, an important role of manufacturing process design in product development, and a focus on fast time-to-market. The term CE, however, was not adopted until the 1980s. Coming from manufacturing engineering, for example, Nevins and Whitney (1989) were among the earliest to demand concurrent design of the product and the manufacturing system. Others emphasize the cross-functional aspect by observing a " … need to transfer decision making authority from managers to teams" (Gerwin and Susman, 1996).

Concurrent engineering became popular in operations management literature with the work by Imai *et al.* (1985) and Takeuchi and Nonaka (1986). These studies contrasted two practices in product development: on the one hand, most Western organizations followed a more sequential process, similar to running a relay race with one specialist passing the baton to the next. This approach has also been referred to as "batch processing" (Blackburn 1991, Wheelwright and Clark 1992). In contrast, some high performing development organizations, most of them in Japan, followed a "rugby team" approach with a strong emphasis on cross-functional integration (a cross-functional team running in a staggered fashion). These early examples were motivated mainly from the camera and automobile industries (Takeuchi and Nonaka 1986).

In a landmark study on the automobile industry, Clark and Fujimoto (1991) extended this pioneering work and provided a detailed empirical analysis of product development processes, their impact on development performance and their differing use across regions. This study operationalized many important variables of CE, including measures for task overlapping and cross-functional communication. Clark and Fujimoto observed that companies with short time-to-market combined activity overlap with intensive information transfer, a practice they referred to as "integrated problem solving." These ideas were refined in further studies, e.g., Wheelwright and Clark (1992).

In this article, we propose a managerial framework of CE as posing four key managerial challenges, each of which has previously been addressed separately (see Figure 1).

- Defining development tasks based on the product architecture.
- Timing of activities with emphasis on task overlapping.
- Defining coordination and integration mechanisms among development tasks, which ensure adequate information exchange.

- Establishing support processes and an appropriate organizational context.

The framework with its four managerial challenges is summarized in Figure 1.

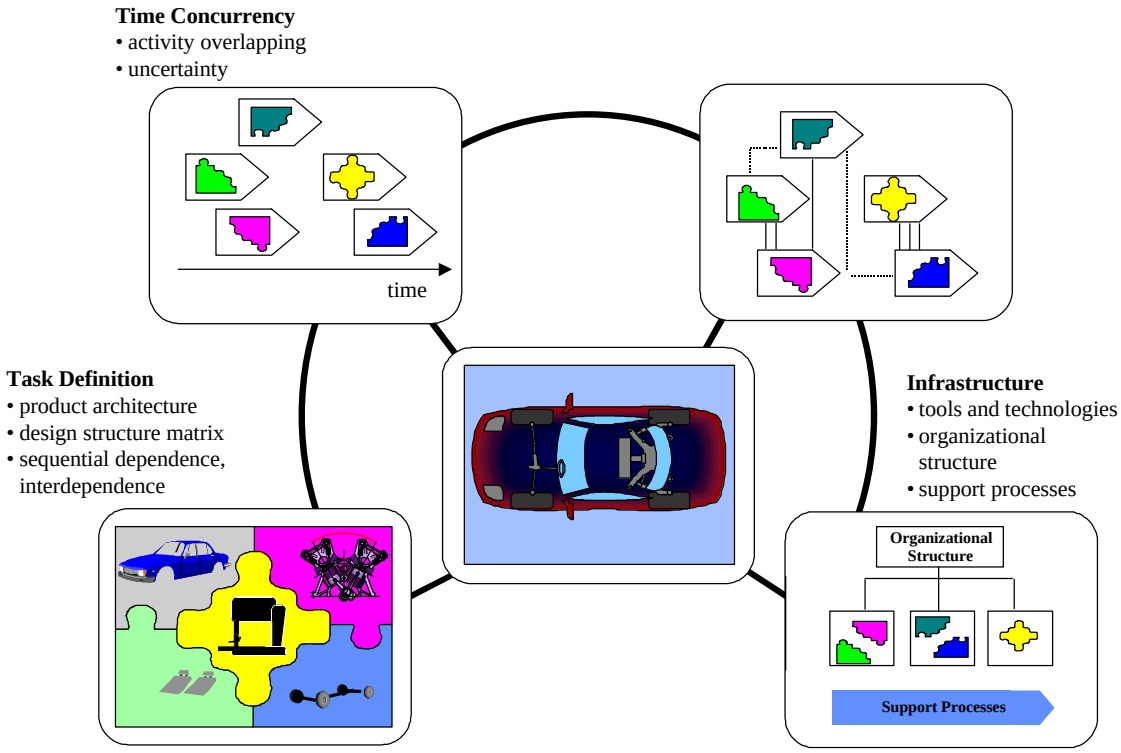

Figure 1: Operations Management Issues in Concurrent Engineering

Task Definition in Complex Design Problems

When faced with product development on a large scale, often hundreds of people work simultaneously on separate portions of the overall development effort, organized in many small teams (Eppinger *et al.*, 1994). The division of the complex overall design task into smaller, separately manageable portions depends on the couplings between these portions, which are largely determined by the product architecture.

The *product architecture* identifies the basic building blocks, their relationships to the functionality of the product ("what they do"), and the interfaces among them. A modular architecture is one with standardized interfaces and a one-to-one correspondence between

building blocks and functions: a module roughly corresponds to a function. The other extreme is an integrated architecture, with complex interfaces and a one-to-many correspondence between blocks and functions: functions are spread out over several building blocks, or several building blocks share one function (Ulrich 1995).

In terms of the final product, a modular architecture offers the possibility of component standardization, easily achievable product variety, and add-ons. An integrated architecture, on the other hand, may increase the compactness of the product, or allow better adaptation to specific customer needs, higher performance, or longer life through the tight integration of its building blocks. Moreover, the architecture heavily influences the development process, since modularity reduces interactions among product components, and thus complexity (Fitzsimmons *et al.*, 1991). In a modular architecture, product components can be developed in a decomposed and parallel manner (Ulrich 1995).

The goal of task definition is to minimize interactions among activities, or at least make them explicit so as to clearly understand the information transfer necessary. Two types of couplings exist, defined by information flows among activities. First, *sequential dependence* refers to a one-way information flow from an upstream to a downstream activity. In this case, one activity is the information supplier and the other the information receiver. Second, task interdependence refers to several tasks requiring information input from one another.

The process of decomposing the overall design problem can be supported by operations management methods. The most important support tool is the design structure matrix (DSM), which groups activities so as to minimize the interdependencies among the groups (Steward,

1981; Eppinger *et al.*, 1994; Smith and Eppinger, 1997). We now present a brief example of the application of this tool.

Based on the product architecture and the resulting component interfaces, the DSM captures interdependencies among the development tasks (corresponding to components), in the sense that tasks need input (physical or informational) from other tasks in order to be completed.

	A	**B**	**C**	**D**	
Task A	**A**				A-B: sequential
Task B	x	**B**			B-C: independent
Task C			**C**	x	C-D: coupled
Task D	x		x	**D**	A-D: sequential

Figure 2: Design Structure Matrix (Ulrich and Eppinger, 1995: 262)

As is shown in Figure 2, information-receiving tasks are listed along the columns, and information-supplying tasks along the rows. Crosses (X) mark information dependencies. Task B is *sequentially dependent* of task A, as the information flow goes only one way. Tasks B and C are *independent*. Tasks C and D require mutual information input and are *coupled* (interdependent). The matrix suggests a plan for the order of the tasks: A, then B in parallel with C and D, the latter two being performed in a closely coordinated way.

Significant Analytical Models. Eppinger *et al.* (1994) extend the DSM concept to include task completion times (marked on the diagonal) and strength levels of dependencies. By grouping the tasks with the strongest couplings together, the DSM can thus also be used to suggest design team formation, since a team is best able to perform coupled tasks in a coordinated manner. The application of the method is demonstrated in a semiconductor design project.

Smith and Eppinger (1997) extend this framework to a *Work Transformation Matrix* (WTM) tool for planning project execution. If one assumes that all activities are performed in parallel, with rework arising stochastically when a task receives information input, then the eigenvalues of the WTM can be interpreted as the *convergence rate* of the project, analogously to a Markov chain. Thus, the completion times of the project can be estimated, and problematic iteration loops among closely coupled tasks can be identified in advance. The method is demonstrated on the example of brake system development in the automobile industry.

Time Concurrence and Activity Overlapping

Task overlapping originated in the automotive (notably, Toyota and Honda), camera (e.g., Canon), and aerospace industries (Takeuchi and Nonaka 1986). It has also been adopted in electronics (Krishnan *et al.* 1997) and the software industry, and recently even in making movies (*New York Times*, May 5, 1997: D1).

Once development tasks have been defined in the overall design problem, their execution over time must be planned. While it is natural to execute interdependent tasks in parallel, it is easiest to perform sequentially dependent tasks in their logical sequence. This results, however, in a lengthy sequential process similar to a relay race. To reduce the time needed, CE attempts to at least partially overlap the sequentially dependent activities.

In the context of classical project planning, overlapping proposes, in effect, to shorten the critical path of a project by "softening" precedence relationships and conducting sequential activities in parallel. This offers a fundamental time advantage, but also has drawbacks. In a fully sequential process, downstream starts with finalized information from upstream, whereas in an overlapping process, it has to rely on preliminary information. This approach can be risky

if the outcome of the upstream activity is too uncertain to be accurately predicted. Under these conditions, overlapping activities creates uncertainty for the downstream activity, which would not exist in a sequential process. Thus, a trade-off arises between time gains from parallel execution and rework caused by uncertainty in the project. An optimal balance between parallelity and rework has been derived via analytical models and confirmed by several empirical studies showing that concurrence benefits decrease with increasing project uncertainty.

Significant Analytical Models. Several modeling efforts have been put forward to address possible trade-offs in applying CE. Ha and Porteus (1995) investigate a situation in which two development tasks are inherently coupled and must be carried out in parallel to avoid quality problems. They develop the "how frequent to meet" problem as a dynamic program. If one design activity proceeds without incorporating information from the other, design flaws and corresponding rework result. Thus, parallel development together with design reviews save time and rework. Similar to a quality inspection problem in production, these gains have to be traded off with the time spent on review meetings. The main question is how to coordinate, *i.e.*, how often to communicate.

Krishnan *et al.* (1997) developed a framework for concurrence in case of sequentially dependent activities. The authors model preliminary information passed from an upstream to a downstream activity in the form of an interval. A parameter, e.g., the depth of a car door handle, is initially known only up to an interval, which narrows over time as the design becomes final. In this framework, two concepts determine the overlap trade-off. "Evolution" is defined as the speed at which the interval converges to a final upstream solution. "Downstream sensitivity" is defined as the duration of a downstream iteration to incorporate upstream changes

associated with the narrowing of the interval. If upstream information is frozen before the interval has been reduced to a point value, a design quality loss occurs.

The authors formulate the problem as a mathematical program and show when overlapping (and thus preliminary information) should be used, and when upstream information should be frozen early. The authors solve an application example numerically and suggest that "generally, a fast evolution and low sensitivity situation is more favorable for overlapping. " The concept is illustrated for a door handle, a pager, and parts of a dashboard (Krishnan 1996, Krishnan *et al.* 1997).

Loch and Terwiesch (1998) conceptualize uncertainty resolution as the distribution of engineering changes (ECs) over the course of the project: the more uncertain the upstream activity, the more ECs are likely to arise. ECs have the universal characteristic that they become more difficult to implement the later they occur. The authors investigate the trade-off between gaining time from overlapping and the downstream rework caused by implementing ECs. Sensitivity analysis on the optimal overlap level shows that gains from overlapping activities are larger if ECs can be avoided, if dependence among activities can be reduced, and if uncertainty (the rate of ECs) can be reduced early in the process.

Implementation. The above-cited models suggest that overlap in product development is not equally applicable in all situations. This is supported by several empirical studies. In their study of the world computer industries, Eisenhardt and Tabrizi (1995) identify substantial differences across different market segments. For the stable and mature segments of mainframes and microcomputers, the authors find that overlapping development activities significantly reduces time-to-market. However, in rapidly changing markets such as printers

and personal computers, overlapping is no longer found to be a significant accelerator. Eisenhardt and Tabrizi argue that compressing the development process through activity overlaps only yields a time reduction if the market environment is stable and predictable.

Terwiesch and Loch (1996) find that project uncertainty in general (caused by the market or the technology) reduces the benefit from overlapping. Their study is based on data from 140 completed development projects across several global electronics industries. Although overlap helps to reduce project completion time overall, it is less helpful in projects with late uncertainty reduction than it is in projects with early uncertainty reduction. Intensive testing and frequent design iterations emerge as effective measures to reduce completion times in projects with late uncertainty reduction.

The effects of *uncertainty* in CE are compounded by product *complexity*. Complexity of a product refers to the number of elements in the system and the level of interactions among these elements (Fitzsimmons *et al*. 1991). The higher the complexity, the more interactions among subsystems exist in the design structure matrix, and the more components are affected by uncertainty, manifested in ECs. This makes intensive coordination among the parallel tasks even more important, which is discussed next.

Information Concurrence: Coordination and Integration

When the subtasks of the overall product design problem are executed, information exchange is required to keep the tasks coordinated. In simple cases, the information exchange may occur prior to the start of the downstream activity. Such *ex-ante* coordination can be achieved by, for example, design rules or preferred parts lists, which are often used in the context of design-for-manufacture. These methods help to anticipate problems in the manufacturing process as early

as possible, but they are not sufficient when the design problem is complex (Wheelwright and Clark 1992). In this case, ongoing coordination is required during task execution. This is consistent with Clark and Fujimoto's (1991) finding that successful task overlapping is supported by intensive information exchange.

Coordination Frequency. Coupled activities are almost always performed in parallel (see the earlier discussion of the DSM), and the key question is how intensively they should be coordinated. Ha and Porteus (1995) offer a model where this coordination is analyzed in terms of the frequency of communication between the two activity teams. The more significant the coupling, i.e., the more quality problems exist and the more severe the rework impact per quality problem, the more frequently the task teams should communicate.

Loch and Terwiesch (1998) show that the same principle holds in the case of sequentially dependent tasks: the more significant the number of changes from upstream and the dependency of the downstream task, the more intensive communication is required. High communication capabilities are thus an important enabler of task overlapping. If shared problem solving activities are performed before the actual design tasks begin, uncertainty levels during development may be significantly reduced in routine development projects. In such cases, the degree of overlapping may be very high.

These analytical results are consistent with empirical evidence: Morelli et al. (1995) examined communication patterns during the development of electrical computer board connectors, and they found that most communication took place among activities that were heavily interdependent. In a more general context, the relationship among dependencies, uncertainty,

and communication has been extensively studied in the organizational sciences (e.g., Thomson 1967).

Coordination Format. In addition to the frequency of communication, an important choice is the format of information exchanged between concurrent activities. Ward *et al.* (1995) report that Toyota in its product development process pursues a seemingly excessive number of parallel prototypes and "communicates ambiguously." They conclude that Toyota uses "set-based" concurrence, where whole sets of possible design solutions are explored in parallel, and over time the set of solutions considered is narrowed down (as a rule, not broadened). For example, key body dimensions are not fixed until late during development. However, when a fixed solution is reached, it no longer changes unless absolutely necessary. Ward *et al.* contrast the set-based approach to "iterative" or "point-to-point" design, where design teams fix solutions for their respective components and then coordinate among teams by iterating.

One cannot, however, conclude that a set-based approach is always superior. Terwiesch and Loch (1997) examined climate control system development at a different automobile manufacturer and encountered both approaches, set-based and iterative, the choice depending on the nature of the component developed. The trade-offs involved in the choice of approach are summarized in Figure 3, where the two communication approaches are classified according to the nature of the information exchanged. Information precision refers to whether it is communicated as a range or a precise value, while stability refers to whether the information tends to remain stable afterwards or whether it is likely to be changed again.

If the cost of making the wrong choice, or in other words, the value of flexibility in the design is very high, and if the number of candidate solutions that must be pursued is relatively small (and

thus less expensive), then the set-based approach offers advantages. On the other hand, if the development of a component is very flexible, i.e., engineering changes are very easy to incorporate, then the iterative approach is preferable.

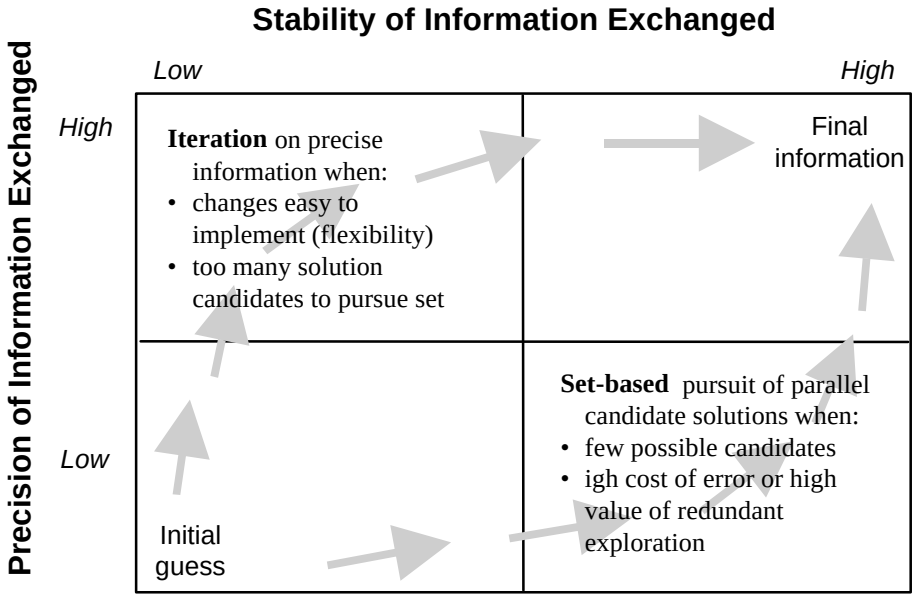

Figure 3: Set-based and Iterative Approach to Communication

Thus, the choice between set-based and iterative concurrence depends on the nature of the product component developed, and often a combination is used. The trade-offs involved, and thus the choice, are influenced by the technological and process capabilities of the organization. This is further examined in the next Section on "Supporting Infrastructure and Implementation."

Supporting Infrastructure and Implementation

The previous three aspects of CE (task definition, time and information concurrence) must be viewed within the context of a supporting infrastructure (Fitzsimmons *et al.* 1991). Three infrastructure components are discussed below: technologies, project organization, and support processes.

Technology and Tools. New problem solving technologies can play an important role not only in diminishing the cost of iteration, but also in anticipating and eliminating project uncertainty in the first place. Examples of such new technologies are CAD, simulation tools, or rapid prototyping. These technologies have the potential of fundamentally changing the trade-offs involved in time and information concurrence. Thomke (1996) studies integrated circuit design and distinguishes between flexible design technologies, which allow incorporation of design changes quickly and cheaply, and inflexible design technologies. He finds that projects using flexible design technologies are more efficient than projects using inflexible technologies. The difference stems from the lower cost of direct iterations (design changes), but also from the reduced need for costly resource investments to reduce the risk of design changes.

Thomke's (1996) study offers evidence of new technologies shifting the information exchange trade-off away from the necessity of set-based toward iterative concurrence (see Figure 3). In particular, powerful CAD systems allow the frequent information sharing and coordination necessary for activity overlapping in complex projects (a widely quoted example is the Boeing 777 jet, which was developed with heavy CAD usage).

Communication technologies also have the potential of influencing CE practices (e.g., Allen 1986). E-mail, portable phones, and shared databases all have the effect of making communication more time-efficient and independent of personal meetings between busy parties. However, even without considering implementation problems of forever incompatible data formats and protocols, communication technologies do not make communication instantaneous and painless. Whether it occurs orally in a meeting or through a database, communication among engineers involves mental set-ups and effort, inevitably leading to batching and delays.

Project Organization. Often, the structure of the project organization mirrors the logical structure of the design problem: the architecture determines communication patterns and physical proximity. At the same time, organizational capabilities determine which architectures can be developed (Gulati and Eppinger 1996, Allen 1986).

Commonly, four main types of project organization are distinguished (e.g., Wheelwright and Clark 1992, Chapter 8): in a *functional structure*, people are grouped by discipline, and projects are coordinated via *a priori* agreed-upon specifications, complemented by occasional meetings. In a *lightweight structure*, each function has a representation in a coordinating committee, led by a project coordinator. A *heavy weight structure* makes the coordinator a real project manager, with control over budgets and direct supervision of the functional representatives. Finally, the *autonomous structure* pulls the whole project outside of the regular organizational structure, and the project manager becomes like a line manager, who controls all resources and evaluates all employees.

The strength of the autonomous structure is focus and speed. However, it often suffers from integration problems with the rest of the organization, which may result in rejected output and disrupted career paths, as well as lost commonality (of parts and designs) across different products. The functional structure, on the other hand, is weak in its ability to coordinate complex projects with tight time-to-market goals, because communication and coordination is difficult across functional boundaries. However, it supports deep expertise in specialty disciplines, which is important in research projects with high technical uncertainty.

The project organization also influences behavioral aspects of the parties involved. Clark and Fujimoto (1991: 212f.) point out the importance of *attitudes* in CE, which requires a willingness

to share information and to cooperate. Hauptman and Hirji (1996) confirm this observation in a multi-industry study of development projects using CE. They find that CE must build on integration in the project organization, i.e., on overcoming the differences resulting from functional specialization: the parties involved must understand and trust one another sufficiently to be willing to share information. Once two-way communication happens, it tends to improve trust and employee satisfaction. For example, misunderstandings may arise from two engineering groups speaking different technical languages and having different perceptions of the same problem (e.g., Terwiesch and Loch, 1998).

In addition, Hauptman and Hirji find that CE requires a tolerance for releasing and using preliminary information, information that is imprecise and/or unreliable. Workers tend to minimize the risk of having to re-do tasks or of being proven wrong later by suppressing information, procrastinating, *etc*. Active management is necessary to overcome such risk-avoiding behavior.

Support Processes. CE requires a number of effective support processes, of which we briefly discuss three: interfaces with suppliers, engineering change orders, and interfaces among multiple development projects.

Clark and Fujimoto (1991) observed that close *integration with suppliers* in product development is necessary for successfully applying CE. This was further studied by Liker et al. (1995), who identified important success patterns. *1*. Simplify the coordination problem by outsourcing self-contained chunks. *2*. Communicate clearly and often (e.g., by having resident engineers, a practice now widely adopted). *3*. Use a simple, stable development process that is well understood by both sides. *4*. For suppliers participating in CE: develop the capability of

solving technical problems within their components without external help ("full service capability").

A general principle is that the investments and open information sharing required for involving suppliers in CE only work in long-term relationships with mutual commitment that prevents abuse and cheating. Such mutual commitment can be supported by investments in common infrastructure and systems (such as CAD or order management) and by establishing a mutually agreed-upon and understood development process, which should describe who is responsible for which deliverables, what information must be exchanged, and what milestones must be met.

Engineering change orders are an almost unavoidable consequence of CE, as tasks start based on preliminary information leading to mistakes. ECs arise also during component integration, or as a design is fine-tuned to improve market success. They often consume significant resources and require, therefore, an effective management process. Five principles can help to make the EC management process effective (Terwiesch and Loch, 1998):

- Give clear responsibilities and avoid unnecessary handoffs in the process.
- Manage capacity to reduce congestion. Technical personnel often have multiple responsibilities, leading to congestion and wait. For example, capacity can be made flexible and workloads can be shifted dynamically.
- Set-ups occur every time an engineer switches from one activity to a different one, for example, because files need to be loaded, or simply because he/she needs to become familiar with the problem again. To avoid set-ups, engineers (just like manufacturing people) typically batch their work, which can lead to substantial delays. Reducing set-ups can help to reduce batches, for example, by better IT systems, or by assigning one person responsibility for one EC all the way through.

- A complex product with a highly integrated architecture makes EC management more difficult, as ECs "snowball" to affect other product components. In this situation, it is important to educate all personnel involved about the key interactions among components (using, for example, the DSM as a tool) and to give strong incentives for fast problem communication and resolution. Fast problem resolution avoids incompatible changes being performed in parallel on multiple components. Incentives need to be given by explicit estimates of the value of time, in order to enable engineers of making trade-offs between optimization and time savings.

Interfaces among multiple development projects arise in all organizations of sufficient size. In many cases, much of the engineering work is performed in functional organizations of specialists who serve several projects at the same time. On the one hand, projects compete for the same resources, leading to congestion and interfering with the intensive communication and coordination within a project that are required by the practice of CE (Adler *et al.,* 1995). This requires a clear prioritization of projects, not only once at the aggregate level when projects are initiated, but also during execution, when concrete resource conflicts arise. Clear rules are necessary (e.g., degree of lateness) to settle conflicts that may otherwise be decided politically.

On the other hand, pursuing multiple projects in parallel supports expertise building of the people and integration and commonality across projects. Nobeoka and Cusumano (1995) find that overlapping development projects for consecutive product generations, rather than running them sequentially, facilitates the sharing of specifications and the transfer of knowledge.

Outlook: From Efficiency to Organizational Learning

The operations management literature on CE has largely emphasized efficiency of product development, seeking to reduce time-to-market or development costs, with product quality only mentioned on the side. Recently, the question of knowledge transfer is receiving more attention again, which was, interestingly enough, already mentioned in Takeuchi and Nonaka (1986).

Nobeoka and Cusumano (1995) find that parallel execution of whole projects is advantageous not for efficiency reasons, but to ensure effective knowledge transfer from one project to the next. Pisano (1997) reports that in pharmaceutical development, overlapping between product and manufacturing process development may be necessary entirely for learning reasons. If the scientific basis of a new product is clear (as, for example, in a purely chemical entity), the manufacturing process needs can be anticipated in development, and overlapping is not necessary. Pisano refers to this mode as "learning before doing." If, however, the basis of the molecule is not understood (as, for example, in a biochemical entity), then overlapping may be necessary to experiment with the manufacturing process during development, only to anticipate major problems that cannot otherwise be foreseen.[1] This mode is called "learning by doing." Overlapping allows the organization to learn, building its knowledge about the scientific basis of its products and processes and about the interactions between them.

As the state of the art in product development (albeit not yet the bulk of organizations) is turning its attention from efficiency to capability building and organizational learning, operations management needs to shift its focus from static efficiency to dynamic capability building (Jaikumar and Bohn, 1992). In light of the evidence that CE is closely connected with

[1] The model by Ha and Porteus (1995) described above could be used to analyze this situation. However, Ha and Porteus viewed this as a pure quality problem and did not see the implications for knowledge transfer and learning.

how organizations solve problems and spread knowledge, there is reason to believe that this paradigm shift will open up many opportunities for further research on CE for years to come.

Cross Reference

Product architecture, modular architecture, work transformation matrix, convergence rate of a project, downstream sensitivity, time to market, flexible design technologies, light weight project manager, heavy weight project manager, autonomous teams.

References

Adler, P. S., A. Mandelbaum, V. Nguyen, and E. Schwerer. "From Project to Process Management: An Empirically Based Framework for Analyzing Product Development Time," *Management Science*, 41, 1995, 458 - 484.

Allen, Th. J. (1986). "Organizational Structure, Information Technology, and R&D Productivity." *IEEE Transactions on Engineering Management*, EM 33, 212 - 217.

Blackburn, J.D. (1991). *Time Based Competition*, Homewood: Business One Irwin.

Clark, K. B. and T. Fujimoto (1991). *Product Development Performance: Strategy, Organization and Management in the World Auto Industry*. Harvard Business School Press, Cambridge.

Eisenhardt, K.M., and B. N. Tabrizi (1995). "Accelerating Adaptive Processes: Product Innovation in the Global Computer Industry." *Administrative Science Quarterly*, 40, 84-110.

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

Fitzsimmons, J. A., P. Kouvelis, and D. N. Mallik. "Design Strategy and Its Interface with Manufacturing and Marketing: a Conceptual Framework," *Journal of Operations Management*, 10, 398 – 415.

Gerwin, D., and G. Susman (1996). "Special Issue on Concurrent Engineering." *IEEE Transactions on Engineering Management*, 43, 118 - 123.

Gulati, R. K., and S. D. Eppinger (1996). "The Coupling of Product Architecture and Organizational Structure Decisions." MIT Working Paper 3906.

Ha, A. Y., and E. L. Porteus (1995). "Optimal Timing of Reviews in Concurrent Design for Manufacturability." *Management Science*, 41, 1431-1447.

Hauptman, O., and K. K. Hirji (1996). "The Influence of Process Concurrency on Project Outcomes in Product Development: an Empirical Study with Cross-Functional Teams." *IEEE Transactions in Engineering Management*, 43, 153 - 164.

Imai, K., I. Nonaka, and H. Takeuchi (1985). "Managing the New Product Development Process: How the Japanese Companies Learn and Unlearn." In: Clark, K. B., R. H. Hayes, and C. Lorenz (eds.). *The Uneasy Alliance*. Harvard Business School Press, Boston.

Jaikumar, R. and R. Bohn (1992). "A Dynamic Approach to Operations Management: An Alternative to Static Optimization." *International Journal of Production Economics*.

Krishnan, V., S. D. Eppinger, and D. E. Whitney (1997). "A Model-Based Framework to Overlap Product Development Activities." *Management Science*, 43, 437 - 451.

Liker, J. K., R. R. Kamath, S. N. Wasti, and M. Nagamachi (1995). "Integrating Suppliers into Fast-Cycle Product Development." In J. K. Liker, J, J. E. Ettlie, and J. C. Campbell (eds.). *Engineered in Japan: Japanese Technology Management Practices*. Oxford University Press, Oxford.

Loch, C. H., C. Terwiesch (1998). "Communication and Uncertainty in Concurrent Engineering," *Management Science* 44, August.

Morelli, M. D., S. D. Eppinger, and R. K. Gulati (1995). "Predicting Technical Communication in Product Development Organizations." *IEEE Transactions on Engineering Management*, 42, 215 - 222.

Nevins, J. L., and D. E. Whitney (1989). *Concurrent Design of Products and Processes*. McGraw Hill, New York.

Nobeoka, K., and M. A. Cusumano (1995). "Multiproject Strategy, Design Transfer, and Project Performance." *IEEE Transactions on Engineering Management*, 42, 397 - 409.

Pisano, G. (1997). *The Development Factory*. Harvard Business School Press, Boston.

Smith, R. (1997). "The Historical Roots of Concurrent Engineering Fundamentals." *IEEE Transactions on Engineering Management*, 44, 67 - 78.

Smith, R. P., and S. D. Eppinger (1997). "Identifying Controlling Features of Engineering Design Iteration." *Management Science*, 43, 276 - 293.

Steward, D. V. (1981). *Systems Analysis and Management: Structure, Strategy and Design*. Petrocelli Books, New York.

Takeuchi, H., and I. Nonaka (1986). "The New Product Development Game." *Harvard Business Review*, 64, 137-146.

Terwiesch, C., C. H. Loch, and M. Niederkofler (1996). "Managing Uncertainty in Concurrent Engineering," Proceedings of the 3rd EIASM Conference on Product Development, 693 - 706.

Terwiesch, C., and C. H. Loch (1997). "Management of Overlapping Development Activities: a Framework for Exchanging Preliminary Information." Proceedings of the 4th EIASM Conference on Product Development, 797 - 812.

Terwiesch, C., and C. H. Loch (1998). “Managing the Process of Engineering Change Orders: The Case of the Climate Control System in Automobile Development.” Forthcoming in the *Journal of Product Innovation Management*, 1998.

Thomke, S. (1996). “The Role of Flexibility in the Design of New Products: An Empirical Study.” Harvard Business School Working Paper 96-066.

Ulrich, K. (1995). “The Role of Product Architecture in the Manufacturing Firm.” *Research Policy*, 24, 419 - 440.

Ulrich, K., and S. D. Eppinger (1995). *Product Design and Development.* McGraw Hill, New York.

Ward, A., J. K. Liker, J. J. Cristiano, D. K. Sobek II (1995). “The Second Toyota Paradox: How Delaying Decisions Can Make Better Cars Faster.” *Sloan Management Review*, Spring, 43 - 61.

Wheelwright, S. C., and K. B. Clark (1992). *Revolutionizing Product Development*. The Free Press, New York.