

Exchanging Preliminary Information in Concurrent Engineering: Alternative Coordination Strategies

Christian Terwiesch • Christoph H. Loch • Arnoud De Meyer

*Operations and Information Management Department, The Wharton School, University of Pennsylvania,
Philadelphia, Pennsylvania 19104-6366*

INSEAD, Boulevard de Constance, 77 305 Fontainebleau Cedex, France

INSEAD, 1 Ayer Rajah Avenue, Singapore 138676

terwiesch@wharton.upenn.edu • christoph.loch@insead.edu • arnoud.de.meyer@insead.edu

Abstract

Successful application of concurrent development processes (concurrent engineering) requires tight coordination. To speed development, tasks often proceed in parallel by relying on preliminary information from other tasks, information that has not yet been finalized. This frequently causes substantial rework using as much as 50% of total engineering capacity. Previous studies have either described coordination as a complex social process, or have focused on the frequency, but not the content, of information exchanges. Through extensive fieldwork in a high-end German automotive manufacturer, we develop a framework of preliminary information that distinguishes information precision and information stability. Information precision refers to the accuracy of the information exchanged. Information stability defines the likelihood of changing a piece of information later in the process.

This definition of preliminary information allows us to develop a time-dependent model for managing interdependent tasks, producing two alternative strategies: iterative and set-based coordination. We discuss the trade-offs in choosing a coordination strategy and how they change over time. This allows an organization to match its problem-solving strategy with the interdependence it faces. Set-based coordination requires an absence of ambiguity, and should be emphasized if either starvation costs or the cost of pursuing multiple design alternatives in parallel are low. Iterative coordination should be emphasized if the downstream task faces ambiguity, or if starvation costs are high and iteration (rework) costs are low.

(Preliminary Information; Concurrent Engineering; Communication; Coordination; Problem-Solving Strategies; Product Development; Information Processing)

Introduction

Concurrent engineering, the practice of executing coupled development activities in parallel, has become the

common mode of product development as time-to-market has gained in importance over the last 15 years (Takeuchi and Nonaka 1986, Wheelwright and Clark 1992, Krishnan and Ulrich 2001). Given tight project schedules, many engineers cannot afford to wait until all required information input is available, and have to start “in the dark,” requiring close *coordination* with other interdependent activities.

Coordination among tightly coupled (interdependent) and parallel tasks forces parallel teams to share preliminary information about work in progress. Production tool orders have to be based on rough sketches of product designs, product concepts must be developed while uncertainty remains about the customer’s needs, and components must be specified while interacting systems are still under development.

This kind of coordination often proceeds in an informal, ad hoc manner. It is hard to tell if the right information is being shared at the right time, as in the place of factual data (“the total vehicle mass is 3,126 pounds”); there is only vague preliminary data (“at present, we expect the vehicle mass to be around 3,000 pounds”). As one automotive executive put it, “Designing a car is much like building a house: you cannot afford to suspend the kitchen planning until you have put up the walls. But, if you start the kitchen planning too early, using preliminary floor plans from the architect, you are likely to do it twice. [. . .] We need a new way of exchanging information between the architect and the kitchen planner. Currently, our kitchen planner’s idea of concurrent engineering is that they should receive the floor plans as they did in the past, just six months earlier. They don’t understand that the nature of the information has changed!”

This highlights the two fundamental coordination problems addressed in this article. First, the uncertainty¹ facing the kitchen planner with the floor plan may not necessarily arise from technologies or markets, but may be a consequence of the project manager's decision to overlap (execute in parallel) two sequentially dependent activities. But how can the architect (upstream) inform the kitchen planner (downstream) that the information is only preliminary in nature?

Second, we need to understand how the downstream party should use the preliminary information. If treated as final information, it is likely to lead to costly rework (if you plan the kitchen twice, you order a large appliance but end up not having the space to put it in). At the other extreme, not releasing any information until it has "converged" basically holds up the kitchen planning until the walls are up—an approach which avoids rework but sacrifices any time gains from parallel task execution.

Coordination strategies outlined in the existing organizational literature have primarily focused on finding appropriate organizational structures to respond to uncertainty and interdependencies (Brown and Eisenhardt 1995). However, as most of these models have been static in nature (Adler 1995), they cannot fully capture the concept of concurrency, which is by definition time dependent. Prior studies have also left the concept of preliminary information itself undefined, despite numerous recommendations to define it (e.g., Clark and Fujimoto 1991).

In the case of the kitchen planner, the existing literature would recommend forming a cross-functional team and engaging in frequent information exchanges. Although an important first step in dealing with uncertainty and interdependence, this fails to answer the fundamental question of what to communicate.

The key questions when coordinating concurrent tasks are not how often to exchange information, but rather what information to exchange at what moment in time, and how to react to it. Moreover, preliminary information exchange, which results from the combination of interdependence and concurrency, is a time-dependent construct which is gradually finalized as upstream advances in its problem solving (or, if the uncertainty is caused by external events, as these events unfold). A model addressing this issue must be dynamic in nature.

Theoretically our work extends a classical line of research on information exchange, uncertainty, and interdependence (e.g. Thompson 1967, Galbraith 1973) which has frequently been used as a theoretical foundation for the literature in the emerging field of new product development, such as Clark and Fujimoto (1991), Sobek et al. (1999), Krishnan et al. (1997), and Loch and Terwiesch

(1998). More recently, detailed empirical studies of new product development projects have not only *applied* existing organizational theories, but successfully *extended* them (e.g. Adler 1995, Staudenmeyer 1999, Eisenhardt and Tabrizi 1995).

In this article we present a qualitative study of an engineering project facing several situations of interdependence and concurrency, thus heavily dependent on preliminary information exchange. Using data collected from 10 engineering decisions traced on-site in a vehicle development project, we develop a dynamic model of coordination that hinges on the concept of preliminary information exchange. We study this exchange from three perspectives: that of the information-providing party, the information-receiving party, and of the system designer, as is reflected in our three research questions:

- How does the information provider transmit preliminary information, and how is it revised over time?
- How do downstream activities adjust to changes in the preliminary information received?
- What trade-offs are relevant for downstream when using preliminary information, and specifically, can preliminary information be traded off against budget, time, or system performance?

Based on these perspectives, we present a time-dependent model for coordinating interdependent tasks which in turn helps to define two alternative coordination strategies that we label *iterative* and *set based*. Second, we address the trade-offs faced by a development team in choosing a coordination strategy and how they change over time. This allows an organization to match its problem-solving strategy with the interdependence it faces. Relying on preliminary information too early can lead to rework in the form of costly iterations. Conversely, waiting for information to reach a desired level of certainty foregoes the time gains that come with parallel problem solving.

We start by clarifying the theoretical problem of coordinating concurrent interdependent tasks and how it relates to existing literature in organizational theory and new product development. Secondly, we present our research methodology and a detailed description of the engineering decisions requiring coordination. Building on on-site observations, the three perspectives of preliminary information are presented. From this we derive the two coordination strategies and identify the managerial trade-offs involved in choosing between them.

Theoretical Problem and Literature Background

Consider a very simple example of a development process where two activities are overlapped to reduce de-

velopment leadtime. The overlap allows the information-absorbing downstream operation (e.g., stamping die development) to start before the information-supplying upstream activity (e.g., product design) is completed, thereby potentially reducing the overall cycle time due to concurrency benefits.²

In a fully sequential process (Figure 1, left), no information is released to downstream until upstream has gained full knowledge of its task. When downstream finally starts, it can rely on finalized information from upstream. This is symbolized by a formal release milestone in the process (the diamond shape in Figure 1) corresponding to the classical stage-gate model.

Although the overlap creates a direct time gain by downstream starting early, it is not without drawbacks.³ Executing two activities concurrently (Figure 1, right) forces downstream to sacrifice quality of information and use preliminary information. If there is no concurrency, information can be exchanged in its final form. If there is no dependence, there is no need to exchange information. Thus preliminary information is the direct consequence of the interaction between task concurrency and dependence.

Such preliminary information tends to be based on a low-to-medium level of upstream knowledge, symbolized by lighter shading in Figure 1. The earlier downstream starts, the higher the risk of future changes, especially if the outcome of the upstream activity is hard or impossible to predict. In this case, overlapping activities risk creating additional engineering effort in the form of rework (Stoy 1996). Rework can consume up to 50% of the engineering capacity and up to one-third of the development budget (Clark and Fujimoto 1991, Soderberg 1989).

The well-known NPD process models (such as stage-gate, waterfall, or evolutionary models known from engineering and software development, see, e.g., Cooper 1993 or Boehm 1981) show how uncertainty is resolved over time but do not address how to coordinate parallel

activities. Within product development, Eastman (1980) was the first to discuss the benefits and dangers of committing engineering resources to nonfinal specifications. This work was further refined by Clark and Fujimoto's (1991) studies in the automotive industry, specifically in the context of stamping die development. Taking an information-processing perspective,⁴ the authors identified intensive communication as a key driver of development performance. Instead of batching the information created by one activity and handing it on to the next, it was found to be more effective to release preliminary information early and let downstream use it to start in parallel.

Although this line of research has substantially improved current understanding of development processes, looking at communication frequency and organizational structures alone cannot fully resolve the managerial challenges of dealing with preliminary information. Indeed, most companies have implemented many aspects of cross-functional integration over the last decade. In contrast with the large volume of literature on communication frequency, little work has been done on the *format* and *timing* of the information exchange. This is consistent with Scrivener et al. (2000), who emphasize the importance of what is communicated, as opposed to the media used or the frequency of the exchange. Unlike Scrivener et al., our focus is on need for information exchange, not on the mechanisms and media used to exchange it. Needs for information content are media independent (Scrivener et al. 2000, p. 349).

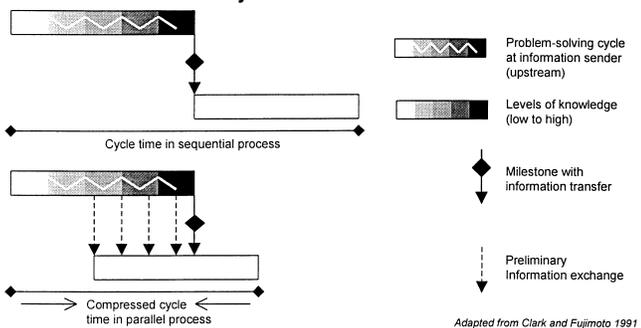
Allen (1977) observed that communication changes over the course of a project, both literature research and personal advice, decrease over time, although the latter surges again toward the end due to interface problems. Since this study, the dynamic nature of the interdependence resulting from task concurrency has been neglected with the exception of Adler (1995), who points out that uncertainty and interdependence can dynamically change in the course of a project. We extend Adler's work by looking at situations of interdependence and concurrency, adding a further level of complexity.

Our research seeks an operational definition of preliminary information, needed not just from an academic standpoint, but also, as the initial kitchen example illustrates, for effective information exchange in practice.⁵ Secondly, it explores how downstream reacts to preliminary information. This implies the presence of costs and trade-offs, which is the third area explored.

Research Methodology

The automotive industry is a natural candidate for research on the coordination of concurrent activities: Car

Figure 1 Coordinating Parallel Activities Requires the Use of Preliminary Information



development projects combine novelty with complexity (and thus task interdependence), and use task concurrency widely, thereby creating the coordination problem that interests us. Moreover, previous studies of concurrent engineering in this industry allow results to be compared (e.g., Clark and Fujimoto 1991, Cusumano and Nobeoka 1998). Task concurrency is, of course, used in other industries, such as electronics (Terwiesch and Loch 1999b), airplane design (Sabbagh 1996), and film making (Trip 1997).

Our effort to extend previous research on preliminary information⁶ started with a visit to our host company, where we presented results from our own modeling (Loch and Terwiesch 1998) and survey-based (Terwiesch and Loch 1999b) research. While the audience did not disagree with any of the arguments presented, including the need for greater concurrency and for frequent information exchange between parties working concurrently, they cited other problems, including the introductory quotation. At a follow-up meeting, we agreed to explore preliminary information further and defined specific research objectives (similar to those presented in the previous section).⁷

Research Site

We decided to focus on the climate control system (CCS) of a new vehicle under development. We narrowed our focus to one subsystem rather than the overall vehicle, as field-based research (see data collection methods below) required an in-depth observation of how engineering decisions evolved over time and how engineers exchange preliminary information. By examining such a “microcosm” we could interview all the engineers involved with the actual CCS design including its interfaces (about 40 individuals).

A detailed system overview of the CCS is provided in Figure 2. The CCS contains all components and development activities related to the passenger’s climate environment, including air ventilation, air cleaning, warm up, and cool down. It was chosen over other subsystems because of the strong need for coordination and information exchange. Referring to the air intake of the car (part of the CCS), one interviewee explained: “Here at the air-intake you find all the problems we have in the development of new vehicles: coordination with other components (e.g., fire-wall, engine) and information release to tooling.” Together with the dashboard, the CCS is the subsystem with the most interfaces to other activities.

Selected Cases

The unit of analysis in our study is the exchange between an information-supplying upstream activity and an

information-receiving downstream activity. Given our research focus, we are only interested in cases where downstream starts its work at a point before upstream has finalized its problem solving. With the help of the host company, we identified 10 such cases within the scope of the CCS, each relating to different components.

For each case, we focused on one piece of information that fulfilled two criteria. First, the information had to move, over the course of our observation, from an initial estimate to a finalized design decision. Second, it had to relate to an important interdependence, thereby causing substantial rework downstream if not transmitted to the receiver. We then identified the sender (upstream) who had to forward this information to a receiver (downstream) prior to completing its problem solving. Thus, the communication dyad faced a situation similar to that described in Figure 1.

Data Collection

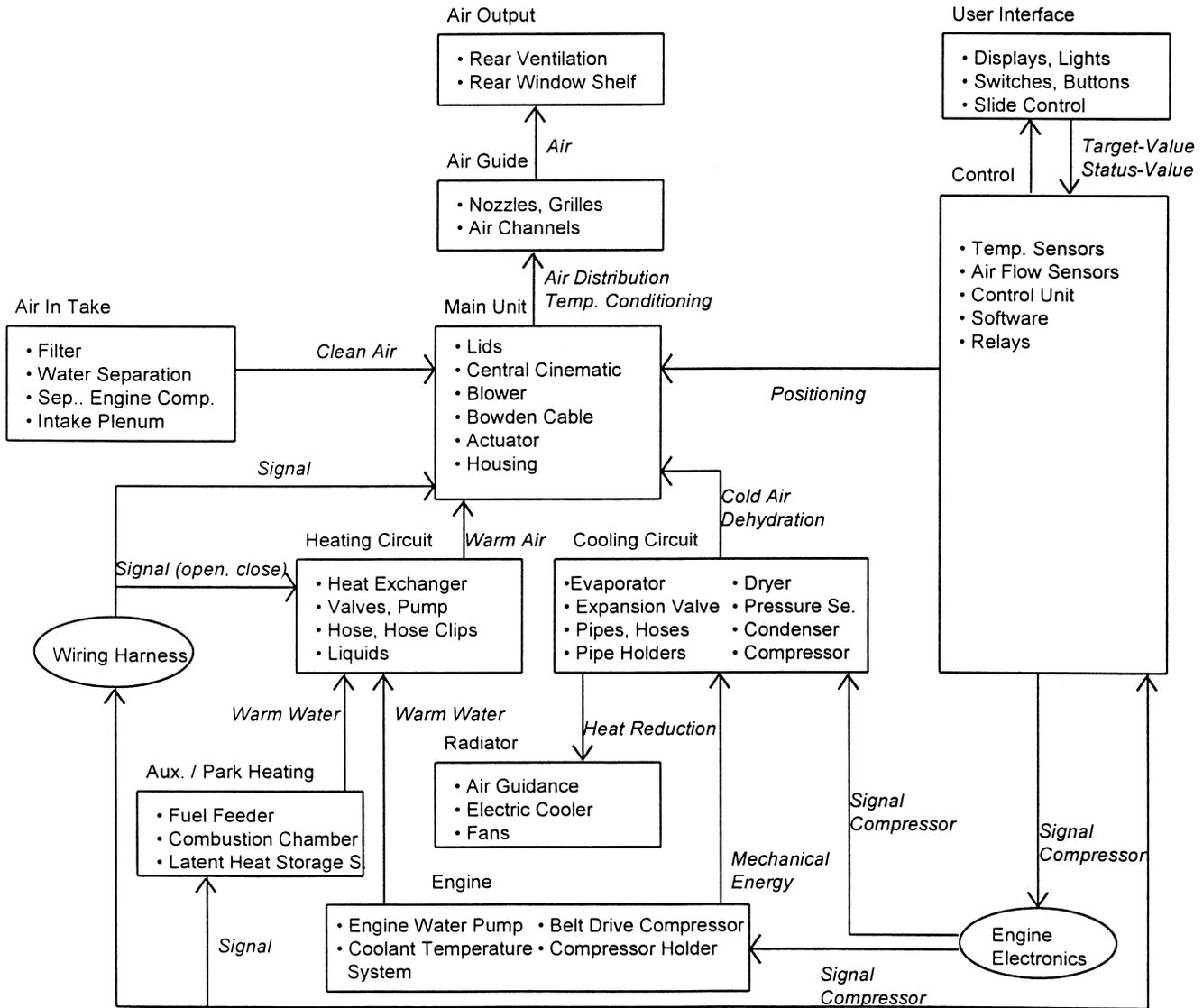
Data collection was performed over four months, during which the first author stayed on-site full time. This permanent presence enabled us to monitor the evolution of information and gain access to data sources usually closed to outsiders. Data were collected from multiple sources, including interviews, participation in relevant meetings, and internal project management information systems.

About 100 semistructured interviews were conducted with 40 engineers and management, including upstream (information-sending) and downstream (information-receiving) parties for every component. Initial individual interviews lasted from one to two hours. Subsequently, one to three follow-up interviews lasting between 15 and 45 minutes helped to capture what had happened since the previous interview.

We passively participated in all relevant meetings on the 10 components. A typical week included five meetings (for all components together). About two-thirds of them occurred on a routine basis (e.g., every Tuesday afternoon). All meetings were cross-functional and, in most cases, included an outside supplier. They were held on the company’s development campus, either in conference rooms or the prototyping lab. In addition to those related to CCS development, we also participated in weekly package meetings on overall vehicle development, where space allocation among various modules was discussed. Each component was owned by one engineer, who had to coordinate with engineers of interacting components (for example at the package meetings) or manufacturing engineers (including suppliers), and get approval for changes from the CCS project manager.

Problem solving in our host organization was supported by a number of information systems. In addition

Figure 2 System Structure of the CCS



to the CAD system, the most important systems for our research were an engineering change management system and a quality control list. The engineering change management system provided a database of all ongoing engineering changes and the anticipated cost and time delay associated with each one. The quality control list provided project management with a database of detected development problems still to be resolved, rather like a big “to do” list. Each open problem was assigned an evaluation of the risk it posed for the overall development process, a responsible engineer, and the next steps to be taken. Both systems were invaluable in that they allowed

us to generate data and identify new leads for additional interviews.

We followed the evolution of specific design decisions and their corresponding engineering changes over time. Following the paradigm of grounded research (Miles and Huberman 1984, Eisenhardt 1989), our analysis built on detailed field notes—interview notes, transcripts of engineering meetings, and company documents—compiled into detailed case studies for each engineering decision. This process was more iterative than sequential as the emerging cases were frequently updated after follow-up discussions with respondents. This included two rounds

of final presentations to engineers involved in the study as well as to the senior managers who had given us the go-ahead. Based on the cases, we contrasted various paths of information evolution, as described below.

Technical Characteristics of the Five Engineering Problems

Below we provide a detailed description of five of the 10 design decisions, as summarized in Table 1, beginning with the context of the information exchange, including sender, receiver, and content.

The five cases are representative of the range of issues encountered and allow a thorough description of the information exchanged, the situation of the information-sending party, and the consequences for the information-receiving party. They help to convey some of the richness and complexity of CCS development and show how our framework for preliminary information is grounded in the data collected.

Much of the thermal energy used for heating the passenger cabin is supplied by the engine, the remainder coming from auxiliary heaters. Crucial information to the development of the CCS is thus the amount of *warm water* supplied by the engine. This is not fully available before engine development is finalized. Engine development supplies preliminary information about the volume of warm water per second to CCS development, which then uses a rough estimate to determine its approach to auxiliary heating.

The *geometry of the air-intake/filtering system* is highly dependent on the engine geometry. Packaging space between engine, fire wall, and dashboard is extremely limited, requiring careful management of geometrical dependencies. This is particularly relevant for air flows as the amount of fresh air entering the cabin is a function of this geometry.

An innovative feature of the CCS is a *latent heat storage system* (LHSS). Once the engine is running at its operating temperature, the LHSS can chemically store its heat for up to several days. This can be used to rapidly warm up the engine and passenger cabin the next time the car is started. At the time of our study, the LHSS was a highly innovative component with unproven demand, and management was not sure whether to include it. Given its substantial size, packaging the LHSS into the vehicle is a rather difficult task which completely changes the layout of the engine compartment. In this case, uncertainty and the associated preliminary nature of the information stemmed from the external market rather than the overlap of activities.

Cars in general, and the CCS in particular, are increasingly sophisticated in their functionality. The CCS contains a substantial amount of *software* controlling dozens

of precision motors, fans, and pumps to create a comfortable cabin environment. Software development depends on testing results from CCS prototyping in order to fine-tune the control system.

The fire wall of the car is a solid metal sheet separating the passenger cabin from the engine compartment. It is a key component in providing body stiffness, but at the same time must have many holes to allow air and water from the engine compartment to enter the cabin, rather like a “Swiss cheese.” The *positions of the holes* are important pieces of information, as they provide the interface between fire-wall development/stiffness, climate control, and stamping die development for the car body.

The Exchange of Preliminary Information from Three Perspectives

To derive our two coordination strategies and the trade-off between them, we first need to describe how preliminary information is exchanged. The description takes three perspectives: the format of the information as transferred by upstream, the downstream adjustment costs to changes, and possible substitutes for preliminary information that may be better overall.

Format of the Information Passed from Upstream to Downstream

The five cases share the same problem: A downstream development activity has to start with preliminary information from an upstream activity. For each of the five cases we describe in what format the preliminary information was passed on, how this uncertainty was resolved over time, and what format the final information took.

The information needed for an auxiliary heating concept is the amount of water supplied by the engine. This information can be captured in a single number. While the first information transfer from the engine group occurred in the form of a number, there was a tacit agreement that this number should be understood more as an interval. This corresponds to the Krishnan et al. (1997) set-based framework, with the uncertainty being resolved in the form of an interval narrowed down over a series of engine tests.

The information required for the package coordination between engine and air intake is more complex. The interface is defined not by a single number but through a complex three-dimensional geometric structure that evolves with the development of the engine. Not all this information is relevant for constructing the air intake—for the development of the air intake/filter box, the geometric details *within* the engine are of little importance—but the *hull* of the engine heavily influences the design

Table 1 Summary of the Five Cases

Case	Information Piece	Sender	Receiver
Auxiliary Heating Concept	<i>How much warm water is supplied by the engine?</i>	<i>Engine Development</i>	<i>Development of CCS in Determining an Auxiliary Heating Concept</i>
Packaging Air Intake/Engine LHSS	<i>Where to find space in the engine compartment for air intake and filter box?</i>	<i>Engine Development, Packaging Team</i>	<i>Development of CCS in Determining Air Flow Concepts</i>
LHSS	<i>Will there be a LHSS for the car?</i>	<i>Management, Based on Market Experience for a Predecessor</i>	<i>Development of CCS and all Package-Critical Activities</i>
CCS Software	<i>What are the software specifications for the control unit?</i>	<i>Development of CCS and Testing Teams</i>	<i>Development of Control Unit</i>
Fire-wall Holes	<i>Where to put the holes in the fire-wall for air and water throughput?</i>	<i>Packaging Team, Development of CCS</i>	<i>Fire-wall Development, Die Development</i>

solution, which aims to provide sufficient air throughput without wasting a cubic centimeter of space. This allows the engine development team to provide the geometry of the hull around the engine as preliminary information to the CCS developers.

In the LHSS case, the required information is extremely concise: One bit (one binary unit of information) suffices to capture the “yes/no” answer. This information became available only after market input from a predecessor project roughly one year before the launch. As a year was too short to redesign the entire packaging of such a large component, the information was explicitly transferred as “yes *and* no,” meaning that the CCS engineers were expected to be prepared for both scenarios. This is mathematically similar to the hull of the engine or the confidence interval for warm water.

The CCS control software needs detailed data from the CCS prototyping as to how changes in certain control variables (e.g., rotation speed of a fan) translate into passenger perceptions (e.g., fresh air, noise from fan). Testing and SW development proceed in parallel, relying on a series of SW prototypes, each of which is developed according to the latest testing results.

Finally, the relevant information concerning the fire-wall holes can be described by about 50–100 numbers describing their geometric position and size. In contrast to the interval approach of Cases 1 to 3, these data were often given with high precision initially and then iteratively modified (corresponding to the iterative framework of engineering change orders in Loch and Terwiesch 1998). An additional way of exchanging preliminary information on holes was to work with “locking zones.” If, for example, for reasons relating to stiffness, it was undesirable to have holes in some areas of the fire wall, zones could be “locked,” thus prohibiting definition of

holes in the CAD system. Information for the five cases is summarized in Table 2.

Comparing the five cases in Table 2 points to two characteristics of a piece of preliminary information. First, its accuracy, which we will refer to as information precision. For example, the information on the fire-wall hole location “hole at $x = 123\text{mm}$, $y = 34\text{mm}$, diameter = 5mm” is precise, whereas the information on the warm water supply “19–21 liters/hour” is not. A measure of precision for the latter information could be defined as a range of 2 liters/hour divided by 20 liters/hour (the midpoint of the interval), or a range of 10% around the midpoint. More generally, a measure of precision can be derived by comparing the range of outcomes that is communicated with the range of all possible outcomes (for discrete sets), or by comparing the range of communicated outcomes to the distance of the range from zero (for parameter intervals).

The second characteristic of preliminary information is the likelihood of it no longer changing through the remainder of the process, which we refer to as *information stability*. The information for the LHSS is fully stable, as “yes” and “no” provide all possible outcomes, and thus the statistical probability that this information will remain unchanged is 100%. This contrasts with the parameters for the software systems, which will almost certainly be changed. The fire-wall hole positions are also very likely to be changed, but by using the locking zones described above, the engineers can reduce changes to the geometry and thereby create some information stability.

Technically, an accurate measure of stability is only possible ex post, after potential changes have occurred. However, our interviews and observations of the engineers’ actions suggest that they form beliefs based on experience, and thus do estimate the stability construct ex ante.

Table 2 Upstream Uncertainty Resolution

Case	Uncertainty Resolved Through	Format of Preliminary Information	Format of Final Information
Auxiliary Heating Concept	<i>Engine testing</i>	<i>Rough guess with tacit understanding of a confidence interval</i>	<i>One number</i>
Packaging Air Intake/ Engine LHSS	<i>Engine development</i>	<i>CAD models of the hull</i>	<i>Detailed geometry (almost impossible to capture in numbers)</i>
CCS	<i>Market observation</i>	<i>Explicit inclusion of both alternatives</i>	<i>Yes/No</i>
Software	<i>Evolutionary prototyping; adjusting parameters based on test results</i>	<i>Prototype</i>	<i>One program and a set of parameters</i>
Fire-wall Holes	<i>Iterative modification</i>	<i>Intermediate solutions; use of locking zones</i>	<i>50–100 numbers describing position and diameters</i>

For a given amount of knowledge, information precision and information stability are in conflict with each other, as the following everyday scenario illustrates. A traveler flying from Philadelphia to Paris wants to arrange a pick-up cab at the Paris airport. The day before the journey, the arrival time in Paris is uncertain. Thus, any information forwarded to the cab driver will be preliminary in nature. The traveler can ignore this uncertainty and communicate an arrival time of 14:34, which is precise information but unlikely to be stable. Or, she can focus on information stability and say she will arrive between 12:00 and 18:00. As the journey unfolds (e.g., prior to boarding, after take-off, at the baggage claim), the uncertainty of the arrival time is reduced and the preliminary information is revised repeatedly until it is fully stable and precise (as she leaves the airport).

Initially, little information on the resolution of the design decision is available (low level of knowledge), and information is neither stable nor precise. As problem solving on the design decision progresses (level of knowledge increases), information is repeatedly communicated with changing levels of precision and stability. At the end of the problem-solving process (high knowledge level), the design solution is in place and functional both upstream and downstream. Now, information is both stable and precise.

While the beginning and the end point of this path are fixed, the question arises at the intermediate levels of knowledge: When should one focus on precision, and when on stability? Before we can answer it, we first need to understand how the information receiver (downstream) reacts to preliminary information and its updates.

Adjustment Costs Downstream

Upstream uncertainty resolution and the release of preliminary information have consequences for the downstream activity. Thus, before we can create a framework

on how preliminary information should be exchanged and used, we need to understand the mechanisms by which information is needed by downstream and what costs the preliminary nature of the information creates.

The auxiliary heating concept does not require highly precise information. A flow deviation of less than 5% creates no major problem. However, larger deviations require a new auxiliary heating concept, which is both costly and time consuming. In addition, downstream adjustment becomes more difficult if modifications occur later. Even minor modifications, if they occur late, can be very costly and potentially delay the launch of the overall project.

The only information that matters for air-intake development is the hull of the engine. As long as information changes do not affect this interface, downstream is insensitive to the changes. Hull modifications become more costly over time because tools for the air intake have very long leadtimes (special plastic tools) and are difficult to change. This will be analyzed more formally below.

The information on the LHSS contains only one bit, but it affects the packaging of the entire engine compartment. Thus, any deviation (e.g., saying “no” first and then changing to “yes”) has major consequences for all development activities involved. A late change may delay the launch of the car. By saying “yes and no,” the development department has to develop two engine compartments in parallel, one with and one without the LHSS.⁸

The CCS software has a modular product architecture which separates a set of parameters from the actual software itself. These parameters are stored in a separate memory module. Given this chip design (details discussed below), the parameters can be changed easily and inexpensively, however late in the process. In fact, even an adjustment to customer needs is feasible during after-sales service. Thus, the cost of adjusting downstream to new information is minimal.

Downstream adjustment costs for the fire wall vary widely depending on the positions of the holes to be changed. Similar to the filter box tools, the stamping dies for the fire wall are expensive to change late in the project. Repositioning holes becomes much more costly once tools have been developed: Downstream’s flexibility to adjust to new information decreases over time. However, unlike the filter box, it is possible to create stamping dies for prototype production from softer steel which can be reworked more easily. Table 3 includes a summary of the downstream adjustments.

Table 3 suggests two types of couplings between the information-supplying and the information-receiving activities. *Rework* occurs if downstream commits resources based on upstream information that later turns out to be wrong; i.e., was not stable. Rework becomes more costly the further downstream has progressed in its problem solving. Almost universally, it is more expensive to make changes late than early.

For the three air-intake components, there were a total of 18 design changes after upstream had released its first precise (but preliminary) information. Like many development organizations, our host company required that such postrelease changes follow a formal process of filing

for an engineering change order (ECO). Formal ECO documents as well as the company internal tracking system for ECOs were used to investigate the relationship between downstream adjustment costs and the timing of a change (see Figure 3).

Figure 3 displays the total cost of each ECO as a function of its occurrence time for the three components of the air intake. Component 1 had six changes (labeled in the sequence of their occurrence as A through F), Component 2 had four changes (A through D), and Component 3 had eight changes (A through H).

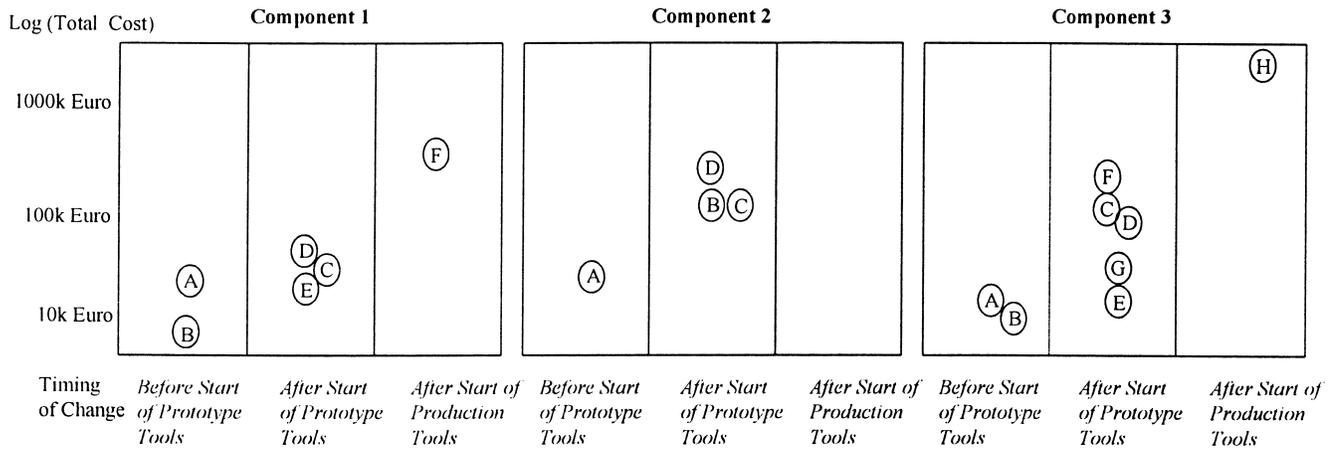
According to the timing of their occurrence, ECOs are classified into three phases: before the start of prototype tools, before the start of production tools, and after the start of production tools. For example, Change F in Component 1 occurred after production tools had been started, at a cost of 190,000 euro.⁹ Change B in Component 3 creates a total cost of 10,000 euro.¹⁰

Figure 3 shows for all three air-intake components that a later detection time of an ECO substantially increased downstream adjustment cost. Moreover, this increase is faster than linear (the cost scale in Figure 3 is logarithmic). The CCS project manager who had to sign off each ECO referred to this as the “10x-rule”: the cost resulting

Table 3 Downstream Implications of Changes in Preliminary Information

Case	Downstream Adjustment Cost if Information Is Not Stable		Downstream Cost if Information Is Not Precise	
	How flexible is downstream with respect to upstream deviations?	Does downstream flexibility change over time?	Starvation: Can downstream continue based on preliminary information?	Duplication: Can downstream prepare for multiple outcomes?
Auxiliary Heating Concept	<i>A bit more or less does not matter.</i>	<i>Increase in adjustment cost due to long leadtime for heating concepts.</i>	<i>Little starvation, as information was sufficiently precise.</i>	<i>Developers sketched out some ideas for the “worst-case scenario”; nothing formal was prepared.</i>
Packaging Air Intake/ Engine	<i>All that matters is the hull; deviations from that can be expensive.</i>	<i>Sharp increases of adjustment costs over time; usage of “soft-dies” not possible.</i>	<i>Waiting is possible, risks however that an inexpensive part shifts the overall critical path.</i>	<i>A continuum of different outcomes / geometry; makes duplication impossible.</i>
LHSS	<i>Packaging of the entire engine compartment depends on this information.</i>	<i>If one said NO initially and then changed to YES, the launch would be delayed.</i>	<i>Would shift LHSS on the overall critical path.</i>	<i>Team prepared for both outcomes.</i>
CCS Software	<i>Modular, program remains stable, only parameter adjustments.</i>	<i>Very flexible, SW adjustments possible even in after sales service.</i>	<i>No, full precision is required for coding.</i>	<i>Not necessary, given the ease of change.</i>
Fire-wall Holes	<i>Varies, depends on hole position.</i>	<i>Sharp increases of adjustment costs over time; during prototyping, this increase could be reduced by “soft dies.”</i>	<i>Some waiting cannot be avoided.</i>	<i>A continuum of different outcomes / geometry; duplication not possible.</i>

Figure 3 Relationship Between Timing of a Change and Rework Cost



from a change with identical content increased by a factor of 10 from one phase to the next.¹¹

While rework is a result of too little stability, the second downstream adjustment cost, which we call *starvation*, is a result of too little precision. Starvation occurs if the downstream activity runs out of work because there is too little information available to start or to continue the task. Starvation is costly, as valuable project time passes while the downstream activity remains idle waiting for additional information, off-setting at least partially the benefits of concurrency.

Whether or not starvation occurs depends on the information needs of the downstream activity. If exact details are not necessary for the downstream work to commence, starvation can be avoided. This is particularly the case if only a subset of tasks within the downstream activity requires detailed information, while others are reasonably “generic.” For example, engineers working on the warm water system could start designing a circuit of pipes for moving warm water from the engine to the main heating units as well as a fixture to hold the auxiliary burner. This design work is fairly generic with respect to changes in the warm water supply. However, the design of the auxiliary heating unit itself (especially its capacity sizing) required more precise information about the warm water supply, and at that point only smaller deviations (about 5%) could be tolerated.

Sometimes starvation can be avoided by *duplication*. In the LHSS case it was avoided by developing two alternative packaging concepts of the front end in parallel. Obviously, such an approach is costly in terms of engineering resources and prototyping hardware. However, by enumerating the complete outcome set of the infor-

mation (yes and no), the team could achieve high precision while retaining stability, thereby maintaining the benefits of concurrency.

Duplication requires the set of possible outcomes to be relatively small. It could not, therefore, be applied to the CCS software, where for combinatorial reasons the number of possible parameter constellations was in the millions. The only other element of duplication among the five cases was the warm water case:¹² While not going as far as developing two complete designs, engineers did prepare a “back-up plan” in case the volume of water supplied was too low.

Substitutes for Information: Decoupling and Increasing Downstream Flexibility

The cost of downstream adjustment can be reduced or even eliminated if it is possible either to weaken interdependence between upstream and downstream, or to make downstream so flexible that adjustment becomes almost costless.

In discussing the information format and downstream adjustment cost, we have treated interdependence as exogenous. However it can, in many cases, be weakened, allowing for the decomposition of the design problem, thereby reducing the need for coordination,¹³ or it may be possible to increase downstream’s flexibility to absorb changes (Thomke 1997). While both mechanisms are clearly desirable from an information-processing perspective, they come at the price of reduced system performance, increased manufacturing cost, or increased development budget and are therefore referred to as substitutes for information. Table 4 summarizes the five cases.

Consider first the CCS software. It has low adjustment

Table 4 Opportunities for Substituting Information (Reducing the Need for Coordination)

Case	Can the interdependence be weakened?	How can downstream's flexibility in absorbing changes be increased?
Auxiliary Heating Concept	<i>Rather than relying on heat from the engine, all heat could be created by an extra burner; penalty in the form of higher weight and cost.</i>	<i>If the heater is sufficiently powerful, any reduction in warm water supply can be absorbed with minor changes.</i>
Packaging Air Intake/ Engine LHSS	<i>Almost impossible, given the dense package.</i>	<i>Soft dies are impossible to use even during prototyping given the material requirements.</i>
LHSS	<i>Modular design possible; would substantially reduce package density or product performance.</i>	<i>Change could only be absorbed by a modular design (see left).</i>
CCS Software	<i>Separation of algorithm and parameters; thus, only a few numbers need to be modified.</i>	<i>By using a rewriteable memory for storing the parameters, the traditional ASIC redesign is avoided; Unit cost increased by about 30%.</i>
Fire-wall Holes	<i>Some small holes can be created independent of die design, at the cost of requiring an additional process step.</i>	<i>Use of soft dies possible during prototyping.</i>

cost (Table 3), but at a price. Traditionally it was “hard-wired” on an ASIC chip, with little opportunity for adjustment. More recently, the software was moved to a chip with rewriteable memory to store the parameter set (similar to a PC, although much smaller in storage capacity) at an increased cost of three euro per unit (a significant cost considering the large volumes involved). Moving the entire software (not only the parameter set) to a rewriteable memory would have cost eight euros more per unit. In this case, interdependence was weakened by separating the parameters from the remaining software, and the adjustment flexibility was increased by working with rewriteable memory instead of ASIC technology.

In the LHSS case, dependence on results from marketing could have been reduced had the team designed a more modular front end with a “slot” for the LHSS, to be used or left empty depending on the marketing decision (similar to the design of most desktop PCs), thus giving the team a design option (Baldwin and Clark 1997). However, it would have meant reduced package density, requiring either a longer car or a higher engine compartment, both of which were unacceptable to our host company.

In principle, the auxiliary heater also offered an opportunity for increasing downstream’s flexibility. If it had been sufficiently powerful, it could have produced any amount of warm water independently of the engine. However, such “over-engineering” increases cost (higher per-unit procurement cost for the heater) and reduces product performance (excess weight).

Finally, the soft dies discussed in conjunction with Table 3 offer yet another way to make downstream more

flexible in reacting to change. As long as changes to the fire wall occur prior to or during prototype tooling, they can be incorporated relatively easily as the steel used is easier to shape. The cost of this flexibility lies in the purchase of the soft dies, which lack durability for volume production and therefore have to be scrapped after prototyping. Interdependence could be weakened by introducing an additional process by which holes in the fire wall are created by an intelligent/programmable press.

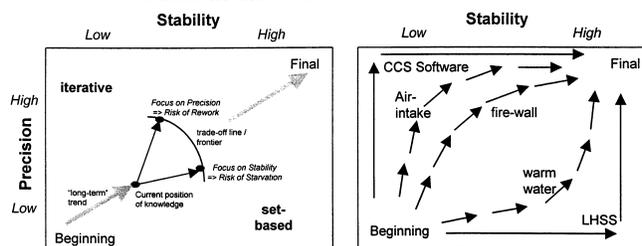
Discussion: Set-Based and Iterative Coordination Strategies

Having explored the tension between information precision and stability, and understood the economic impact of preliminary information on the downstream activity, we now integrate the two perspectives into a coherent framework.

Set-Based and Iterative Coordination

Figure 4 (left) shows that, at each point of information exchange, a decision must be made about the degree of

Figure 4 Trading Off Information Precision with Information Stability in Theory (Left) and for the Five Observed Cases in the CCS



precision and of stability. Taking as fixed the level of knowledge of the problem (the design progress status), there are two options. On the one hand, one can communicate the information in a way that is very stable (i.e., the probability of change is less than some threshold value, e.g., 1%). While this strategy allows downstream to commit resources based on the information without running the risk of rework, given the low precision it may not be able to continue at all (i.e., starvation) or have to perform duplication (redundant work), as in the LHSS case. This corresponds to moving sideways to the right in Figure 4.

On the other hand, one can communicate the information in a way that is very precise. Downstream can then proceed with operational information, but at the risk of having to iterate and redo part of the work if modifications occur, as we have seen in the case of the air intake or the CCS software. This corresponds to moving upward in Figure 4. Obviously, any intermediate point can also be chosen. However the team decides to resolve this trade-off, there is a limit (in the sense of a frontier) to how far it can move along in one period of time. For example, the endpoint of the upper right is out of reach for most information exchanges.

Choosing between stable and precise communication repeatedly over time results in a path through the matrix defined in Figure 4 (left). This path summarizes the coordination strategy between upstream and downstream. Figure 4 (right) illustrates this path for our five cases (summarizing the discussion under Tables 3 and 4).

While the beginning (low level of knowledge) and end points (complete knowledge) of each path are fixed, different trajectories are observed in between. A path which gravitates towards the lower-right quadrant emphasizes information stability at the risk of possible starvation or duplication of downstream effort. We call this *set-based coordination strategy*. In contrast to this, the path gravitating towards the upper left, called *iterative coordination strategy*, emphasizes precision at the expense of increasing rework.

As can be seen in Figure 4 (right), an iterative strategy was used for the CCS software, where information was always communicated precisely, whereas the number of changes clearly indicates a low emphasis on stability. The auxiliary heating concepts case illustrates a more set-based approach, where an interval was communicated first, then narrowed over time. As the interval was not communicated explicitly, and therefore might have widened again at some point, the information was not fully stable.

The LHSS provides the most extreme case. Full stability was maintained from the beginning, since the set of possibilities (“yes” or “no”) did not change. Precision

remained low until the engine compartment packaging was two-thirds complete, then jumped to 100% on the day of management’s decision to include the LHSS.

Pure set-based or iterative strategies are the extremes. Intermediate choices are available, as the air-intake and the fire wall holes show. In both cases, there were a large number of engineering changes, suggesting a more iterative strategy. At the same time, both cases used certain set-based elements in communicating preliminary information: In the air-intake case the engine group communicated the hull around the engine. In the fire wall case the locking zones followed the set-based paradigm.

Choosing Between Iterative and Set-Based Coordination Strategies

Unlike previous studies, which either assert the superiority of one problem-solving strategy over the other (e.g., Sobek et al. 1999), or suggest a link between problem solving and the nature of its technology and/or environment (Eisenhardt and Tabrizi 1995, Terwiesch and Loch 1999a), our study suggests that an organization should master and apply *both* coordination strategies *within* a single development project. With this capability, it can then choose its coordination strategy for every situation of concurrency and interdependence encountered over the course of a project.

In each instance, the choice between the two strategies is driven by the cost of rework relative to the cost of duplication and starvation. As the relative cost of rework grows, stability becomes more attractive. In addition to this cost-based argument, other conditions can facilitate or hinder the effective deployment of a coordination strategy.

Attractiveness of an Iterative Coordination Strategy. An iterative strategy requires that interdependence between upstream and downstream is either relatively weak (typically as a result of a modular design), or that downstream is flexible in absorbing changes (as in the rewritable memory for the CCS software). As discussed in conjunction with Table 4, both provide the means to substitute the need for information by paying the price of lower system performance or higher cost, hence, they favor an iterative strategy.

As a result of its focus on precision (at the cost of stability), an iterative strategy is likely to lead to numerous engineering change orders. Our study found that the formal approval process for ECOs was cumbersome—between 10 days and a whole year for an ECO to be signed off by all parties.¹⁵ This is clearly a severe handicap to applying the iterative strategy.

The approval process reflected the engineers’ desire to have change unfold in an orderly manner, consistent with

their current mode of “sensemaking” (Weick 1993). As a case in point, the development engineer for the air intake had been constructing a particular component for over a year, based on design assumptions (such as the available space) that were formally written down and “frozen” in previous information exchanges. Subsequently, he had to cope with a total of 18 ECOs (see Figure 3), many of them based on elements beyond his horizon, which thus had no obvious logic. As a result, his sensemaking collapsed, leaving him in severe stress, and he took extended sick leave. (Weick calls this a cosmology episode).

Weick (1993) presents several sources of resilience, including “improvisation and bricolage” and “respectful interaction.” The more experienced engineers accepted the need for late rework and understood that the company sold cars thanks to strong engines, not CCS systems. They were able to retain their creativity despite the new circumstances and increased pressure caused by late ECOs. Their reaction was that of a “bricoleur”—tinkering with whatever constraints were imposed late in the process. In contrast, functional engineers who were used to working on CCS components only could not accept the disruption and instead of working around it, took a more defensive approach. Indeed, in the course of our research, this group hired external consultants to help further document and standardize the process (with the aim of imposing control and curbing unexpected engineering changes).

Attractiveness of a Set-Based Coordination Strategy. Having a large search space (a large number of alternatives to be considered) makes effective duplication (as proposed by set-based communication) more expensive and thus favors the iterative strategy. Set-based communication also requires the existence of “natural sets,” that is, to what extent the uncertainty can be communicated with a few bits (e.g., the LHSS with 0/1, or the warm water supply with a confidence interval).

This points to another important prerequisite of the set-based approach: In order to be able to define a meaningful set, upstream must know which elements of its design problem will create interdependence with downstream. Schrader et al. (1993) point to the difference between uncertainty and ambiguity in engineering problem solving. They define a situation as uncertain if the problem solver understands the structure of the problem (including the set of relevant variables), but lacks knowledge concerning the value of these variables. A situation is ambiguous if neither the variables themselves (as opposed to just their realized values) are known nor the problem-solving mechanisms to increase the knowledge.

In the case of the warm water supply, both parties knew that the amount of warm water supplied represented the

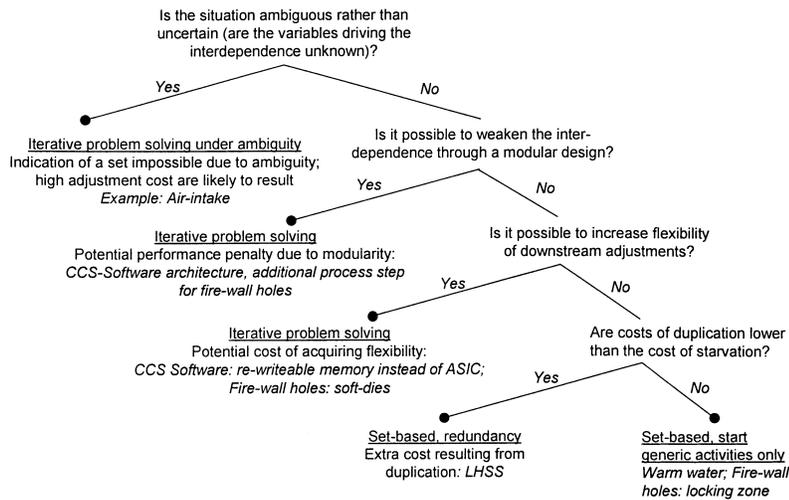
only interdependence between upstream and downstream. Thus, the variable was known and only its value missing. In this situation, the set-based approach ensured downstream that no other interdependence existed, which left downstream with uncertainty, but no ambiguity.

Similarly, the software example was one of little ambiguity (all parameters were defined in the software, the mechanism to find the optimal parameters; i.e., engine testing, was known) but high uncertainty. The team’s choice of an iterative coordination strategy was a result of the trade-off related to the downstream adjustment cost. While it would have been possible to rely on a set-based approach, it was not economical.

In contrast to these two examples, the situation of the air intake was substantially more ambiguous. Being the first air intake of this type, there was no experience to build upon. The engineers initially evaluated the situation as one of uncertainty, with the engine geometry being the only relevant information. But many additional, unexpected variables surfaced, leading to the numerous ECOs described earlier.¹⁶ Thus, while a set-based strategy seemed attractive from an economic perspective, the nature of the problem (at least at the given knowledge level) forced the team into an iterative strategy.¹⁷ This was not consciously chosen by the team but simply “happened” after their set-based strategy failed. This underlines the importance of prior experience with a similar interdependence when choosing between the two coordination strategies. Figure 5 summarizes the choice between the two coordination strategies.

As Schrader et al. (1993) discuss, the difference between uncertainty and ambiguity can have implications for the way engineers go about framing a problem. Historically, downstream has the luxury of commencing the design work after a predefined milestone, at which upstream would hand over all relevant information (stable and precise). In the language of Weick (1993), engineers could be seen as engaging in “conditioned viewing”; i.e., passive and relying on analyzable data coming from upstream, partitioned in defined milestones. As tasks overlap to reduce cycle time, the role of milestones shrinks and engineers must change their interpretive behavior of information they receive. If this change does not occur, a collapse in sensemaking can occur (see above).

If well-defined parameters can be identified that incorporate the dependence between the tasks, the set-based coordination strategy can turn ambiguity into uncertainty: It forces all parties to explicitly identify the range of upstream outcomes, for which one can, in principle, assign probabilities. This facilitates sensemaking and allows engineers to continue conditioned viewing, as the information remains fully stable. Thus, one benefit of the set-based approach is that uncertainty becomes explicit in the

Figure 5 Choosing Between Iterative and Set-Based Coordination

information exchange. It is better to have information labeled as uncertain than to commit resources based on assumptions that later turn out to be wrong. As the introductory quotation shows, understanding the preliminary nature of information, manifested as instability or imprecision, is crucial but often lacking.

This is often not achievable, for two reasons. First, when the design problem is novel (as for the filter box), dependencies cannot be described beforehand. Second, when design knowledge in the upstream task is tacit, the upstream designer may be unable to communicate his anticipated solution to the downstream task. Thus, downstream is left ambiguous about upstream changes and their impact (e.g., Nonaka 1991).

When ambiguity cannot be avoided, *upstream problem solving* must combine multiple parallel approaches (to find one that works) with iteration (to adjust the best approach as knowledge emerges). This is what Sobek et al. (1999) call set-based exploration (see also Pich et al. 2002). Similarly, Bucciarelli (1994) in his ethnographic study of three design projects demonstrated ambiguity and showed how engineers negotiated an emerging shared meaning in a social process, combining multiple solutions with iteration.

Our study, however, concentrates on concurrent activities between an upstream and a downstream task. The very nature of ambiguity makes it virtually impossible for downstream to anticipate all possible upstream solutions, and thus to pursue a set-based approach. Thus, one may not want to proceed concurrently at all, letting upstream finish before starting downstream. Indeed, some project management literature counsels against concurrency in highly novel projects (e.g., Morris and Hugh 1987, Loch

and Terwiesch 2000). Concurrency is appropriate when the design is far enough advanced to largely eliminate ambiguity.

If one does proceed concurrently in the presence of ambiguity (as was the case for the air intake), iteration is the only possible approach: Trial and error, learning and subsequent adjustment lead to an emerging solution (e.g., Thomke 1998). This situation requires that engineers change their modes of interpretation and are flexible in reworking solutions that have already progressed part of the way (which the air-intake engineer was not capable of). This iterative process can be greatly facilitated if the system architecture has reduced the cost of iteration (as is discussed in Table 4).

Figure 5 summarizes the choice between the iterative and set-based coordination strategies in a stylized “decision tree.” If the dependence of the concurrent tasks is ambiguous, only iteration is de facto possible. If only uncertainty is present, the choice is determined by decoupling (modularity) and the comparison of downstream adjustment costs, costs of duplication and of starvation.

Summary and Conclusion

In this article, we develop a dynamic model of coordinating parallel development activities based on an in-depth field study. It rests on an operational definition of preliminary information, consisting of information precision and information stability. This complements the rapidly growing literature on concurrent engineering, which extensively refers to the construct without providing a clear definition.

Our objective is to identify variables that together represent the complex construct of preliminary information

exchange in concurrent engineering projects and to explore how these are connected with the engineers' problem-solving patterns. This is consistent with Miles and Huberman's (1984, p. 7) definition of grounded theory. Our study is based on multiple sources of data gathered on-site in a detailed qualitative study. However, we are concerned more with the development and refinement of theory than with a descriptive report of the developers "Lebenswelt."¹⁸ We believe that Figures 3, 4, and 5 and the analysis presented in this article do provide an extension of the previous state of theory (as was summarized Figure 1). While the initiation of our study was collaborative and did lead to improvements in the practice of concurrent engineering in our host company, this work does not, it should be noted, represent action research or a formal field experiment.

Figure 6 summarizes some of our findings. The left part illustrates the iterative strategy, in which upstream relies on high-precision preliminary information (displayed as Σ), but faces the risk of rework. The middle part illustrates a set-based strategy which focuses on stability (displayed as \square), which symbolizes the common denominator among all possible outcomes at this point in

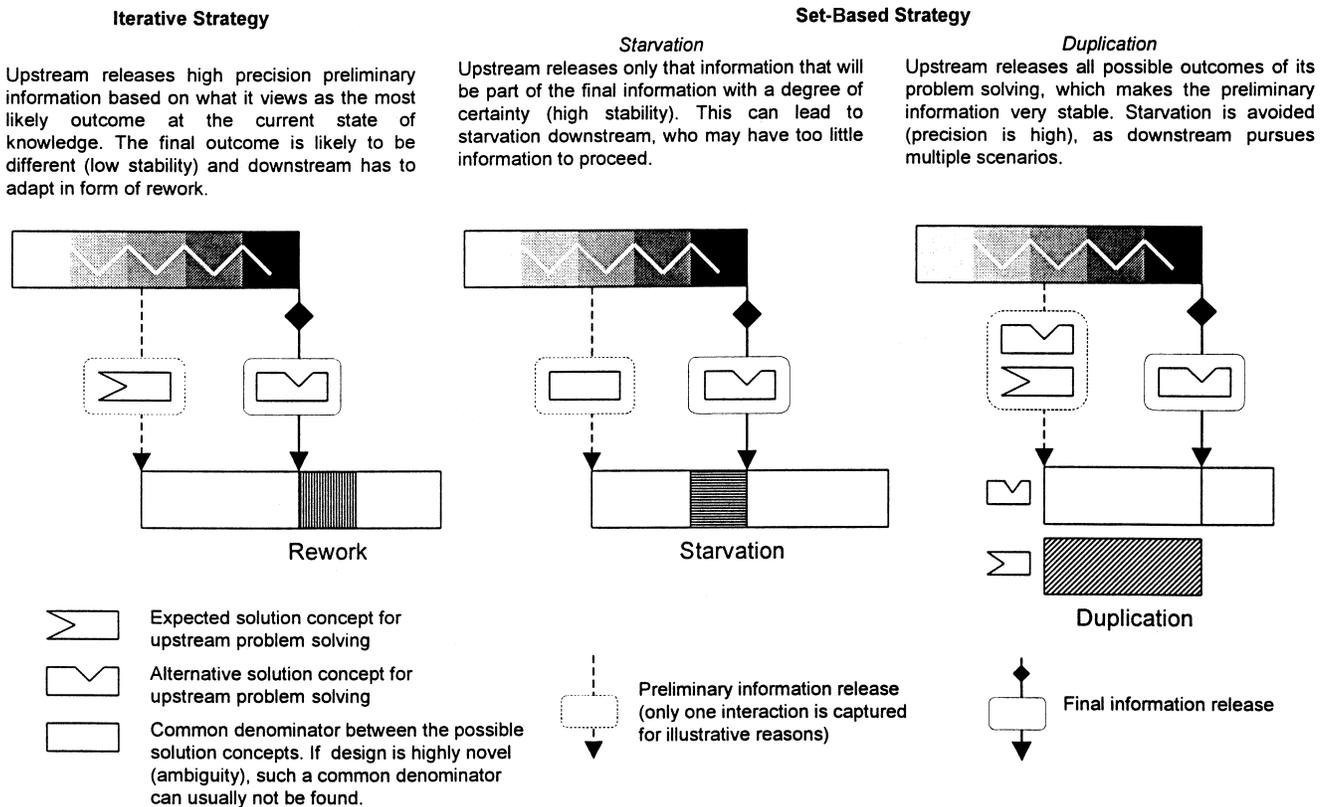
time); the insufficient precision can, however, lead to starvation of downstream. The right part of Figure 6 summarizes a set-based strategy that uses duplication (displayed in the form of downstream working on multiple outcomes in parallel) to avoid starvation.

The two strategies outlined are of direct managerial importance, as they shift from the traditional question "how much should we communicate" (to which the response is generally "a lot" among strongly coupled tasks) to the question of defining a trajectory over time between information stability and information precision.

Finally, by shedding light on the underlying trade-offs linked to the usage of preliminary information, we provide a simple tool to guide concurrent engineering teams in their choice between an iterative and a set-based information exchange strategy. We describe various costs of downstream adjustment as well as substitutes for information that facilitate especially the iterative strategy.

Our study extends the earlier work by Adler (1995) by exploring and refining the temporal dimension in information-processing situations. Specifically, we investigate a situation of concurrent task execution, which creates an additional level of complexity in managing the

Figure 6 Three Modes of Coordinating Parallel Activities



information exchange. In such a situation, it is necessary to look at the information content (not only at the timing and frequency of an exchange), including how preliminary information impacts the downstream task and the team's problem solving. We also refine the work by Clark and Fujimoto (1991), by providing the first operational definition of preliminary information and addressing trade-offs related to its usage.

By grounding our study in detailed data of engineering problem-solving, collected on-site, we also complement the existing literature from a methodological perspective. Over the past years, most studies have relied on interviewing, survey data, or mathematical modeling, methodologies that tend to lose at least some of the richness of the underlying phenomena. While on-site observation allowed us to discover new phenomena in concurrent engineering, it raises issues characteristic of single-company studies: biases, reproducibility, and, most importantly, generalizability. By following methodological guidelines in our data collection, we seek to minimize these problems. More research is needed to ultimately settle the generalizability question.

This article focuses on preliminary information exchange in a concurrent engineering environment. However, we believe that our findings are of relevance to other domains of managing interdependent tasks under time pressure. For example, preliminary information about sales forecasts is now routinely shared in supply chains, enabled by advances in communication technologies. Some of the latest mathematical models in the related literature include constructs that are conceptually similar to our information precision and information stability. Sales forecasts have historically been shared as if they were final information (high precision), but recently researchers have used concepts like a minimum purchase commitment (of at least the specified quantity), which leads to increased stability for the seller (e.g., Tayur et al. 1999).

Several additional opportunities for future research remain. First, our exploratory investigation provides emerging variables and trade-offs that are important in managing preliminary information. Our framework implicitly states a number of hypotheses that can be tested by future research such as the definition of measures and the collection of additional data, which also would overcome the limited generalizability referred to. Second, we have only touched upon the interesting issue of sensemaking and how it relates to the exchange of preliminary information as well as how it influences the engineer's framing of the problem under consideration.

Finally, it is important to explore the impact of increasingly powerful information technologies in product development. Contrary to numerous statements in the business press, new technologies or organizational forms (as in the supply chain example cited earlier) by themselves do not reduce the need for coordination. Indeed, the importance of preliminary information is likely to grow in a world of unlimited bandwidth. New support technologies such as CAD or rapid prototyping may substantially change the cost structure of engineering, and potentially reduce *both* downstream adjustment costs (as simulated designs can be changed quickly) and duplication costs (as virtual prototypes are much cheaper). Thus they reduce the cost of task concurrency, encouraging its use and amplifying the need for coordination of parallel activities (e.g., Baba and Nobeoka 1998, Bensaou 1999). However, the impact of the new technologies on the relative merit of iterative or set-based coordination might go either way. Making decisions based on such preliminary information will therefore remain a key managerial challenge. Investigating its usage in other environments seems a worthwhile research effort.

The surface has simply been scratched, and much work lies ahead. This article contributes to moving the coordination of concurrent activities and preliminary information, in Clark and Fujimoto's (1991, p. 236) words, from "something of an art" to a science.

Acknowledgments

The authors thank Senior Editor William Glick and three anonymous referees for their help with this manuscript.

Endnotes

¹The relationship between uncertainty and ambiguity is discussed in §5.

²Note that by labeling one activity upstream and the other downstream, we are assuming that one activity supplies information to the other during a given information exchange. This does not imply that the information is "thrown over the wall" to downstream. Coordination prior to or during the problem solving can occur, similar to Adler (1995).

³In this study, we take the degree of overlap as exogenously given and focus our attention on the information exchange. We refer the interested reader to the work by Krishnan et al. 1997, Loch and Terwiesch 1998, and Terwiesch and Loch 1999b for the question of when and how much to overlap.

⁴See Thompson 1967, Galbraith 1973, Van de Ven et al. 1976, Victor and Blackburn 1987.

⁵The need for additional research related to the preliminary information concept is summarized by Clark and Fujimoto (1991): "There remains something of an art involved, for example, in leaving a slightly thicker cutting margin in an area of the die more likely to be affected by design changes, or in recognizing that the location of holes in a door panel change more frequently than the anticipated shape."

⁶As our host company was German, it did not use the words preliminary information, but talked about “unscharfe Informationen.” This translates to “fuzzy information.” We decided not to use the word fuzziness, as it would suggest an application of Zadeh’s concept of fuzzy logic (e.g., Zadeh 1994 for a recent summary), which is only remotely related to our study.

⁷We received no pay or financial support from our host company. We were permitted access to the company based on prior relations between ourselves and the host company, and because of the joint curiosity that was created during our initial meeting.

⁸Readers familiar with the auto industry know that this duplicated effort costs several million euros.

⁹One euro is slightly less than one United States dollar.

¹⁰Depending on its occurrence time, an ECO causes different types of rework. Early changes are likely to only cause additional engineering/design hours. A later change, however, typically comes with changes in prototyping tools and sometimes even production tools.

¹¹While this is in accordance with the linear line on a logarithmic scale, the factor 10 (opposed to any other real number) represents a heuristic. Similar rules of thumb are also reported in development guidelines within the company, as well as by other vehicle-producing companies and even other industries.

¹²We observed one other case of duplication in one of the decisions not reported in this article. In the absence of detailed user preference data related to the ergonomics of the CCS interface, which was under development by a marketing group, the team developed four different interfaces concurrently.

¹³This is in line with Staudenmeyer (1999), who observes that interdependence among design teams is mitigated by the architecture of the system under development. Sometimes interdependencies can be minimized, using a modular product architecture, which—in general—comes at the cost of reduced system performance (Ulrich 1995).

¹⁴Due to the high pressure needed to make air-intake prototype parts, soft dies were not an option here.

¹⁵To keep development work from spiraling out of control, design changes must be approved in a formal process. Long ECO leadtimes result from a complicated formal approval process, combined with extensive paperwork, problem-solving responsibilities dispersed across different groups, and congestion effects. In these cases, substantial waiting times result, increasing the cost of rework (details of this part of the study are reported in Loch and Terwiesch 1999).

¹⁶Examples of other, nongeometric, interdependences included noise and vibration problems, problems in the assembly sequence, and problems with engine maintenance in the after-sales environment.

¹⁷This is related to Eisenhardt and Tabrizi 1995 and Terwiesch and Loch 1999b, who distinguish between an experiential, iterative problem-solving strategy and a highly structured strategy. The authors report that in rapidly changing markets and technologies, structured approaches are likely to fail.

¹⁸See, e.g., Bucciarelli (1994) for an ethnographic study of three engineering projects that leads to new reflections on the dynamics of technological choice. The study also creates a less rational and more sociocultural picture of how engineering occurs.

References

Adler, P. S. 1995. Interdepartmental interdependence and coordination: The case of the design manufacturing interface. *Organ. Sci.* **6**(2) 147–167.

- Allen, T. 1977. *Managing the Flow of Technology*. The MIT Press, Boston, MA.
- Baba, Y., K. Nobeoka. 1998. Towards knowledge-based product development: The 3D-CAD model of knowledge creation. *Res. Policy* **26** 643–659.
- Baldwin, C. Y., K. B. Clark. 1997. Managing in an age of modularity. *Harvard Bus. Rev.* **75**(September–October) 84–93.
- Bensaou, M. 1999. Collaboration support technologies in interorganizational relationships: An empirical exploration in buyer-supplier joint design activities. Working paper 99/78/TM/ABA, INSEAD.
- Boehm, Barry W. 1981. *Software Engineering Economics*. Prentice Hall, Englewood Cliffs, NJ.
- Brown, S., K. M. Eisenhardt. 1995. Product development: Past research, present findings, and future directions. *Acad. Management Rev.* **20**(2) 343–378.
- Bucciarelli, L. L. 1994. *Designing Engineers*. The MIT Press, Boston, MA.
- Clark, K. B., T. Fujimoto. 1991. *Product Development Performance: Strategy, Organization and Management in the World Auto Industry*. Harvard Business School Press, Cambridge, MA.
- Cooper, R. C. 1993. *Winning at New Products: Accelerating the Process from Idea to Launch*. Addison Wesley, Cambridge, MA.
- Cusumano, M. A. 1997. How Microsoft makes large teams work like small teams. *Sloan Management Rev.* **39**(Fall) 9–20.
- , K. Nobeoka. 1998. *Thinking beyond lean: How multi-project management is transforming product development at Toyota and other companies*. The Free Press, New York.
- Eastman, R. M. 1980. Engineering information release prior to final design freeze. *IEEE Trans. Engrg. Management* **EM-27** 37–41.
- Eisenhardt, K. M. 1989. Building theories from case study research. *Acad. Management Rev.* **14** 532–550.
- , B. N. Tabrizi. 1995. Accelerating adaptive processes: Product innovation in the global computer industry. *Admin. Sci. Quart.* **40** 84–110.
- Eppinger S. D., D. E. Whitney, R. P. Smith, D. A. Gebala. 1994. A model-based method for organizing tasks in product development. *Res. Engrg. Design* **6** 1–13.
- Galbraith, J. 1973. *Designing Complex Organizations*. Addison Wesley, Cambridge, MA.
- Iansiti, M., S. D. Eppinger, D. E. Whitney. 1997. A model-based framework to overlap product development activities. *Management Sci.* **43** 437–451.
- Krishnan, V., K. Ulrich. 2001. Product development decisions: A review of the literature. *Management Sci.* **47**(1) 1–21.
- Liker, J. K., D. K. Sobek II, A. C. Ward, J. J. Cristiano. 1996. Involving suppliers in product development in the United States and Japan: Evidence for set-based concurrent engineering. *IEEE Trans. Engrg. Management* **43** 165–178.
- Loch, C. H., C. Terwiesch. 1998. Communication and uncertainty in concurrent engineering. *Management Sci.* **44**(8) 1032–1048.
- , ———. 1999. Accelerating the process of engineering change orders: Capacity and congestion effects. *J. Product Innovation Management* **16** 145–159.
- , ———. 2000. Product development and concurrent engineering. P. M. Swamidass, ed. *Encyclopaedia of Production and Manufacturing Management*. Kluwer Academic Publishing, Dordrecht, Germany, 567–575.

- Miles, M., A. M. Huberman. 1984. *Qualitative Data Analysis*, Sage Publications, Beverly Hills, CA.
- Morris, P. W. G., G. H. Hugh. 1987. *The Anatomy of Major Projects*. Wiley, Chichester, U.K.
- Nonaka, I. 1991. The knowledge-creating company. *Harvard Bus. Rev.* **69**(November–December) 96–104.
- Pich, M. T., C. H. Loch, A. De Meyer. 2002. On ambiguity, uncertainty and complexity in project management. Working paper, INSEAD, Fontainebleau, France.
- Sabbagh, K. 1996. *Twenty-First Century Jet*. Scribner, New York.
- Schelling, T. C. 1978. *Micromotives and Macrobehavior*. W. W. Norton, New York.
- Schrader, S., W. M. Riggs, R. P. Smith. 1993. Choice over uncertainty and ambiguity in technical problem solving. *J. Eng. Tech. Management* **10** 73–99.
- Scrivener, S. A. R., A. Woodcock, L. J. Ball, eds. 2000. *Collaborative Design*. Springer, Berlin–New York.
- Sobek, D. K., A. C. Ward, J. K. Liker. 1999. Toyota's principles of set-based concurrent engineering. *Sloan Management Rev.* **39**(Winter) 67–83.
- Soderberg, L. G. 1989. Facing up to the Engineering gap. *McKinsey Quart.* **25**(Spring) 3–23.
- Staudenmayer, N. 1999. Webs of interdependency: Strategies for managing multiple interdependencies in new product development. Working paper, Duke University, Durham, NC.
- Stoy, R. 1996. Assembled product development. M. D. Rosenau, A. Griffin, G. A. Castellion, and N. F. Anschuetz, eds. *The PDMA Handbook of New Product Development*. John Wiley, New York.
- Takeuchi, H., I. Nonaka. 1986. The new, new product development game. *Harvard Bus. Rev.* **64**(January–February) 137–146.
- Tayur, S., R. Ganeshan, M. Magazine. 1999. *Quantitative Models for Supply Chain Management*. Kluwer, Dordrecht, Germany.
- Terwiesch, C., C. H. Loch. 1999a. Managing the process of engineering change orders: The case of the climate control system in automobile development. *J. Product Innovation Management* **16** 160–172.
- , ———. 1999b. Measuring the effectiveness of overlapping development activities. *Management Sci.* **45**(4) 455–465.
- Thomke, S. H. 1997. The role of flexibility in the design of new products: An empirical study. *Res. Policy* **26** 105–119.
- . 1998. Simulation, learning and R&D performance: Evidence from automotive development. *Res. Policy* **27** 55–74.
- Thompson, J. D. 1967. *Organizations in Action*. McGraw Hill, New York.
- Trip, G. 1997. Turning rough takes into summer's big hits. *New York Times* (May 5, D1).
- Ulrich, K. 1995. The role of product architecture in the manufacturing firm. *Res. Policy* **24** 419–440.
- Van de Ven, A. H., A. L. Delbeq, R. Koenig. 1976. Determinants of coordination modes within organizations. *Amer. Sociological Rev.* **41** 322–338.
- Victor, B., R. S. Blackburn. 1987. Interdependence: An alternative conceptualization. *Acad. Management Rev.* **12** 486–498.
- Weick, K. E. 1993. The collapse of sensemaking in organizations: The Mann Gulch disaster. *Admin. Sci. Quart.* **38**(4) 628–652.
- Wheelwright, S. C., K. B. Clark. 1992. *Revolutionizing Product Development*. The Free Press, New York.
- Zadeh, L. 1994. A fuzzy logic, neural networks, and soft computing. *Comm. ACM* **37** 77–84.