

# **Adaptive Decisionmaking**

**A white paper**

**Jeff Rothenberg  
The RAND Corporation  
January 2005**

## **Abstract**

Adaptive Decisionmaking is a novel approach to managing and performing complex, multi-faceted endeavors in which decisions often have to be reversed. Examples of such endeavors include business process re-engineering, designing and implementing large-scale systems, conducting environmental or ecological studies, and performing long-term policy or systems analyses. Adaptive Decisionmaking consists of two integrated ideas: (1) an iterative “Cyclic” process model for all of the activities involved in an endeavor and (2) a Decision Knowledge Base (DKB) that enables activities to coordinate their progress. Each activity performs a series of cycles, each of which attempts to produce an increment of progress; at the start of each of its cycles, each activity uses the DKB to determine what relevant new factors (events, decisions, constraints, assumptions, etc.) have emerged from other activities since its last cycle, allowing it to adapt the goal of its current cycle accordingly. The DKB is used in subsequent steps of each cycle to coordinate and synchronize with other activities as needed.

## **Introduction**

Adaptive Decisionmaking is a set of techniques intended to help manage and perform complex, multi-faceted endeavors involving multiple stakeholders and multiple, co-evolving activities. Examples of such endeavors include business process re-engineering of large organizations or institutions, designing and implementing large-scale systems and “systems of systems” (SoSs), conducting environmental or ecological studies, and performing ongoing, long-term policy or systems analyses. Such endeavors typically have many active participants—as well as other, more peripheral stakeholders—each of whom has distinct perspectives, goals, and concerns. These participants perform numerous, disparate activities, such as management, strategic planning, analysis, design, implementation, training, community-outreach, interaction with other, related endeavors, etc. These activities co-evolve with each other in dynamic and unforeseen ways.

The traditional approach to performing such endeavors typically adopts the convenient but false assumption that they will progress in a linear, monotonic fashion and that their outcomes can be predicted from the start. This unwarranted assumption frequently leads to a failure to meet expectations and may produce results that are of doubtful value—or at least cause extensive overruns and delays. In contrast, Adaptive Decisionmaking is based on the insight that complex, co-evolving endeavors rarely unfold as predicted; the false assumption of monotonic progress is made, not because anyone really believes it, but rather because facing reality is too complex. Much of this complexity arises from the interdependence of the many activities that comprise such an endeavor and the fact that each activity is affected by decisions, discoveries, and events that occur in other activities. Participants in each activity have a hard enough time trying to make progress within their own activity without constantly responding to changes emanating from other activities. In reality, participants are frequently forced to respond to such changes, but they often do so in an ad hoc manner, receiving little help from the processes and techniques they employ, which are based on the fiction that such responses will rarely be necessary. Adaptive Decisionmaking addresses this problem head-on by providing a disciplined way to coordinate and synchronize the efforts of multiple activities while enabling them to make meaningful—though possibly non-monotonic—progress. In particular, it allows each activity to adapt its behavior to factors that arise from other activities within the endeavor or even from outside the endeavor.

Adaptive Decisionmaking consists of two integrated ideas: an iterative “Cyclic” process model for all activities and a Decision Knowledge Base (DKB) that enables activities to coordinate their cyclic progress. Each activity performs a series of cycles, each of which attempts to produce an increment of progress; at the start of each of its cycles, each activity first uses the DKB to determine what relevant new factors (events, decisions, constraints, assumptions, etc.) have emerged from other activities since its last cycle, allowing it to adapt the goal of its current cycle accordingly. Subsequent steps in each cycle use the DKB to coordinate and synchronize with other activities as needed. The Adaptive Decisionmaking paradigm thereby

- **Facilitates adaptation and co-evolution**
- **Operationalizes risk-management**
- **Increases traceability, accountability, and reproducibility**

## **The problem**

There are many factors that make complex endeavors difficult. In any undertaking involving more than a few people, it is difficult if not impossible for participants to keep all relevant factors in mind, especially since these factors—and their relevance—change dynamically. Yet few endeavors explicitly capture more than a fraction of their assumptions, decisions, constraints, rationales, and actions or represent dependencies among these factors and among the activities that control them and rely on them. Leaving such factors implicit makes them difficult to analyze, question, communicate, and share across—or even within—activities. Similarly, this reliance on implicit knowledge makes it costly and inefficient to bring new participants on board, since they must waste considerable effort reexamining issues that have already been laid to rest and pursuing dead-ends that have long since been abandoned. (A fresh perspective can sometimes breathe new life into an abandoned option, but this process is more efficient and effective if it is informed by an explicit rationale for why the option was discarded originally.)

In addition, it is difficult to understand the consequences of decisions in a complex endeavor, and this is exacerbated by a lack of explicit dependencies among assumptions, constraints, and previous decisions. Rationales for decisions are often at most an afterthought, yet they are vital for tracing the implications of previous choices and predicting where new decisions are likely to lead. Furthermore, in the absence of explicit dependencies, it is extremely difficult to see the relevance or implications of changes and therefore to respond appropriately, or even to know when a response is necessary.

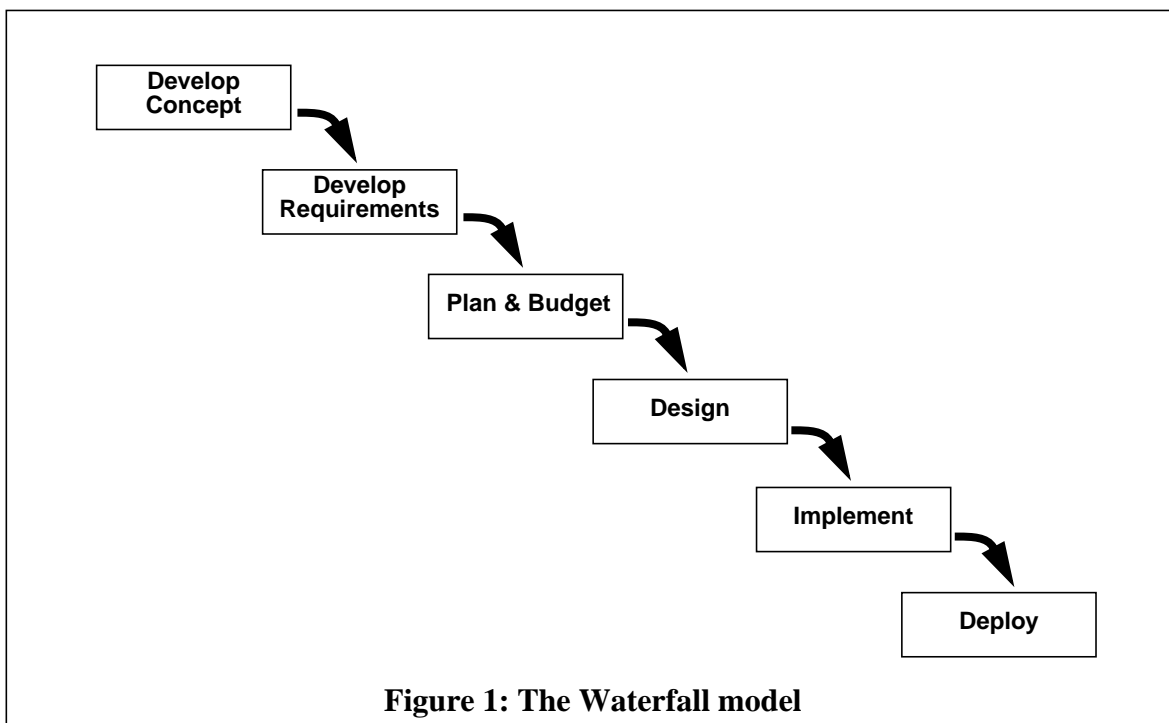
Finally, it is naive to imagine that we can plan more than the broad outlines of any complex endeavor far in advance. Many initial conditions of an endeavor are really just assumptions, which can easily be violated, and many decisions should be treated as tentative, to be revised or even reversed when appropriate, on the basis of further experience. It is understandable to want to plan an endeavor up front, both in order to gain some idea of where it is headed and to help convince others that it is worth pursuing (and funding); but it seems axiomatic that it makes little sense to try to plan farther in advance than we know enough to plan. In fact, as argued above, it is often unrealistic to expect a complex endeavor to progress monotonically, let alone to be able to plan its course.

One of the key strategies that is used in approaching complex endeavors is what is sometimes called divide-and-conquer. When faced with something large and complicated, we break it up into smaller, more tractable pieces. Though it is hard to see any alternative to this approach, it isolates each activity from all the others. Indeed, this is the crux of the rationale for dividing problems: limiting the scope of each activity is designed to make it unnecessary for those solving a given sub-problem to consider factors outside their scope. In practice, however, factors from many other activities outside of a given activity's scope impinge on it in all sorts of ways. Changes in funding or other resources affect everything, while management decisions concerning one aspect of an endeavor affect the management of other aspects, design decisions concerning one subsystem affect other subsystems, management decisions affect design, design decisions affect training, implementation decisions affect schedule, etc. No matter how hard activities try to ignore each other, they are bombarded by a constant stream of decisions and actions taken by other activities,

many of which affect them profoundly. This is sometimes called the boundary problem, since it involves information that must cross the scope boundary of each activity.

To some extent the boundary problem is inherent in the divide-and-conquer strategy. But it is compounded by the absence of explicit representation of assumptions, constraints, decisions, rationales, and other events in most endeavors; this makes it extremely difficult and costly to see clearly beyond the boundary of a given activity. Various management strategies can be seen as attempts to solve this problem, ranging from the sequential scheduling of activities to the specification of well-defined interfaces or “contracts” between activities. However, most of these strategies have trouble coping with the co-evolving activities that are the norm for many complex, dynamic endeavors, such as those that involve the use of cutting-edge technology, which often changes within the lifetime of an endeavor. In order to cope with co-evolution, the activities in an endeavor must be able to adapt their behavior, change their plans and the directions in which they are proceeding, and coordinate, integrate, and—when necessary—synchronize their progress with that of other relevant activities.

The sequential, “waterfall” model (of which a generic version is shown in Figure 1) emerged from the systems development arena rather than the wider context of this discussion, and it is not always used in its pure form even in its original domain; but it serves as a useful strawman for elaborating these ideas. As its name implies, it is a unidirectional process, intended to flow downward, with each activity being performed to completion before the next is begun. This assumes that the dependencies among activities are quite limited: the only inputs to an activity are the outputs of the activity immediately above it in the waterfall. No activity can affect any activity that is above it in the waterfall, and even direct downward effects are limited to adjacent activities (though indirect effects may be transmitted via intermediate activities). The waterfall paradigm assumes that requirements for the desired solution can be specified in detail in advance and that monotonic

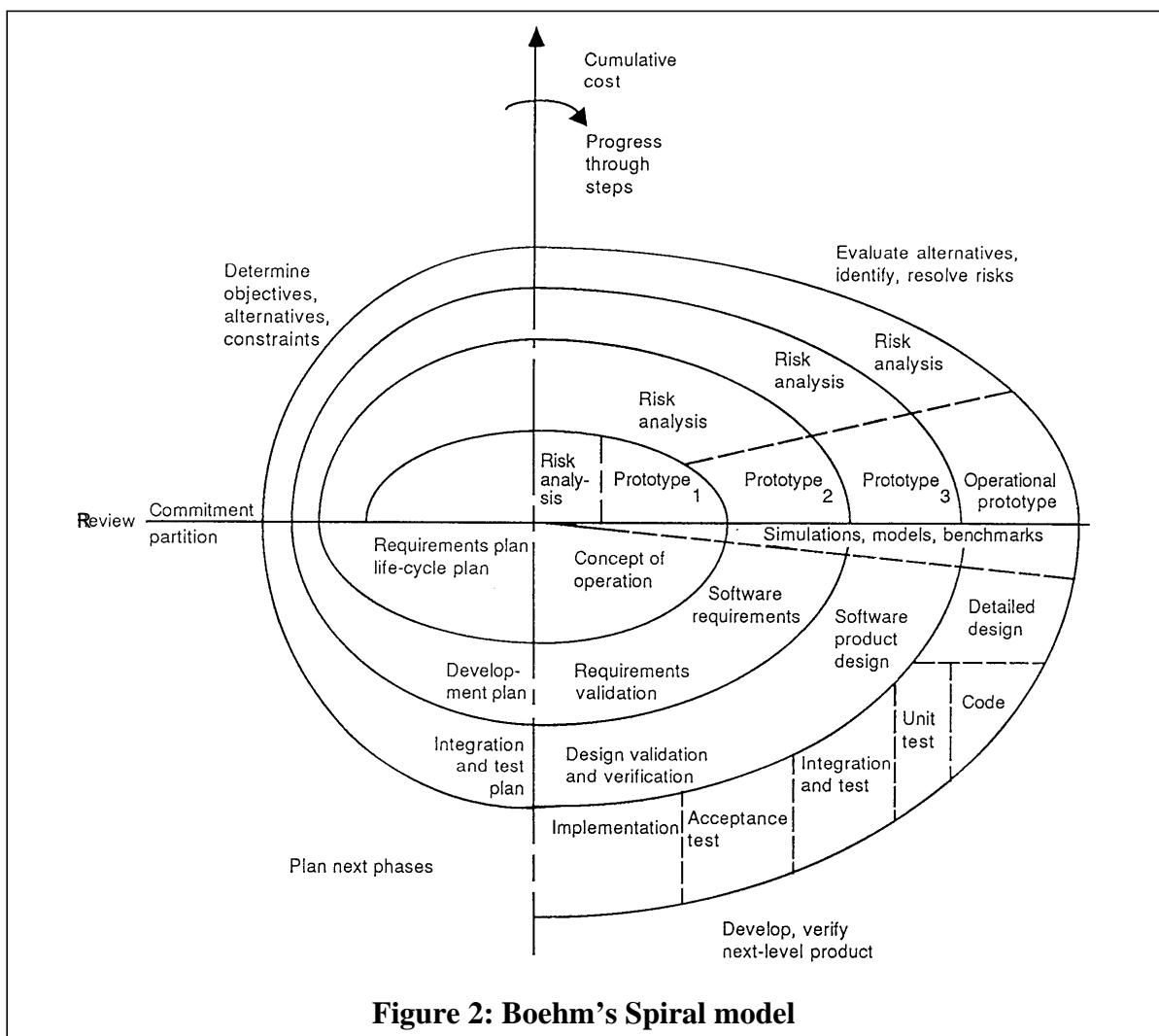


progress can be made toward this solution. These assumptions make it very costly to back up and redo previous activities in response to problems or opportunities that arise later in the process. (The waterfall model is sometimes amended by the addition of a “salmon ladder” to allow information to go upstream when necessary, but the metaphor is overly apt, since many ideas that try to buck the current die in the attempt.) In practice the waterfall paradigm often leads to sub-optimal results or even complete failure.

### A solution: Adaptive Decisionmaking

Adaptive Decisionmaking attempts to solve the boundary problem in a principled way, while also abandoning the assumption of monotonicity. It offers a flexible, Cyclic process model for all of the activities in an endeavor, and its Decision Knowledge Base (DKB) enables participants in each activity to maintain awareness of those events arising in other activities that affect them, thereby allowing activities to coordinate and synchronize their actions as necessary.

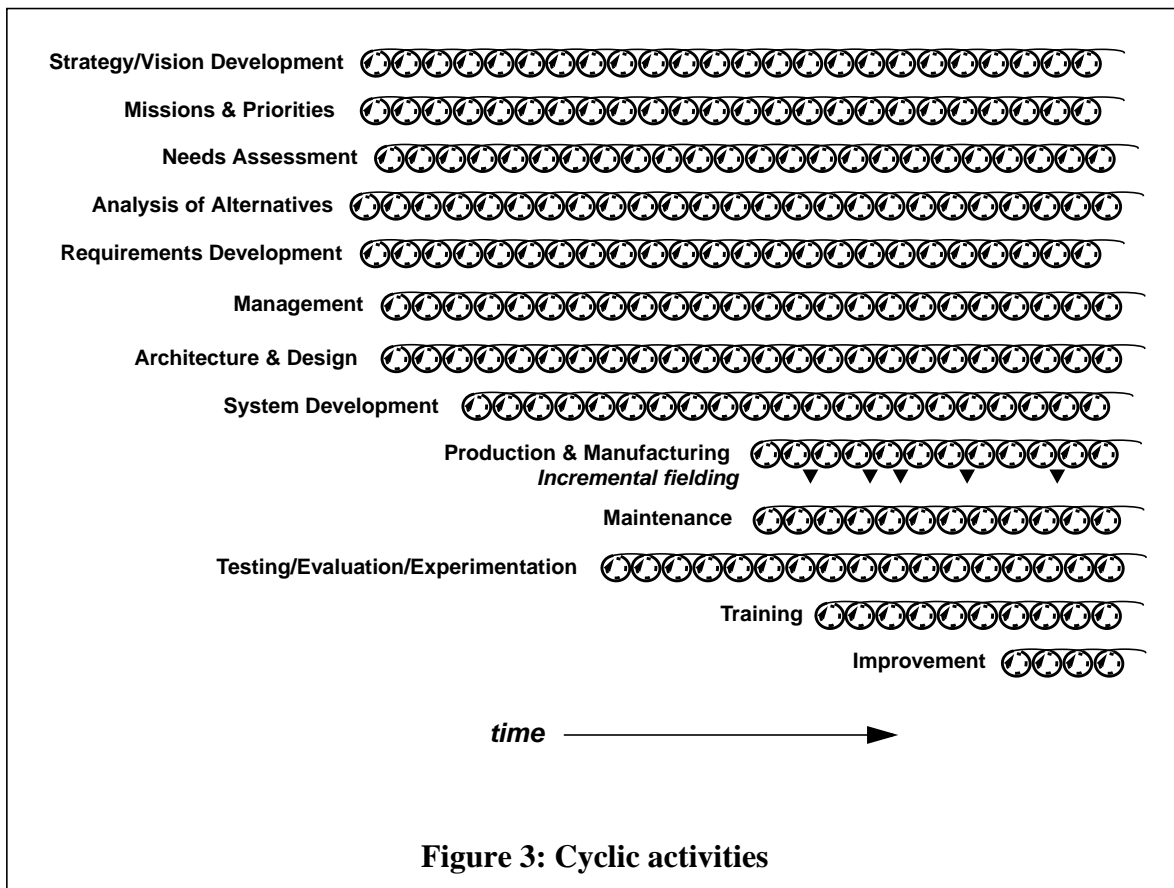
The Cyclic process model is an outgrowth of Boehm’s “Spiral” model of software development (shown in Figure 2), which was designed to explicitly counteract the deficiencies of the waterfall



**Figure 2: Boehm’s Spiral model**

paradigm [1, 2]. Because the Spiral model is tailored to software development, however, it is not directly applicable to such disparate activities as management, marketing, training, etc. Moreover, like the waterfall model that it replaces, the spiral model still implicitly assumes monotonic progress: it provides no explicit mechanisms for reexamining or revising past decisions or assumptions. In contrast, the Cyclic model requires each iteration of each activity to consult the DKB to see what has changed during the activity's previous cycle and to plan its current cycle based on the new state of the endeavor implied by these changes.

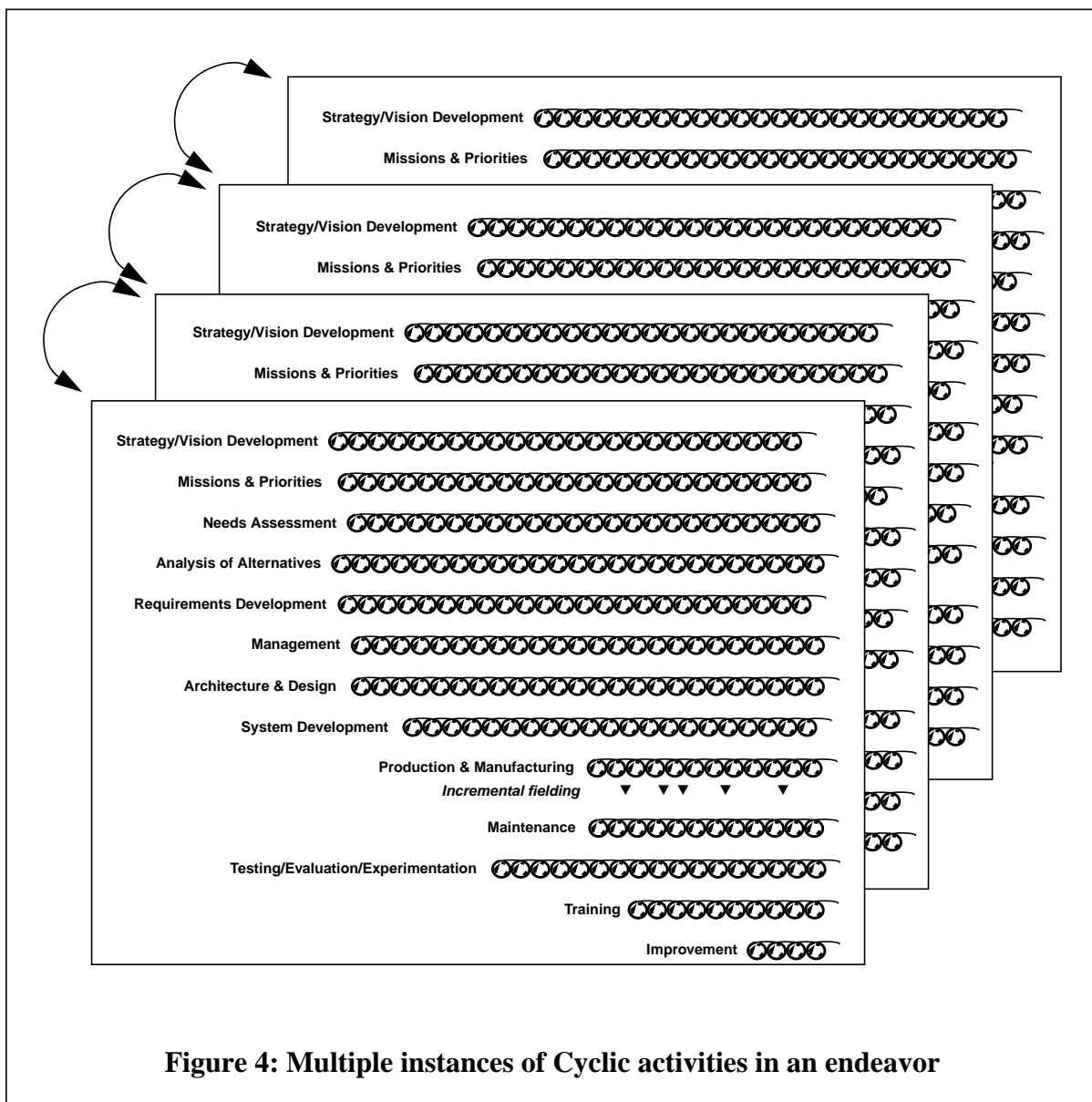
Figure 3 shows a notional collection of activities that comprise a typical complex endeavor. Each activity is shown as a sequence of cycles, indicating that it is to be performed according to the Cyclic process model. (For simplicity, the figure shows all cycles as the same size, though in practice their sizes and durations will vary.) The temporal overlapping of activities in the figure indicates that some activities may start before or last longer than others; but unlike the waterfall model in which each activity must finish before the next begins, most activities in the Cyclic paradigm proceed in parallel. However, a given activity may perform different tasks at different stages. For example, an architecture/design activity may initially focus on architectural issues, later devoting most of its effort to lower-level design issues. This allows each activity to be available to other activities as the endeavor unfolds, which is essential if activities are to adapt to each other and co-evolve; for example, the ongoing architecture/design activity can be called on to re-examine



**Figure 3: Cyclic activities**

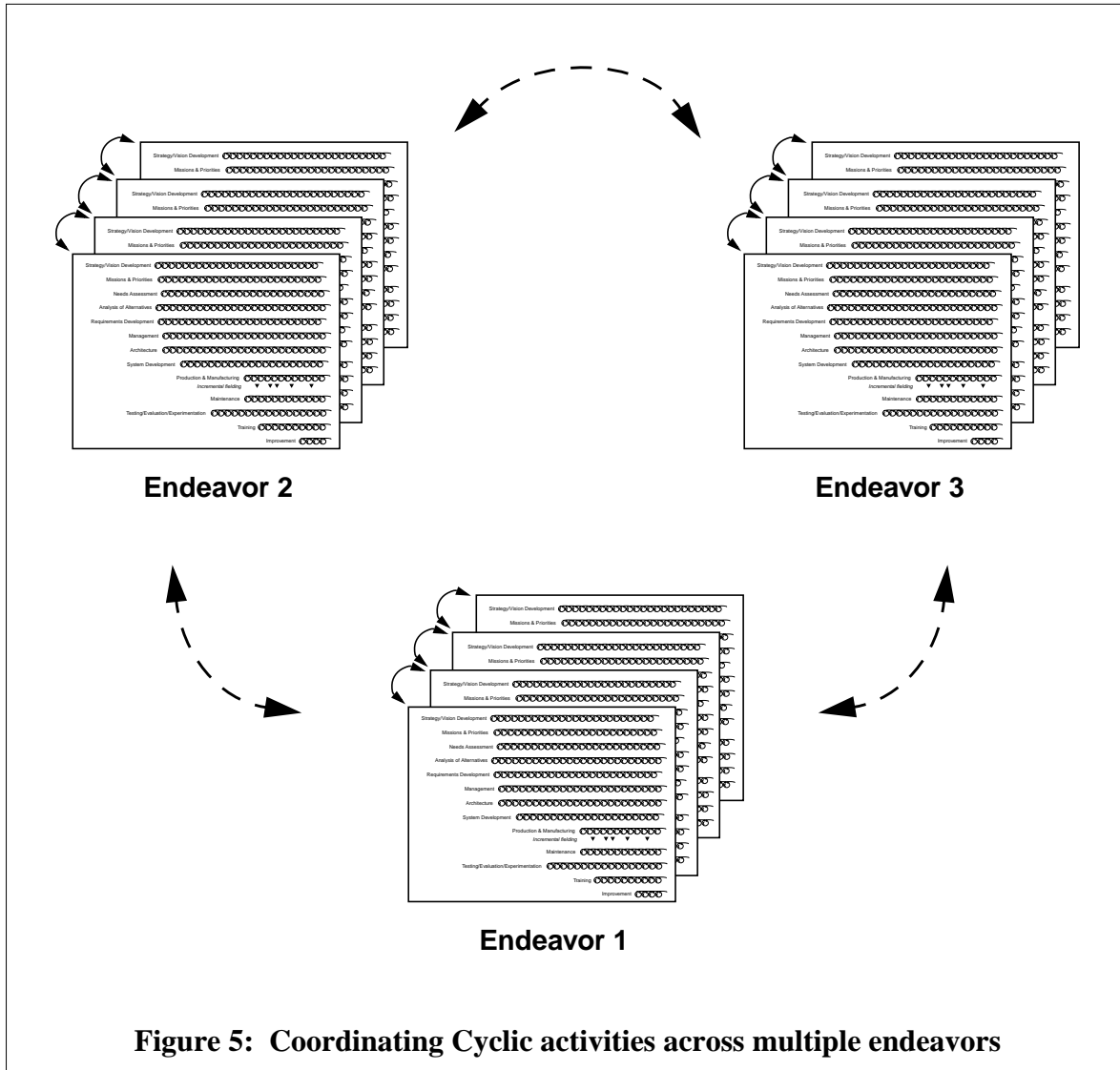
and revise architectural issues if problems are encountered during implementation or if requirements change as the endeavor proceeds.

The precise set of activities will of course be specific to the endeavor at hand, but most endeavors will include needs assessment, problem analysis, analysis of alternative solutions, requirements determination, and management, and many will include some subset of strategic vision development, architecture and design, implementation, market analysis, marketing, training, maintenance, interaction with other endeavors, outreach, etc. Some endeavors may choose to combine or split some of these activities (for example, separating out funding or planning from management or combining design and implementation), but any complex endeavor is likely to have a fair number of such activities. Furthermore, as illustrated in Figure 4, a given endeavor may include multiple instances of each such activity: for example, the development effort for a large-



**Figure 4: Multiple instances of Cyclic activities in an endeavor**

scale system (or system of systems) will include multiple components or subsystems, each of which may have its own management, design, implementation, training, and other activities. Finally, an endeavor may have interfaces and interactions with other endeavors, making the complete collection of relevant activities even more complex, as illustrated by Figure 5.

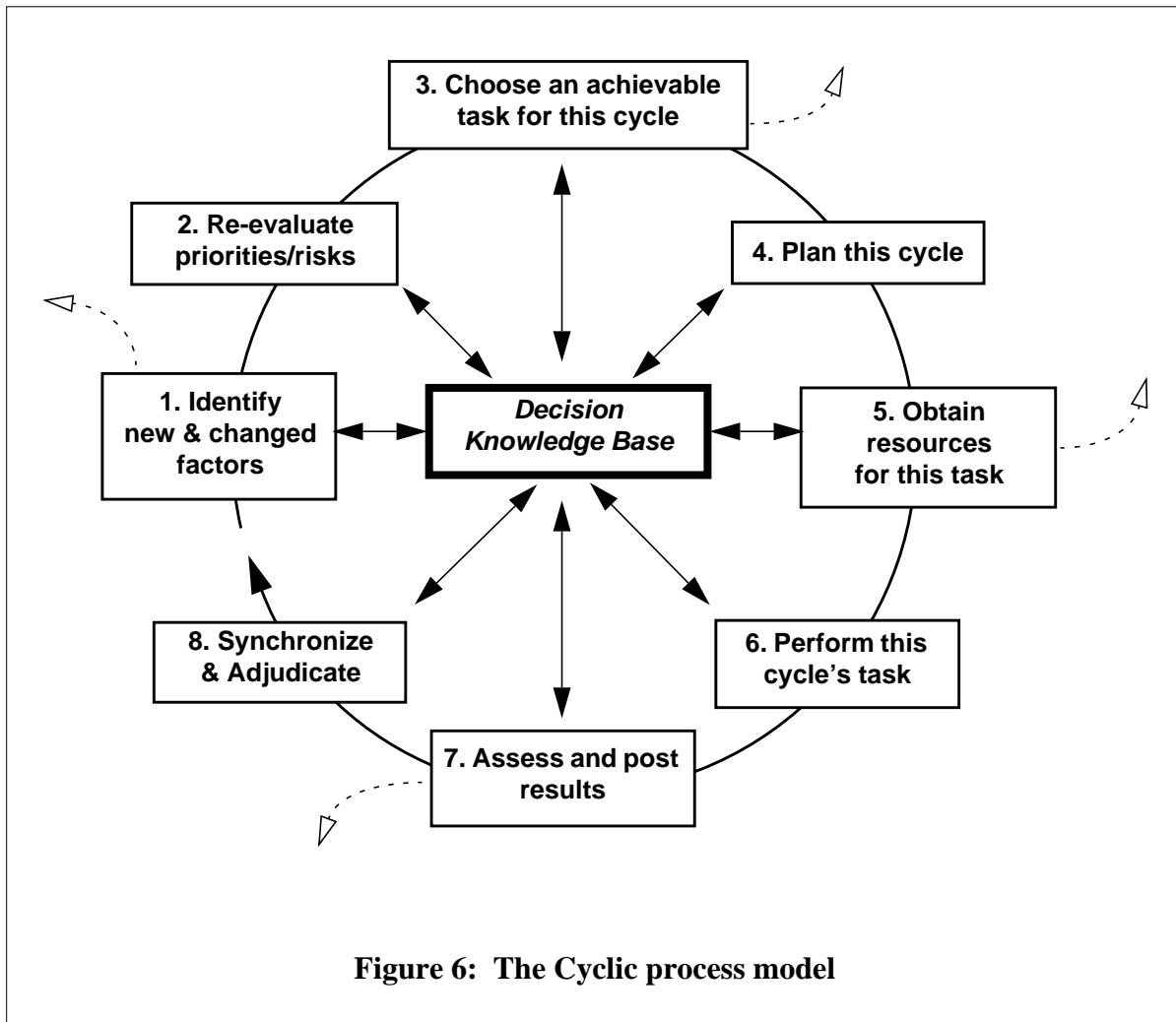


**Figure 5: Coordinating Cyclic activities across multiple endeavors**

In order to be adaptive, each activity must be able to keep track of relevant factors that arise in other activities and to interact and coordinate with those activities as necessary. These interactions are not shown in the above figures, since they are so ubiquitous that they would make the diagrams unreadable; however, Adaptive Decisionmaking provides specific mechanisms to support such interaction, as elaborated in the following discussions of the Cyclic process model and the DKB.

## Overview of the Cyclic process model

Adaptive Decisionmaking provides a single, generic Cyclic process model that is applicable to any activity. Since it is generic, this model does not attempt to describe the substantive work to be done by each activity; rather, it provides an iterative framework that allows each activity to perform its own work at its own pace while taking into account relevant events that occur in other activities so as to coordinate and synchronize its actions with those other activities. The generic cycle consists of eight steps, as shown in Figure 6. Not every activity will devote equal effort to each of these steps in every one of its cycles, but each cycle should address the concerns of each step as appropriate. Note that each step uses the DKB as a communications mechanism for coordinating with the ongoing cyclic processes of other activities, thereby addressing the boundary problem. Similarly, since each cycle reexamines the state of the endeavor and reevaluates its own course, monotonicity is not assumed. In addition, any step of a cycle can spawn new sub-cycles to pursue subsidiary or parallel issues: a cycle can suspend itself pending further input from these sub-cycles, or it can continue with some other aspect of its work, revisiting the pending issue in a later cycle, after its sub-cycles have returned their results via the DKB.



### **Step 1: Identify new and changed factors**

The first step of each cycle uses the DKB to identify new or changed internal or exogenous factors—such as changes in stakeholders, assumptions, constraints, decisions, and other events relevant to this activity. This step analyzes the implications of these new factors for the activity and posts the results of this analysis to the DKB for access by other relevant activities. (Throughout this discussion, “other relevant activities” include other users within the same activity. Also note that all such postings to the DKB constitute new assumptions or decisions in their own right.) Drastically new or changed exogenous factors—such as major programmatic changes or loss of funding—may lead to the termination of this activity, as indicated by the dashed arrow.

Note that the initial cycle of any activity will be somewhat different from all later cycles. In particular, the first step of any such first cycle may involve considerable research and investigation to identify all factors that are initially relevant to the activity at hand.

### **Step 2: Reevaluate priorities/risks**

The second step of each cycle uses the DKB to perform an updated assessment of all of the issues facing the activity at hand, and it prioritizes these and posts its results to the DKB for access by other relevant activities. This prioritization should be based on urgency, opportunity, and importance as well as risk. Step 2 merges the new or changed factors identified in Step 1 with any previous and ongoing concerns of the given activity, and it re-evaluates the activity’s priorities and risks in this context. Though it is fashionable to focus on risk, what is really at issue is what are the most important tasks to do next. High-priority tasks may be ones that are important or urgent or require long lead time or have the greatest short or long term impact, whether or not they are inherently risky. This step should avoid spending undue effort trying to identify the absolute highest-priority issue facing the activity. Furthermore, in many cases, the highest priority issue will still be the one that was the highest priority for the immediately preceding cycle.

Step 2 allows each activity to find its own balance between stability and responsiveness. For example, if changed factors imply that an activity should be redirected, but redirecting it would sacrifice some desired result that appears to be achievable within the next few cycles, then an activity may (tentatively) decide in this step to continue along its current path rather than redirecting itself.

### **Step 3: Choose an achievable task for this cycle**

The third step of each cycle chooses a specific, achievable task for the current cycle, which will advance the activity in some tangible and meaningful way within an acceptably short timeframe and at an acceptable cost; having chosen a task, this step then defines evaluation criteria for the task and posts the results of this step to the DKB for access by other relevant activities.

The chosen task should be achievable in the sense that there is a high degree of likelihood that it can be accomplished in a reasonably short timeframe—where a “reasonably short timeframe” is one that is short enough to be reasonably certain that relevant external factors are unlikely to

change during that timeframe, or that if they do, they can be safely ignored until the start of the next cycle. Of course, major perturbations can always invalidate this assumption, requiring a cycle to be aborted or restarted in the middle; however, such interruptions should be greatly diminished by the use of short cycles. Cycle length will vary for different activities and for different cycles within a given activity. If no appropriate task can be defined, this step may terminate the activity.

#### **Step 4: Plan this cycle**

Based on steps 1-3, step 4 plans the remainder of the current cycle of this activity, updates the activity's overall plan, and posts the revised plan for access by other relevant activities.

#### **Step 5: Obtain resources for this task**

Based on the plan developed in step 4, step 5 obtains commitments to proceed—including additional funding or other resources, if necessary—and posts relevant information about these commitments (or its failure to obtain them) for access by other relevant activities.

Obtaining such commitments may require repeating steps 1 through 5 until closure is reached. If commitment to proceed is ultimately denied, the activity may terminate or be put on hold.

#### **Step 6: Perform this cycle's task**

Step 6 does the actual work needed to perform the task defined in step 3 of this cycle. This may include subtasks such as identifying alternative approaches, selecting the most appropriate one (based on objectives, constraints, and available resources), and following the selected approach to perform the specified task. Any such subtask may be performed by spawning a new sub-cycle, which performs its own Cyclic process. In general, this step will also involve coordination with other activities, which is done via the DKB.

#### **Step 7: Assess and post results**

Step 7 first performs a precursor of the next cycle's step 1, examining any new factors external to this activity that may have invalidated or otherwise affected the work done by the current cycle. In light of this analysis and using the evaluation criteria developed in step 3, this step then assesses and evaluates the results of step 6, including any anomalies or surprises. It then analyzes the implications of those results, including the possible need to refine the overall goals, scope or structure of this activity or to terminate it, partition it into multiple processes, merge it with other activities, etc. The conclusions of this assessment, evaluation, and analysis are posted for access by other relevant activities. In addition, these results become new factors for the next cycle of this activity itself, enabling its subsequent cycles to continue or refine the task performed in the current cycle. Note that this step may terminate this activity.

## **Step 8: Synchronize and adjudicate**

Step 8 utilizes the DKB to perform any required explicit synchronization with other activities, adjudicating disputes if necessary. In order to maximize overall progress, synchronization should be performed only when strictly required, to allow each activity to proceed at its own pace. This step may therefore often be skipped, but it is crucial that each cycle allow for synchronization when it is needed.

A given Cyclic process may be terminated in several ways, as indicated by the dashed arrows in Figure 6 emanating from steps 1, 3, 5 and 7. These are principled bail-out points, where a Cyclic process may decide to terminate or restart itself if it encounters unforeseen obstacles or determines that it is no longer relevant. Step 1 checks that the assumptions underlying the given process are still sufficiently valid to perform another cycle; if this is not the case, the process will be terminated at this point. However, once this determination has been made and a new cycle has begun, it will not normally be interrupted. Step 2 takes any new and changed factors into account in deciding which issues have the highest priority, thereby ensuring that each cycle is based on the latest factors. As described in step 3, the task of each cycle is then chosen to be achievable within a timeframe that is short enough to ensure that most of the external factors that change during this timeframe can be safely ignored until the start of the next cycle. This allows each individual cycle to make progress without constantly needing to verify that its assumptions are still valid. (As pointed out above, major disruptions may still interrupt a given cycle and demand that it be aborted or restarted, but the use of short cycles should minimize such occurrences.) If step 3 cannot find a suitable task, it may suspend or terminate the process. If a commitment to proceed cannot be obtained in step 5, the process may repeat steps 1 through 5, or it may terminate or be put on hold. Finally, the assessment in step 7 of the results obtained in step 6 may lead to the conclusion that the process should be terminated, for example, if insurmountable obstacles are encountered or if the process has been invalidated by external factors that have changed since step 1 of this cycle.

The key features of the Cyclic process model are:

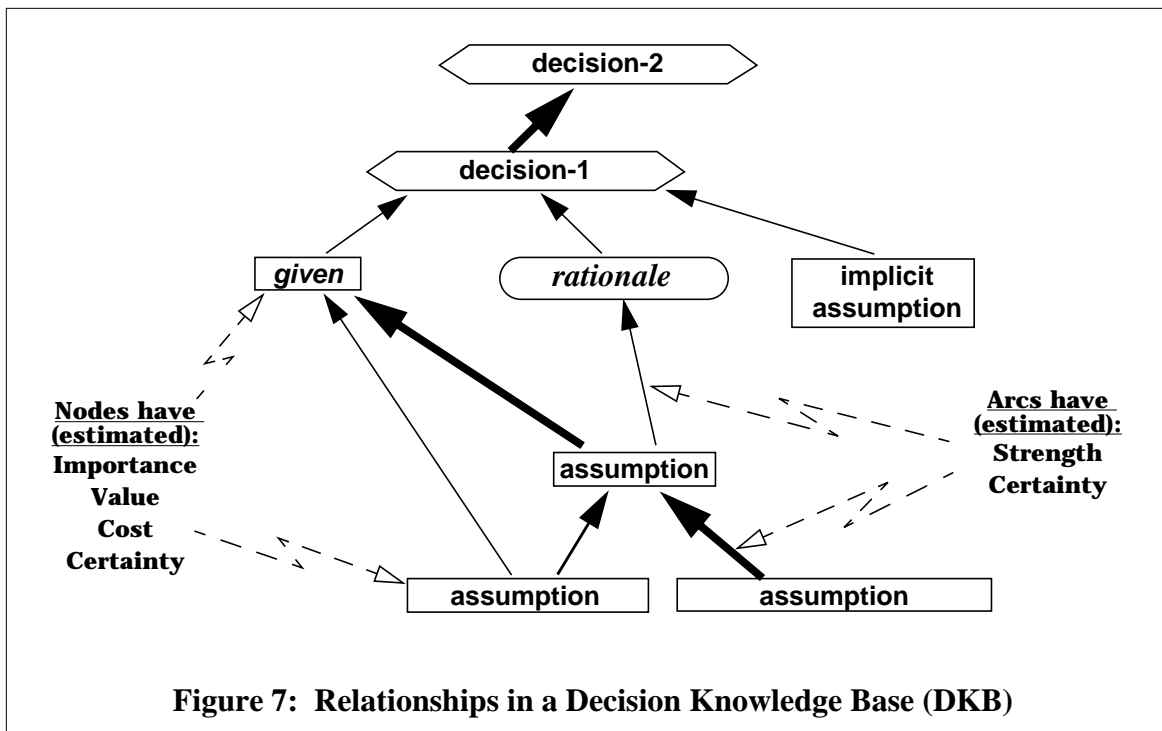
- It considers new or changed factors and reevaluates risks, assumptions, and prior decisions at the start of each cycle. Exogenous factors outside the given activity's boundary are thereby explicitly taken into account, and monotonicity is not assumed.
- Each cycle sets itself a specific, achievable task, evaluation criteria for that task, and a specific timeframe and cost estimate, derived from a prioritized assessment of all issues confronting the activity. Each cycle's timeframe is kept short enough to be normally uninterruptible. Each activity may spawn multiple Cyclic sub-processes.
- Each cycle is planned dynamically, based on the results of previous cycles (except for the first) and on any new or changed exogenous factors. Since the Cyclic process does not assume that all aspects of an activity can be predicted in advance, it avoids making crucial decisions prematurely; instead, it focuses earlier cycles on actively learning whatever is relevant to such decisions before making them. The overall goals of each activity are reassessed and refined near the end of each of its cycles (in Step 7).

## The Decision Knowledge Base (DKB)

The Decision Knowledge Base is the mechanism that enables Cyclic processes to coordinate their actions and co-evolve with each other. The Cyclic process model and the DKB depend intimately on each other. The DKB is used to record all the decisions that each activity makes, along with the rationale for each decision and its dependencies on other decisions and factors, whether internal or external to the activity. The DKB makes this knowledge available to all other activities that require it, either passively or actively. The DKB passively allows decisionmakers in all activities to consult it to answer any questions they may have about factors that concern them; but in addition, it actively notifies decisionmakers of any changes they have previously asked to hear about (using a “publish-and-subscribe” model), as well as notifying them of any changes that appear to affect them, based on its own automated reasoning mechanisms or at the instigation of anyone involved in the endeavor.

Considerable research over the past few decades on capturing and representing design rationales and argumentation—as well as work on “truth maintenance systems” that can represent constantly evolving logical implications—has laid the foundation for the DKB. Figure 7 presents a notional view of the kinds of relationships that should be represented. The DKB should help decisionmakers record and monitor decisions, assumptions, constraints, etc., detect when they change, and reexamine the rationales of any decisions that depend on them.

The DKB must be much more than simply a collection of text or data: it must be a formal representation of facts, decisions, rationales, assumptions, actors, events, and other entities and the relationships among them that are relevant to the endeavor at hand. It must allow its users to define and maintain a controlled vocabulary and formal relationships, such as dependencies, temporal



ordering, etc. The DKB must have a formal logical structure rather than relying on unstructured text, so that it can perform automated inferencing, such as following chains of dependency, sorting events in temporal order, or deriving the implications of changes. It must also provide alert mechanisms (such as publish-and-subscribe) to enable it to inform appropriate parties when something changes or occurs.

Creating and maintaining a DKB will require a special set of skills that are unlikely to be present in many organizations. Specially trained “Knowledge Representors” may be required to encode decisions, rationales, issues, and events in appropriately formal ways. Although this places a new burden on each activity, the DKB should more than repay this burden by improving each activity’s awareness of relevant changes and events outside its own scope and by enhancing its ability to make progress while reducing the effort its participants must devote to attending meetings, producing reports, etc. This is analogous to introducing disciplined accounting practices into an organization: although gathering and maintaining the information in an accounting system requires dedicated ongoing effort, it is likely to improve the organization’s understanding of its financial situation, leading to better decisions and enhanced performance. Adaptive Decisionmaking will succeed only if sufficient incentives are put in place to create and maintain a meaningful DKB.

One of the key advantages of the DKB is that it makes it easy for anyone in any activity who becomes aware of a new or changed factor to check whether it has already been represented in the DKB and, if not, to register it there. Among their other responsibilities, Knowledge Representors would be automatically alerted to new factors that have been added to the DKB by others, and they would encode in the DKB appropriate triggers for these factors, indicating which activities should be alerted to their existence and to any future changes in these factors. Although it is unrealistic to expect all new or changed factors to be immediately represented in the DKB, the Cyclic model does not rely on such immediacy: whenever such factors become apparent and are recorded in the DKB, they will be caught in Step 1 of some future cycle.

No automated system can be expected to detect changes in the real world that may affect assumptions or decisions represented within an endeavor. Human effort will therefore be required to monitor when such assumptions are violated or external decisions have changed. This is especially true for changed factors that are external to an activity. Although truth maintenance techniques and other inference methods may be able to help track the implications of such changes (by identifying dependent decisions that may change as a result), they cannot be expected to identify changes in external factors such as the market, funding climate, political context, etc., any of which may have serious implications for an endeavor. While most endeavors perform such monitoring to some extent, they generally do so in an ad hoc manner and do not typically formalize the process or use it to infer the implications of changes. Adaptive Decisionmaking requires that the results of such human monitoring and assessment of changed external factors be formally represented in the DKB so that they become available to all activities within the endeavor.

Similarly, once a changed assumption or decision is detected, its impact on all decisions that depend on it must be evaluated. This requires understanding the rationale for each such decision, to see if the change implies that that decision should itself be changed. It is not yet feasible to fully automate the reasoning required to reexamine the rationale for each decision that is potentially affected by a change. Although automated inference mechanisms may be able to perform some of

this analysis, human expertise and judgment will be required in most such cases for the foreseeable future. Finally, once the impact of a changed assumption or decision is assessed, human intelligence must be brought to bear to decide what to do on the basis of this assessment.

Therefore, although the formal representation and automated reasoning capabilities of Adaptive Decisionmaking should greatly facilitate human assessment and decisionmaking, they cannot be expected to replace them. Intelligent, well-informed people will always be required to make major decisions. Nevertheless, Adaptive Decisionmaking should act as a significant enabler for such decisionmakers, providing them with the kinds of information they need to act effectively.

### **Key advantages of Adaptive Decisionmaking**

There appear to be three key advantages of the Adaptive Decisionmaking paradigm. First, it facilitates adaptation and co-evolution by providing disciplined, pervasive mechanisms that enable each activity in an endeavor to maintain an awareness of factors in other activities that affect it and to adapt and revise its actions accordingly, in a potentially non-monotonic manner. It accepts the fact that external factors change continuously, and it offers a rational way of dealing with such change by revisiting assumptions and deciding how to proceed incrementally at the start of each new cycle. Furthermore, the DKB provides a direct channel for coordinating and synchronizing interdependent, co-evolving processes; this offers a more efficient and less burdensome alternative to the current widespread use of Integrated Process or Product Teams (IPTs), which consume vast amounts of person-time and travel budget.

Second, Adaptive Decisionmaking operationalizes risk management rather than just espousing it. The Cyclic process model identifies and reassesses problems, obstacles, and risks in steps 1 and 2 of every cycle. Each cycle performs a reprioritization and analysis of alternatives, and cycles are easily spawned to perform experiments whose results can inform later cycles. In addition, the Cyclic process model recognizes that knowledge is often incomplete: by encouraging short, frequent cycles, it avoids the necessity (and illusion) of planning farther ahead than current knowledge justifies. Moreover, by re-examining changed factors in step 1 of each cycle, the approach encourages decisionmakers to reevaluate previous decisions and revise or reverse those that are no longer appropriate, which further reduces risk.

Third, Adaptive Decisionmaking should greatly improve the traceability, accountability, and reproducibility of complex endeavors. Since the DKB is used to record rationales for all decisions, it can intelligently trace why complex decisions were made. Because the DKB automatically alerts decisionmakers to changes in assumptions, decisions, and other factors that they have declared to be relevant, it can be used to track what is known by which participants at any given time. The added accountability that this affords should make it easier to reproduce the processes employed by successful endeavors. Finally, the DKB records progress more dynamically and accurately than paper documents, allowing virtual documents to be derived from it (as database “views”), thereby reducing the documentation burden on an endeavor.

## **Caveats**

If Adaptive Decisionmaking is to work, decisionmaking incentives and culture must be changed. First, decisionmakers must be motivated to share information rather than hide it from each other. In many environments, this requires a significant cultural shift; in addition, information sharing must be balanced against legitimate concerns about secrecy, privacy, and security. Second, decisionmakers must be encouraged to identify problems as soon as possible rather than ignoring or denying them. And finally, decisionmakers must be rewarded for reversing decisions when appropriate rather than being penalized for having made decisions that must later be reversed. Although these changes are daunting, Adaptive Decisionmaking itself may help enable them: for example, the DKB might be used to automatically “score” how well individual decisionmakers share information, identify problems, and revise decisions that have become inappropriate.

Significant resources are needed to encode and maintain a DKB. As noted above, this probably requires specialized “Knowledge Representors” who are comfortable working in the necessary logical formalism. Just as the introduction of a formal accounting system may appear to place a new burden on an organization, Adaptive Decisionmaking may appear to complicate endeavors; but in practice, it should merely make explicit how complex they already are, while offering concrete mechanisms to simplify them.

## **Status**

Adaptive Decisionmaking has been elaborated for use in the acquisition of military systems of systems (SoSs) for the Army [3], but it has not yet been tried out in a real or pilot endeavor. As part of the Army effort, a prototype interactive tool has been developed, which leads a user through the eight steps of a cycle, each step utilizing a rudimentary DKB, implemented as a relational database. Although this prototype tool is far from providing a fully operational Adaptive Decisionmaking environment, it is flexible enough to serve as a starting point for implementing these ideas in nearly any application domain. Nevertheless, both the Cyclic process model and the DKB must be tailored to each new domain and each specific endeavor, in order to represent the appropriate activities and decision factors for that domain and endeavor.

A second prototype effort involved the creation of a stand-alone DKB to represent and analyze the history of decisions and events in a large-scale acquisition program. This DKB was used to encode knowledge about entities such as persons, projects, organizations, offices, documents, roles, scenarios, action items, concepts, facts, alternatives, assumptions, decisions, constraints, events, and tasks, as well as the relationships among these entities. The resulting knowledge base facilitated the analysis of the acquisition program’s history. For example, the DKB was able to produce “closures” such as the set of all facts, assumptions, etc. that supported a given conjecture or decision. In addition, the DKB made it easy to identify “critical nodes” (i.e., those that appeared on the greatest number of chains of inference supporting or refuting a given decision) or any facts, assumptions, etc. that indirectly implied conflicting or inconsistent conclusions. The DKB also enabled dynamic analysis such as seeing the effects of adding new facts or assumptions or modifying previous ones (i.e., performing “What-ifs?”). Finally, the DKB made it possible to view program history at any time in the past, i.e., what was implied when, what was known when—and

by whom, what was decided when, etc. Although this effort was discontinued prior to completion due to insufficient funding, it nevertheless suggested the feasibility and utility of the approach.

Adaptive Decisionmaking should be applicable in either of the above two modes, i.e., as a complete decisionmaking paradigm (using both the Cyclic process model and the DKB) or as an analytic tool (using a stand-alone DKB). The next phase of this work should be to identify endeavors that are suitable to serve as testing grounds for this promising set of ideas.

## **Conclusion**

Adaptive Decisionmaking is a radically new paradigm for conducting and coordinating the multiple co-evolving activities that characterize many complex endeavors. Its combination of a generic, tailorable Cyclic process model and a Decision Knowledge Base that records and communicates assumptions, decisions, rationales, and other factors across all of an endeavor's activities should enable the endeavor to make incremental—though possibly non-monotonic—progress, while maintaining unprecedented levels of traceability and accountability and effectively coping with dynamism and other sources of risk.

## REFERENCES

1. Boehm, Barry, "A Spiral Model of Software Development and Enhancement," *Computer*, Vol. 21, No. 5, May 1988, pp. 61-72.
2. Boehm, Barry, Alexander Egyed, Julie Kwan, Dan Port, Archita Shah, and Ray Madachy, "Using the WinWin Spiral Model: A Case Study," *IEEE Computer*, July 1998.
3. Rothenberg, J., E. Harris, I. Kameny, *Adaptive Acquisition: Improving the Acquisition of Army Software-Intensive Systems of Systems*, The RAND Corporation, MR-1813-A (forthcoming).