Chapter 7

Collective Strategy

In chapter 6, three types of processes were discussed that are used by collaborative groups to develop the contents of the collective long-term memory. These included computer-, group-, and conference-mediated cognition cycles, comprised of short sequences of social and/or cognitive actions interspersed with evaluative tests. The emphasis there was on identifying the underlying architectural form of these processes and to illustrate that form through example cycles. Each process or type of process was treated separately, and no attempt was made to consider the relationship of one process to another as they occur over time and within the flow of a group's work. In this chapter, I look at patterns among processes that result from and reveal the strategies groups use to accomplish their goals.

This distinction between the structural form of processes and their situated occurrences is similar to the distinction computational linguists make between types and tokens in language. *Types* refers to the words in a vocabulary for a given language; *tokens* refers to individual occurrences of those words in sentences or other expression generated by a speaker or writer of that language (Herdan, 1960). For example, the unique configuration of letters — *c a t* — denotes the word type, *cat*, whereas its occurrence in the sentence, "The *cat* scared the dog" represents a token or instance of the type. Thus, there may be many tokens of a given word type in a particular sample of text.

The set of collaborative behaviors known to a group is analogous to the set of words that constitute a vocabulary; an instance of one of those behaviors within a group's flow of work is analogous to a token within a particular expression. Just as words are not used randomly to communicate an idea, but in coherent expressions, so behaviors within a group do not occur randomly, but are composed into coherent, purposeful sequences.

The issue, then, is to identify the patterns of behavior that occur in collaborative groups that give coherence to the actions of their individual members and, in turn, to the artifacts they build.[1] How are coherent sequences of actions generated? How can we, as researchers, recognize them and understand their effects? These questions go to the heart of collaborative work. We form intellectual groups to increase the pool of effort available for tasks too large to be done by one person and to increase the pool of expertise for tasks that transcend the knowledge and skills of any single individual. Consequently, groups must be able to decompose tasks into separate subtasks that can be done by individual members or teams; but groups must also assemble the separate pieces of work that result from decomposition into a coherent whole. If the synthesized artifact is to be coherent, the group must, in general, work coherently. Good work does not happen by accident, at least not for large projects. To argue otherwise would be to argue that we should be able to routinely throw paint at a canvas and expect to produce artistically valid paintings — theoretically possible, but not very likely.

Coherence is not, of course, a black or white matter. All groups exhibit at least a minimum degree of coherence; otherwise, they would not be regarded as a group in the first place. Rather, it is a matter of degree. It is reasonable to expect that as groups exhibit more or less coherence, they also work more or less efficiently and produce higher or lower quality products. If this hypothesis turns out to be true, it would indicate that technology and training that increase coherence in groups will also increase their productivity. Consequently, it is important that we develop the tools and techniques needed to observe collective behavior and to identify patterns within it.

A number of factors can help groups work more coherently. These include standard procedures, prescribed design and development methodologies, required milestones and design documents, and the organizational culture, itself, with its projected expectations of behavior. These instruments can be effective for routine tasks, but they work less well for tasks that are unusual or that are particularly complex. For these situations, groups must rely on more flexible means. One term for a flexible, but coherent approach to a task is

---

[1] I use the term *coherence* with regard to collaborative behavior to mean that the individual actions of a member or of a group contribute in a constructive and orderly way to achieving a desired goal. With regard to the artifact, coherence refers to the orderly and logical relationship of its parts.

strategy. Because this discussion is primarily concerned with groups doing nonroutine tasks for limited periods of time, rather than with ongoing organizations with established procedures, strategy is particularly important concept.

In the remainder of this chapter, I consider strategy as it applies to collaborative groups and, thus, to a concept of collective intelligence. I look, first, at the general concept of strategy. After that, I discuss strategy in individual work and then extend the discussion to groups. The goal of this last section is to identify a set of constructs and a methodology that can be used to recognize and understand strategic behavior in groups.

Concept of Strategy

Strategy is both a familiar and an elusive concept. We all have at least an intuitive sense of strategy, but it is hard to define the term precisely. One reason is that it is used to refer to at least three different things. *Abstract strategy* exists as a set of rules, heuristics, visual images, or other forms that can be applied to a large number of specific situations. Frequently this type of strategy is associated with a particular type of task, such as writing or programming; thus, one has a strategy for planning and writing a document or planning and writing a computer program. People who have developed an abstract strategy for a task usually have the sense that they will know how to perform that task when an occasion arises, and they are likely to have a ready set of procedures they can draw on to do so. Thus, although abstract strategy is often task specific, it is also general in the sense that it is not restricted to any particular instance of its use.

Although individuals or groups may develop abstract strategies on their own, frequently particular strategies are taught as part of teaching a skill. For example, many writing instructors teach a strategy called the *stages model*; it advocates that writers follow a particular sequence of actions — planning, followed by drafting, followed by editing. A similar approach to computer programming is called the *waterfall model*. Thus, abstract strategy is often prescriptive, indicating a particular procedure that one is urged to follow. Both the stages and waterfall strategies have been criticized as being too rigid (i.e., too linear), reflecting what an ideal writer or programmer might do, but not what most writers and programmers actually do (Hayes & Flower, 1980; Boehm, 1988). But this need not be the case. Abstract strategies can exist as flexible models that provide the person or group with varied sets of options from which to select. I will illustrate this form of strategic model in the next section by describing a strategy for writing that is based on a general graph structure, rather than a strict linear sequence.

A second concept of strategy arises when an abstract strategy is applied to a given set of circumstances, for example, when a person sets out to write a particular document or a particular program. These events frequently generate a concrete product — a *plan of action* — that records the result of the strategic process. Because a plan describes particular sequences of actions whose completion would accomplish the task, we sometimes refer to the plan, itself, as a strategy. An outline for a document is an example of a plan of action, because writing all the various sections identified in the outline will, ultimately, produce the document as a whole. PERT charts and CPM diagrams are more temporally oriented schema and can describe parallel sequences of actions that can go on independently.

A third concept of strategy refers to *patterns of behavior* exhibited by an individual or group. Imagine looking down on someone carrying out a task. The person's behavior may at first be incomprehensible, but, in time, we would begin to recognize patterns and steps in his or her behavior. When this happen, we might describe those patterns in terms of the strategy we infer the person to be following. For example, chess matches between expert players are often described this way when a player is thought to be following a well-known strategy, such as Ruy Lopez or the Sicilian Defense.

If we step back and look at all three concepts of strategy, we can see that they are closely related. Abstract strategy applied to a given task produces concrete intermediate products, that is, a plan of action. Abstract strategy also produces patterns in the observable behavior of a group and its individual members. Thus, understanding the abstract strategies collaborative groups use is key to understanding the behavior of groups and the ways they go about constructing artifacts.

The information flow model illustrated in Fig. 2.1 made distinctions among tangible, intangible and ephemeral forms of information. Tangible products were further divided into target and instrumental, whereas intangible knowledge was divided into private and shared knowledge. Differences in the behavior of groups can be characterized in terms of differences in the information products they produce. For example, some individuals or groups will routinely develop particular instrumental products before they develop target products; others will bypass or spend less time on instrumental products and move more directly to target products. Thus, different paths through the framework point to different strategies being followed. As we try to draw an individual's or a group's strategy into sharper focus, we should look closely at the various transformations suggested by the flow model and at the evolving artifact in which those changes are visible.

The distinction between abstract and observed strategy is similar to the distinction between language competence and language behavior. Individuals are thought to possess a vocabulary of words, a semantically structured memory keyed by words in the vocabulary, and a grammar. When confronted by particular situations and needs, they draw on these constructs to generate specific expressions that address those needs. We can imagine an analogous concept of strategic competence. An individual or group possesses a vocabulary of actions, a long-term memory that stores information about the task content domain, and a strategic grammar. When confronted with a particular task or problem, they generate specific sequences of behaviors that address those needs. Although we seem to be born with the basic apparatus that underlies linguistic competence, we must develop the specific knowledge that comprise the vocabulary, memory contents, and grammar that enable it. Similarly, we are probably born with basic problem-solving and planning apparatus, but we develop specific methods and behaviors and learn how to use them through experience and training.

In the rest of this chapter, I consider how we might go about uncovering the strategic grammar that underlies coherent collaborative behavior. In doing so, I first consider strategy with respect to individuals, then groups.

Implicit in this discussion is both a paradigm for research and a long-term agenda. As emphasized in chapter 4, I suggest that we begin with observable data and concentrate, first, on building analytic and descriptive models of strategy. Expressed as grammars, those models will enable us to parse behaviors and produce structural descriptions of them. Those descriptions, in turn, will enable us to make comparisons between individuals working alone and within groups, to test hypotheses about what does or does not affect strategy, and to see whether or not differences in strategy make a difference in productivity or quality of work. Thus, valid analytic models would produce a number of practical benefits. Over time, they should also account for finer and finer grains of behavior and incorporate an increasing number of situational factors. That detailed knowledge should eventually provide the basis from which to infer the mechanisms that underlie observable behaviors, leading to generative models. Generative models should, in turn, lead to detailed predictive studies. Without a prior base of knowledge built from studies that use analytic models, predictive models are likely to be able to address only small, disjointed behaviors, with no realistic prospect of ever getting to the really hard and interesting problems found in the work of actual collaborative groups.

Individuals

I begin with individual strategy for two reasons. First, individual work comprises a large fraction of most collaborative projects. Second, although models of individual strategy are not simple, they are more tractable than detailed models of collective strategy will be. If we can build the first, we may be able to extrapolate from them to handle the more complex strategies of groups.

The discussion should be understood within the terms and constructs introduced earlier in chapters 4, 5, and 6. I assume that a complex conceptual construction task is being carried out by an individual working in close conjunction with a computer system, preferably one designed to amplify the user's intelligence with respect to the given task (or set of similar tasks). Consequently, thinking will occur largely in terms of a succession of computer-mediated cognition cycles. The strategy followed by this user can be described in terms of a strategy that applies to an individual processor for tangible knowledge. Because that processor operates in close conjunction with the artifact that serves as a form of long-term memory, the results of the user's strategy will be visible in the succession of small-grain changes made to the artifact. Thus, the primary issue is uncovering patterns that occur within the sequence of operations that produce those changes and the rules and other factors that are responsible for them.

Because the discussion is oriented toward analytic or descriptive models, we must, first, consider the *data* — the sources and types that reflect strategic behavior and that will be analyzed by the model. This is important because the data represent all of the information available to the model with respect to a given instance of a task. Second, we must consider the analytic *framework* that will be used and the terms and categories it provides. Third, we must consider the *formalism* in which the model will be expressed.

To make the discussion concrete, I use expository writing as the illustrative task. Thus, the discussion is oriented toward the strategies writers use to plan and write technical or scientific documents. As the example system, I use the Writing Environment (WE) described in chapter 4. The discussion also draws on the mode-based model of writing discussed there. Focusing on writing strategy, rather than strategy in general, is a convenience; the approach and concepts apply to a broad range of conceptual construction tasks.

Data

Because the data strongly influence what the researcher can infer about the user's cognitive behavior, choosing the right kind of *data* — at the right level of granularity and with the right descriptive parameters — is crucial. Several options are available. In this section, I briefly discuss four kinds of structured data, or *protocols*, that have been used to study strategy as well as other cognitive and human-computer interaction issues. These include *concurrent think-aloud, events, video,* and *action protocols.*

Concurrent Think-Aloud Protocols. This method was developed by Newell, Simon, and others at Carnegie Mellon University during the 1960s to study complex, problem-solving behaviors (Newell & Simon, 1972). It produces a written record of subjects' trains of thought based on the subjects' own verbalizations of their thinking while they perform the task being investigated. Tasks that have been studied using think-aloud data include writing documents and computer programs, solving arithmetic problems, assembling physical devices, playing chess, and, more recently, using various computer systems. Under laboratory conditions, subjects are prompted to continuously narrate their thoughts; under naturalistic conditions, such as a subject writing a paper at home, prompting is impractical. Consequently, think-aloud protocols often differ significantly in the levels of detail reported by different subjects under different conditions.

Think-aloud protocols have been debated on several grounds (Nisbett and Wilson, 1977; Ericsson & Simon, 1980). Subjects do not always know what they are thinking. The act of thinking aloud can, under some circumstances, distort thinking. The data are hard to encode in a consistent manner and doing so requires considerable time and effort, making them expensive. And, as noted previously, subjects produce protocols that differ significantly from one another in completeness and detail. In spite of these problems, think-aloud protocols can, when used judiciously, provide richly detailed insights into a person's motives and intentions not available from other types of data.

Events Protocols. To address problems of cognitive interference and the laboriousness of manual preparation associated with verbal protocols, some researchers have used the computer system itself to collect data. For command-driven systems, data is collected in the form of keystrokes (Card, Moran, & Newell, 1983). For direct manipulation systems, protocols can be collected for a wider variety of input and display events, such as movements of the mouse and graphic information displayed on the screen. So-called X-Windows protocols are an example of this type of data. I refer to both keystroke and low-level direct manipulation protocols as events protocols.

Events protocols solve several problems raised by think-aloud methods, but they also raise others. Because they are recorded by the system, they do not require initial manual input. However, because the data are so fine-grained, it is difficult to infer what the keystrokes or mouse events add up to in terms of system commands, and inferring their effects on the conceptual structures being developed by the users may be impossible. Consequently, before conceptual or cognitive analyses can be done, events data must be manually categorized by a trained analyst, raising problems of consistency and costs.

Video Protocols. This method has been used alone as a supplement to both think-aloud and events protocols. Subjects are video taped as they perform a task. For studies involving a computer, these data can show what a person is doing when not thinking aloud or interacting with the computer. They can also show what is displayed on the screen — information not available through think-aloud or keystroke records. Thus, a major benefit of video data is that users' behaviors are captured in a more complete context (Mackay, Guindon, Mantei, Suchman, & Tatar, 1988). However, video protocols require extensive analysis and coding, raising similar problems of consistency, time, and labor.

Action Protocols. A fourth type of protocol data records the user's actions with respect to a system's data objects and control functions, rather than low-level keystrokes or mouse events (Smith, Smith & Kupstas, 1993). Action protocols record movements of the mouse from one window to another — rather than from one screen coordinate to another — and selections of identifiable data objects — versus simply reporting that the mouse was clicked at a particular screen location. They also report menu options selected and character strings typed in as names or labels for objects. Thus, action data are far richer than events data in terms of the semantics of the task.

Currently, the approach is limited to studies that involve computer systems for which the researcher has access to the source code or can persuade those who do to make the necessary modifications. Nevertheless, action protocols offer many advantages over other forms of data. Like events protocols, they solve the problem of cognitive interference raised by think-aloud methods by being passive and unobtrusive; and because they are recorded in machine-readable form, they eliminate the need for manual transcription, required of both think-aloud and video protocols. But unlike events protocols, action protocols provide data in which content objects are identified, enabling analyses at a conceptual or cognitive level without manual coding. Because they represent a more abstract level of activity with respect to the task, they are an order of magnitude less numerous than events protocols. Although not a panacea, they offer a number of advantages that make them especially well-suited for studies of strategy. Consequently, they figure prominently in the discussions that follow.

Action protocols are normally recorded one action per record. Each record includes several parts. A symbol identifies which of a designated set of actions has occurred; for example, *create_node* might indicate that the user has created a new node on the display. Each record also includes the time the action begins, its duration, and parameters relevant for the particular action, such as the identification number of the node and its location on the display, or the character string typed in. Thus, different actions have different parameters. For the Writing Environment, the set of designated actions numbers approximately 50. We can think of this set of identifiable actions as a vocabulary of action *types* from which individuals working with the system can select.

As a users works with a system that is recording action protocols, he or she generates a sequence of action records. Each action within the flow of work can be viewed as a *token* of its corresponding action type. In a language, words are normally organized into sentences or expressions that serve some purpose. Similarly, actions are normally used to accomplish some purpose with respect to the task. If we are to understand a user's strategy, we must understand what these sequences of actions are telling us about the user's intentions with respect to the intellectual construct he or she is creating. One way to approach this problem is to build a model of strategy in the form of a grammar that can be used to parse expressions in the language of action protocols, just as conventional grammars can be used to parse expressions in English, French, or Pascal. In the sections that follow, I describe how this can be done.

Framework

To build a model of strategy, we must first decide on an *analytic framework*. This is an important decision because it includes the terms and categories in which the model will be defined. Ultimately, this set of categories is arbitrary, because one can always construct alternative categories with which to describe a given set or type of data. However, because the framework in which the model is built provides the mental constructs in which we think and talk about strategy, it should be given some care. First, the categories within the framework should denote the "seams" in the phenomena to be considered, the "natural" breaks in the data. Second, we should think of these categories as falling within different levels of abstraction. If the model takes the form of a grammar, the categories will constitute the nonterminal symbols in the grammar, analogous to terms such as *noun* and *noun phrase* in natural language grammars. Thus, lower level symbols, denoting fine-grained behaviors, will be composed to form higher level symbols, denoting more extensive behaviors; conversely, higher level symbols will be decomposed into finer grained symbols. Finally, because category names are unlikely to be original, they will carry associations with other models; we should be sure that such associations are intended and reinforce desired relationships.

In chapter 4, several analytic frameworks were discussed. The GOMS framework included four components: goals, operations, methods, and selection rules. As emphasized there, the selection rule component requires that GOMS models be predictive models. They are capable of handling short, independent sequences of actions that occur over durations of a few seconds to 1 or 2 minutes; but they are not currently capable of handling sequences that take place over extended periods of time

Consequently, I discussed an alternative framework, based on cognitive modes, that is better suited for analyzing and characterizing strategic behaviors that extend over several hours. Conceptual behavior is viewed as a sequence of cognitive modes, each composed of different combinations of goals, products, processes, and constraints. I do not repeat that discussion here, but for the remainder of this section, I use the categories included in the modes framework as the building blocks with which to construct models of strategy.

Formalisms

The third major component needed to build strategic models is a *formalism* in which to describe relationships among the terms and categories of the analytic framework. Models have been developed in many different formalisms, including algebraic expressions, set theory, production rules, and graph theory. Although the choice of formalism is ultimately arbitrary, some are inherently more powerful than others, and some are better suited for dealing with particular types of data and the phenomena they represent. I focus on strategic models that take the form of formal grammars. However, a number of different options are available for expressing grammar rules. I discuss two: *production rules* and a graph-based form, called *augmented transition networks* (ATN). My colleagues and I have developed several different strategic models for writing using these formalisms, which I draw on for illustrations. But, I want to emphasize, the formalisms are general, and other models could be built within these same formalisms by defining alternative sets of rules.

WE Production Rule Grammar. Our first grammar was expressed as a set of production rules. Production rules denote a relationship between a single symbol on the left side of the expression and a pattern of one or more symbols on the right (Newell & Simon, 1972). The pattern may be a sequence of symbols or a Boolean expression, such as an *and* or an *or* expression. A rule is normally interpreted as a mapping between the two sides. If it is used in a recognition procedure, an occurrence of the pattern on the right within an expression can be replaced by the single symbol on the left. Conversely, if it is used in a generative procedure, the symbol on the left can be expanded and replaced by the more detailed pattern on the right. For example, the rule, $S ::= NP + VP$, says that the symbol, S, can be replaced by the pattern, $NP + VP$. More detailed rules might map NP onto expressions that involve nouns — the class of which might be denoted N — and, in turn, N might be mapped onto a set of specific nouns — such as *dog* or *cat*.

When used with a parsing program, the rules map specific expressions onto a parse tree that denotes the abstract grammatical structure of the expression. Conversely, when used with a generative program, the rules produce valid expressions in the language defined by the grammar. One way of looking at the parse tree is that it represents the sequence of rules that could be used, starting with the most abstract symbol in the grammar, to generate the expression that was parsed.

Production rule grammars are context free. Because studies involving computers must often take into account the context of a given behavior, the formalism must be extended if it is to provide this capability. The WE production rule grammar included some half-dozen functions that recognize contextual relationships among the data. For example, they recognize the particular type of graph object involved in an action, such as disconnected sets of nodes versus connected graphs; they also recognize distance relationships between data objects, such as the distance between two nodes on the screen. Consequently, the same user action may be interpreted differently under different conditions. For example, the WE grammar infers that a different cognitive operation has occurred when a user crates and places a new node close to an existing node on the screen, versus placing it in a location with no other nodes nearby. (Smith et al., 1989).

The nonterminal symbols in the WE production rule grammar form a hierarchy. The start symbol is a *session*. It is composed of a series of cognitive modes in which a sequence of cognitive processes are used to produce particular cognitive products or changes to products. The top four levels of the grammar describe the cognitive behavior of the user from a modal point of view and are general with respect to the task.

The bottom two levels link the cognitive portion of the model to a specific writing system — WE. To transfer changes in cognitive products, which take place in the head, to the representation of those products, within the computer, the user performs a series of system operations, each composed of one or a short sequence of system actions. Actions are the protocol data recorded by the system, as described previously, and constitute the terminal symbols of the grammar. Operations represent a more general view of the user interface. For example, the *create_node* operation requires four user actions: designating a screen location with the mouse, highlighting and then selecting the create node option from a pop-up menu, and typing a name or label for the node.

The WE production rule grammar is illustrated in Fig. 7.1. The figure is divided into two parts. At the bottom is a partial representation of the user's display. At the top of the screen is a portion of the grammar's parse of the user's actions that produced the display. The illustration is focused on the actions that resulted in the creation of the node labeled *n*. Because the protocol for the entire session is available, we can look down on it and see both forward and backward in time. In this case, the user has recently created nodes *n-1* and *n-2*, in two previous operations, and will soon create nodes *n+1* through *n+4*. Five of these seven nodes are interpreted by the grammar as forming a *cluster*, whereas two — nodes *n+3* and *n+4* — are interpreted as "solo" nodes. This distinction is based on the relative distances between the various nodes, determined by the grammar's context recognition functions.

In the portion of the parse tree shown at the top of the figure, we can see the actions that resulted in the creation of node *n* as well as the grammars interpretation of the user's cognitive behavior that resulted in nodes *n+1* through *n+4*. To create node *n* requires the four system actions, shown at the lowest level of the parse tree — opening a menu, highlighting and selecting the create node option, and typing a name for the new node (i.e., "Introduction"). These actions comprise the system operation, *create_node*, shown in the second level of the tree.

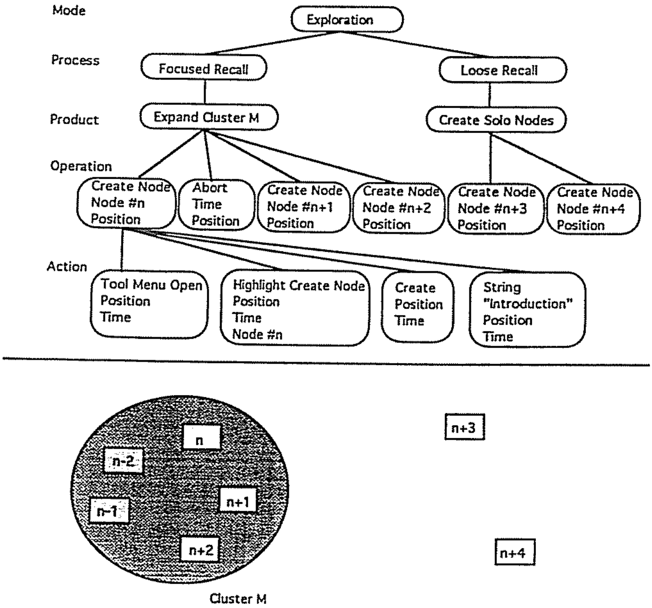

Fig. 7.1. Illustration of Writing Environment (WE) production rule grammar, showing a sequence of actions and their parsed interpretation by the grammar. Note that the same *create_node* operation can indicate different cognitive processes, depending on context or position of the node relative to other nodes.

Within this particular context, this operation is interpreted as an indication that the cognitive product, *Cluster M*, has been expanded (level 3), as a result of a *focused recall* cognitive process (level 4). We can also see that the user's activities that produce nodes *n+1* and *n+2* are interpreted similarly. However, the operations that produced nodes *n+3* and *n+4*, by virtue of their locations on the screen, are interpreted differently — they indicate that a *loose recall* cognitive process has occurred. Thus, the same system operation may be interpreted differently by the grammar, supplemented by the recognition functions, depending on contextual factors such as the location on the screen where a particular operation takes place. Finally, all of the activities involved in creating nodes *n-2* through *n+4* are interpreted by the grammar as occurring within an *exploration* cognitive mode. The illustration stops at this level; the overall session is comprised of a number of such modes.

The WE production rule grammar has been used to analyze some 5 user protocols, each representing 2- to 3-hour sessions recorded in our laboratory under seminaturalistic conditions. Results of these studies have been described in Lansman and Smith (1993); Smith and Lansman (1992); Lansman (1991); and Smith and Lansman (1989). Figure 7.2 shows the strategic behavior of two subjects as they planned, wrote, and revised short, 3–4 page documents. On the left side of both displays is a taxonomy of actions defined by the researcher; actions have been organized into three categories — *explore, organize,* and *write.* On the right are frequency distributions for each category as well as each operation within a category. Time extends from left to right. Individual operations are represented by tick marks at the time each begins, and a horizontal "tail" on the tick mark indicates its duration.

The computer tool used by researchers to produce these displays allows them to divide a user's session into segments, indicated by vertical lines that extend from top to bottom in each display. In the figure, Subject 1's session has been divided into four large segments, and Subject 2's display has been divided into two segments. Histogram at the bottom of each display show the total number of operations for each category of operations for each such segment.

The data, grammar, and display tools let us literally see users' strategies as they are reflected in patterns in their cognitive behavior that extend over several hours. Subject 1 first created a group of exploratory products. He then constructed the top of his tree and wrote blocks of text for each node in the tree. Finally, he went back and filled out the bottom of the tree and wrote text for those nodes. Thus, the strategy followed by this first user is almost a classic stages or waterfall model in which the one first plans, then writes, and finally, edits the draft.

This strategy is markedly different from that of Subject 2, who characteristically created a node, in exploration mode, and then immediately wrote a block of text for that node, in writing mode. This writer constantly moved back and forth between the two modes, rather than spending substantial time in one before moving to the other. The same pattern predominated in the second part of the session, but between organizing and writing modes.

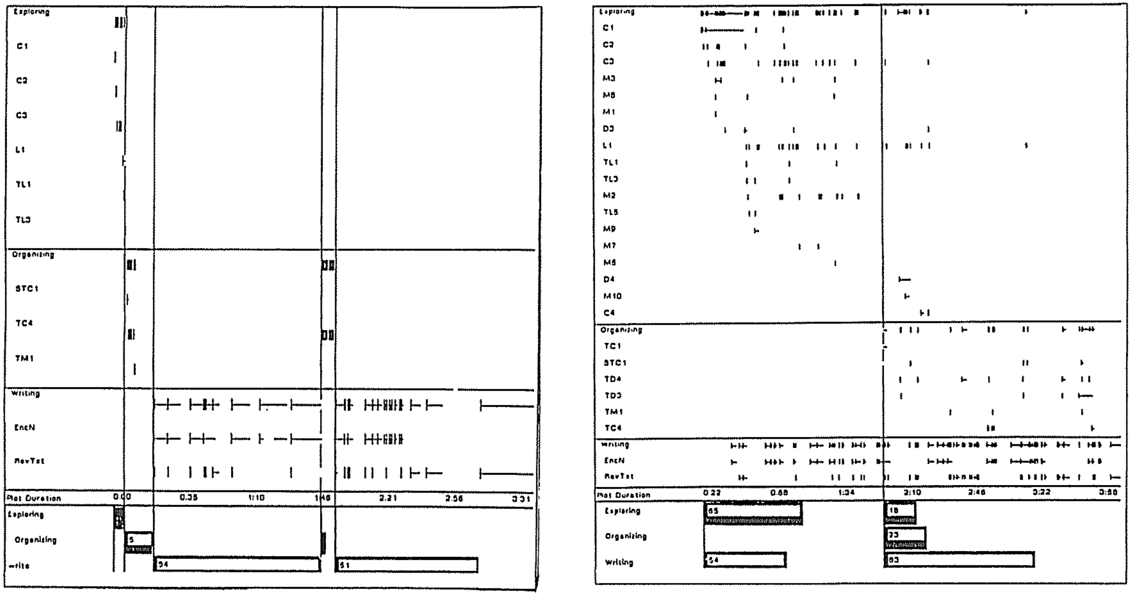

Fig. 7.2. Two writers' strategies. One shows a pattern of sustained planning followed by sustained writing; the other shows a continuous movement back and forth between planning and writing.

One possible explanation for these very different strategies is that the first subject may be more abstract in his or her thinking whereas the second is more verbal. The first strongly separated brainstorming and planning activities from linguistic encoding; for the second, brainstorming and planning were consistently interleaved with actual writing. Additional research is needed to understand these differences in strategy and their causes and effects. The point to be emphasized here is the access to users' strategies provided by this methodology and its associated tools.

WE ATN Grammar. The production rule grammar described previously works satisfactorily, but the architecture of the computer program in which it was implemented is awkward, making refinement of the model difficult. In addition, the recognition functions that provide context sensitivity are not well integrated into the grammar. To solve these problems and to extend our thinking about strategic behavior, we have defined a second grammar, using the Augmented Transition Network formalism (Woods, 1970). Although this formalism has been used primarily in natural language understanding systems, ATN grammars are well-suited for analyzing human-computer interaction data.

An ATN grammar is expressed as a set of graph structures whose nodes represent *states* and whose links represent *transitions*. The states are arbitrary and abstract, whereas the descriptive labels attached to the links/transitions denote the nonterminal symbols of the grammar. *Tests* may be attached to the transitions, providing a convenient mechanism for checking contextual parameters, and values may be written to *registers* by one part of the grammar and later accessed by another. ATN registers, which are simply global variables, are not to be confused with computer hardware registers.

A parse is performed by a program that reads a sequence of symbols from a language and, for each symbol, traverses the ATN graph structures, checking for conditions, performing tests, and recording information in registers. A traversal of a link in one graph may also result in a call to a different graph; when the traversal of the latter graph is complete, control returns to the link/transition from which the original call was made. Of course, not all transitions will be successful, and the parser must be able to backtrack when it reaches a dead-end and try an alternative path. The parse is complete when the complete sequence of symbols has been read and the parsing program has reached a special *stop* state. After the parse has been completed, a parse tree that represents the grammatical structure for the string can be constructed from the labels along the path of successful transitions that led to the stop state.

Our decision to use ATNs was motivated by several factors. Its register and test capabilities make it convenient to incorporate tests on parameter values — such as the spatial coordinates of a node — into the rules of the grammar. This allowed us to incorporate the same functionality provided by the context recognition functions used by the production rule grammar directly into the ATN grammar. Second, because ATNs have the power of a Turing machine, no other formalism is more powerful. Thus, ATNs offer an attractive formalism for studies of strategies.

To gain a feel for the ATN model, consider Fig. 7.3. The model of writing strategy defined in the ATN grammar differs from the production rule model in several ways. Most obvious is that the ATN model includes a varying number of levels. In the production rule grammar, all sequences of actions were interpreted in terms of a six-level tree, ranging from session and modes at the top to operations and actions at the bottom. In the ATN grammar, some modes are "deeper" than others. A second distinction is that the computer system, itself, has been incorporated as a fundamental part of the task model. This is done by including the user's attention to the tool — manifest as a sequence of actions that address the system rather than the conceptual artifact being developed — as a mode of thought analogous to substantive modes — such as exploring or organizing.

At the highest level, users are assumed to be *working*, as indicated in the first graph. For whatever reason, they decide to write a document or to continue work on a document they were working on earlier. To do so, they engage a strategic mode that contains a model of the writing process. In the grammar, this is represented by beginning a traversal of the link marked *Write Document*, which generates a call to a separate ATN graph having the same label. When all activity associated with this graph is completed (including any additional calls to lower level graphs), the traversal of the link in the top graph will be completed and the grammar will stop. This pattern of descending and ascending by starting to traverse a link, calling a lower level graph, completing its traversal, and returning to the higher level graph to complete the traversal of the original link is repeated throughout the ATN grammar. Let's now follow one possible path through the grammar down to a terminal symbol.

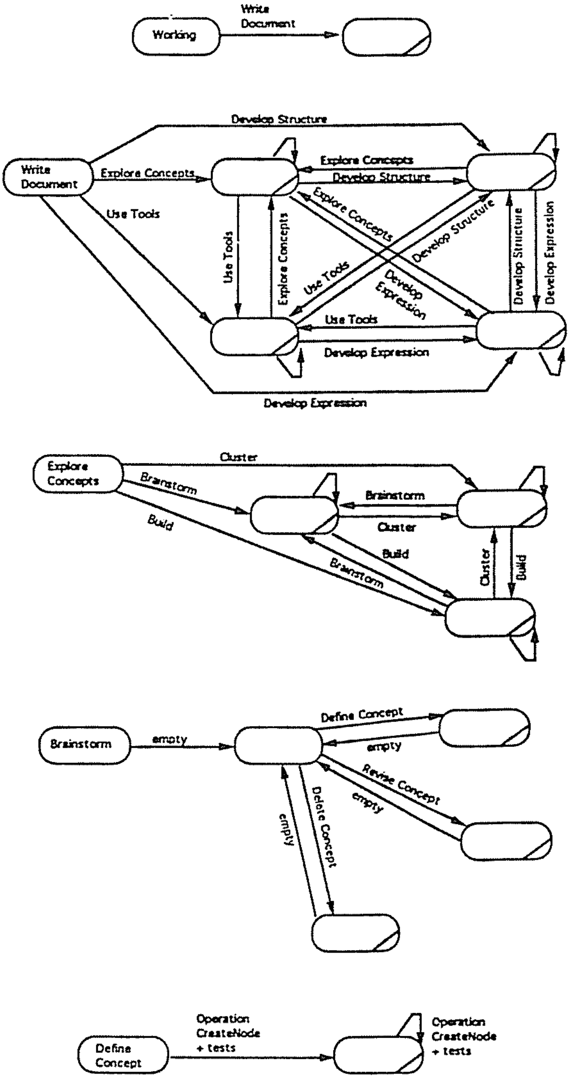

Fig. 7.3. A portion of the Writing Environment (WE) Augmented Transition Network (ATN) grammar. It shows a possible vertical path through the grammar, from initial intention to write, through brainstorming mode, to the define_concept action. The path could appear as a branch in the parse tree for a user's session in which a define_concept action occurred.

Within the context of the Write Document graph, users can explore concepts, develop the structure of the document, develop its expression, or they can consciously address the tool. Assume that our writer decides to explore concepts. This produces a call to the Explore Concepts graph. In the cognitive mode associated with this graph, one may brainstorm, build clusters of ideas, or build component structures. Each choice leads to a lower level graph for the corresponding activity. If the user decides to brainstorm, resulting in a call to that subgraph, he or she can represent a concept (define it), revise an existing concept, or discard (delete) an existing concept from further consideration. Defining a concept is viewed as a basic cognitive process in the model. As the final graph structure shows, the define concept operation and the change in the artifact it produces are subjected to tests. If the conditions are satisfied, the transition in this lowest level graph is completed. Presumably, the user then goes on to another cognitive or conceptual process that is reflected in the next action in the protocol sequence. If the test fails, the grammar backs up and tries an alternative path through the ATN network.

Basic cognitive processes, such as the define concept process, are examples of the computer-mediated cognition cycles discussed in chapter 6. Thus, the rules and tests in the grammar that recognize these elements map between basic cognitive operations, which occur in the head and produce changes in the structure of concepts active in human short-term memory, and basic system operations, which produce changes in the data structure represented in the display. The particular cmc cycle — *define_concept* — used for purposes of illustration here can be seen in Fig. 6.3. Thus, the grammar maps patterns of user behavior as they are manifest in observable system actions to low-level cognitive processes, which, in turn, are linked through several levels of abstraction to the session as a whole. As a result, high-level behaviors, which may last for as long as several hours, can be decomposed into sequences of lower level behaviors, which last for only a few seconds. Conversely, lower level behaviors can be composed to form higher level behaviors, at several levels of abstraction.

In this section, I have outlined a methodology for observing and analyzing the strategic behavior of individuals carrying out complex, conceptual construction tasks, such as writing a document. These steps included observing users' behaviors and obtaining suitable protocol data, selecting an analytic framework that includes the categories in which the model will be cast, and identifying a formalism in which to define relationships among those terms. Action protocols were selected because they provide both fine-grained data and sufficient contextual information to support a limited form of context-sensitive analysis. The cognitive mode framework was used, for reasons discussed in chapter 4. Two formalisms were discussed — production rules, supplemented by context recognition functions, and Augmented Transition Networks.

Collaborative Groups

In this section, I extend the discussion of strategy from individuals to collaborative groups. The fundamental problem faced by collaborative groups is, on the one hand, to divide their work into semiautonomous tasks so that they can take advantage of the parallel efforts and the individual expertise of its various member and, on the other hand, to synthesize their respective contributions to form a coherent whole. The problems we face in trying to study and/or understand collaborative behavior is to observe these various activities, to construct analytic frameworks that fit the data, and to relate the categories and terms of the framework to one another to form analytic models.

As in the preceding section, I assume that a complex knowledge-construction task is being carried out and that much of the work is done in conjunction with a computer system. In this case, the work is being done by a collaborative group, rather than an individual, and the group is assumed to be using a collaboration support system, such as ABC. We can gain important insights into collaborative strategy by observing groups use the system to build the artifact, similar to the approach outlined in the preceding section for an individual writing a document. However, because many group activities do not directly address the artifact, the methods we use to collect data, to analyze it, and to model strategic behavior must also address activities, such as meetings, in which intangible knowledge is developed.

The approach taken for individual strategy was to focus on the individual processor and the various computer-mediated cognition cycles that operate within it. To consider collective strategy, we must extend the discussion from individual processor to the multiple independent processors that function within a group. We must also include the processor for shared intangible knowledge and the hybrid processor. In a collaborative group, all three forms of processing will operate at one time or another and, for larger groups, often at the same time. Thus, a concept of collective strategy must recognize patterns not just within the individual instances of these processors but also in their interactions with one another.

The discussion begins with the data. The issue is collecting and managing fine-grained data comparable to that available for the individual processor but extended to the other types of processors. Second, I consider a basic framework that provides categories and terms in which to describe the social as well as conceptual behaviors of collaborative groups. Finally, I consider a formalism that can be used to build analytic models of collective strategy.

Data

The goal is to collect data over the duration of a project that will enable us to both examine the fine-grained behavior of the group and see its overall strategy. To meet these requirements, we must record the behavior of all three types of collective processors at the level of individual mediated cognition cycles. This situation is illustrated in Fig. 7.4. Three members of a group are working independently, shown on the left of the figure, while three others are working together in a computer conference, shown at the top. Meanwhile, we can see five additional members holding a meeting, at the bottom of the figure. Also shown are two observers — one observing and taking notes on the meeting, the other doing the same for the computer conference. Let's look more closely at each of these three types of activity.

Multiple Independent Processors. Group members working independently at their respective workstations, shown on the left of the figure, comprise a set of multiple independent processors. As they work with a collaboration support system — for example, the ABC system — the system records each action they perform. Thus, each users' ABC session produces a protocol similar to those described previously for the WE system. Each such protocol is identified by the group member who produced it, the time it was produced, and other

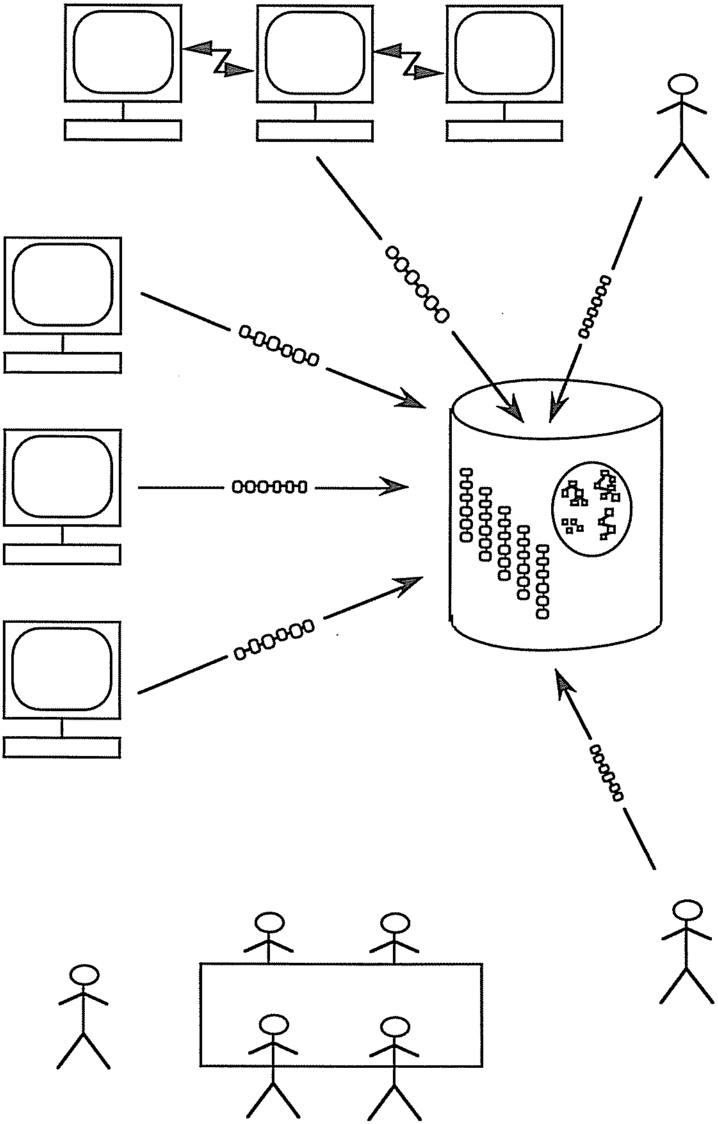

Fig. 7.4. Multiple streams of protocol data for group members working independently, in meetings, and in computer conferences.

similar information, followed by the sequence of records that identify the user's individual actions. Protocol data are stored in the same hypermedia data storage system that holds the artifact, shown at the right of the figure. Each such protocol, and the actions that comprise it, is suggested by an arrow; several prior protocols, along with the artifact, are also visible in the database.

Processor for Intangible Knowledge. Data that record actions that take place in meetings and other similar situations must be recorded and coded by a human being, preferably a trained observer who can identify different types of actions — that is, group-mediated cognition cycles — in a reliable and consistent manner (Cain & Reeves, 1993). Manual coding is needed whether the observer is actually in the room when the activity takes place, as suggested in this illustration, or views a videotape of the activity and codes it later. Individual behaviors are coded as to which of a given set of behavior types has been observed, the time the behavior takes place, and relevant parameters. This type of protocol has the same basic form as the machine-recorded protocols, but differs in the specific actions identified. Consequently, these data can also be kept in the same storage system and many of the same tools can be used with them. An example of a study based on this type of observation is described in the next section; a more thorough discussion can be found in Holland, Reeves, and Larme (1992).

Hybrid Processor. Data is also needed for the 1xn hybrid processor, shown at the top of the figure. In this example, three members of the group are participating in a computer conference in which they take turns working on the same part of the artifact. They can also talk to and see one another using supplemental voice and video channels. The protocol stream for the chairperson includes symbols that identify actions for all of the conference participants that affect the part of the artifact that is the subject of the computer conference, along with his or her actions before and after the conference. Actions of the other participants that take place in nonconferenced windows — representing individual actions that take place during the conference but are not part of the conference — are included in their respective protocol streams but not in the protocol stream for the conference.

The verbal and visual interactions that occur through supplemental communication channels are not included in the machine-recorded protocols. However, an analyst can observe the computer conference, including these channels, and take notes similar to those for meetings. These data can then be coded and entered into the database. Thus, as the figure suggests, a computer conference, when supplemented by video and/or voice channels, will (potentially) produce two protocols — one, machine-recorded; the other, manually coded observations. These two data streams can subsequently be merged or correlated with one another using the time parameter included in each record.

Thus, protocol data can be collected that include a record of all actions that affect the artifact and a substantial number of those that contribute to the development of shared intangible knowledge. Although not complete, these data are extensive and can provide a detailed understanding of a group's fine-grained behavior as well as broad, strategic patterns.

Framework

Earlier, I discussed cognitive modes as a framework in which to describe the cognitive behavior of individuals. Because the individual processor is part of both the multiple and the hybrid processors, cognitive modes are also relevant to collective behavior. However, to describe behaviors that take place in meetings and other situations where intangible knowledge is developed, we need a framework that incorporates social as well as cognitive factors. In this section, I will describe a construct, called a *mode of activity,* that does this.

Western psychology, with its emphasis on controlled experiments, has tended to focus on individual cognitive functions. Although this research has produced detailed knowledge of isolated processes, those processes have seldom been examined under real-world conditions. Thus, it has not addressed the quick oscillations between conceptual and social processes that occur in collaborative groups and lie at the heart of group-mediated cognition.

An alternative tradition, known as *activity theory,* developed in the first half of this century in the Soviet Union. Associated with Vygotsky, Leont'ev, and their followers, it assumed that mental processes are always situated in broader cultural and social contexts and should be studied within those contexts, rather than in isolation as required by controlled experiments.[2] Activity theory includes several key concepts that are useful for understanding collaboration and for building a concept of collective intelligence. These include *situated activity, mediating devices, higher* and *lower mental functions,* and the *zone of proximal development.*

Activity theory views all mental processes as being *situated,* because it asserts that individual cognition always takes place in, and is responsive to, socially created activities. Even when alone, individual thinkers typically interpret the issue at hand in relation to a mental activity learned from others. Thus, conceptualization is embedded in the culture, with respect to both the symbols and concepts that are the substance of thought as well as basic mental processes.

To explain how mental processes can be influenced by social factors, Vygotsky differentiated between what he called lower and higher mental functions. The key to this distinction lies in the role symbols play in abstract thought. Symbols function in the mental world as tools function in the physical world. Thus, symbols mediate between one's mental states and processes and one's environment. For example, the basic act of remembering, made possible by one's neural apparatus, is a "lower level" function; however, when people learn to use mediating devices as tools for remembering — such as mnemonic associations derived from their language and culture — their memory capacity is increased and they have more conscious control over memory-related processes. Thus, mnemonic-assisted remembering is a form of "higher" mental function.

Vygotsky argued that mediating devices are largely invisible, or *fossilized*, under normal circumstances, because once learned or developed, they become habitual and individuals are unaware that they are using them. Consequently, mediated cognition can best be observed when new abstract devices are being developed or when new technologies are being introduced, and before the new forms of cognitive behavior that will inevitably develop become routine. Thus, there is a narrow window of opportunity when new mediating devices can most easily be observed, often occurring as an individual or group attempts to resolve a problem or snag.

---

[2] This discussion is based on an earlier summary included in Smith et al. (1990) which, in turn, was derived from Holland and Valsiner (1987) and from Vygotsky (1962, 1978, 1987).

Finally, Vygotsky argued that before we can carry out a task by ourselves, we must first learn the skill in proximity to another person. New skills are usually learned during work episodes that involve at least one (relative) neophyte and at least one (relative) expert. As the neophyte's ability develops, the expert curtails his or her participation, leading to the development of higher mental functions in the neophyte. Vygotsky called this situation the *zone of proximal development*, or *zoped*, for short.

Let's look back at collaborative behavior through the prism of these concepts. First, activity theory asserts that all mental behavior is situated within cultural and social contexts and is affected by those contexts. For collaborative groups, this context normally includes the organization in which the group functions, the group itself, and the physical settings in its various activities take place. For example, an activity carried out in a conference room (e.g., a meeting) is different from one carried out in a hallway (e.g., a chance conversation), although both may contribute to the development of shared intangible knowledge. Consequently, because context affects behavior, the analytic framework in which collaborative behavior is to be considered should incorporate parameters that characterize the particular situation in which the activity takes place.

Second, mediating devices are crucial in both activity theory and in collaborative groups. In activity theory, the principal form of mediating device is the symbol; for collaborative groups, mediating devices are more diverse. The computer system used to develop the artifact is probably the strongest mediating device (i.e., computer-mediated cognition). However, the ideas voiced by other participants during a meeting that influence one's own thinking are another form of mediation (i.e., group-mediated cognition). Other common mediating devices include audio/visual equipment, the whiteboard, and metaphors used to explain abstract concepts. Thus, although symbols and language continue to function as mediating devices, the tools and systems used as well as the group, itself, also mediate collaborative thinking. Consequently, the analytic framework should include mediating devices as a basic category.

Third, the Vygotsky distinction between lower and higher mental functions can be applied to a group's collaborative skills. As groups learn to work together more effectively, the successive stages they go through may be considered forms of higher mental function. Indeed, the coherent, integrated behavior I have referred to as collective intelligence can be viewed as a form of *collective higher mental* *function*. Thus, it can serve as a goal groups should aim to achieve. One reason they have trouble achieving this level of performance may be because there are few, if any, "expert" groups a "novice" group can work with to develop its collaborative skills. The notion that one group can serve as a mentor for another less skilled group, creating a zone of proximal development for collaboration skills, is a training strategy worth considering.

We can now define a *mode of activity* framework that is based on the cognitive mode framework but includes several new components from activity theory. We should keep in mind that cognitive modes were defined with respect to individual cognition and, in the terms of this discussion, an individual processor. The mode of activity construct is intended to address collaborative behavior. Consequently, it must account for the multiple processor, the hybrid processor, and the processor for intangible knowledge that, together, comprise the collective processor. Because an individual processor is one part of the multiple processor, which, in turn, is a component of the collective processor, we should think of mode of activity as an extension of cognitive mode and, conversely, a cognitive mode as a subset or projection of a mode of activity.

A *mode of activity* is defined as a multilevel structure that includes configurations of six basic components: goals, products, processes, constraints, situation, and mediating devices. Let's consider this definition, first, as an extension of cognitive mode, and then, as a multilevel structure.

Like cognitive modes, modes of activity include configurations of goals, products, processes, and constraints. *Goals* represent the abstract intentions of the group. Normally, they result in the production of some tangible *product* or in the further development of shared intangible knowledge. Thus, the types of products developed and/or used by collaborative groups are more extensive than those for an individual carrying out a similar task. This expanded set of information types requires a larger set of mental *processes* to build and access them. These include computer-, conference-, and group-mediated cognition cycles. Finally, *constraints* shape the situational matrix in which specific modes function. They limit the processes that can be used in a given mode, but they also define what is possible. More important, they define the norms of behavior that give a particular mode its identifying characteristics. For example, a brainstorming mode will carry a different set of expectations than a presentation mode. Part of this difference lies in the structures of the two modes and in the different sets of processes included in them, but a more fundamental difference lies in their respective atmospheres — one is likely to be formal and restrained, whereas the other is likely to be more relaxed, freewheeling, creative.

Thus, modes of activity are similar to cognitive modes with respect to these four constituent categories, although each category includes a more extensive set of possible values. However, modes of activity include two additional categories — the *situation* in which the mode occurs and *mediating devices* that may be used within the mode. I can best explain these extensions through an example.

Many meetings include or are oriented around a presentation. Although they may comprise an entire meeting, presentations are more commonly one activity within a larger agenda. For example, a presentation is often preceded by other business and followed or interrupted by discussion. Thus, the presentation is one mode of activity within a sequence of modes. Presentations normally take place in some form of conference or meeting room where participants gather at the same time and same place. The situation will be different and, as a result, the behavior of the group will be different if the presentation is made by video or takes place in an auditorium. Presentations usually include some form of audio/visual equipment, such as an overhead projector, a slide projector, or a VCR. The behavior of the group — including their understanding of the material and/or their agreement with the ideas presented — is influenced by these mediating devices. Thus, for a presentation mode of activity, we can identify the following additional categories and parameters:

- *situation*
  - conference room
  - same time, same place

- *mediating devices*
  - overhead projector
  - whiteboard
  - slide projector
  - flip chart

We can see the subset/extension relationship between cognitive mode and mode activity by recognizing that in the earlier discussion of writing modes, a particular computer system — WE — was assumed; thus, it functioned as a mediating device but was not explicitly incorporated in the task model because it was a constant — all writers covered by the model used WE. Similarly, the situation for the studies cited was constant — our computer lab and the conditions under which subjects performed the task.

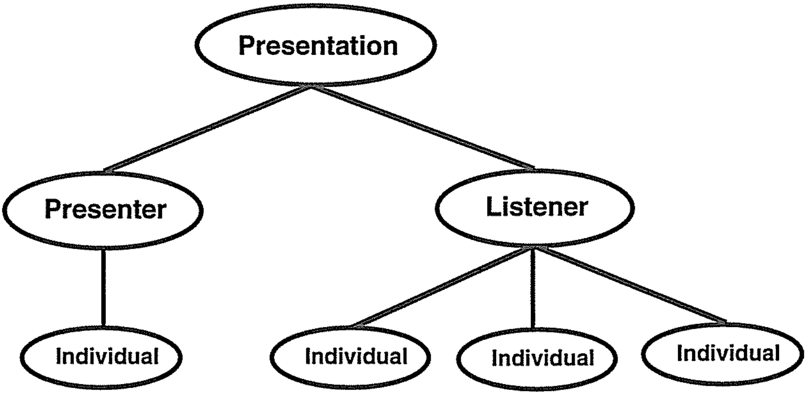

Fig. 7.5. Presentation mode of activity, comprised of one individual functioning as presenter and several as listeners.

The third extension is more fundamental. Modes of activity are multilevel structures. The mode in which a group as a whole is engaged is often comprised of several submodes in which subgroups and/or individuals participate. The multilevel structure for a presentation is shown in Fig. 7.5. At the top is the overall *presentation* mode, itself. It, in turn, is comprised of two submodes that operate concurrently — a *presenter* submode and a *listener* submode. At the bottom is a third level — the individual modes engaged by the participants. Each individual will presumably be involved either as a presenter or as a listener. An individual not participating in the mode of activity — for example, by daydreaming — can be viewed as having engaged a cognitive mode that is not part of the current group mode.

Other modes of activity will have different structures. But they all have at least two levels — the top one denoting the mode of activity for the group as a whole and the bottom one denoting the individual modes for the members participating in the activity. A variable number of intermediate submodes may be included, depending on the particular activity.

To show how the modes of activity framework can be used to characterize the behavior of a group, Kim Blakeley carried out an exploratory study during the summer of 1990. Over a 2-month period, she observed five meetings of a group during the early stages of a new project, taking detailed ethnographic notes similar to those described in the preceding section (Blakeley, 1990). Her analysis of these data identified three basic types of modes of activity: *discussion, presentation,* and *delegation.* Two of these modes, however, included variations on the basic form. In all, she observed the following seven modes or variations:

- Discussion
  - Discussion, in its basic form
  - Conflict-Resolution
  - Brainstorming

- Presentation
  - Presentation, in its basic form
  - Summary
  - Demo

- Delegation

Figure 7.6 shows the specific constituents that comprise these seven modes.[3]

---

[3] Blakeley's original table included only the four cognitive mode categories; situation and mediating devices were identified in the narrative description of each mode. I have updated the table to include these two additional categories of information.

Mode	Variation	Goal	Product	Process	Constraint	Situation	Mediating Device
Discussion	**Discussion**	• externalize information	• group-level awareness of information	• dialog • analysis	• take turns talking	• conference room • same time	• whiteboard
	Conflict-Resolution	• externalize information	• group-level awareness of information • resolving a conflict	• dialog • analysis	• take turns talking • one topic addressed • subset of group involved • different opinions encouraged	• conference room • same time	• whiteboard
	Brainstorm	• externalize information	• group-level awareness of information • generating ideas	• dialog • analysis	• take turns talking	• conference room • same time	• whiteboard
Presentation	**Presentation**	• introduce information	• group-level understanding of information	• teach and inform • listen, learn, question, evaluate	• one individual controls	• conference room • same time	• whiteboard
	Summary	• introduce information • receive same message	• group-level understanding of information	• teach and inform • listen, learn, question, evaluate	• one individual controls • carried out by leader	• conference room • same time	• whiteboard
	Demo	• introduce information	• group-level understanding of information	• teach and inform • listen, learn, question, evaluate	• one individual controls	• lab • same time	• computer system
Delegation		• assign task	• understanding of work responsibility	• delegate and explain • listen and evaluate	• senior member delegates	• conference room • same time	• whiteboard

Fig. 7.6. Seven Modes of Activity, including the processes, products, goals, constraints, situation, and mediating devices for each. (Adapted by permission from Blakeley, 1990.)

The description that follows is taken from Blakeley's summary and analysis of the group's second meeting,[4] following an earlier "kick-off" meeting for a new project. The purpose of the meeting was to identify a set of tools the group could use to help them build a new computer system. In these segments, we can see Blakeley's use of the modes of activity framework to characterize both the overall behavior of the group as well as the behavior of individual participants.

---

[4] Blakeley italicized her direct descriptions of events; her discussion and analyses of events are shown in normal type.

The second meeting of this group took place on 5-23-90. The only co-leader present was George[5] (Sam was out of town). Bill and Paul were the only other faculty members present. Graduate students Fred and Tim also attended. The purpose of this meeting was to begin determining the software tools needed to build the collaboration system.

George opened by asking, "Who's here?" George immediately assumed the leadership role, but did ask if "anyone (had) an agenda for the day." When no one came up with an agenda, he announced that he would like to discuss the tools and applications needed to build the collaboration system.

George's opening comment reflected his concern in the first meeting about who would be joining the team. George's asking the group for an agenda served to give the other members a feeling that they shared control of the project. Perhaps this would further attract the individuals attending the meeting.

Fred was the first to offer a software tool needed in the new system. George went to the white board and began recording the information offered by the other group members. After several tools were listed on the board, George began checking off and circling items as an indication of those he thought important.

Thus, the group began the second meeting in a brainstorming activity. This activity was initiated by the leader as he solicited ideas from the group. Because the project was in its infancy and little was firm, the group seemed to be in an exploratory mode. Thus, the constraints on the discussion were few, resulting in a free-form flow of conversation among group members. Although George was soliciting ideas from the others, he was still in control of how these ideas were presented on the board. George was thus in a slightly different cognitive mode than the other group members. He was interpreting while the individuals presenting candidate tools were brainstorming.

The group as a whole was in a type of discussion mode, with the following characteristics:

---

[5] Not the real name of the participant; all names were changed to protect confidentiality.

Mode type: Brainstorming

Goal
- to identify candidate tools needed in the collaboration system

Product
- a list of tools on the whiteboard

Processes
- Overall: active, balanced exchange of ideas; although not everyone participated, no one person did a majority of the talking
- George: interpreting, writing
- Others: brainstorming, offering items

Constraints
- general topic identified
- low censorship with respect to relevance

George then reiterated that the group was "deciding what tools are needed to complete a project." Tim asked him to define "project." The leader answered, "Good point. What kinds of projects should we aim for?"

Thus, the group was still not sure about the purpose of the collaboration system and was engaged in an ongoing process to define the system and its goals. In addition to the goals of each mode of activity that the group went through, the group seemed to share a general goal of developing a common understanding of the project.

About 15 minutes into the meeting, Paul stated that it is important "not to confuse the possible multiple roles of the computer." He explained that "building things for the computer and using the computer to build are two different things, and we shouldn't confuse the two." George gave a short argument to Paul's statement. Paul's response to George was "possibly." George then drew a diagram on the board, showing where he thought the project should be on a scale between "design" and "CASE tools."

Thus, through the diagram on the board, Paul and George had an understanding of the other's concept, although they may not have agreed on the subject. This example shows how differences in opinion can be acknowledged and understood. In a conflict-resolution mode, the individuals with the differing opinions debate until a common ground is located, or until positions are understood or left unresolved by agreement. This particular scenario had the following characteristics:

Mode type: Conflict-Resolution

Goal
- to further specify the purpose of the project

Product
- a group-level awareness of the issue

Processes
- dialog, debate

Constraints
- a limited subset of the group participated in dialog
- a particular topic was addressed
- different points of view were encouraged

As the meeting continued, Fred again brought up the issue that he had introduced in the first meeting: the goal of virtual proximity versus the goal of collaboration augmentation. He stated that a system with the goal of collaboration augmentation would require more prerequisite cognitive work, something that he indicated the "other group members (were) more interested in." He also described that a software system used to achieve virtual proximity would be more easily accepted by users.

Fred's comments again show that the group was divided in its interests. Fred was on the "virtual proximity" side of the house and was pushing the project in that direction. Thus, coalitions of individuals within the group were emerging. There seemed to be two groups, each defined by a separate goal for the collaboration project. Fred and George were in one group, while Sam was separated from them by his different goal for the project.

The meeting continued with more discussion of tools needed in the collaboration system. Fred explained that with the new system, users should be able to share documents although they may use different text editors. He pointed out a benefit of this as he said, "Today people get together and bite the bullet and agree to use Tex even though they hate it."

About 45 minutes into the meeting, George presented a general summary of the meeting. He stated, "This discussion has been useful to me and has helped me understand the scope of what we are really talking about here." He also summarized a goal he had for the project.

> George's summary seemed to be an attempt to bring the meeting to a close. It also seemed to be an attempt to provide the other members of the group with the feeling that their time spent attending the meeting had been put to good use. Thus, the meeting had turned from a discussion to a summary with the following characteristics:
>
> **Mode type: Summary**
>
> **Goals**
> - to bring the meeting to a close
> - to assure group members of time spent profitably
>
> **Product**
> - a general description of the meeting
>
> **Processes**
> - George: summarizing, recalling, evaluating, talking
> - Others: recalling, listening
>
> **Constraint**
> - presentation/statement made by the group leader
>
> (Blakeley, 1990, pp. 28-31; reproduced by permission)

The meeting did not end at this point, but, rather, went through several additional mode shifts. These included another period of discussion followed by a second summary that did bring the meeting to a close. Fig. 7.7 shows the complete sequence of modes engaged during this meeting.

Blakeley's study demonstrates that ethnographic techniques can be used to observe in close detail a group's social and intellectual actions, that the resulting data can be analyzed in terms of an appropriate set of modes of activity, and that the overall strategic behavior of the group can be seen in the sequence of modes engaged by the group. As we attempt more comprehensive studies, the set of modes will have to be expanded to include behaviors other than discussions, presentations, and delegation. But the basic mode of activity framework, with its six categories of components, appears to provide a viable set of building blocks with which to develop models of collective strategy and to characterize the behavior of collaborative groups.

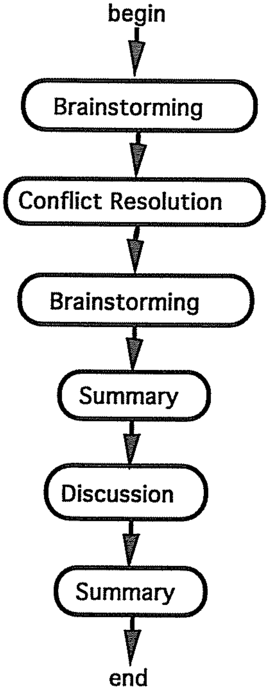

Fig. 7.7. Sequence of modes of activity observed during one meeting of a group. (Adapted by permission from Blakeley, 1990.)

Formalism

To understand the behavior of groups in a detailed way, we need to understand the interacting effects of its members with respect to one another and to the products and information structures they work with. Earlier, we saw that models of individual strategy could be built using the cognitive modes framework and the formalisms of production rule and ATN grammars. To model collective strategy, we need comparable, but extended, mechanisms that can deal with the more complex behavior of groups. In the preceding section, I suggested that we could use the modes of activity framework as the basic construct in which to define models of collective strategy. Here, I suggest a similar extension with respect to grammatical formalisms.

The system that implements a model of collective strategy will have to parse or otherwise analyze the multiple protocol streams that result from the different activities of the group. These protocols could be derived from individuals working alone, several members participating in a computer-supported conference, and groups involved in meetings or other similar activities. By analogy, imagine a natural language grammar that can parse not just a single input stream representing a single source of sentences but, instead, the multiple input streams representing all of the conversations going on in a room at one time, as might occur during a reception, including making sense of references in one conversation to remarks overheard in another. This is the nature of the task that an analytic model of collaboration must deal with.

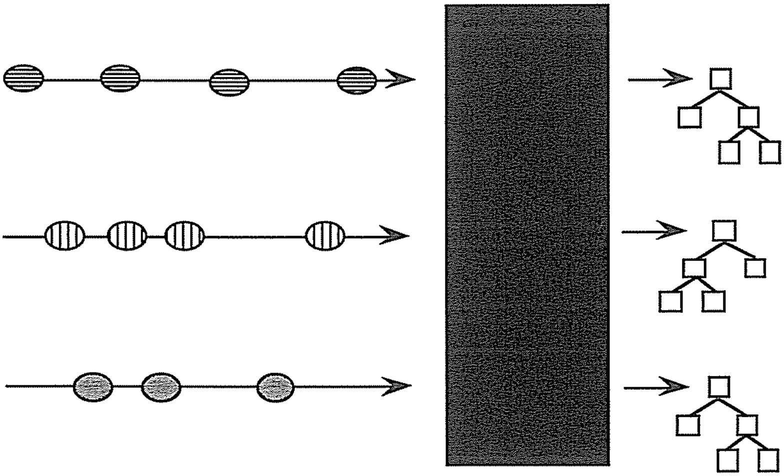

Fig. 7.8. N-dimensional grammar producing n separate parses. The grammar cannot identify effects one member's actions have on another.

To address these issues, our project is extending the ATN formalism to include rules that apply to multiple protocol streams. We refer to this formalism as an *n-dimensional grammar* to indicate its acceptance of *n* concurrent input streams, rather than the single steam of symbols accepted by a conventional grammar. N-dimensional grammars could take several forms. Fig. 7.8 illustrates one possibility. Three simultaneous protocol streams are input to the parser, represented by the shaded box in the figure. The parser applies its grammar to each protocol stream independently, producing three separate parses, shown as the three separate parse trees emerging on the right.

Rules for the three different types of processors — computer-, group-, and conference-mediated cognition — are written either as separate grammars or as an integrated set of rules, portions of which can be selectively applied to the three corresponding types of protocols. Thus, we could think of the grammar as three specialized grammars — one for each type of protocol — that have been yoked together so that they operate in tandem and as a single system. Following this approach, no attempt would be made by the grammar to infer the effects actions in one protocol stream have on actions in another. Instead, such inferences would be left to the researcher to derive from the analyzed data.

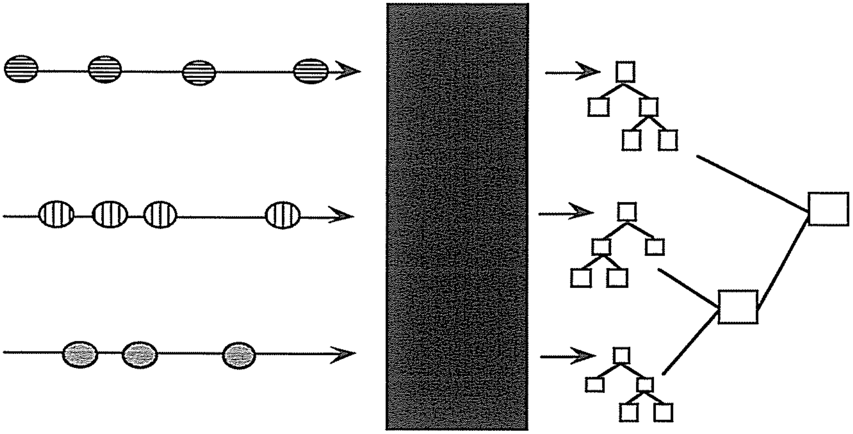

Fig. 7.9. N-dimensional grammar that produces n separate parses plus a group parse that identifies effects of interaction within the group.

A second, and more powerful, approach would include additional rules that identify patterns across multiple protocol streams. A parser based on this concept of n-dimensional grammar might still produce individual analyses for the different protocol streams — perhaps during a first pass through the data — but it would also produce a more general parse that spans all of the protocol streams, identifying interactions across individuals and subsets of the group. This second form of the grammar is illustrated in Fig. 7.9, in which three low-level parses have been further analyzed in relation to one another to produce a higher-level parse that represents the overall strategy of the group.

Although the first form of the grammar is appealing because of its relative simplicity, it is unlikely to be satisfactory, even for individual protocols. To see why, let's look more closely at some of the issues that must be addressed for each of the protocol types.

Multiple Individual Processors. Because the protocols for individuals working alone will be expressed in terms of computer-mediated cognition cycles and because the higher levels of the model can be expressed in terms of modes, we might expect this part of the grammar to differ from grammars that deal with individual behavior largely in the details of the task model — for example, a model for planning and writing computer code versus one for planning and writing a short document.[6] Similarly, the fact that the total system would have to accept multiple instances of this type of protocol would place additional requirements on the design of the parsing program but not on the grammar rules, themselves. Thus, at first glance, this part of the grammar seems similar to the grammars described previously that address individual behavior, that is, writers using the WE system.

However, this form of grammar cannot deal with the effects one member's actions have on another, either at the moment or at some future time. Momentary interactions for individuals working alone have to do with concurrency effects. If one member of the group is working on a particular part of the artifact and, at that very moment, another member wants to work on the same portion, under some conditions he or she may not be allowed to do so by the collaboration support system's concurrency control mechanisms. For example, ABC permits multiple members to read the same portion of the artifact, but only one member at a time may edit or change that part. If another user attempts to gain so-called write access to a segment for which another member already holds write access, he or she will be blocked. If a member asks for read access while another has write

---

[6] To be consistent with other parts of the collective model, we would replace cognitive modes with modes of activity. This will require additional rules to take into account situation and mediating device parameters, but this extension would not change the fundamental form of the grammar.

access to that segment, he or she will be provided with a copy of the segment as it existed at the time the person with write access began work on the segment. Consequently, the reader's version of the segment may already be out of date if the writer has made a change but has not yet saved his or her version of the segment. Thus, the actions of an individual working alone on the artifact may be affected by the simultaneous, but logically independent, actions of other members of the group. A grammar whose rules apply only to a single individual could not accurately interpret situations such as this that result from the actions of another user.

Long-term interactions can take several forms. Of particular concern is the evolution of the artifact. If a group grants more than one member permission to write or change a given segment, although not at the same time, one member's work may be undone or changed by another's. For example, if member A adds a paragraph to a project document, member B could read that paragraph several days later, disagree with it, and delete or edit it. When member A comes back to the task, he or she would find his or her earlier work changed or missing. Because a parser that analyzes actions from an individual processor takes into account the effects of those actions on the artifact, it would find a discontinuity with respect to the data from one session to the next. Thus, it would have difficulty interpreting A's subsequent actions with respect to the artifact.

Issues such as these pose two different kinds of problems. First, they pose technical problems for the parser. If the grammar were built as three logically independent grammars, as illustrated in Fig. 7.8, it would include rules that treat such events as "black" events — they occur; the grammar produces a superficial analysis of the user's responses to them; but it could not develop a rationale for the user's subsequent actions based on the semantics of the black event. Let me illustrate this by continuing the first example. If a member of the group was denied access to a segment — either repeatedly or continuously for some period of time — that person would likely try to find out who else was working on that segment and then either arrange, informally, times when he or she could get access or initiate a computer conference so that they could work together on the segment. A grammar of the first type could not make an explicit connection between the access event and these responses to it. On the other hand, the second form of the n-dimensional grammar could include rules that do make connections between one user's working on a given segment and another user's attempts to gain access to that segment.

If the members of a project frequently encounter events such as these, it may indicate more fundamental problems. At the least, the situation suggests problems of inefficiency — individual work is being delayed or even destroyed. But it may indicate that the decomposition of the design and/or work assignment is faulty. It is precisely behaviors such as this that we would like a model of collective strategy to identify. A comprehensive perspective that looks down on the project as a whole and can see interactions such as these can only be gained from n-dimensional grammars of the second type.

Processor for Intangible Knowledge. The actions in this type of protocol occur in situations, such as meetings, where intangible knowledge is developed and consist of group-mediated cognition cycles. Consequential, they must be coded by an observer. The observer records individual behaviors and, if a mode of activity framework is being used for the study, the submode or role, such as presenter and listener, for the action. These behaviors can be recorded either as separate protocol streams, one for each participant and/or submode, or the respective actions can be interleaved to form a single protocol for the entire situation, with parameters identifying both individual and submode for each action. The two options are, of course, equivalent, because one can be mapped to the other.

If we think of the protocol streams as divided by participant, then the data look like that for multiple independent processors. An analysis will probably look for patterns of coherence within the behavior of individuals, in temporary coalitions, and in the group as a whole. Thus, rules will have to be included that focus on actions generated by each of these units. However, the grammar will also have to include rules that look at interactions across the various protocol streams, including behaviors influenced by submode or role. These are the same requirements as those just discussed for multiple independent processors, but the interactions are much stronger and more frequent, and the influences are more direct. Thus, although the rules for these two types of processors will be different because their information products and basic processes are different, the overall structure for both of these portions of the grammar will be similar.

1xn Hybrid Processor. Computer-based conferences can produce two protocol streams. A machine-recorded protocol recrods the actions produced by the various participants as they take turns working on the artifact. Rules that analyze this work will be similar to those for the multiple independent processors but will have to take into account the different origins of individual actions, because they can originate with different conference participants. Computer-based conferences may also produce a second protocol — a researcher's hand-coded observations of the social/conceptual interactions that accompany the conferencing group's work on the artifact. This second protocol will resemble that of the processor for intangible knowledge, described previously, and rules for analyzing these data will be similar in form. However, they will have to take into account the strong mediating effects of the communications channel(s) and the conferencing system, itself. Additional rules will also be needed to identify patterns of interaction between these two forms of behavior and their respective protocol streams.

Although the rules for an n-dimensional grammar will be complex and their development will pose a significant challenge to researchers, a more basic problem lurks in the very concept of strategy when it is applied to large groups. A fundamental assumption with regard to individual strategy is that a person engages a single cognitive mode at a time, although he or she may rapidly shift from one mode to another. Thus, the person is brainstorming, planning, coding, and so on, but he or she is not brainstorming and coding at the same time. Consequently, we can describe a person's strategy by tracing his or her shifts from one mode or activity to another. This is not necessarily true for groups.

If the group is small — consisting of five or six members — the behavior of the group may be largely coherent. For example, a small group in the early stages of developing a new computer system might assign several members the task of going off and analyzing specific issues — such as the database system to be used or user interface requirements. After they report back, the group as a whole might then work out the high-level architecture for the system. If we analyzed this portion of the group's work, we might describe the group as experiencing a shift from brainstorming to early planning. Thus, although individuals or teams may engage in different activities, their individual actions come together at frequent intervals to support a single purpose for the group as a whole. Later, the group may shift to largely writing and/or coding activities. Eventually, we might expect to find the group primarily involved with testing and, perhaps, developing user documentation. Thus, although the work of small groups will not generally be as coherent as that of individuals, nevertheless, we may be able to identify the group's predominant mode of activity most of the time, while recognizing a diversity of behavior for individual members and/or subgroups. This was the case for the group writing congressional testimony, described in the second scenario in chapter 2. A study of design teams that found similar patterns of behavior is (Olson, Olson, Carter, & Storrosten, 1992)

The situation is more complex when the group is large. A project that involves 25–30 people is likely to include five or six such groups, as was the case for the software development project described in the third scenario in chapter 2. Although each individual team may exhibit coherent strategic behavior, it is not clear that the group as a whole will do so. Imagine a situation in which three teams are building three separate parts of a system, a fourth is building a set of analytic tools, and a fifth is studying the behavior of potential users of the system. Although the system- and tool-building teams must interact with one another to some degree, they may work independently for considerable periods of time. Similarly, the team concerned with human users may have only infrequent interaction with the others. Thus, at any one time, different teams may be at different stages relative to their respective parts of the project as a whole and, as a result, engaged in different modes of activity. What, then, is the overall strategic state of the group as a whole at any given time?

If it makes little sense to talk about a large group being engaged in a single overall mode of activity at any one time, we cannot expect to produce parse trees that describe the group's overall strategic behavior. In the parse trees described previously, a session was composed of the sequence of cognitive modes engaged by the subject while performing the observed task. If we took the same approach with groups, the penultimate level of the tree — or, if the grammar spanned multiple sessions, some high, but not penultimate, level — would consist of symbols that identify the overall mode of activity for the entire project at a given time. Because this will not be the case for large groups, we need a different way of representing collective strategy.

One possibility is to think of the strategy followed by a large group as resembling a multistrand cable, as suggested in Fig. 7.10. Each strand represents a single coherent activity extending over time, which runs parallel to the strands. Some strands represent individual behaviors, others collective behaviors such as meetings and computer conferences. When the activities of one group or one individual strongly interact with those of another, their respective strands merge for the duration of the interaction to form a single coherent, but larger, behavior/strand. When their activities diverge, the merged strand separates back into parallel and independent strands.

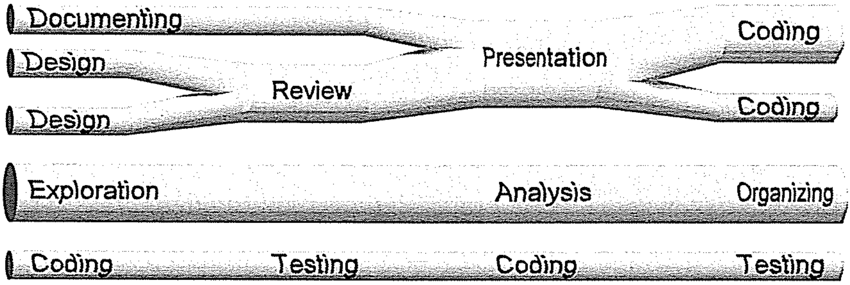

Fig. 7.10: The strategic behavior of large groups can be thought of as multiple strands of coherent behavior, parallel to one another in time. Strands for individual members merge during periods of collective work and separate during periods of independent work.

In chapter 6, I suggested that IPS models of collective behavior could be built more easily and more appropriately using a large-grain distributed systems architecture rather than the Von Neumann architecture used in models of individual cognition. Within this architecture, each strand of independent behavior within a group would correspond to a separate process in the distributed systems model of strategy. Each represents an instance of one of the three types of processors included in the collective processor — the processor for tangible knowledge, intangible knowledge, and the hybrid processor. Thus, for example, when an individual member of the group works alone on the artifact, a corresponding individual processor in the distributed system model comes into existence for the duration of work. Similarly, analogous processes are launched for meetings (and other similar situations in which intangible knowledge is developed) and for computer conferences.

Each processor runs the parsing program and parses the protocol generated by the corresponding activity in the group. Because processes can exchange messages in this architecture, the overall system can interpret interactions between separate activities. Thus, the overall behavior of the group is modeled by the dynamic topology of the distributed system and the flow of messages within it. The behavior of the distributed systems model, in turn, can be represented by the multistrand cable, where each strand corresponds to a separate thread of control in the system and, hence, a separate coherent behavior in the group. Within each strand, detailed behavior can be represented by a conventional parse tree, because each such behavior is coherent.

Considerable research will be needed to build distributed systems models of strategy as suggested here, the n-dimensional grammars that run in its processes, and display programs that allow researchers to visualize the results of the model. I suspect, however, that any approach that is simpler will miss much of the complexity and interaction that occur within large collaborative groups and, hence, be of limited value.

To summarize briefly, I have discussed three components needed to build models of collective strategy. First, data must be collected that can provided a comprehensive view of the fine-grained activities of a group. These data must capture the actions of individual members working at their respective workstations, their interactions with the system and with one another during computer-based conferences, and their behavior during meetings and other situations where shared intangible knowledge is exchanged and developed. We can't expect to get a total and complete record of a group's behavior, but we must aspire to comprehensive collection if we are to build comprehensive models. Occasional samples, retrospective accounts, or records of a single type of activity will not be sufficient.

Second, I described a basic framework — the mode of activity — in which to describe collective behavior. Its multilevel structure and constituent categories can provide the high- and intermediate-level symbols for models of collective strategy.

Third, I suggested that we can extend the basic ATN formalism to include rules that identify patterns of behavior across as well as within individual protocol streams. The resulting n-dimensional grammars can function as the analytic components that operate within a distributed systems model of collective strategy. However, to develop the methodology and tools needed to build actual models capable of analyzing and characterizing the work of large, multiteam collaborative projects will require considerable research.

Issues for Research

The long-term research goal identified in this chapter is to develop comprehensive, yet detailed models of collective strategy and to use those models to study collaborative behavior. Here, I will discuss several smaller tasks that continue the line of research outlined at the end of Chapter 6 and that contribute to that goal.

- *Identify specific modes of activity that occur in particular collaborative tasks.*

At the end of chapter 6, I suggested as topics for research identifying the habitual activities engaged by groups and, then, the fine-grained processes that occur within them. Because the mode of activity construct described in this chapter includes both, I suggest further testing of it as a framework in which to describe those activities. The next step would be to identify each mode of activity that occurs in the target group(s) and the constituents of each mode — for example, the goal, constraints, and so on, as well as its multilevel structure. To test the construct, we would need to answer several questions. Does the set of modes account for all mediated cognition cycles observed? Can the behavior of different groups working in different situations be described in terms of a specific set of modes? If the modes framework holds up, then we can use it as a basic architectural construct with which to build models of collective strategy. If it does not, the studies that discredit it may suggest alternative constructs.

- *Build a methodology and a set of tools through which to study collaboration.*

To address the research issues described here and elsewhere in this chapter, we will need better methods and better tools. Observing groups and collecting detailed ethnographic-type notes are difficult and time consuming tasks. Observers need training to help them see what is going on in a group and to control their own preconceived notions. They also need guidelines on issues ranging from ethics to consistent coding. On the other hand, machine-recorded protocols pose different problems. Ideally, commercial developers would include tracking functions in their products that conform to some technical standard, but this will be long coming; when and if it does, it will raise many thorny issues concerning uses of the data. In the meantime, we must look to those building research and experimental collaboration systems for tools of this sort. Regardless of their origin, instrumented systems will produce a flood of data that will have to be stored, managed, and analyzed. Thus, we will need automated analysis tools, such as the grammars described previously, that can integrate and analyze multiple types and instances of protocols. We will also need tools to represent results in visual and dynamic forms that enable human researchers to literally see a group's strategic behavior. Consequently, we should regard methodology, itself, as a legitimate area of research and one that will need its own supporting systems.

- *Build specific models of collaboration.*

The goal of this methodology should be to build detailed, yet comprehensive, models of collaboration for particular kinds of projects, such as group writing and software development. Initially, these models will be quite specific — covering a single project, using a particular set of tools, working within a particular organizational structure and context. With experience, we may be able to extend initial models across different groups, tasks, and conditions. The ultimate goal is to develop a general process model of collaboration; the path we follow to do this should include iterative cycles of specific studies followed by generalization and testing.

- *Use insights derived through these studies to build better support systems, educational curricula, and organizational structures.*

A model of collective strategy should not be an end in itself but, rather, an instrument for developing knowledge about collaborative behavior which can be put to useful purposes. It should lead to better collaboration support systems and to new educational curricula that help groups work together more effectively. If we knew the circumstances under which groups work more effectively, we should be able to adjust organizational structures in specific ways so that they support those circumstances as a matter of policy and intent.