

Why Are Some Diagrams Easier to Work With?
: Effects of Diagrammatic Representation on the Cognitive Integration
Process of Systems Analysis and Design

Jungpil Hahn

Information and Decision Sciences
Carlson School of Management
University of Minnesota
Minneapolis 55455 Minnesota

Jinwoo Kim

Department of Business Administration
Yonsei University
Seoul 120-749 Korea

Primary Contact Person

Jinwoo Kim

Department of Business Administration, Yonsei University
Seodaemun-ku Shinchon-dong 134, Seoul 120-749, Korea
Telephone: 82 - 2 - 361 - 2628
Fax: 82 - 2 - 313 - 5331
Email: jinwoo@base.yonsei.ac.kr

ABSTRACT

Various diagrams have been used heavily in systems analysis and design without proper verification of their usability. However, different diagrammatic representations of the same information may vary in the computational efficiency of working with these diagrams. The objective of this research was to explore the effects of diagrammatic representations on the task of integrating multiple diagrams. The domain of systems analysis and design was used to generate examples and test the theory. A cognitive model of diagram integration was proposed, and an experimental study was conducted to explore the effects of representational features of diagrams on the cognitive process of diagram integration. Results of the experiment show that the representational features of the diagrams acted as the criteria for selecting among various methods for analyzing and designing the integrated diagram. In addition, the difference in the selected methods resulted in different task performances in terms of analysis and design errors. This paper concludes with the implications of the results for the development of cognitively compelling diagrams.

KEYWORDS

Diagrammatic Representation, Visual Grammar, Diagrammatic Manipulation, GOMS

Why Are Some Diagrams Easier to Work With?

: Effects of Diagrammatic Representation on the Cognitive Integration Process of Systems Analysis and Design *

1. INTRODUCTION

Diagrams are an effective medium of human thinking and problem solving (Kosslyn, 1980; Larkin and Simon, 1987; Tversky, 1997). The role and effectiveness of diagrams have been investigated in various areas such as geometric problem solving (Koedinger and Anderson, 1990), qualitative reasoning in artificial intelligence (Forbus, Nielson, and Faltings, 1991), reasoning about economic markets (Tabachneck and Simon, 1992; Tabachneck, Leonardo and Simon, 1994), medical diagnosis (Rogers, 1995; Rogers, 1996), scientific discovery (Cheng and Simon, 1995), understanding mechanical systems (Narayanan, Suwa, and Motoda, 1995), computer programming (Modugno, Corbett and Myers, 1997) and systems analysis and design (Kim and Hahn, 1997).

In systems analysis and design, the diagram has become an important artifact as a medium of knowledge representation, because current trends in software engineering place more emphasis on the design diagram rather than on the final program code (Winograd, 1995). Nowadays, diagrams are used as a means of actually developing a system by directly manipulating the diagrams themselves, in addition to being a means of representing knowledge about the system. As a result, many software engineering methodologies have developed various diagrams and detailed prescriptive models of how to manipulate these diagrams for systems analysis and design. Some even provide multiple diagrams to represent the same information which differ only in terms of their visual appearance (Rational, 1997). The visual appearance of a diagram influences its level of computational efficiency and may thus produce different outcomes (Larkin and Simon, 1987). This becomes a major source of confusion for the users (i.e., software engineers) and also for the designers (i.e., developers of software engineering methodologies) of these diagrams. The confusion results from the fact that the prescriptive models proposed by the methodologies lack any theoretical framework for the cognitive processes involved in manipulating these diagrams. The users are on their own when choosing the diagram to use without knowing which diagrammatic representation is more appropriate for a particular task. The designers will face the problem of devising cognitively compelling diagrams without having any guidelines or principles to assist them. Therefore, a theoretical framework and an empirical exploration of diagrammatic manipulation in the domain of systems analysis and design are in dire need.

* This research was supported by the Korean Ministry of Information and Telecommunications research grant # 96095-IT2-II and based on the master's thesis of the first author.

The objective of this study is to explore the representational effects of diagrams on the cognitive processes of manipulating external diagrammatic representations in systems analysis and design. Various aspects of a system, such as data, functions, objects, and processes, can be represented by diagrams (Pressman, 1992). This study will focus on the diagrammatic representations of *processes*. A process can be defined as a collection of activities that takes one or more kinds of input and creates an output (Hammer and Champy, 1993). The diagrammatic representation of processes is an interesting issue both theoretically and practically in that the knowledge to be represented is invisible and dynamic by nature and that various diagrammatic representations of a single process are extensively used in systems analysis and design. The main theme of this paper is to present an empirically validated view of how we read and combine diagrams for systems analysis and design, and how different diagrammatic representations might affect our behavior and behavior, in turn, affect our performance in a diagram manipulation task. A cognitive model of process development through diagram manipulation is proposed, and the empirical results are presented of an experimental study conducted in order to understand how different diagrammatic representations affect the cognitive behavior of developing system processes.

This paper is organized as follows. The next section reviews prior research on diagrammatic knowledge representation. Section three presents a detailed cognitive model of a diagrammatic manipulation task and discusses the consequences of different diagrammatic representations in process development. Section four describes the experiment designed to explore the effects of different representations on the cognitive process of diagram manipulation. Section five presents the results of the experiment. Finally, this paper concludes with a discussion about the implications of the study and directions for future research.

2. DIAGRAMMATIC KNOWLEDGE REPRESENTATIONS

The diagrammatic representation uses diagrams to represent data and knowledge, and diagrammatic reasoning uses inspection and direct manipulation of the diagram as the primary means of inference (Kulpa, 1994). As Simon pointed out “... *solving a problem simply means representing it so as to make the solution transparent* ...” (Simon, 1981), the diagram can be regarded as a means of representing knowledge so as to facilitate problem solving by making the solution transparent. A diagram is said to be well represented when the diagrammatic representation supports the cognitive processes in reasoning with the diagram. The diagram should not only be complete by expressing fully all the facts that need to be represented, but also effective in presenting the information in a way that makes it easy to perceive and also easy to reason with (Mackinlay and Genesereth, 1985). This section reviews previous literature on diagrammatic knowledge representation focusing on why some diagrams can be easier than other *diagrams* for human problem solving (Larkin and Simon, 1987).

Even though we are very used to working with diagrams, representing knowledge via diagrams is not as trivial a task as it seems, especially when we want to represent things that are inherently not visible and dynamic such as processes. Information that is visible and static is easier to represent because the relevant objects of interest may be mapped onto the presentation space (e.g., on paper or on screen) in accordance with how we perceive them in the real world. For example, when we draw a diagram for someone wanting directions to some place, we map roads, buildings, traffic lights, zebra crossings etc. from our memory of the surroundings. Furthermore, the directions (e.g., east-west or north-south) and distances between relevant objects in the real world can also be directly mapped onto the presentation space (e.g., as left-right or top-bottom accordingly). However, information about processes which are inherently invisible and dynamic are more difficult to represent because there does not exist a direct mapping from the real-world knowledge to the diagrammatic representation of that knowledge (Tversky, 1997). The process of representing such information entails a number of complex decisions to be made in order to construct a valid diagrammatic representation. These decisions are governed by rules that translate the relevant information into a diagrammatic representation. These rules may be regarded as the grammar of a visual language for diagrammatic representations (Engelhardt, Bruin, Janssen and Scha, 1996). Generally, the decisions required in representing knowledge via diagrams can be described in terms of two basic rules, which concern the *decomposition* (Tabachneck-Schijf, Leonardo and Simon, 1998) and *layout* (Engelhardt et. al, 1996) of the knowledge. The first rule governs the way real world knowledge is decomposed into graphical primitives, while the second rule dictates the positioning of the graphical primitives on the presentation space¹.

In order to create a diagrammatic representation, we first need rules of *decomposition* that map the different types of data for the knowledge to the various graphical primitives that represent them. In other words, decomposition refers to how knowledge is divided into meaningful units, each represented as a separate graphical primitive. The various data can be represented as grouped together (i.e., entities and actions are represented as visually aggregated elements), or separately as individual elements (i.e., each component as a different visual element). For example, if we wanted to represent a simple process such as "John sells a book to Mary, Mary pays John for the book, and then Mary gives the book to Bob", we may depict this by representing "John sells book to Mary" as a graphical token representing an event, "Mary pays John for the book" as another token representing the second event, "Mary gives the book to Bob", with arrows from the first to the second and the second to the third to show the sequence of occurrence of the events (Figure 1a). However, this form of knowledge representation is not the only possible configuration of information

¹ There are other grammatical rules (e.g., color of graphical primitives, etc). However, we focus on these two rules because of their relative importance in constructing the diagrams used mostly in the domain of systems analysis and design.

decomposing about the same process. Another possible way to depict the same process would be to represent "John", "Mary" and "Bob" as separate graphical tokens representing the actors with an arrow from "John" to "Mary" stating "sell book", an arrow from "Mary" to "John" stating "pay for book" and an arrow from "Mary" to "Bob" stating "give book" representing the actions (Figure 1b). The two diagrammatic representations convey the same meaning, but with a different application of information decomposition.

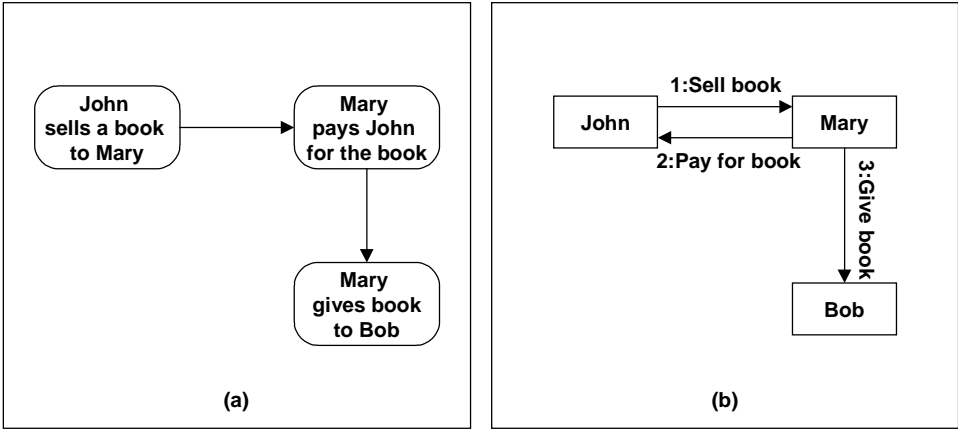


Figure 1 - Information Decomposition

Next, rules of layout organization are needed to determine how to provide meaningful information with the layout of the presentation space. The presentation space can be divided into separate areas and visual elements assigned to one of the sub-areas, thereby expressing a categorization (Moher, Mak, Blumenthal and Leventhal, 1993). Rules of layout organization that determine how we divide a presentation space and how we position visual elements in the spaces will change the layout structure of the final diagrammatic representation (Engelhardt et. al, 1996). For example, the graphical token representing "sells a book to Mary", "pays John for the book" and "gives book to Bob" may be positioned within partitions of the presentation space representing "John" and "Mary" (Figure 2a), or simply be randomly positioned (Figure 2b). The difference in layout will result in different diagrammatic orientations of the same knowledge.

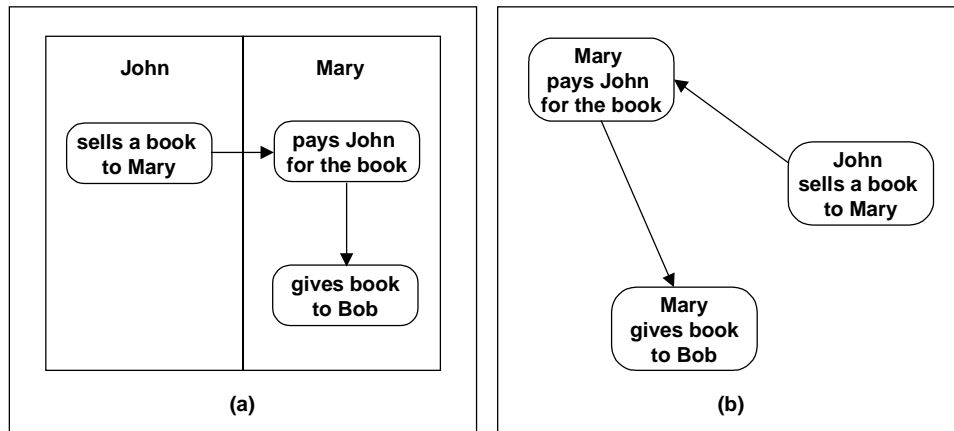


Figure 2 – Layout Organization

The above discussion concerning the visual grammar for diagrammatic representation implies that various combinations of grammatical rules may be applied in representing the same knowledge. Applying different information decomposition and layout organization rules will result in a different diagrammatic format and orientation of the same information. Even though the different representations are informationally equivalent (i.e., represent the same amount of information), the different diagrams will not necessarily be computationally equivalent in reasoning with them and in manipulating them (Larkin and Simon, 1987) because different diagrammatic representations will provide different perceptual cues that affect the amount and effort of search that is required for problem solving with diagrams (Zhang, 1997). Thus, applying different grammatical rules will produce different diagrammatic representations, which in turn will differ in terms of computational efficiency for diagrammatic reasoning tasks. The effects of different representations on the process integration task are discussed in more detail in the next section.

3. REPRESENTATIONAL EFFECTS OF PROCESS MODEL DIAGRAMS

As discussed earlier, the diagram has become an important and useful artifact in modeling systems, especially in modeling processes. Process models are abstractions of real world processes aimed at understanding the activities and tasks that comprise the processes (Davenport, 1993). Prior research on process modeling has suggested the information a process model should convey. A process model should express the work being done (i.e., the sequence of process activities to execute the process), the entities involved in the work process and the relationships between these entities (Curtis, Kellner and Over, 1992). In its simplest form, a process can be modeled as a sequence of activities (i.e., who does what to whom in a predefined order) where each activity is performed by a specific actor — the sending object (i.e., who) — and has a recipient — the receiving object (i.e., to whom) — who receives the outcome of the actor's activity — the message (i.e., does

what)².

Process model diagrams are important in that systems engineers commonly use these diagrams in analyzing and designing system processes (Beedle, 1995). The process model diagrams are used not only for visualization of the target processes but also for manipulating them. For example, process models may provide valuable insight as to how to make the processes more efficient (Jacobson, 1995; Taylor, 1995). A good process model gives an exact description of the processes, which enables thorough understanding of the current situation, which in turn helps identify problem areas for improvement. In business process reengineering (BPR), for instance, models of business processes are analyzed in order to assess their efficiency and effectiveness. This kind of analysis usually involves the integration of multiple processes to obtain a comprehensive view of the overall processes (Davenport, 1993). This study will focus on this task of process integration through diagram manipulation.

Integrating processes with diagrams is a complex problem solving activity of analysis and design where the problem solver has to fully understand the individual process diagrams, infer the underlying relationship between the processes represented in separate diagrams (i.e., analysis), and finally externalize the integrated process in the form of a single diagram depicting the relevant information (i.e., design). The problem solver has to represent all the components of the process model, such as the relevant entities and the activities performed by the entities in the correct sequence.

Even though the diagram manipulation task with different diagram involves the same process of analysis and design, the difference in the notation of the diagrams will have an effect on how the integration task is performed because certain cognitive operations needed for the analysis and design are harder in some diagrammatic representation than in others (Gilmore and Green, 1984). In the next section, we will formalize the model of process integration as a formal GOMS model (Card, Moran and Newell, 1983) of diagram integration. Emphasis will be placed on understanding how different diagrammatic representations will affect the problem solving behavior and performance.

3.1. GOMS Model of Diagram Integration

The task analysis model of process integration is presented in the form of a GOMS model at the functional level (Figure 3). The problem solver begins the process integration task with the top level goal of integrating the individual processes into a single integrated process (GOAL: INTEGRATE-PROCESS). This is accomplished

² The terms sender, receiver and action will be used interchangeably with sending object, receiving object and message, respectively.

by integrating the individual diagrams that portray the sub-processes into a single diagram that illustrates the integrated process. The top level goal of integrating the processes is further partitioned into two sub-goals; analyzing the individual processes (GOAL: ANALYZE-PROCESS) and designing the integrated process (GOAL: DESIGN-PROCESS).

The second level sub-goal of analyzing the process (GOAL: ANALYZE-PROCESS) is accomplished by reading the individual diagrams and integrating the information acquired from reading into an internal representation of the integrated process (GOAL: READ-AND-INTEGRATE-DIAGRAMS). The third level sub-goal of reading and integrating the diagrams can be achieved by means of two alternative methods. These methods are 1) reading each individual diagram fully first then integrating them once all the information has been fully analyzed (BATCH-METHOD) and 2) integrating *while* reading the individual diagrams (ALTERNATE-METHOD). The BATCH-METHOD integrates the information presented only after having investigated all the information provided in the individual diagrams. For example, the problem solver will take a diagram to investigate, read off all the information presented in that diagram, take the other diagram, read off all the information presented there, and then try to integrate all the information into a coherent internal representation of the integrated process. On the other hand, the ALTERNATE-METHOD involves a purposeful exploration among the information presented in the different diagrams during the process of reading the individual diagrams. For example, the problem solver will first take a diagram and read by following the sequence of process activities. When a reference to another diagram appears in that diagram, he/she will switch to the other diagram and search for the referent. Once the referent is identified, he/she will compare the two diagrams in order to infer the relative order of occurrence between the diagrams, then continue to follow the sequence of process activities until reference to the first diagram appears. Such a process of following process activities and switching diagrams will continue until all the information in all the diagrams have been read.

The other second level sub-goal of designing the integrated process (GOAL: DESIGN-INTEGRATED-PROCESS) consists of externalizing the internal representation of the process constructed from the analysis phase into a valid diagrammatic representation (GOAL: DRAW-PROCESS-COMPONENTS). Externalizing the internal representation of the integrated process is basically achieved by drawing all the components of the process model correctly. Normally, when drawing a diagram, we do not proceed mechanically from the top of the page to the bottom without regard to the semantics of what is being drawn. The drawing activity is a more meaningful process of illustrating the objects of interest in an organized manner. In the case of process diagrams, the drawing activity will be a sequence of specifying the components of the process model — sender, receiver and action (i.e., who does what to whom) — in the order of occurrence of the actual process activities. This drawing activity can take the form of 1) drawing the components of the process model in a strictly sequential manner (i.e., sequentially specifying all the "who does what to whom" starting from the first activity until the last activity) or 2) building a template of all the entities first (i.e., senders) and then

specifying the activities (i.e., actions) in relation to the template already drawn. Thus in the task analysis model, the goal of drawing the components of the process model can be achieved by sequentially drawing the components of the process model (SEQUENTIAL-METHOD) or by drawing a basic template first and then specifying the remaining components needed (TEMPLATE-METHOD).

Furthermore, different methods may apply in drawing the sequence of activities (GOAL: DRAW-ACTIVITIES). The components for each process activity (i.e., sender, receiver and action) can be 1) specified sequentially as separate graphical tokens (DRAW-SEPARATE-METHOD) or 2) expressed together as aggregated graphical tokens (DRAW-BULK-METHOD).

```

GOAL: INTEGRATE-PROCESS
.  GOAL: ANALYZE-PROCESSSS
.  .  GOAL: READ-AND-INTEGRATE-DIAGRAMS
.  .  [select GOAL: USE-ALTERNATE-METHOD
.  .  .  SELECT-DIAGRAM
.  .  .  FOLLOW-ACTIONS
.  .  .  FOLLOW-ACTIONS-ON-OTHER-DIAGRAM
.  .  .  GOAL: USE-BATCH-METHOD
.  .  .  SELECT-DIAGRAM
.  .  .  FOLLOW-ACTIONS
.  .  .  TAKE-OTHER-DIAGRAM
.  .  .  FOLLOW-ACTIONS
.  .  .  COMPARE-ORDER]
.  GOAL: DESIGN-PROCESS
.  .  GOAL: DRAW-PROCESS-COMPONENTS
.  .  [select GOAL: USE-TEMPLATE-METHOD
.  .  .  IDENTIFY-SENDERS
.  .  .  DRAW-SENDERS
.  .  .  GOAL: DRAW-ACTIVITIES
.  .  .  .  IDENTIFY-ACTION
.  .  .  .  [select DRAW-SEPARATE
.  .  .  .  .  IDENTIFY-SENDER
.  .  .  .  .  IDENTIFY-RECEIVER
.  .  .  .  .  SPECIFY-ACTION
.  .  .  .  DRAW-BULK
.  .  .  .  .  IDENTIFY-SENDER
.  .  .  .  .  SPECIFY-ACTION/RECEIVER]
.  .  .  GOAL: USE-SEQUENTIAL-METHOD
.  .  .  .  GOAL: DRAW-ACTIVITIES
.  .  .  .  .  IDENTIFY-ACTION
.  .  .  .  .  [select DRAW-SEPARATE
.  .  .  .  .  .  IDENTIFY-SENDER
.  .  .  .  .  .  SPECIFY-SENDER
.  .  .  .  .  .  IDENTIFY-RECEIVER
.  .  .  .  .  .  SPECIFY-RECEIVER
.  .  .  .  .  .  SPECIFY-ACTION
.  .  .  .  .  DRAW-BULK
.  .  .  .  .  SPECIFY-SENDER/ACTION/RECEIVER]]

```

Figure 3 - GOMS Model of Diagram Integration

3.2. Effects of Diagrammatic Representation on the Cognitive Process of Diagram Integration

The above task analysis model of process integration through diagram integration casts important insights as to how different diagrammatic representations can have an effect on the cognitive behavior of integrating the process diagrams. Basically, behavior in the diagram integration task depends upon which methods are selected in order to achieve the lower level sub-goals of analyzing the individual diagrams (GOAL: READ-AND-INTEGRATE-DIAGRAMS) and of designing the integrated diagrams (GOAL: DRAW-PROCESS-COMPONENTS). In other words, the criteria for selecting among alternative methods will determine the diagram integration behavior. This paper argues that method selection will be guided by the diagrammatic representation of the process diagrams. In other words, the decomposition and organized layout of information will influence the selection of which method to apply both in analyzing the individual diagrams as well as in designing the

integrated process.

First, the perceptual explicitness arising from decomposition is expected to provide effective visual cues that trigger the recognition of relevant information from different diagrams (Green, Petre and Bellamy, 1991). For example, as can be seen in Figure 1, the entity “John” can be more easily recognized in Figure 1b than in Figure 1a. Therefore, if a diagram has a reference to the entity “John”, it would be more easily recognized when information decomposition is supported than when not supported. Once such recognition is triggered, the problem solver will be induced into switching diagrams in order to relate the relevant information among different diagrams (Narayanan, Suwa and Motoda, 1995). Therefore, decomposition of the process components will induce the problem solver into applying the *ALTERNATE-METHOD* rather than the *BATCH-METHOD* for reading and integrating the processes because the visual cues will motivate the problem solver to switch diagrams to integrate relevant information. Otherwise, if decomposition is not supported, the same information may not look visually similar because of the different perspectives between the diagrams (e.g., the same activity "John sells book to Mary" may be represented as "Mary buys book from John" in another diagram). Therefore, the diagrammatic representation will induce the problem solver into applying the *BATCH-METHOD*, because without explicit visual cues it would be difficult for the problem solver to recognize the point to switch to the other diagram. In addition, decomposition can also have an effect in the design phase of the process integration task. If decomposition is supported, then the components of the process model can be specified separately in sequence (*DRAW-SEPARATE-METHOD*), whereas the components need to be specified together in bulk if decomposition is not supported (*DRAW-BULK-METHOD*).

Second, if some elements from the components of the process model can be spatially organized with a layout, then these will be represented in advance. For example, if the entities of the process model (i.e., the senders and/or receivers) can be represented as divided spaces or axes, these entities will be represented first as a basic template, and only the activities (i.e., the "does what") will have to be specified in relation to the template to complete the process model. Therefore, the provision of organized layout will induce the problem solver into applying the *TEMPLATE-METHOD* rather than the *SEQUENTIAL-METHOD*. Otherwise, if the diagrammatic representation does not support layout organization, the diagrammatic representation will induce the problem solver into following the *SEQUENTIAL-METHOD* for specifying the process activities and relevant entities, because separate space divisions or axes cannot be assigned to entities or activities a priori. The effects of the visual grammar of the diagrammatic representation are summarized in Figure 4.

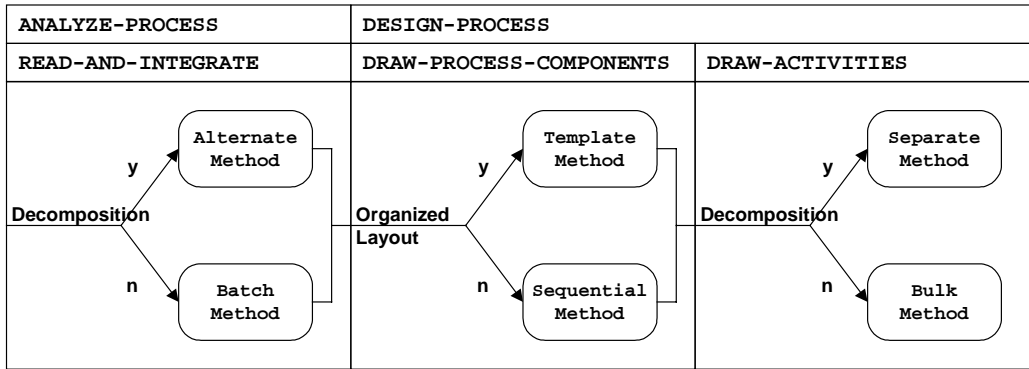


Figure 4 - Effects of Diagrammatic Representation on Diagram Integration Behavior

3.3. Effects of Visual Representation on Diagram Integration Performance

From the above discussion, we may notice that there exist alternative methods for achieving the same sub-goal. The sub-goal of READ-AND-INTEGRATE-DIAGRAMS can be achieved by the ALTERNATE-METHOD or the BATCH-METHOD, and the sub-goal of DRAW-PROCESS-COMPONENTS can be achieved by the SEQUENTIAL-METHOD or the TEMPLATE-METHOD. Moreover, the third level goal of DRAW-ACTIVITIES can be achieved by the DRAW-SEPARATE-METHOD or the DRAW-BULK-METHOD. In this section, we explain how different methods for achieving the same sub-goal may result in different performance outcomes of the diagram integration task.

In the case of the sub-goal of READ-AND-INTEGRATE-DIAGRAMS the ALTERNATE-METHOD will have a higher probability of successful integration than the BATCH-METHOD. Since the problem solver does not know a priori how the diagrams of the individual sub-processes should be integrated, deliberately trying to integrate while reading the diagrams will help the problem solver induce the critical inferences such as inferring where the process starts and where the different diagrams meet. This information will be attended to and thus located in working memory while reading and integrating the individual diagrams. Consequently, the BATCH-METHOD will have a lower probability of successful integration because this method requires a vast amount of information to be retained in working memory in order to induce the critical inferences. Since working memory has a limited capacity and a fast decay rate (Baddeley, 1986), some of the information needed in inducing the critical inferences may not be present in working memory or will be difficult to recall to working memory when needed.

In the case of the sub-goal of DRAW-PROCESS-COMPONENTS, the TEMPLATE-METHOD will be easier than the SEQUENTIAL-METHOD in terms of processing load and memory load. Processing and memory load will be reduced since the problem solvers will not have to keep track of all the components involved in each process activity. This is because many components are already drawn in the template and can easily be identified with

the `TEMPLATE-METHOD` (Kalyuga, Chandler and Sweller, 1997), therefore the probability of omitting necessary components is reduced. Therefore, the `TEMPLATE-METHOD` will have a higher likelihood of success than the `SEQUENTIAL-METHOD`.

In addition, the specification of the activities will also have a higher probability of success if components can be specified separately one at a time as individual graphical tokens (`DRAW-SEPARATE-METHOD`) rather than in bulk (`DRAW-BULK-METHOD`). With the `SEPARATE-METHOD`, components of the process model have to be separately and explicitly specified. The explicit specification acts as an external constraint on the design activity (Zhang and Norman, 1994) and makes the omission of process components highly improbable. Whereas with the `BULK-METHOD` these constraints no longer apply and the problem solver is more prone to omitting some of the components. The effects of the decomposition and layout organization structures of the diagrammatic representation are summarized in Table 1.

Table 1 - Effects of Visual Representation on Diagram Integration Performance

Sub-goal	Method	Performance	
		Analysis	Design
READ-AND- INTEGRATE-DIAGRAMS	ALTERNATE-METHOD	High	
	BATCH-METHOD	Low	
DRAW-PROCESS- COMPONENTS	TEMPLATE-METHOD		High
	SEQUENTIAL-METHOD		Low
DRAW-ACTIVITIES	SEPARATE-METHOD		High
	BULK-METHOD		Low

4. EXPERIMENTAL STUDY

An experiment was conducted to investigate the effects of diagrammatic representation on the cognitive behavior of integrating process diagrams. This next section presents the experimental design.

4.1. Subjects

Thirty-two students participated in the experiment for course credit. All subjects were junior or senior undergraduate students at Yonsei University, and were enrolled in a systems analysis and design course. The course required the subjects to analyze and model real-world business system processes. The experiment was conducted near the end of the semester-long class by which time all the subjects had completed several homework assignments and a semester long project. The subjects were therefore very comfortable with the diagrammatic representations used in the experiment. None of the subjects had prior knowledge of the system processes used as the experimental stimuli.

4.2. Experimental Design

A 4×4 analysis of variance (ANOVA) design (Latin Square Confounded Factorial Design: LSCF 8-4²) was used as the experimental design (Kirk, 1995). Each subject performed four process integration tasks with four different system processes, each modeled with a different diagram. The sequence in which the diagrams were presented was completely randomized. Ultimately, each process-diagram combination was performed by eight subjects³.

4.3. Procedure

The experimental sessions were divided into four sections. First, subjects were given instructions about the general nature of the experiment and told that verbal protocols would be collected. Second, the subjects were trained to *think-aloud* with several traditional training tasks until they were comfortable with thinking aloud (Ericsson and Simon, 1993). The experimenter then presented the subjects with a one-page overview of the experimental material that presented background information about the business system whose processes were modeled and also about the individual processes he/she was to integrate. Finally, the subjects were provided with the two diagrams for each integration task. The subjects were asked to understand the given process as a whole and to integrate the two diagrams by drawing the integrated process into a single diagram. In order to control experimenter bias, the experimenter provided all the instructions following a standard script. Finally, there was no time limit for the integration tasks.

4.4. Experimental Material

For each task, the subjects were provided with two diagrams depicting a single process modeled from two different systems' perspectives. One of the diagrams was drawn from a company's headquarters' perspective, while the other was from the company's branch's perspective. Each diagram depicted a portion of the whole system process in that the activities within the other systems were encapsulated. The two diagrams were presented on paper and the final integrated process diagram was also drawn on paper by hand, thus could be spread out physically in parallel

Four real-world system processes modeled with four different diagrams were used as the experimental

³ A pure between subject design would have been more effective for the purpose of the study. However, considering the limited number of subjects available for the experiment a within subject design was inevitably adopted. The latin-square design was also employed so that some of the carry-over strategies could be minimized. Preliminary analyses of the data showed that the order of diagrams presented did not have an effect on the integration performance ($F(7, 24)=0.94$, ns).

material. Several criteria were employed in selecting the system processes. First, the activities of the process needed to be equally distributed between the two diagrams. Second, the process needed to be difficult enough so that the integration task was not too easy, but also simple enough so that the subjects could finish the integration tasks within the experimental session. The complexity of the processes was as follows. There was an average of 9.5 entities in each process, with an average of 12.5 messages between these entities. Three of the four processes had one overlap between the diagrams, whereas the remaining process had two between diagram overlaps. There was also an average of 1.5 cross-diagram messages for each process.

The diagrams used in modeling the system processes were four process model diagrams. The four diagrams were the sequence diagram (SD), the activity diagram (AD), the collaboration diagram (CD) and the activity flow diagram (AFD). Three of the diagrams (SD, AD and CD) were adopted from an existing object-oriented software engineering methodology: the Unified Modeling Language, UML 1.0 (Rational, 1997), whereas one (AFD) was newly designed for the purpose of the experiment. The diagrammatic representation of each diagram is explained in detail below.

The sequence diagram (SD) models relevant objects represented by vertical lines with the name of the object at the top of the vertical line. Interactions between objects are conveyed via messages sent to and from objects depicted as arrows between the vertical lines. The sequence of messages is represented through spatial placement of the messages. The messages are processed from top to bottom (earliest to latest). An example of a sequence diagram is provided in Figure 5. The example process starts with a recursive message of the branch warehouse manager who performs the inventory check. Then the branch warehouse manager performs an inventory replenish order to the branch clerk, who in turn inputs the order data in the branch information system. The branch's system transfers the order to the headquarters, which in turn faxes the order list to the suppliers. The supplier then delivers the goods ordered to the branches' warehouse where the warehouse manager receives the goods and its receipt. The receipt is transferred to the branch clerk who finally enters the data in the branch's information system. We may see from the example that the sequence diagram supports decomposition since the components of the process model are separately represented (i.e., the senders and receivers as vertical lines and actions from sender to receiver as arrows with the message label). Furthermore, the sequence diagram supports an organized layout in that all the objects are organized vertically showing the origin or the destination of the messages.

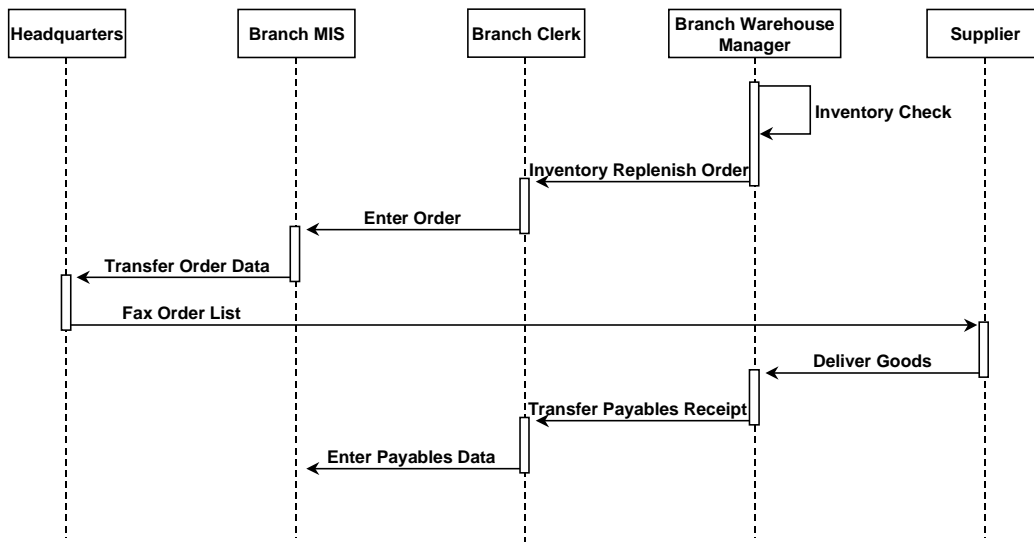


Figure 5 - Example of the Sequence Diagram (SD)

The activity diagram (AD) also models processes in another diagrammatic manner. In the activity diagram the objects are represented as swimlanes with the names of the objects at the top. However, unlike the sequence diagrams, where interaction is conveyed via messages, the activity diagram communicates the interaction via objects' activities depicted as bubbles with textual labels. The sequence of activities is represented by the arrows to and from activities. Since interactions are conveyed via activities only the senders are represented in the activity diagram; the receivers need to be explicitly stated within the activity bubbles. In Figure 6 an example of the activity diagram is provided for the same process used in Figure 5. From the example activity diagram, we may see that decomposition is not supported in that the receivers and actions are represented together within the activity bubble. However, layout organization is supported in that the senders are organized vertically showing the ownership of each process activity (i.e., all activity bubbles within a swimlane is performed by the actor specified as the owner of the swimlane).

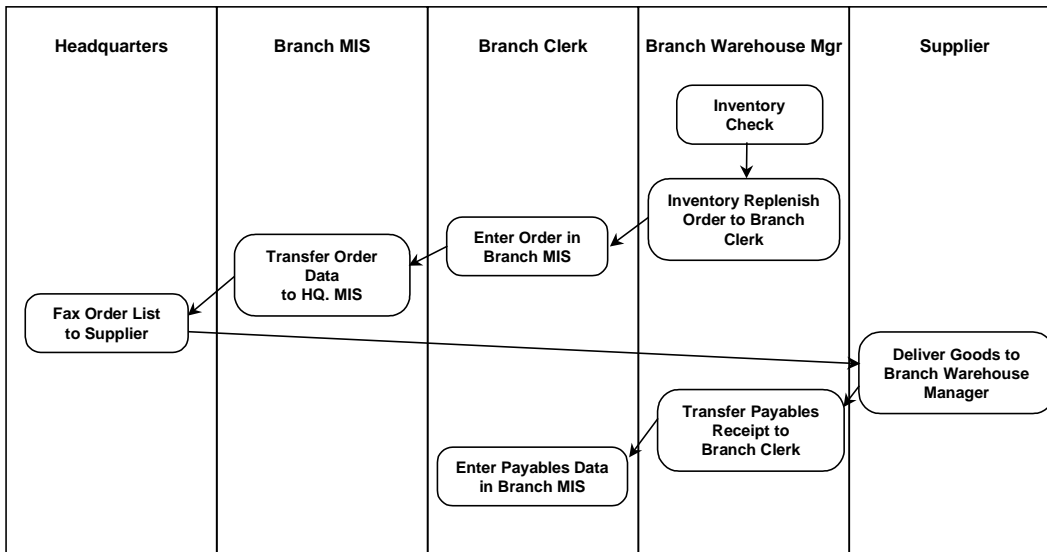


Figure 6 - Example of the Activity Diagram (AD)

The collaboration diagram (CD) is a diagram consisting purely of boxes and arrows diagram. Each box represents an object while the lines between the boxes represent the relationship between the objects. Interactions between objects are conveyed via messages sent to and from objects and are depicted as arrows with textual labels showing the name of the message being sent. The sequence of the messages is represented by a sequence number placed at the beginning of the label for each message. An example of a collaboration diagram is provided in Figure 7. We may see from the example collaboration diagram that the collaboration diagram supports decomposition since the components of the process model are separately represented (i.e., the senders and receivers as boxes, and actions from sender to receiver as textual labels with a small arrow indicating the direction of the action). However, an organized layout is not incorporated in the collaboration diagram since the components are arranged randomly within the presentation space.

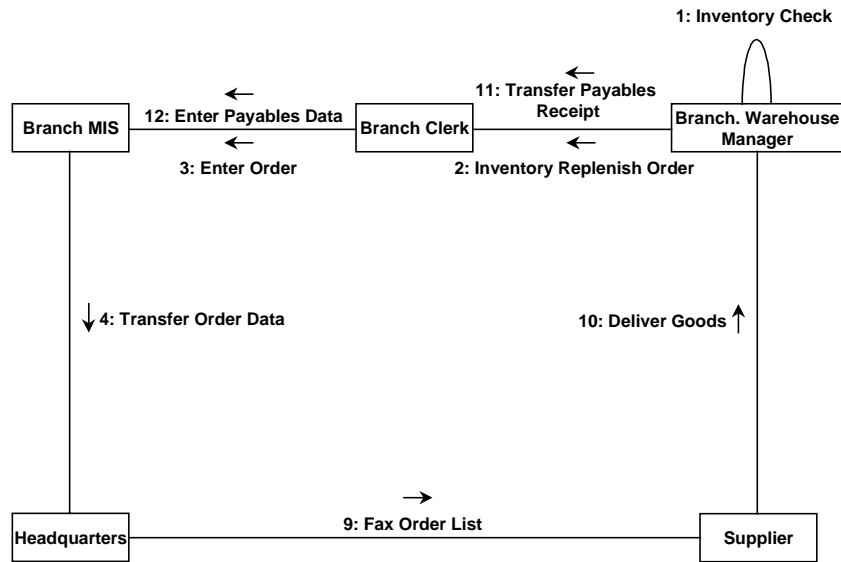


Figure 7 - Example of the Collaboration Diagram (CD)

The activity flow diagram (AFD) is similar to the activity diagram in that the interaction is conveyed via activities, however the activities are not arranged within swimlanes. The sender objects are specified as textual labels within the activity bubbles. The sequence of activities is also represented via arrows to and from activities. An example of an activity flow diagram is provided in Figure 8. With the activity flow diagram decomposition of the process components is not supported in that the sender, receiver and action components for each process activity is expressed together within each activity bubble. Even though the activities are organized from top to bottom following the order of the process, the components are not organized in space in that the senders and receivers do not cover any separate area of the presentation space to show the ownership of the activities.

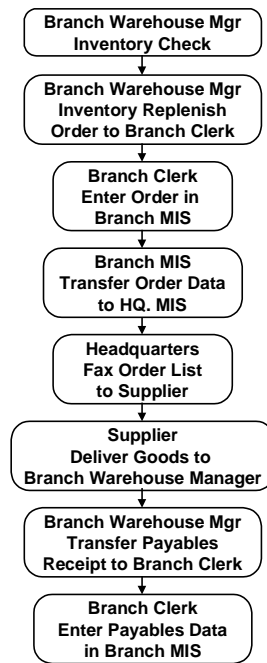


Figure 8 - Example of the Activity Flow Diagram (AFD)

In summary, the diagrams used as the experimental material differed with respect to how they decompose and organize the process components in terms of an organized layout. The four diagrams can be categorized as supporting (or not supporting) decomposition and/or organized layout of process components. The manipulation of the experimental material is summarized in Table 2.

Table 2 - Visual Language of the Experimental Diagrams

		Decomposition	
		Yes	No
Organized Layout	Yes	SD	AD
	No	CD	AFD

4.5. Hypotheses for Diagram Integration Performance

As discussed in the task analysis model of process integration the manipulation of the diagram treatment is hypothesized to have an effect on the final integration performance. The GOMS model of diagram integration predicts that providing decomposition will support the internal integration activity in the analysis phase and also the specification activity in the design phase, whereas providing an organized layout will support the design activity. Therefore, we hypothesize that decomposition will have a main effect on the analysis and design performance and that decomposition and organized layout will affect the design performance. The effects are summarized in Table 3; which is a combination of Tables 1 and 2. The arrows in the task performance effects refer to the cell to which the performance is compared. For example, the

analysis performance when both decomposition and organized layout are supported will be higher than when only organized layout is supported.

Table 3 - Hypothesis for Diagram Integration Performance

		Decomposition	
		Yes	No
Organized Layout	Yes	<u>Diagram ; SD</u> <u>Selection of Methods :</u> ALTERNATE TEMPLATE SEPARATE <u>Task Performance :</u> Analysis : Higher than (→) Design : Higher than (→) Higher than (↓)	<u>Diagram ; AD</u> <u>Selection of Methods :</u> BATCH TEMPLATE BULK <u>Task Performance :</u> Analysis : Lower than (←) Design : Lower than (←) Higher than (↓)
	No	<u>Diagram ; CD</u> <u>Selection of Methods :</u> ALTERNATE SEQUENTIAL SEPARATE <u>Task Performance :</u> Analysis : Higher than (→) Design : Lower than (↑) Higher than (→)	<u>Diagram ; AFD</u> <u>Selection of Methods :</u> BATCH SEQUENTIAL BULK <u>Task Performance :</u> Analysis : Lower than (←) Design : Lower than (↑) Lower than (←)

4.6. Data Analysis

Two types of data were collected from the experiment: subjects' performance data and behavior data. The performance data were collected from the subjects' finally integrated diagrams and the behavior data were collected from the subjects' concurrent verbal and action protocols.

4.6.1. Integration Performance Data

The purpose of the performance data analysis was to check whether the experimental stimuli had an effect on the subjects' diagram integration performance. The steps of the analysis were coding of the subjects' final design outputs, computation of the kappa ratio for inter-coder reliability, and finally between group comparison of the integration task performance.

First, each subject's final design outputs were evaluated independently by two coders. The evaluation consisted of counting the number of analysis and design errors present in the integrated diagram. The analysis errors were concerned with the subjects' inability to construct a correct internal representation of the integrated process. Therefore, in order to infer from the subject's final integrated design that a subject failed

to construct a correct internal representation, errors indicating the subject's failure to specify a correct sequence of activities in the final process diagram were coded as analysis errors. This is because the sequence of activities was believed to be the most direct measure of the internal representation. The design errors concerned the subject's inability to correctly represent the components of the process diagram. These include the errors of omitting or incorrectly specifying components that did not influence the sequence of the activities. This is because senders, receivers and actions can be incorrectly drawn even with the correct internal representations.

Second, in order to check the reliability of the coding between the two independent coders, the kappa ratio was computed (van Someren, Barnard and Sandberg, 1994). The two independent codings of the subject's outputs turned out to be significantly reliable ($\kappa = 0.937$).

4.6.2. Integration Behavior Data

In order to understand how the different diagrammatic representations affected the subjects' analysis and design behaviors, exploratory protocol analysis was performed to analyze the subjects' cognitive processes (Ericsson and Simon, 1993; van Someren, et al., 1994). Verbal utterances during problem solving were the major source of data in the protocol analysis. Action protocols were also collected along with the verbal protocols, since the use of both verbal and action protocols was expected to provide a more complete trace of problem solving behavior (Rist, 1989). Two important preconditions that must be met to use protocol analysis are to identify an appropriate unit of analysis, and to develop an objective coding system for each unit. We elected to use the episode as the unit of analysis considering the volume of protocol data from the experiment. Episodes are small self-contained phases of highly organized activity (Newell and Simon, 1972). The coding system is discussed below.

The behavior data in the analysis phase was coded based mainly on the subjects' verbal protocols because the subjects' analysis behaviors could be faithfully captured only by the verbal utterances of the subjects. The analysis data was concerned with which part of the diagram the subjects were investigating at a given time. All the information presented in the diagrams was translated into natural language and then indexed (e.g., information concerning the first process activity in the branch diagram was indexed as "b1", and information concerning the first activity in the headquarters diagram as "h1" and so on). The subjects' verbal protocols were coded as the corresponding indices. The analysis data was summarized into an Analysis Behavior Graph (ABG) which is similar to the Problem Behavior Graph (PBG) proposed by Newell and Simon (1972). The ABG indicates the sequence in which subjects investigated the information presented in the diagrams. An example of the ABG is shown in the results section (Figures 8 and 9). With the ABGs, the subjects' analysis behaviors were categorized as applying either the `ALTERNATE-METHOD` or the `BATCH-METHOD`. If the subjects

showed explicit transitions to another diagram before all the information on one diagram was fully analyzed, the subjects' analysis behavior was coded as `ALTERNATE-METHOD`. Otherwise, if the subjects' analysis behavior showed a transition between the diagrams only after having fully analyzed all the information in one diagram, it was coded as `BATCH-METHOD`. The distinction between these two methods were based on the ABGs that shows the analysis behavior until all the information presented in the diagrams were investigated at least once, because this portion of the analysis behaviors clearly shows the impact of the perceptual cues of the diagrammatic representation without the influence of conceptual inferences.

The design behavior data from the subjects' design activity was coded mainly based on the action protocols because most of the subjects' design behavior was the actual drawing of the diagram on paper. The design data was concerned with which component of the process model the subjects were drawing at a given time. In order to code the design data objectively, all the information from the diagrams were indexed by component of the process model. For example, entities presented in the branch diagram was indexed as "BR-e *n*", entities in the headquarters diagram as "HQ-e *n*", and the actions as "action *n*" where *n* refers to an index number because there were multiple entities and actions. The subjects' action protocols were coded in terms of the corresponding indices. The design data was summarized in the form of a Design Behavior Graph (DBG) which portrays the sequence in which the subjects drew the components of the process model to design the final integrated process diagram. An example of the DBG is also provided in the results section (Figures 10 to 13). Based on the DBGs, the subjects' design behavior was categorized as applying either the `TEMPLATE-METHOD` or the `SEQUENTIAL-METHOD`. If the subjects drew all the entities before specifying any process activities, then the subjects' design process was coded as `TEMPLATE-METHOD`. On the other hand, if the subjects specified the relevant entities and actions for each process activity in sequence then the subjects' design behavior was coded as `SEQUENTIAL-METHOD`. Furthermore, when specifying the sequence of activities, if the remaining components were specified in sequence the design behavior was coded as `DRAW-SEPARATE-METHOD`. Otherwise, if they were specified in bulk, the behavior was coded as `DRAW-BULK-METHOD`.

5. RESULTS

5.1. Integration Performance

This section presents the results of the experiment in terms of integration performance of analysis and design.

5.1.1. Integration Performance - Analysis Errors⁴

The subjects' analysis performance is summarized in Table 4 which shows the number of subjects that committed at least one analysis error for the integration task. For the thirty-two subjects in each group, three subjects in the SD group and two subjects in the CD group failed to construct the correct sequence of activities for the integrated process. On the other hand eight subjects in both the AD and AFD groups committed analysis errors. A Cochran-Mantel-Haenszel test on the subjects' analysis performance data showed that the subjects that used diagrams that supported decomposition of the process components (SD and CD) showed higher performance in constructing the correct integrated process than the subjects that used diagrams that did not support decomposition (AD and AFD) ($CMH=6.79^5$, $df=1$, $p<0.01$).

Table 4 - Frequency of Error/Correct Subjects (Analysis)

Decomposition	Organized Layout	Diagram	Analysis	
			Error	Correct
Y	Y	SD	3	29
Y	N	CD	2	30
N	Y	AD	8	24
N	N	AFD	8	24

We also compared the average number of analysis errors committed during the integration task, which is summarized in Table 5⁶. We may see that there was a difference in the average number of analysis errors between the diagram groups ($F(3,39)=4.04$, $p<0.05$). When the performance was compared through decomposition of the process components and layout organization, we may see that decomposition had an effect on the average number of analysis errors ($F(1,39)=11.24$, $p<0.005$), while layout organization did not have any effect ($F(1,39)=0.18$, ns).

Table 5 - Average Number of Analysis Errors

		Decomposition	
		Yes	No
Organized Layout	Yes	SD (0.0938)	AD (0.2813)
	No	CD (0.0625)	AFD (0.3750)

⁴ The problem solving times were measured. The difference in problem solving times was not statistically significant between diagrams ($F(3, 39)=1.59$, ns). The difference was also insignificant when compared between distribution and organized layout (Distribution : $F(1, 39)=3.09$, ns; Organized layout : $F(1, 39)=0.87$, ns).

⁵ The results of the statistical analysis should be interpreted with caution for the empirical data in Tables 4 and 6 does not fully comply with the assumptions of the Cochran-Mantel-Haenszel (CMH) statistic, which, like the Chi-square statistic, has the assumption that the frequencies of all the cells in the contingency table should be at least 5.

⁶ The average number of errors is very small because of the small number of subjects that actually committed analysis

In summary, diagrammatic representations supporting decomposition resulted in less analysis errors committed than when not provided. The results are consistent with the GOMS model of diagram integration in that decomposition had an effect on the analysis performance, but that layout organization did not have such an effect.

5.1.2. Integration Performance - Design Errors

The design performance of the subjects was also analyzed first by comparing the number of subjects that committed design errors for the integration task. The subjects' analysis performance is summarized in Table 6. Eleven out of thirty-two subjects in the SD group and sixteen in the CD group committed design errors, whereas nearly all subjects in the AD (thirty subjects) and AFD (all subjects) groups committed design errors. A Cochran-Mantel-Haenszel test on the subjects' design performance data showed that the subjects given the diagrams that supported decomposition of the process components (SD and CD) showed higher performance in correctly designing process components than the subjects that used diagrams that did not support decomposition (AD and AFD) ($CMH=45.11$, $df=1$, $p<0.001$). In addition, the use of diagrams having an organized layout also resulted in less design errors than when diagrams did not have an organized layout ($CMH=2.75$, $df=1$, $p<0.1$).

Table 6 - Frequency of Error/Correct Subjects (Design)

Decomposition	Organized Layout	Diagram	Design	
			Error	Correct
Y	Y	SD	11	21
Y	N	CD	16	16
N	Y	AD	30	2
N	N	AFD	32	0

The average number of design errors was also compared between groups. We can observe from Table 7 substantial differences in the average number of design errors among the four diagrams ($F(3,39)=53.46$, $p<0.0001$). Decomposition of process components also had an effect on the average number of design errors ($F(1,39)=131.61$, $p<0.0001$). However, unlike the analysis performance, the design performance was also affected by layout organization ($F(1,39)=22.51$, $p<0.0001$).

Table 7 - Average Number of Design Error Types

		Decomposition	
		Yes	No
Organized Layout	Yes	SD (0.531)	AD (2.156)
	No	CD (0.938)	AFD (3.468)

errors.

The above results suggest that diagrammatic representations supporting decomposition and layout organization resulted in less design errors than those not provided. Results of the design performance are again consistent with the GOMS model of diagram integration in that both decomposition and layout organization of the process components had an effect on the design performance.

These results of the subjects' analysis and design performance confirm the GOMS model of diagram integration and establish causal relationships between the decomposition / layout organization of the diagrammatic representation and the subjects' final integration performance. The next section presents the results of the protocol analysis of the subjects' behavior in an attempt to understand in detail how the different decomposition and layout affected the actual cognitive processes, resulting in such differences in analysis and design performances.

5.2. Process Results

Although comprehensive analysis of the verbal and action protocols most accurately shows the dynamic nature of the diagram integration behavior, it is impractical to analyze all the data in detail for all the subjects because of the tremendous amount of verbal and action protocol data. Therefore, for each diagram four representative subjects were chosen for detailed exploratory protocol analysis. In selecting the representative subjects, the average error rate was calculated for each process-diagram combination. Then each subject's deviation from the mean was computed and the subject with the least deviation from the mean was selected as the representative subject for that group. The protocol analysis results of the 16 representative subjects are discussed in the next section.

5.2.1. Integration Process - Analysis Behavior

The GOMS model of diagram integration presented in section 3.1. predicts two different methods for reading and integrating the multiple diagrams (i.e., the *ALTERNATE-METHOD* and the *BATCH-METHOD*). In order to determine which of the two methods was chosen, the subjects' Analysis Behavior Graphs (ABG) were analyzed. Figure 9 and Figure 10 show the ABGs for two representative subjects. Subject 11 (Figure 9) was in the SD group (decomposition supported) and subject 15 (Figure 10) was in the AFD group (decomposition not supported). In the ABGs, the Branch and Headquarters partition represents different diagrams. The second column represents the different information presented in the respective diagrams. Each black box indicates the specific information a subject was investigating. Time flows from left to right to show the dynamics of the analysis process. An explicit effort of integration is coded when a subject uttered two pieces of information in different diagrams in one episode. The gray shadings in the ABGs represent these explicit efforts of integration.

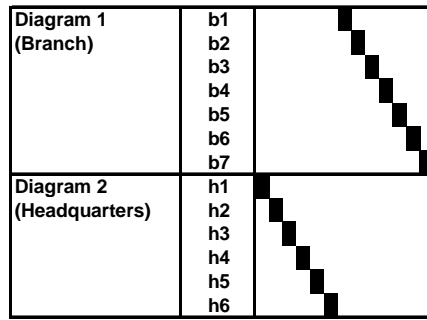


Figure 10 – Analysis Behavior Graph (ABG) of Sbj 15

The integration behaviors of all the other representative subjects were analyzed and the results are summarized in Table 8. Table 8 shows the methods the subjects applied for reading and integrating the diagrams. The GOMS model prediction for method selection is also presented in the table at the third column. Since SD and CD support decomposition, and AD and AFD do not, the model predicts the selection of the ALTERNATE-METHOD for SD and CD, and the selection of the BATCH-METHOD for AD and AFD. The overall fit between the GOMS model prediction and the subjects' actual behavior was 75% (12 out of 16 representative subjects). The data for the SD group showed a perfect fit with the GOMS model prediction (100%), the fit was average for the CD and AFD groups (75%), and lowest for the AD group (50%). The results show that decomposition of the process components resulted in more subjects applying the ALTERNATE-METHOD rather than the BATCH-METHOD ($CMH=4.00$, $df=1$, $p<0.05$).

Table 8 - Method Selection for Read and Integrate Diagrams

Decomposition	Diagram	Predicted Method	Correct	False	Model Fit (%)
Yes	SD	ALTERNATE	4	0	100
	CD	ALTERNATE	3	1	75
No	AD	BATCH	2	2	50
	AFD	BATCH	3	1	75
Total			12	4	75

In order to infer why the application of different methods had an effect on the number of analysis errors, the diagram transition data was analyzed⁷. Diagram transition refers to the number of times a subject switches between the two diagrams. It was assumed that visual mapping would induce the subjects to perform more transitions, which in turn would enable the critical information to be readily available in working memory.

⁷ The total number of episodes was also compared between the experimental groups. No significant differences were found between diagram groups ($F(3,9)=0.24$, ns), or for contrasts of distribution ($F(1,9)=0.07$, ns) and organized layout ($F(1,9)=0.01$, ns). Therefore, we may infer that it was not the amount of exploration but how the exploration was performed that influenced the integration performance.

Table 9 shows the percentage of transition episodes from the whole protocols.

Table 9 - Percentage of Diagram Transitions (%)

Decomposition	Diagram	Diagram Transitions (%)	Average
Yes	SD	33.10	32.63
	CD	32.16	
No	AD	25.03	23.19
	AFD	21.35	
Average		27.91	

The subjects in the SD and CD groups showed a higher percentage of diagram transition than the AD and AFD groups. The difference was significant between the four diagram groups ($F(3,9)=5.10, p<0.05$) and also between contrasts of decomposition (SD and CD vs. AD and AFD; $F(1,9)=14.16, p<0.005$). However, layout organization did not have an effect on the amount of diagram transitions (SD and AD vs. CD and AFD; $F(1,9)=0.85, ns$). These results imply that the decomposition of the process components enabled subjects to perform more transitions between those diagrams. This in turn made the relevant information readily available while integrating the diagrams, thereby resulting in better analysis performance.

5.2.2. Integration Process - Design Behavior

The GOMS model of diagram integration predicts different methods for designing the integrated process based on layout organization and decomposition. Based on layout organization, the subjects select either the `TEMPLATE-METHOD` or the `SEQUENTIAL-METHOD`. Based on decomposition, different methods of `DRAW-ACTIVITIES` can be applied, `DRAW-SEPARATE-METHOD` or `DRAW-BULK-METHOD`. As discussed earlier, the behavior in the design phase was analyzed with the subjects' Design Behavior Graphs (DBG) showing which component of the integrated process model the subject draws at a given time. The design behavior is represented by plotting the various components of the process model on the DBG. The entities are plotted as either "s" for senders or as "r" for receivers. All the actions are represented as "a". As with the ABGs the dynamics of the design behavior is conveyed by the time sequence from left to right.

Figure 11 shows the DBG of subject 15 and Figure 12 that of subject 29. The subjects were in the SD and AD groups respectively, therefore their diagrams had an organized layout of the process components. We may see from the subjects' DBGs that the design behaviors of subjects were in accord with the model. They show the application of the `TEMPLATE-METHOD` because all the senders ("s") were specified prior to the specification of the receivers ("r") and actions ("a"). As for the goal of drawing the process activities, subject 15 applied the `DRAW-SEPARATE-METHOD` (i.e., specifying only the "a"), whereas subject 29 adopted the `DRAW-BULK-METHOD` (i.e., specifying the "r" with the "a"). Again, these behaviors are in accord with the model because the diagrammatic representation of subject 29 (AD) did not support decomposition while that of subject 15 did

(SD).

Entities	BR	
	BR-e1	s
	BR-e2	s
	BR-e3	s
	HQ	
	HQ-e1	s
	HQ-e2	s
	HQ-e3	s
	A1	s
Actions	action1	
	action2	a
	action3	a
	action4	a
	action5	a
	action6	a
	action7	a
	action8	a
	action9	a
	action10	a
	action11	a
	action12	a
	action13	a
	action14	a

Figure 11 – Design Behavior Graph (DBG) of Sbj15 (SD)

Entities	BR	
	BR-e1	s
	BR-e2	s
	BR-e3	s
	HQ	
	HQ-e1	s
	HQ-e2	s
	HQ-e3	s
	A1	s
Actions	action1	a
	action2	a
	action3	a
	action4	a
	action5	a
	action6	a
	action7	a
	action8	a
	action9	a
	action10	a
	action11	a
	action12	a

Figure 12 – Design Behavior Graph (DBG) of Sbj29 (AD)

Figure 13 shows the DBG of subject 19 and Figure 14 that of subject 17. The subjects were in the CD and AFD groups respectively, therefore their diagrams did not have an organized layout of the process components. Again, the subjects' DBGs reveal that the design behaviors of subjects were in accord with the GOMS model. They show the application of the SEQUENTIAL-METHOD because a template of all the senders was not drawn prior to the specification of the activities. The drawing behavior followed the sequence of process activities while specifying the senders ("s") and receivers ("r") relevant to each activity ("a"). As for the goal of drawing the process activities, subject 19 adopted the DRAW-SEPARATE-METHOD (i.e., specifying the "s", "r" and "a" sequentially when needed), whereas subject 17 applied the DRAW-BULK-METHOD (i.e., specifying the "s" and the "r" with the "a"). Again, these behaviors are in accord with the model because the diagrammatic representation of subject 17 did not support decomposition (AFD) while that of subject 19 did (CD).

Entities	BR	
	BR-e1	s
	BR-e2	r
	BR-e3	r
	HQ	
	HQ-e1	r
	HQ-e2	r
	HQ-e3	r
	A1	r
Actions	action1	a
	action2	a
	action3	a
	action4	a
	action5	a
	action6	a
	action7	a
	action8	a
	action9	a
	action10	a
	action11	a
	action12	a
	action13	a
	action14	a

Entities	BR	r
	BR-e1	s
	BR-e2	r
	BR-e3	r
	HQ	r
	HQ-e1	s
	HQ-e2	r
	HQ-e3	r
	A1	r
Actions	action1	a
	action2	a
	action3	a
	action4	a
	action5	a
	action6	a
	action7	a
	action8	a
	action9	a
	action10	a
	action11	a
	action12	a
	action13	a
	action14	a

Figure 13 – Design Behavior Graph (DBG) of Sbj19 (CD)

Figure 14 – Design Behavior Graph (DBG) of Sbj17 (AFD)

The design behaviors of all the representative subjects were analyzed and summarized in Table 10. Table 10 shows the methods the subjects selected for designing the integrated process. The GOMS model prediction for method selection is also presented in the table. The overall fit between the GOMS model prediction and the subjects’ actual behavior for designing the integrated diagram was 75% (12 out of the 16 subjects). The data for the SD group showed a perfect fit with the GOMS model prediction (100%), the fit was average for the CD and AFD groups (75%), and lowest for the AD group (50%). The results show that the organized layout of the process components resulted in more subjects applying the `TEMPLATE-METHOD` rather than the `SEQUENTIAL-METHOD` ($CMH=3.73, df=1, p<0.1$).

Table 10 - Method Selection for Design Integrated Business Process

Organized Layout	Diagram	Predicted Methods	Correct	False	Model Fit (%)
Yes	SD	TEMPLATE SEPARATE	4	0	100
	AD	TEMPLATE BULK	2	2	50
No	CD	SEQUENTIAL SEPARATE	3	1	75
	AFD	SEQUENTIAL BULK	3	1	75
Total			12	4	75

6. CONCLUSION AND DISCUSSION

6.1. Summary of the Results

The results of the experiment reveal that the way knowledge is decomposed and organized through layout in diagrams does have an effect on the manipulation task of process integration. As predicted by the cognitive model of diagram integration, decomposition of the process components had a positive effect on both the analysis and design activities. The average number of analysis and design errors was reduced when decomposition was furnished. Furthermore, layout organization had a positive effect on the design activities in that design errors were reduced with an organized layout.

Detailed protocol analysis of the representative subjects revealed how and why decomposition and layout organization of the diagrammatic representations might have affected the task performance of the analysis and design activities. First, in terms of *decomposition*, the subjects’ analysis behavior graphs (ABGs) show that when the components of the process model were decomposed the subjects undertook an alternating

integration behavior which had a higher potential of success than the non-alternating batch behavior. The perceptual explicitness arising from the decomposed information representation was expected to provide effective visual cues to alternate between different diagrams to look for relevant information in the other diagram (Narayanan, Suwa and Motoda, 1995). This alternating analysis behavior might also have reduced working memory load because the critical information in integrating the diagrams was more likely to be readily available as a result of the alternation. At the same time, the subjects' design behavior graphs (DBGs) show that decomposition played an important role in keeping track of important process components while drawing the integrated process diagram. When the diagrammatic representation of the process components supported decomposition, the representation forced the problem solver into explicitly specifying all the components, which made it hard to omit information. Whereas, when decomposition was not supported, the subjects tended to omit certain components in their designs, which finally led to more design errors. Second, in terms of *layout organization*, the design behavior graphs (DBGs) show that the subjects were induced into specifying the components that could be organized with a layout first as a basic template and then inserting the remaining process components later. We infer that the basic template provided the subjects with effective external memory aids for keeping track of necessary components and thereby, reduced errors when designing the integrated process.

In summary, the diagrammatic grammar of decomposition and layout organization was an important factor in selecting between diagram manipulation method of analysis and design when multiple were possible for attaining a particular goal. The selection of different methods resulted in different cognitive behaviors, which in turn produced different task performance in the diagram manipulation task.

6.2. Discussions and Implications

The limitations of this study stem from several factors, such as diagrammatic factors, GOMS model factors, and experimental factors. First, the results of the study show that how knowledge is represented in diagrams has an effect on how we manipulate these diagrams for problem solving. In particular, this study focused on the effects of decomposition and layout organization of information on the task of process integration. However, it must be noted that several additional diagrammatic factors may also have influenced the subjects' problem solving behavior. First, other grammatical rules (except decomposition and layout organization) such as color coding of the graphical primitives and spatial orientation have been found to influence reasoning with diagrams (Sweller, 1997). For example, an unanticipated consequence of visual grammar can make some of the diagrams more horizontally or vertically oriented whereas others do not have any specific orientation. Such factors would affect subjects' problem solving behavior, since they treat the time dimension differently, which is a very important factor in process diagrams. Second, factors concerning the medium of presentation can also have an effect. With the current proliferation of powerful computers, more and more

tasks are performed directly at a computer terminal rather than with pencil and paper. And the computer interface may provide different direct manipulation methods for diagrammatic problem solving, which may cause important navigational issues among multiple diagrams. For example, the computer VDU may only be able to display one diagram at one time instead of multiple diagrams laid out in parallel as was with our pencil-and-paper experiment conducted in this study. Whether the displays are provided in series or in parallel may be an important factor that may alter the diagram integration behavior (Woods and Watt, 1997). Further studies are required to investigate the impact of these factors that could not be effectively controlled in our experimental diagrams.

Second, the GOMS model presented in this paper also needs to be expanded to explain the experimental results more faithfully and to account for additional diagrammatic features. For example, the experimental data on the subjects' behavior was not perfectly consistent with the prediction of the GOMS model. Especially, we observed that decomposition and layout organization were less accurate in the prediction of the behaviors of the AD group in which half of the representative subjects did not follow the rule selection criteria proposed by our model. This may have been because the organized layout of the activity diagram (AD) caused a partial decomposition of the process components. Since the sender entities were organized as swimlanes, the specification of the senders was separated, but the action and receiver components remained aggregated. The partial nature of the decomposition (i.e., only one of the three components was decomposed and separately represented) may have caused the subjects to choose either method. Furthermore, our GOMS model is focusing on the simplest form of diagram integration. For example, the task in our experiment was to integrate only two diagrams. However, in reality, most systems development projects involve integrating more than two processes. When the number of diagrams to integrate exceeds two, the process integration behavior may become much more complex than the GOMS model presented in this paper. For example, our model assumes that the problem solver knows the diagram to switch to because there are only two. However, when there are more than two diagrams to integrate, this assumption no longer holds. Such a situation will also affect the behavior depending on whether the diagrams can be laid out in parallel in front of the problem solver or whether the diagrams are presented on a computer VDU. The choice of diagrams to investigate at a certain time and the process of navigation to that diagram may have confounding effects on the behavior of diagram integration, and the model needs to be further refined to explain the effects faithfully.

Finally, important considerations arise with the design and analysis of our experiment. In terms of experimental design, several important judgments had to be made in order to balance the trade-off between internal and external validity of the experiment. For example, we selected actual diagrams used in practice as our experimental material in order to provide a natural task environment. However, these diagrams may be different in terms of some dimensions rather than layout organization and decomposition. At the same time, we limited the number of diagrams to integrate at the minimum (two) so that the scope of the task would not

be exceedingly large, since the subjects must be able integrate the given processes within the experimental session without any side-effects such as fatigue. The level of expertise our subjects had in terms of the process diagrams might also be an important factor in the diagrammatic process integration task. Expert analysts may respond differently to the visual representation of the diagrams, because they may internally process information that is presented with a certain diagrammatic representation in a different way than novices. However, we believe that this problem resulting from the level of expertise could be minimal for several reasons. First, the students that were recruited for the experiment were not completely novices, since they did several homework assignments and conducted a semester-long project with the diagrams. In addition, the experimental task was greatly simplified so that expertise in the real-world software development context was not required in the experimental task. However, given the fact that the GOMS analysis has been usually conducted with experts, an additional experiment with expert analysts may provide more reliable empirical results. In terms of the analysis of the experiment, the analysis of only the representative subjects should be interpreted with caution and be regarded as a pilot analysis. Even though the analysis of the representative subjects did provide valuable insights into why differences in diagrammatic representation may cause behavioral differences, comprehensive protocol analysis of all the subjects will be able to provide greater statistical support to the process results.

Despite these limitations, this research has both strong theoretical and practical contributions. Theoretical contributions from this study will deepen our understanding of human behavior in manipulating diagrams directly as a means of inference. This research not only confirms but also extends the current line of research in diagrammatic reasoning. Previous research on diagrammatic reasoning focused on the advantages of the diagrammatic representation over propositional representation (e.g., text) in that the diagrammatic knowledge representation can facilitate problem solving by providing effective search and recognition cues, and also by enabling powerful perceptual inferences that are natural to the human (Larkin and Simon, 1987). This research extends the current framework of diagrammatic reasoning by comparing the computational efficiencies *between* different diagrams based on how knowledge is represented and by dealing with the direct manipulation of diagrams than just reasoning with given diagrams.

This paper also extends the current framework of cognitive dimensions of notational systems (Green, 1989; Green and Petre, 1996) with respect to two aspects. First, this paper identified important diagrammatic representation factors (i.e., decomposition and layout organization) that may be regarded as the underlying design rationale behind the cognitive dimensions of notations. The cognitive model of diagrammatic manipulation proposed in this paper suggests that the effects of various cognitive dimensions (e.g., diffuseness, hidden dependencies, premature commitment, secondary notation and visibility) may be caused by the low level design of the notation (i.e., the visual grammar) such as decomposition and layout organization. Second, this paper provides empirical evidence to the general cognitive dimensions framework

that features of the notation do in fact affect how people perform integration tasks with diagrammatic notations.

Furthermore, it should also be noted that in addition to the integration task, a wide variety of natural diagrammatic manipulation tasks could be performed. They include dividing a diagram into several partitions or deleting elements from diagrams in order to obtain a diagram that is easier to understand. This would call for an “*algebra of diagrams*” where problem solving consists of various direct-manipulation operations on diagrams such as adding, subtracting, multiplying, and dividing diagram components. This study found that decomposition and layout organization are important factors in the diagram addition task. The other diagrammatic manipulation tasks may be affected by other visual-grammatical rules in addition to the factors identified in this study. The results of this study calls for more research for the comprehensive understanding of the visual language that affects the diverse types of diagrammatic manipulation.

Practical contributions of this research hold for users and designers of diagrammatic representations in software engineering. As mentioned in the introductory section, current systems analysis and design methodologies provide a wide variety of different diagrams that represent the same information in a diagrammatically different form and organization. The systems analysts — the users who work with these diagrams — will be able to select particular diagrammatic representations based on the task at hand (Green, Petre and Bellamy, 1991; Good, 1996). Some tasks may be easier to perform or less error-prone with a particular diagrammatic representation. Therefore the possibility of selecting an appropriate diagrammatic representation will be of great assistance. Furthermore, the results of this study may be applied in designing the training material for using the systems analysis and design methodologies. The users should not only be taught how to use the different diagrams but also how different diagrammatic representations might systematically cause errors and faults in using these diagrams so as to foster an awareness that will help reduce analysis and design errors. From the designer's perspective, the findings of this research may be used to develop guidelines for designing diagrams that are cognitively compelling. These guidelines may be used in the design of diagrams in systems analysis and design methodologies and also in CASE (Computer Aided Software Engineering) tools.

REFERENCES

- Baddeley, A. D. (1986). *Working Memory*, Oxford, Clarendon.
- Beedle, M. A. (1995). Object-Based Reengineering, *Object Magazine*, March-April, pp. 53-58.
- Card, S. K., Moran, T. P. and Newell, A. (1983). *The Psychology of Human Computer Interaction*, Lawrence Erlbaum Associates, Hillsdale, New Jersey.

- Cheng, P. C.-H. and Simon, H. A. (1995). Scientific Discovery and Creative Reasoning with Diagrams, In S. Smith, T. Ward, and R. Finke (Eds.), *The Creative Cognition Approach*, MIT Press, Cambridge, Massachusetts., pp. 205-228.
- Curtis, B., Kellner, M. and Over, J. (1992). Process Modeling, *Communications of the ACM*, September 1992, vol. 35, no. 9., pp. 75-90.
- Davenport, T. H. (1993). *Process Innovation: Reengineering Work Through Information Technology*, Harvard Business School Press, Boston, Massachusetts.
- Engelhardt, Y., de Bruin, J., Janssen, T. and Scha, R. (1996). The Visual Grammar of Information Graphics, *Artificial Intelligence in Design (AID'96)*, in the Workshop on Visual Representation, Reasoning and Interaction in Design. 24-27 June, Stanford University, California.
- Ericsson, K. A. and Simon, H. A. (1993). *Protocol Analysis: Verbal Reports as Data*. The MIT Press, Cambridge, Massachusetts.
- Forbus, K., Nielson, P. and Faltings, B. (1991). Qualitative Spatial Reasoning: The CLOCK Project, *Artificial Intelligence*, 51(1-3).
- Gilmore, D. J. and Green, T. R. G. (1984). Comprehension and Recall of Miniature Programs, *International Journal of Man-Machine Studies*, Vol. 21, pp. 31-48.
- Green, T. R. G. (1989). Cognitive Dimensions of Notations, In A. Sutcliffe and L. Macaulay (Eds.), *People and Computers V*, Cambridge University Press, Cambridge, pp. 443-460.
- Green, T. R. G., Petre, M. and Bellamy, R. K. E. (1991). Comprehensibility of Visual and Textual Programs: A Test of Superlativism Against the 'Match-Mismatch' Conjecture, In J. Koenemann-Belliveau, T. G. Moher, and S. P. Robertson (Eds.), *Empirical Studies of Programmers: Fourth Workshop*, Ablex Publishing, Norwood, New Jersey, pp. 121-146.
- Green, T. R. G. and Petre, M. (1996). Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework, *Journal of Visual Languages and Computing*, Vol. 7, pp. 131-174.
- Good, J. (1996). The 'Right' Tool for the Task: An Investigation of External Representations, Program Abstractions and Task Requirements, In W. D. Gray and D A. Boehm-Davis (Eds.) *Empirical Studies of Programmers: Sixth Workshop*, Ablex Publishing, Norwood, New Jersey, pp. 77-98.
- Hammer, M. and Champy, J. (1993). *Reengineering the Corporation*, Harper Business, New York.
- Jacobson, I. (1995). *The Object Advantage: Business Process Reengineering with Object Technology*, Addison Wesley, Reading, Massachusetts.

- Kalyuga, S. Chandler, P. and Sweller, J. (1997). Levels of Expertise and User-Adapted Formats of Instructional Presentations: A Cognitive Load Approach, In A. Jameson, C. Paris and C. Tasso (Eds.), *User Modeling: Proceedings of the Sixth International Conference (UM 97)*, Springer, New York, pp. 261-272.
- Kim, J. and Hahn, J. (1997). Reasoning with Multiple Diagrams, *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*, pp. 376-381.
- Kirk, R. E. (1995). *Experimental Design: Procedures for the Behavioral Sciences* (3rd Ed.), Brooks / Cole Publishing Company, Pacific Grove, California
- Koedinger, K. R., and Anderson, J. R. (1990). Abstract Planning and Perceptual Chunks: Elements of Expertise in Geometry, *Cognitive Science*, 14, pp. 511-550.
- Kosslyn, S. M. (1980). *Image and Mind*, Harvard University Press, Cambridge, Massachusetts.
- Kulpa, Z. (1994). Diagrammatic Representation and Reasoning, *Machine Graphics and Vision*, vol. 3, nos. 1/2, pp. 77-103.
- Larkin, J. and Simon, H. A. (1987). Why a Diagram is (Sometimes) Worth Ten Thousand Words, *Cognitive Science*, 11, pp. 65-99.
- Mackinlay, J. and Genesereth, M. R. (1985). Expressiveness and Language Choice, *Data and Knowledge Engineering*, 1, 17-29.
- Modugno, F., Corbett, A. T. and Myers, B. A. (1997). Graphical Representation of Programs in a Demonstration Visual Shell: An Empirical Evaluation, *ACM Transactions on Computer-Human Interaction*, Vol. 4, No. 3, pp. 276 - 308.
- Moher, T. G., Mak, D. C., Blumenthal, B., and Leventhal, L. M. (1993). Comparing the Comprehensibility of Textual and Graphical Programs: The Case of Petri Nets. In C. R. Cook, J. C. Scholtz, and J. C. Spohrer (Eds.) *Empirical Studies of Programmers: Fifth Workshop*, Ablex Publishing, Norwood, New Jersey, pp. 137-161.
- Narayanan, N. H., Suwa, M., and Motoda, H. (1995). Hypothesizing Behaviors from Device Diagrams, In J. Glasgow, N. H. Narayanan, and B. Chandrasekaran (Eds.), *Diagrammatic Reasoning: Cognitive and Computational Perspectives*, AAAI Press, Menlo Park, California., pp. 501-534.
- Newell, A. and Simon, H. A. (1972). *Human Problem Solving*, Prentice Hall, New Jersey.
- Pressman, R. (1992). *Software Engineering: A Practitioner's Approach*, McGraw-Hill, New York.
- Rational. (1997). *The Unified Modeling Language ver. 1.0*, Rational Software Corporation, Santa Clara.
- Rist, R. (1989). Schema Creation in Programming, *Cognitive Science*, 13, pp. 389-414.

- Rogers, E. (1995). A Cognitive Theory of Visual Interaction, In J. Glasgow, N. H. Narayanan, and B. Chandrasekaran (Eds.), *Diagrammatic Reasoning: Cognitive and Computational Perspectives*, AAAI Press, Menlo Park, California., pp. 481-500
- Rogers, E. (1996). A Study of Visual Reasoning in Medical Diagnosis, *Proceedings of the 18th Annual Conference of the Cognitive Science Society*, pp. 213-218.
- Simon, H. A. (1981). *The Sciences of the Artificial*, (2nd ed.), MIT Press, Cambridge, MA., pp. 3-33.
- Smith, E. E. (1995). Concepts and Categorization, In Osherson, D. N. and Smith, E. E. (Eds.), *Thinking: An Invitation to Cognitive Science*, Vol. 3, MIT Press, Cambridge, Massachusetts.
- Tabachneck, H. J. M., Leonardo, A. M., and Simon, H. A. (1994). How Does an Expert Use a Graph? A Model of Visual and Verbal Inferencing in Economics, *Proceedings of the 16th Annual Conference of the Cognitive Science Society*, pp. 842-847.
- Tabachneck, H. J. M., and Simon, H. A. (1992). Effect of Mode of Data Presentation on Reasoning about Economic Markets, *In AAAI Spring Symposium on Reasoning with Diagrammatic Representations*, 25-27 March, Stanford University, AAAI Press, Menlo Park California., pp. 56-61.
- Tabachneck-Schijf, H. J. M., Leonardo A. M., and Simon, H. A. (1998). CaMeRa: A Computational Model of Multiple Representations, *Cognitive Science*, Vol. 21, No. 3, pp. 305-350.
- Taylor, D. A. (1995). *Business Engineering with Object Technology*, John Wiley & Sons. Inc., New York.
- Tversky, B. (1997). Cognitive Principles of Graphic Displays, in *AAAI 1997 Fall Symposium on Reasoning with Diagrammatic Representations*, 8-10 November, Massachusetts Institute of Technology, Cambridge, Massachusetts.
- Winograd, T. (1995). From Programming Environments to Environments for Designing, *Communications of the ACM*, June 1995, Vol.38, No.6, pp.65-74.
- van Someren, M. W., Barnard, Y. F. and Sandberg, J. A. C. (1994). *The Think Aloud Method: A Practical Guide to Modeling Cognitive Processes*, Academic Press, San Diego.
- Zhang, J. (1997). The Nature of External Representations in Problem Solving. *Cognitive Science*, Vol. 21, No. 2, pp. 179-217.
- Zhang, J. and Norman, D. A. (1994). Representations in Distributed Cognitive Tasks. *Cognitive Science*, Vol. 18, pp. 87-122.