A Collaborative Visual Language

Kyung-Deok Kim, *Member, KIMICS*

Abstract—There are many researches on visual languages, but the most of them are difficult to support various collaborative interactions on a distributed multimedia environment. So, this paper suggests a collaborative visual language for interaction between multi-users. The visual language can describe a conceptual model for collaborative interactions between multi-users. Using the visual language, generated visual sentences consist of object icons and interaction operators. An object icon represents a user who is responsible for a collaborative activity, has dynamic attributes of a user, and supports flexible interaction between multi-users. An interaction operator represents an interactive relation between multi-users and supports various collaborative interactions. Merits of the visual language are as follows: supporting of both asynchronous interaction and synchronous interaction, supporting flexible interaction between multi-users according to participation or leave of users, supporting a user oriented modeling, etc. For example, an application to a workflow system for document approval is illustrated. So we could be found that the visual language shows a collaborative interaction.

Index Terms—Distributed multimedia, Visual language, Collaboration, Interaction

I. INTRODUCTION

Recently, there are many researches on visual languages at various fields such as a user interface, information retrieval, information communication, etc. Visual languages represent given problems on a multi-dimension by a combination of object icons and icon operators, and can support convenient usage using a direct manipulation, an explicit representation of relations between objects, a visually feedback, an extensibility, etc[1, 2]. Conventional visual languages are easy to represent various relations between objects, but are difficult to support temporal attributes. However, dynamic visual languages can describe various semantic representations according to a temporal flow, and usually it is used to support interaction between computer and human, or presentation of multimedia objects. Features of these visual languages are as follows; supporting only single user on single host, using static information of a user, supporting a task oriented modeling, etc. So, the most of conventional visual languages are difficult to support collaborative interaction between multi-users on computer networks. Currently, methods for interactions and visual representations between dynamic processes for collaboration on distributed multimedia environment are needed[3]. Also, dynamic variation of collaboration by participation or leave of users should be considered[4], and both of synchronous interaction and asynchronous interaction to support structural collaboration should be considered[5]. So, we suggest a collaborative visual language to support of collaborative interaction between multi-users on a distributed multimedia environment.

The collaborative visual language can support both of asynchronous interaction and of synchronous interaction for representation of various collaborative interactions, flexible interactions between multi-users for a variation of collaborative interaction, a user oriented modeling for representation of explicit relation between multi-users, etc. The visual language is extended from a visual language for collaborative interaction [6].

The related works on visual languages for collaborative interaction are as follows: K. Kim's a visual language *Workflow/VL*[7, 8] to support a workflow design, S. K. Chang's dynamic visual languages[9], K. D. Swenson's a visual language *VPL*[10], J. C. Grundy's a visual language *EVPL*[11], C. Castroianni's *Scarabaeus system*[12], A. Sheth's *METEOR$_2$system*[13, 14], etc.

The *Workflow/VL* is a visual language to describe collaborative activities according to a temporal flow. In the visual language, object icons have coworkers' static information and icon operators represent collaborative interactions between coworkers. Generated visual sentences are transformed into conceptual graphs, and collaboration is performed according to semantics of the graph. However, the visual language can describe only asynchronous interaction between coworkers.

The dynamic visual languages are visual languages for a multimedia presentation. The visual languages can describe and process complex relations between multimedia objects. The visual languages can describe efficiently dynamic attributes between multimedia objects, but they support a task-oriented modeling and are difficult to support collaborative interaction between multi-users.

The *VPL*(Visual Planning Language) is a visual language to describe collaboration. The visual language has stages and directed links. The stage is an object icon that means a collaborative activity. The stages are connected by directed links so that a plan for collaboration is generated. The *VPL* can support to describe relations between collaborative activities in detail, but can not describe synchronous interaction between multi-users.

The *EVPL*(Extended Visual Planning Language) is an

Manuscript received March 6, 2003.

The author is with the Division of Computer and Multimedi a Engineering, Uiduk Uiniversity, Gangdong, Gyongju, 780-713, Korea(email: kdkim@uiduk.ac.kr)

extension of the *VPL*. The visual language separates object icons' roles into an independent user's role and a multi-user's role, and added object icons' role according to icons' images. But, the visual language can not represent synchronous interaction alike the *VPL*, too.

The visual language on the *Scarabaeus* workflow system supports to describe relations between object icons that mean collaborative activities. And, the visual language supports to optimize a designed workflow using a knowledge base. A described workflow is transformed into a Petri-net, and then the Petri-net is executed. The visual language can not describe synchronous interaction between multi-users.

The visual language on the *METEOR$_2$* workflow system supports to describe relations between collaborative activities for hospitals' works. The visual language can not describe synchronous interaction alike above the visual languages. So, a visual language to support collaborative interaction should be considered as follows. It must reconfigure scattered resources on a distributed multimedia environment. It must represent various collaborative interactions between dynamic process. It must support a dynamic variation according to participation or leave of users on collaboration. It must have an environment for efficient description of collaborative interaction. It must have a syntax and semantics analysis method of a generated visual sentence.

This paper is organized as follows. In chapter II, we define collaborative interaction, and describe elements for the collaborative visual language. In chapter III and IV, we analyze a syntax and semantics of a generated visual sentence. In chapter V, a compiling technique is explained. In chapter VI and VII, we explain an application of the collaborative visual language to a workflow system and give conclusions and future works.

II. ELEMENTS OF THE COLLABORATIVE VISUAL LANGUAGE

A. Collaborative Interaction

Collaborative interaction means a series of activities for information exchange to solve a given work in collaboration. The figure 1 represents changes of each user's view according to a point t of time on collaboration. The figure 1 represents collaborative interaction between a user a and a user b. Each user performs his/her work by interaction with a local computer at any time t. And, each user shares his/her information or idea in collaborative interaction at a time t+1 and t+2. The information sharing changes incrementally a visual or auditory view each other.

Collaborative interaction can be classified according to an interactive time, object sharing, information form, and locality[15]. According to an interactive time, collaborative interaction is subdivided into synchronous interaction and asynchronous interaction. Synchronous interaction means a series of collaborative activities between multi-users for a same work at same time. Asynchronous interaction means a series of collaborative activities between multi-users for a same work at different time. According to an object sharing, collaborative interaction is subdivided into explicit interaction and implicit interaction. Explicit interaction is interaction without using shared objects. Implicit interaction is interaction with using shared objects. According to information form, collaborative interaction is subdivided into formal interaction and informal interaction. Formal interaction handles information that is easy to analyze, draw conclusions from, or to process further. Informal interaction deals with information that is extremely hard or even impossible to process in any way. According to locality, collaborative interaction is subdivided into local interaction and remote interaction. Local interaction is restricted to a limited region that may be a building, or the range of a LAN. Remote interaction handles information between users on other sites. The collaborative visual language is based on an interactive time, and an interaction operator represents synchronous interaction or asynchronous interaction. The definition 1 defines collaborative interaction as follows.

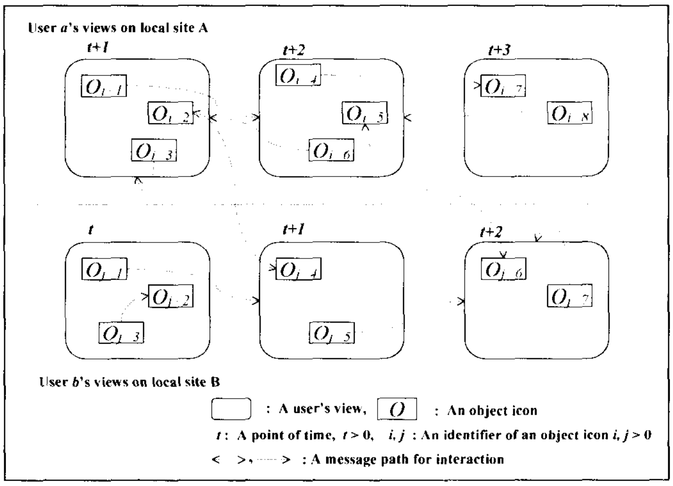

Fig. 1 Collaborative interaction between users a and user b

[Definition 1] Collaborative interaction CI is a 6-tuple, $CI = (US, X, E, OUT, F, IT)$ where: US is a set of users, X is a set of general icons, E is a set of events that triggers interaction, E = {Dragging, Key push, Mouse clicking, Pointing, ...}, OUT is a set of event messages that change users' view, F is a function that produces interaction between a computer and a user, and IT is a function that produces collaborative interaction. Functions, F and IT, are defined as follows;

$$F : US \times 2^X \times 2^E \rightarrow U \times OUT, \quad \text{for } U = 2^{US},$$
$$IT : U \times OUT \times U \times OUT \rightarrow U \times OUT.$$

Therefore, the function F is specified as following equation (1);

$$F(u, \{x_1, x_2, ..., x_n\}, \{e_1, e_2, ..., e_m\}) = (u, \{out_1, out_2, ..., out_l\}) \quad (1)$$
where: $u \in US, x \in X, e \in E, out \in OUT, n > 0, m > 0, l > 0$.

The equation (1) represents local interaction by a user's direct manipulation on object icons. The result of the equation changes a view of a user. And, the function IT is specified as following equation (2);

$$IT(u_i, \{out_{i1}, out_{i2}, ..., out_{im}\}, u_j, \{out_{i1}', out_{i2}', ..., out_{in}'\}) \quad (2)$$
$$= (\{u_i, u_j\}, \{out_{i1}, out_{i2}, ..., out_{im}, out_{i1}', out_{i2}', ..., out_{in}'\}).$$

The equation (2) produces collaborative interaction as its results. The result of the equation changes views of users $\{u_i, u_j\}$. If a user u_i is void, the set of event messages $\{out_{i1}, out_{i2}, ..., out_{im}\}$ is empty and the equation produces local interaction alike the function F.

B. Extended object icons

A general object icon consists of an image part and a semantics part. Using only the two elements, the object icon is difficult to support collaborative interaction, because collaborative interaction has dynamic and interactive aspects. So, in the paper, the collaborative visual language uses an object icon that has an extended logical part than that of a general object icon. The extended object icon can be manipulated easily alike a general object icon.

The extended object icon represents a conceptual entity that means a user who is responsible for a collaborative activity such as an approval, an editing, programming, etc. In addition, it has a value that means participation status of a user in collaboration. The value is used to adjust a point of interactive time between multi-users, and so it can support a flexible interaction. The definition 2 defines an extended object icon.

[Definition 2] An extended object icon EX is a 2-tuple, $EX = (EX_I, EX_M)$ where: EX_I is a set of images, and EX_M is a set of semantics. And, an element ex_m of EX_M is a 7-tuple, $EX_M = (A, U, S, IM, OM, IN\_F, OUT\_F)$ where: A is a set of collaboration activities, U is a set of users who are responsible for given activity, S is a set of user's status value, $S = \{s_{ready}, s_{busy}, s_{absent}\}$, IM is a set of input messages from other users, OM is a set of output messages to other users, IN_F is a function that processes input messages, and OUT_F is a function that processes output messages. Functions, IN_F and OUT_F, are defined as follows;

$IN\_F : 2^{IM} \times S \rightarrow ID \times S$,
for ID is a set of immediate data,
$OUT\_F : ID \times A \times U \times S \rightarrow 2^{OM} \times S$.

When status of a user is a value s_{ready}, the function IN_F can process input messages that are generated by local interaction or collaborative interaction. The meaning of each value in S is as follows; s_{ready} means that a user is ready to work, s_{busy} means that a user is working, and s_{absent} means that currently a user does not participate in collaboration. The status value of a user is changed dynamically according to user's participation or leave on collaboration. Generated output messages from the function OUT_F changes a view of a user. So, the output messages and input messages invoke and support collaborative interaction between multi-users.

Kinds of extended object icons are as follows; a start icon, an end icon, a user icon, and a complex icon. A start icon and an end icon are used to start and end collaborative interaction. A user icon represents a user on collaboration. A complex icon supports an extension and a modulation of visual sentences. Most of object icons, a start icon, an end icon, and a complex icon, are activated automatically by the system. The user icon is only activated by a user in collaboration.

C. Interaction operators

Conventional icon operators compute relations between object icons according to locations of object icons on a working area, a temporal flow, and context between object icons. But, the icon operators are difficult to represent various collaborative interactions between multi-users on a distributed multimedia environment. In this paper, an interaction operator is used to support various interactive relations and it is a conceptual entity that means an interactive relation between multi-users according to an interactive time. And, it is used to combine object icons and consists of a logical operator and a physical operator. The definition 3 defines an interaction operator.

[Definition 3] An interaction operator iop is a 2-tuple, $iop = (IOP_I, IOP_M)$, where: IOP_I is a physical operator that combines images of object icons, and IOP_M is a logical operator that combines semantics of object icons. And, an element iop_i of IOP_I is defined as follows;

$iop_i : EX_I \times EX_I \rightarrow EX_I$, for $iop_i \in IOP_I$
An element iop_m of IOP_M is defined as follows;
$iop_m : EX_M \times EX_M \rightarrow EX_M$, for $iop_m \in IOP_M$.

Therefore, an interaction operator iop is specified as follows;

$$iop(ex, ey) = (iop_i(ex.i, ey.i), iop_m(ex.m, ey.m)) = (ez.i, ez.m), \quad (3)$$

where $ex, ey, ez \in EX$, and i and m represent an image and a meaning respectively.

The equation (3) represents a generation of a new object icon by an operation iop between object icons. So, a visual sentence can be extended incrementally from a combination of interaction operators and object icons. The logical part $ez.m$ of the object icon ez means collaborative interaction between users, $ex.u$ and $ey.u$, who are responsible for collaborative activities of ex and ey. According to an interactive time, kinds of the interaction operators are a synchronous operator iop_s, an asynchronous operator iop_a, a complex operator iop_c, and a partial synchronous operator iop_p.

The synchronous operator is an operator to support collaborative interaction between multi-users at same time. The asynchronous operator is an operator to support collaborative interaction between multi-users at different time. The combination operator represents a combination of a synchronous operator and an asynchronous operator. The partial synchronous operator is a variant of the synchronous operator. Meanings of the combination operator and the partial synchronous operator are explained in detail in the chapter III.

D. Visual grammar VG

We use a visual grammar to check dependency between multi-users or tasks, and connectivity between icon objects and interaction operators. In addition, using an incremental parsing method, visual sentences generated incrementally. The incremental parsing method can support possible continuation for the user as well as determining of the input so far. The definition 4 defines a visual grammar for the collaborative visual language.

[Definition 4] A visual grammar VG is a 5-tuple, $VG = (N, T, S, P, R)$, where N is a set of non-terminal symbols, T is a set of terminal symbols, S is a start symbol, P is a set of production rules, and R is a set of restrictions which is applied to generated visual sentences.

$N = \{S, U, A, B, C\}$

$T = \{ ex_s, ex_e, ex_a, ex_c, iop_a, iop_s, iop_c, iop_p\}$

$P : S \rightarrow US \mid U$

$U \rightarrow ex_a\, A \mid ex_s\, B \mid ex_c\, B$

$A \rightarrow iop_a\, C \mid iop_s\, ex_a \mid iop_c\, ex_a \mid iop_p\, ex_a$

$B \rightarrow iop_a\, C$

$C \rightarrow ex_e \mid ex_a \mid ex_c$

R :

r_1 : A visual sentence has only an ex_s and an ex_e.

r_2 : A visual sentence dose not has iterative relations.

r_3 : A visual sentence has not isolated object icons.

r_4 : An object icon has only one identity for a user.

A generated visual sentence from the collaborative visual language consists of unit visual sentences. A unit visual sentence is a token to analyze syntax of a visual sentence. The definition 5 defines a unit visual sentence.

[Definition 5] A unit visual sentence ($ex\ iop\ ey$) means an operation iop from ex to ey, where: ex, $ey \in EX$, and $iop \in IOP$.

III. SYNTAX ASPECTS

The collaborative visual language produces complex visual sentences that represent various collaborative interactions between multi-users. For analysis of syntax and semantics of a generated visual sentence, first, we classify the visual sentence into semantic units using an interactive type of an object icon. Namely, the visual sentence is subdivided into a visual sentence for asynchronous interaction and one or more visual sentences for synchronous interaction. The interactive type represents connective relations between object icon and interaction operators. Second, each semantic unit is translated into a conceptual graph. Finally, collaborative interaction is performed by semantics of the concept graphs, and interactive media such as audio, video, and multimedia messages.

For example, the figure 2 represents a visual sentence VS_p. In the figure, a rounded small rectangle and a line represent an object icon and an interaction operator respectively.

The visual sentence VS_p consists of 11 interactive relations, namely, $VS_p = \{ex_1 \rightarrow ex_2,\ ex_2 \rightarrow ex_3,\ ex_3 \rightarrow ex_4,\ ex_4 \bullet\!\!\rightarrow ex_5,\ ex_5 \rightarrow ex_{10},\ ex_2 \rightarrow ex_6,\ ex_6 \rightarrow ex_9,\ ex_9 \rightarrow ex_{10},\ ex_3 \circ\!\!\rightarrow ex_9,\ ex_6 \leftrightarrow ex_7,\ ex_6 \leftrightarrow ex_8\}$. Object icons, ex_3, ex_5, and ex_6, are related to both of synchronous interaction and asynchronous interaction. For example, the $ex_3.u$ interacts asynchronously with $ex_2.u$ and $ex_4.u$, and interacts synchronously with $ex_9.u$ at an activation time of ex_3. The symbol u means a user who is responsible for a collaborative activity in an object icon. The other object icons, ex_1, ex_2, ex_4, ex_9, and ex_{10}, interact only asynchronously. Asynchronous operators represent an order of collaborative interaction according to a temporal flow. Meaning of the visual sentence is as follows; first, $ex_1.a$ is started. The symbol a means a collaborative activity of an object icon. When $ex_1.a$ is done, $ex_2.a$ is started. When $ex_2.a$ is done, $ex_3.a$ and $ex_6.a$ are started. When $ex_3.a$ is started, $ex_3.u$ interacts synchronously with $ex_9.u$. When $ex_3.a$ is done, $ex_4.a$ is started. When $ex_4.a$ is done, $ex_5.a$ is started and then $ex_5.u$ interacts synchronously with $ex_4.u$ according to a partial synchronous interaction between ex_4 and ex_5. When $ex_6.a$ is started, $ex_6.u$ interacts synchronously with $ex_7.u$ and $ex_8.u$. When $ex_6.a$ is done, $ex_9.a$ is started. And, $ex_9.a$ dose not affects $ex_3.a$, because ex_9 has a partial synchronous interaction. When both of $ex_5.a$ and $ex_9.a$ are done, $ex_{10}.a$ is started. When $ex_{10}.a$ is done, the collaborative interaction is ended.

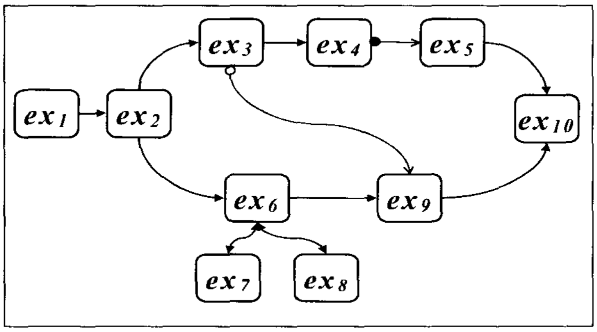

Fig. 2 A Visual sentence VS_p

We define an interaction type that is used to separate a visual sentence into semantics units. The definition 6 defines an interaction type.

[Definition 6] An interaction type it of an object icon defines connective relations with interaction operators. Kinds of interaction types are an asynchronous interaction type it_a, a synchronous interaction type it_s, and a complex interaction type it_c. Let a function Set_iop returns names of connected interaction operators on an object icon. And, a function f to compute an interaction type is as follows;

① $Set\_iop(ex) = \{\{iop_a\}, \{iop_a, iop_p\}\}$
$\Rightarrow f(Set\_iop(ex)) = it_a$, where in the case of iop_p, only the tail of the interaction operator is considered.

② $Set\_iop(ex) = \{iop_s\} \Rightarrow f(Set\_iop(ex)) = it_s$,

③ $Set\_iop(ex) \in \{\{iop_a, iop_s\}, \{iop_a, iop_p\}, \{iop_a, iop_p, iop_s\}, \{iop_a, iop_c\}, \{iop_a, iop_c, iop_s\}, \{iop_a, iop_c, iop_p, iop_s\}\} \Rightarrow f(Set\_iop(ex)) = it_c$,

where in the case of iop_p, only the head of the interaction operator is considered.

According to definition 6, interaction types of object icons in the visual sentence VS_p are as follows. Object icons with it_a are ex_1, ex_2, ex_4, ex_9, and ex_{10}. Object icons with it_s are ex_7, and ex_8. Object icons with it_c are ex_3, ex_5, and ex_6. So, using an interaction type of each object icon, the visual sentence VS_p is subdivided into one asynchronous visual sentence and three synchronous visual sentences. The asynchronous visual sentence is generated by object icons with an interaction type it_a or it_c. The synchronous visual sentence is generated by object icons with an interaction type it_s or it_c. All of generated visual sentences from users consist of one asynchronous visual sentence and zero or more synchronous visual sentences. Semantics units of the visual sentence VS_p are as follows; a visual sentence for asynchronous interaction is represented by a set $\{ex_1 \rightarrow ex_2,\ ex_2 \rightarrow ex_3,\ ex_3 \rightarrow ex_4,\ ex_4 \rightarrow ex_5,\ ex_5 \rightarrow ex_{10},\ ex_2 \rightarrow ex_6,\ ex_6 \rightarrow ex_9,\ ex_9 \rightarrow ex_{10}\}$. And, visual sentences for synchronous interaction are represented by a set $\{\{ex_4 \leftrightarrow ex_5\},\ \{ex_3 \leftrightarrow ex_9\},\ \{ex_6 \leftrightarrow ex_7,\ ex_6 \leftrightarrow ex_8\}\}$. In the case of ex_3, synchronous interaction among $ex_3.u$ and $ex_9.u$ is invoked by an activation of ex_3. In the case of ex_5, synchronous interaction between $ex_4.u$ and $ex_5.u$ is invoked by an activation of ex_5. In the case of ex_6, synchronous interaction among $ex_6.u$, $ex_7.u$, and $ex_8.u$ is invoked by an activation of ex_6.

IV. SEMANTICS ATTRIBUTES

In the chapter III, the visual sentence VS_p is subdivided into semantics units using interaction types of object icons. Each semantic unit is converted into a concept graph. Namely, an object icon and an interaction operator are converted into a concept node and a relation node respectively. And, a join and a simplification of converted concept graphs can generate a concept graph that means semantics of the visual sentence VS_p. A concept graph of each semantic unit is as follows;

G_1: [ex:#1]→(Asyn.)→ [ex:#2] –
(Asyn.)→[ex:#3]→(Asyn.)→[ex:#4]→
(Asyn.)→[ex:#5]→(Asyn.)→[ex:#10],
(Asyn.)→[ex:#6]→(Asyn.)→[ex:#9]→(Asyn.)
→[ex:#10].
G_2: [ex:#3]→(Syn.)→ [ex:#9].
G_3: [ex:#5]→(Syn.)→ [ex:#4].
G_4: [ex:#6]→(Syn.)→ [ex:{7, 8}].

Collaborative interaction is executed using above concept graphs. According to direction of an asynchronous operator in the concept graph G_1, asynchronous interaction between multi-users is executed. If a concept node in the concept graph G_1 means an object icon with a complex type, the concept node becomes a chief node that invokes synchronous interaction. And, all of connected object icons with the object icon are activated at same time.

Kinds of semantic attributes in the collaborative visual language are an interaction attribute and a dynamic attribute. The interaction attributes represent semantics of the function IT in definition 1 according to an interaction operator in a unit visual sentence. And, according to an interactive time, the function IT can be classified into a function IT_A for asynchronous interaction and a function IT_S for synchronous interaction. The interaction attributes can be considered as following four cases according to an interaction operator in a unit visual sentence.

In the case of a unit visual sentence ($ex\ iop_a\ ey$), the semantics can be represented as following equation (4), where the out means output message of an object icon. And, the output message set, $\{outp,\ ey.out\}$, changes a view of $ey.u$ at any time.

$$\begin{aligned}&(ex\ iop_a\ ey)\\&\Rightarrow IT_A(ex.u,\ ex.out,\ ey.u,\ ey.out)\\&= (\{ey.u\},\ \{outp,\ ey.out\}),\\&\text{for output messages } outp \subseteq 2^{ex.out}\end{aligned} \quad (4)$$

In the case of a unit visual sentence ($ex\ iop_s\ ey$), the semantics can be represented as following equation (5). And, the output message set, $\{ex.out,\ ey.out\}$, changes a view of $ex.u$ and $ey.u$ at a same time.

$$\begin{aligned}&(ex\ iop_s\ ey)\\&\Rightarrow IT_S(ex.u,\ ex.out,\ ey.u,\ ey.out)\\&= (\{ex.u,\ ey.u\},\ \{ex.out,\ ey.out\})\end{aligned} \quad (5)$$

In the case of a unit visual sentence ($ex\ iop_c\ ey$), it means a combination of unit visual sentences ($ex\ iop_a\ ey$) and ($ex\ iop_s\ ey$). Namely, an execution of the visual sentence invokes sequentially asynchronous interaction and synchronous interaction between $ex.u$ and $ey.u$. So, the semantics can be represented as following equation (6), where the symbol → represents a temporal flow.

$$\begin{aligned}&(ex\ iop_p\ ey)\\&\Rightarrow (ex\ iop_a\ ey) \rightarrow (ex\ iop_s\ ey)\\&\Rightarrow IT_A(ex.u,\ ex.out,\ ey.u,\ ey.out)\\&\quad \rightarrow IT_S(ex.u,\ ex.out,\ ey.u,\ ey.out)\\&= (\{ey.u\},\ \{outp,\ ey.out\})\\&\quad \rightarrow (\{ex.u,\ ey.u\},\ \{ex.out,\ ey.out\})\end{aligned} \quad (6)$$

In the case of a unit visual sentence ($ex\ iop_p\ ey$), its semantics is similar to that of a unit visual sentence ($ex\ iop_s\ ey$). A difference between the two unit visual sentences is as follows; ($ex\ iop_p\ ey$) means synchronous interaction between $ex.u$ and $ey.u$ when only $ex.a$ is activated. ($ex\ iop_s\ ey$) means synchronous interaction between $ex.u$ and $ey.u$ when the $ex.a$ or the $ey.a$ is activated.

The dynamic attributes represent a variation of collaborative interaction according to temporal flow and participation or leave of users. These variations are generated from an execution of synchronous interaction. So, conceptual graphs, CG_2, CG_3, and CG_4, create 3 sessions. Each session is activated when all of users in the session are participated. So, synchronous interaction is invoked by activation of an object icon with complex interaction type in the session. When the object icon with complex type is inactivated, the session with the object icon is ended, and connected and inactivated object icons with the object icon using asynchronous operator are started. Each session has two or more users and changes users' views. A point of an interactive time is decided flexible using status value of a user. In the case of CG_4, if $ex6.u$ and $ex8.u$ are ready to work and $ex7.u$ is not participated, status values of $ex6.u$, $ex7.u$, and $ex8.u$, are s_{ready}, s_{absent}, and s_{ready} respectively. So, an execution of the session is delayed automatically, and next execution time is searched using schedule information of users. If next execution time for all users is searched, an execution of the session is delayed until the next execution time. Otherwise, the session is closed and the collaborative interaction is closed, too. The searching of the next execution time is performed by an infrastructure for collaborative interaction. Using a status value, the collaborative visual language can support flexible interaction and dynamic variation between multi-users. So, we can drive an equation, $VO_n = IT(VO_{n-1}, VO_a)$ where: VO is a virtual object that means collaborative interaction, IT is a function in definition 1, $n > 1$, and VO_a is an added or removed VO at execution time.

And, using interaction media, users exchange information each other on collaborative interaction. According to interaction media, various collaborative interactions can be generated. The table I represents semantics of an interactive operation according to interaction media. And, the table II represents kinds of collaborative interaction according to interaction media.

TABLE I Semantics of interactive operation according to interaction media

Interaction Media	Interaction Operator	
	Synchronous Operator, Partial Synchronous Operator	Asynchronous Operator
Mouse Pointer	Bi-directional Tele-pointing	Unidirectional Tele-pointing, Local Pointing
Audio Clip	Bi-directional Tele-playing	Local playing
Video Clip		
Audio Stream	Bi-directional Tele-viewing	Unidirectional Tele-viewing, Local viewing
Video Stream		
Text Stream		
Text		Local viewing
Image		
Graphic Elements		

TABLE II Kinds of collaborative interaction according to interaction media

Interaction Media	Collaborative Interaction
Mouse Pointer	Tele-pointing
Video Clip	Conferencing
Audio Clip	Chatting
Video Stream	Electronic Mail
Audio Stream	View Sharing
Text Stream	Tele-viewing
Text	Tele-operation
Image	Tele-presentation
Graphic Elements	....

V. COMPILLING TECHNIQUE

We use an incremental parsing method for syntax and semantics analysis of visual sentences. The most of visual languages use generally an incremental parsing method. The parsing method can support possible continuation for the user as well as determining correctness of the input so far[16].

The visual grammar VG in the section A of chapter II checks correctness of unit visual sentences, connectivity among object icons and interaction operators, and dependency between multi-users. The visual grammar is a context free grammar and a unit visual sentence is a token. Each unit visual sentence is parsed at design time and a visual sentence is created by joints of unit visual sentences.

Generated visual sentences are subdivided into semantic units using an interaction type of an object icon at design time. Each semantic unit is translated into a concept graph. Collaborative interaction is executed by semantics of the concept graph. The figure 3 represents a runtime situation of collaborative interaction. In the figure, each object icon is connected to each user agent. Real communication between multi-users is established by communication between agents.

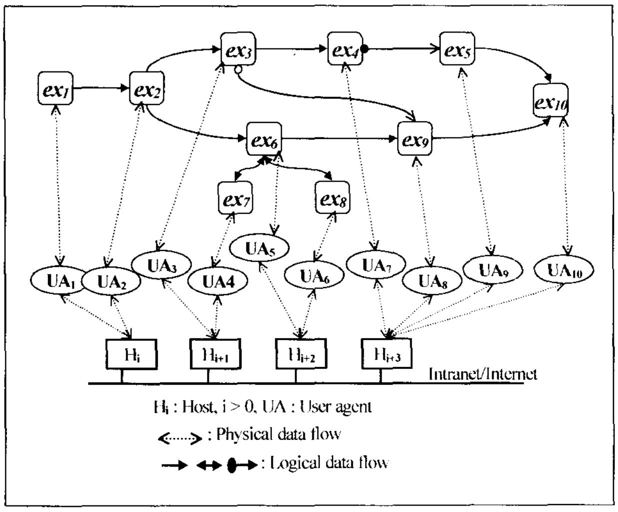

Fig. 3 Runtime situation

VI. APPLICATION TO A WOTKFLOW SYSTEM

For an approval workflow system, for example, the collaborative language is used. The system is implemented on *Solaris 2.4 OPEN WINODW* environment with *OSF/Motif 2.0*, *C++* language, and the *TCP/IP* protocol. An infrastructure of the workflow system was based on *HITE* (the Human Interaction Total Environments) system[17].

The HITE system is a platform for a multimedia Group Ware and has client-server architecture. The system can support a view sharing, a video conferencing, a collaborative editing, etc. The workflow system based on the HITE system consists of a client part and a server part. The client has a user manager, an activity list manager, a multimedia object manager, and a communication manager.

The user manager supports management about an authority and information of each user. The activity list manager supports an order of works that are performed by a user. The multimedia manager supports a management of generated multimedia objects from activities of a user. The communication manager supports communication between a client and a server using the TCP/IP protocol. In the server, a multimedia manager, a user manager, and a communication manager are similar to those of the client part. The workflow manager supports management of users' multi-workflow. The figure 4 represents a structure of the workflow system.

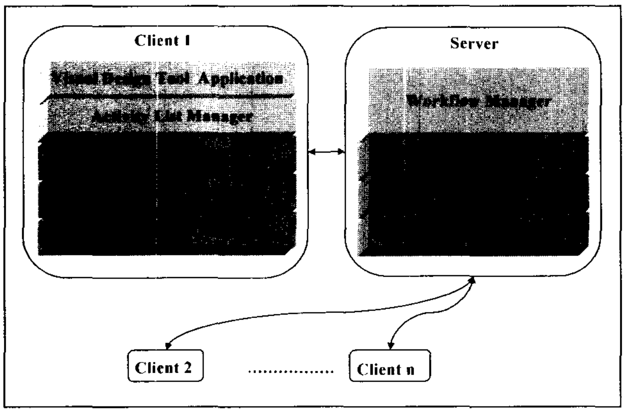

Fig. 4 Structure of the workflow system

The figure 5 represents a designed workflow and its execution.

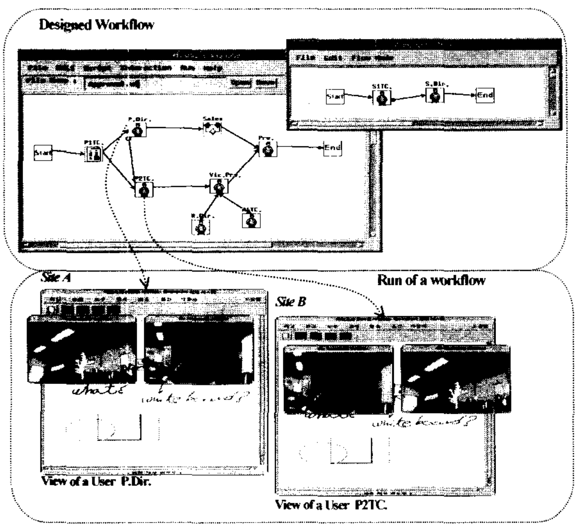

Fig. 5 Collaborative Interaction

The top of the figure represents a visual sentence that means a designed workflow. The bottom of the figure represents collaborative interaction between two users. The small window in the top of the figure represents a sub-workflow using a complex object icon. The meaning of the bottom of the figure is collaborative interaction between a user 'P.DIR' and another user 'P2TC' using a white board and a video at same time. Each user can change a view of the white board. The changes affect views of each other.

VII. CONCLUSIONS

We have presented a collaborative visual language and a syntax and semantic analysis method for the visual language. Conventional visual languages can support an explicit representation between objects, and various processing and representations of multimedia objects with temporal attributes. But, the conventional visual languages are difficult to support representation and supporting of collaborative interaction between multi-users on a distributed multimedia environment. The suggested collaborative visual language is an extension of a conventional visual language and can represent various interactive relations between multi-users using interaction operators according to an interactive time.

The collaborative visual language has following features; conceptual modeling of collaborative interaction, supporting both of asynchronous interaction and synchronous interaction, supporting flexible interaction between multi-users according to participation or leave of users in collaboration, supporting a user oriented modeling, reuse of a generated model, etc.

Future works are supporting real-time changes of relations of collaborative interaction, 3D modeling of collaborative interaction, supporting iterative relation of collaborative interaction, etc.

REFERENCES

[1] S. Chang, G. Costagliola, G. Pacini, M. Tucci, G. Tortora, B. Yu, and J. Yu, "Visual-language System for User Interfaces," IEEE Software, Vol. 12, No. 2, 1995, pp. 33-44.

[2] K. Kim and S. Kim, "Dynamics of Visual Language," Proc. of the 10th Korea Information Processing Society Conf. on Multimedia Industrial Technique, 1997, pp. 250-253.

[3] A. Repenning and T. Sumner, "Agentsheets: A Medium for Creating Domain-oriented Visual Languages," IEEE Computer, Vol. 28, No. 3, 1995, pp. 17-25.

[4] R. Bentley, T. Rodden, P. Sawyer, and I. Sommerville, Architectural Support for Cooperative Multiuser Interfaces, IEEE Computer, Vol. 27, No. 5, 1994, pp. 37-45.

[5] S. Rajan, P. V. Rangan, and H. M. Vin, "A Formal Basis for Structured Multimedia Collaborations," Proc. of the Int. IEEE Conf. on Multimedia Computing and Systems, 1995, pp. 194-201.

[6] K. Kim and S. Kim, "Visual Language supporting Collaborative Interaction," Proc. of the 5th Pacific Workshop on Distributed Multimedia Systems, Taipei, Taiwan, 1998, pp. 98-103.

[7] K. Kim and S. Kim, "Concept-based Visual Language for Workflow Design," Proc. of the IASTED Int. Conf. Artificial Intelligence and Soft Computing, 1997, pp. 172-175.

[8] K. Kim and S. Kim, "A Visual language supporting Workflow", Proc. of HCI '97 Conference, Vol. 6, No. 1, 1997, pp. 189-194.

[9] S. K. Chang, "Dynamic Visual Languages," Proc. of the IEEE Symposium on visual languages, 1996, pp. 308-315.
[10] K. D. Swenson, "A Visual Language to describe Collaborative Work", IEEE Int. Symposium for Visual Languages, Bergen Norway, August 1993.
[11] J. C. Grundy and J. G. Hosking, "Visual Language Support for Planning and Coordination in Cooperative Work Systems," Proc. of the IEEE Symposium on Visual Languages, Boulder, USA, 1996, pp. 324-325.
[12] C. Castroianni, "Development of a Workflow Design Tool for the Scarabaeus Project," Master's thesis, Dept. of Computer Science, Univ. of Twente, 1995.
[13] A. Sheth, Workflow Automation: Applications, Technology and Research, Tutorial notes, SIGMOD Conference, 1995.
[14] A. Sheth, K. Kochut, J. Miller, D. Palaniswami, J. Lyncl and I. hevchenko, "Supporting State-wide Immunization Tracking using Multi-Paradigm Workflow Technology," Proc. of the 22nd Intl. Conf. on Very Large Databases, 1996.
[15] G. Voelksen, "Approach Strategies to Groupware," Proceedings to the Groupware '93 Europe Conferences in Stokholm, London, Frankfurt, The Conference Group, Scottsdale, AZ, 1993.
[16] K. Wittenburg and L. Weitzman, "Visual Grammars and Incremental Parsing for Interface Languages," IEEE Proc. of Workshop on Visual Languages, 1990, pp. 111-118.
[17] S. Kim, J. Kim, T. Kim, J. Lee, and Y. Bae, "HITE: An Object Oriented Multimedia Groupware," J. of Korea Information Science Society (B): Software and Applications, Vol. 24, No. 12, 1997, pp. 1513-1521.

Kyung-Deok Kim
Received the BS degree, MS degree, and PhD degree in computer engineering from Kyungpook National University, Daegu, Korea, in 1989, 1991, and 1999, respectively. During 1991-1996, he was with Institute of Technology, Fast Systems Inc., Korea, as a chief engineer. He is currently with the Division of Computer and Multimedia Engineering, Uiduk University, Gyeongju, Korea, as assistant professor. His current research interests include multimedia authoring, multimedia presentation, and computer languages.