함수적 속성을 가지는 협업 지원 시각언어

김 경 덕†

요 약

본 논문에서는 함수적 속성을 가지는 협업 지원 시각언어를 제안한다. 제안한 시각언어는 객체 아이콘과 연산자들로 구성되는 시각 문장의 집합이다. 객체 아이콘은 협업에 참여하는 사용자를 의미하고 연산자는 사용자간 상호작용 시점에 따른 협업 관계를 의미한다. 시각 문장은 함수적 속성에 따라 연산되며, 연산 결과에 따라 다양한 협업을 지원한다. 함수적 속성은 시각 문장에서 다양한 연산 순서를 지원함으로서 협업 관계의 융통성을 제공한다. 또한, 동기 및 비동기 협업을 함께 표현함으로써 기존 시각언어보다 효율적인 협업을 지원한다. 그리고, 시각 문장의 함수적 속성은 λ 수식으로 분석한다.

A Visual Language supporting Collaboration with Functional Attributes

Kyung-Deok Kim†

ABSTRACT

In this paper, we suggest a visual language supporting collaboration with functional attributes. The visual language is a set of visual sentences that consist of object icons and operators. The object icon is a user who participates in collaboration. And, the operator means interactive relations between users according to a point of collaborative time. The functional attributes that support various computing orders provide flexibility of interactive relations on collaboration. Also, using representation both synchronous and asynchronous relations in collaboration, the visual language supports efficiently collaboration than conventional visual languages. And, functional attributes of visual sentences are analyzed using λ expressions.

1. 서 론

시각언어는 멀티미디어와 인터넷의 발달로 현재 활발히 연구되는 분야이며, 사용자 인터페이스나, 멀티미디어 컨텐츠 저작 등에서 활용되고 있다. 기존의 시각언어는 시각 문장의 확장성과 객체간 산술적인 연산 관계의 표현은 용이하나, 시간적인 연산 관계는 기술하기 어렵다[3]. 이러한 단점을 보완한 동적 시각언어[3]는 객체 아이콘에 시간적 속성을 부여하고, 시간 흐름에 따라 동적인 변화를 지원함으로써, 다양한 미디어 객체들의 처리와 폭넓은 의미 표현이 가능하다. 그러나, 대부분의 시각언어는 단일 사용자 환경에서 컴퓨터와의 상호작용을 위한 인터페이스나 시각 프로그래밍은 지원할 수 있으나, 네트워크 환경에서 협업과 같은 다중 사용자 환경은 지원하기 어렵다[6, 8]. 현재에는 컴퓨터망에서 다양한 프로세스들간의 상호작용과 그에 따른 시각적 표현 방법이 요구되며[6, 8], 연산에서 사용자들의 참여 및 탈퇴에 따른 동적인 변화를 연산에 반영할 수 있어야 한다[1]. 다중 사용자들의 작업을 위하여 동기 및 비동기 협업의 지원이 필요하다[6, 7].

기존에 협업을 지원하기 위한 시각언어에 대한 관련연구로는 J. C. Grundy의 *EVPL*[6], K. D. Swenson

† 정 회 원 : 위덕대학교 컴퓨터공학과 교수
논문접수 : 2000년 1월 21일, 심사완료 : 2000년 9월 5일

의 *VPL*[9], C. Castroianni의 *Scarabaeus*[2], K. Zheng의 *Meteor*$_2$[11], K. Takeda의 *AP*[10], S. Kim의 *COVIL*[8] 등이 있다. C. Castroianni의 *Scarabaeus*는 프로젝트의 작업흐름을 표현하기 위하여 작업 행위를 의미하는 아이콘들과 아이콘들의 연결 관계로 표현된다. 생성된 시각 문장은 페트리 넷으로 변환 후 수행되며 다자간 비동기 협업을 지원한다. A. Sheth의 *Meteor*$_2$는 윈도우 및 웹 기반에서 병원 업무를 지원하는 시각 언어로서 C. Castroianni의 연구와 유사하다. 이 시각 언어는 환자의 등록, 의사의 진단, 간호사의 업무 등에서 발생하는 다자간 비동기 협업을 그래프 형태로 기술한 후, 페트리 넷으로 변환하여 수행한다. S. Kim의 시각언어는 작업 행위를 시간적 흐름에 따라 기술된다. 사용된 아이콘은 작업자의 정보를 가지며, 아이콘 연산자에 의하여 작업자들간의 비동기 협업 관계를 기술한다. K. D. Swenson의 시각언어 *VPL*은 절차적 작업의 수행 관계를 표현한다. 이 시각언어는 각 작업을 스테이지(stage)라는 아이콘으로 표현하고, 스테이지를 방향성 링크로 연결하여 하나의 플랜(plan)을 생성한다. 각 플랜은 해당 작업자가 소유 및 관리하며, 플랜에 표현된 스테이지들의 연결 관계는 이벤트의 전달 경로이다. 이러한 이벤트에 의하여 스테이지가 의미하는 작업이 순차적으로 활성화되면서 협업을 지원한다. J. C. Grundy의 *EVPL*은 *VPL*을 확장한 시각언어로서, 프로세스 스테이지에 구분자를 추가하여 다른 스테이지와의 의존성 표현을 용이하게 한다. 또한, 사용하는 아이콘의 역할을 세분화하고, 이벤트에 따른 각 행위를 정의하는 시각언어와 개발자를 지원하는 질의용 시각언어를 함께 지원한다. K. Takeda의 *AP*(Agent Prototyper)는 융통성 있는 작업 관계를 기술하는 시각언어로서 페트리 넷을 기반으로 한다. 작성된 시각 문장은 에이전트와 사용자들의 관계를 행위, 분기, 문서 등을 나타내는 아이콘과 정보 흐름 및 행위의 우선 순위를 나타내는 링크들의 조합으로 시각 문장을 생성한다.

기존 협업을 지원하는 대부분의 시각언어는 다자간 비동기 협업을 지원한다. 즉, 이러한 시각언어들은 대부분 특정 작업(환자의 등록 업무, 프로그램 버그 수정, 가전 제품 판매 및 A/S 등)의 수행 절차를 지원하는 언어로서, 사용되는 객체 아이콘은 프로그램 모듈과 그 모듈을 수행하는 사용자로 구성되어 협업을 지원한다. 그러나, 이러한 시각언어들은 비동기 협업은 잘 지원하지만, 다자간 다양한 동기 협업의 지원은 어려우며 시간의 흐름에 따라 변화되는 협업 관계의 지원은 더욱 어렵다. 그러므로, 본 논문에서 제안하는 시각언어는 기존 객체 아이콘과 연산자의 속성을 확장함으로써, 비동기 및 동기 협업을 더욱 효율적으로 지원한다. 또한, 사용자간 상호작용 시점에 따른 협업 관계를 λ함수로 분석하고, 제안한 시각언어를 다른 협업 지원 시각언어와 특징을 비교하고 분석한다.

본 논문의 구성은 다음과 같다. 제2장에서는 시각언어의 일반적인 속성을 설명하고, 제3장에서는 시각언어의 함수적 의미를, 제4장에서는 생성되는 시각문장의 인터프리터를 설명한다. 그리고, 제5장에서는 결론 및 연구 방향에 대하여 기술한다

2 시각언어의 일반 속성

기존 대부분의 시각언어는 일반적으로 위치, 시간, 내용 속성을 가진다[4, 5]. 위치 속성은 객체 아이콘이 시각 문장에서 배열되는 위치에 의하여 시각 문장의 의미가 변화됨을 나타내며, 시간 속성은 시간의 흐름에 따라 객체 아이콘의 속성이 변화됨으로써 시각 문장의 의미가 변화됨을 나타낸다. 그리고, 내용 속성은 시각 문장이 객체 아이콘의 증가나 감소에 따라 시각 문장의 의미가 변화됨을 나타낸다.

본 논문에서는 시각언어의 위치 속성과 시간 속성을 확장하여, 다중 사용자 환경에서 협업을 지원하는 시각언어를 정의한다. 즉, 시각언어의 위치 속성은 이차원 평면에 표현되는 객체 아이콘이 평면상의 단순한 위치 정보와 컴퓨터망에 연결된 각 사용자와 일대일의 관계 정보를 함께 가짐으로써 객체 아이콘은 협업을 수행하는 사용자를 나타낸다. 그리고, 시간 속성에서 기존 시각언어는 대부분 정적인 정보를 사용하여 작업을 처리하지만, 제안한 시각언어는 사용자의 참여 상태 값과 같이 수시로 변화 가능한 동적 정보를 사용하여 사용자간 상호작용 시점에 따른 협업 관계를 더욱 융통성 있게 표현한다. 제안한 시각언어에서 사용하는 객체 아이콘과 연산자를 대수적 관계로 정의하면 정의 2-1, 2-2와 같다.

[정의 2-1] 객체 아이콘 EX는 2 튜플로 $EX = (EX_I, EX_M)$이다. 여기서, EX_I는 객체 아이콘을 표현하는 이미지 집합이고, EX_M는 의미 부분이며 7 튜플

로 EX_M = (*A, U, S, IM, OM, IN, OUT*)이다. *A*는 행위의 집합, *U*는 행위를 수행하는 사용자의 집합, *S*는 사용자의 참여 상태 값 집합, $S = \{s_r, s_b, s_a\}$, *IM*은 사용자와 컴퓨터간의 상호작용으로부터 입력되는 메시지 집합, *OM*는 사용자와 컴퓨터간의 상호작용으로부터 출력되는 메시지 집합, 메시지 입력 및 출력 함수 *IN, OUT*는 다음과 같이 정의된다.

$IN : IM^+ \times S \rightarrow ID \times S$, *ID*는 입력 메시지 함수 *IN*에 의하여 생성된 임시 데이터의 집합,
$OUT : ID \times A \times U \times S \rightarrow OM^+ \times S$ §

정의 2-1에서 집합 *S*의 각 원소 s_r, s_b, s_a는 사용자의 협업 수행 가능, 수행중, 부재 상태를 나타낸다. 객체 아이콘은 이미지와 의미 부분으로 구성됨으로, 객체 아이콘에 대한 연산도 두 부분으로 구성된다. 정의 2-2는 연산자에 대한 정의이다.

[정의 2-2] 연산자 *OP*는 2 튜플로 $OP = (OP_I, OP_M)$이다. 여기서, OP_I는 객체 아이콘들의 이미지에 적용되는 연산자이고, OP_M는 객체 아이콘들의 의미 부분에 적용되는 논리적 연산자이다. 연산자 OP_I와 OP_M의 정의는 다음과 같다.

$OP_I : EX_I \times EX_I \rightarrow EX_I$,
$OP_M : EX_M \times EX_M \rightarrow EX_M$ §

제안한 시각언어에서 연산자는 사용자간 다양한 협업을 지원하기 위하여 비동기 및 동기 협업 관계를 제공한다. 연산자의 종류로는 비동기(op_a), 부분동기(op_p), 복합동기(op_c), 완전동기(op_s) 연산자가 있다. 각 연산자는 협업에서 사용자간 상호작용 시점을 기준으로 분류한 것이다. 비동기 연산자는 사용자간 순차적인 협업 관계를 의미하며, 완전동기 연산자는 사용자간 동시적인 협업 관계를 의미한다. 부분동기 및 복합동기 연산자는 완전동기 및 비동기 연산자로부터 파생된 연산자로, 부분동기 연산자는 상호작용 시점에서 조건에 따른 동기적 협업을 지원하는 연산자이다. 복합동기 연산자는 비동기 및 완전동기 연산자의 속성을 모두 가진 연산자로 비동기 연산을 먼저 수행한 후 완전동기 연산을 수행한다. 각 연산자의 의미는 다음 장에서 자세히 설명한다.

제안한 시각언어의 문법 *VG*는 *VG* = (*UG, R*)이며, *UG*는 두 개의 객체 아이콘과 한 개의 연산자로 구성되는 단위 시각 문장을 생성하는 문법이고, *R*은 단위 시각 문장을 결합하는 규칙 집합이다. 즉, 단위 시각 문장은 3 튜플로서 (*OP EX EX*)이다. 문법 *UG*는 *G*(*UG*) = (*N, T, P, S*)이며, *N*는 비단말 기호의 집합으로 *N* = {*S, C, C1, S1, Main, Sub, SubC, OP*}, *T*는 단말 기호의 집합으로 *T* = {  ,  ,  , ,  ,  ,  ,  ,  ,  }, *P*는 생성 규칙의 집합으로 $P = \{p_1, \cdots, p_7\}$, *S*는 시작 기호이다. 단말 기호 집합의 각 원소 의미는 다음과 같다. 기호  ,  는 협업의 시작과 마침을 의미하며,  ,  는 부분적인 협업의 시작과 마침을 의미하고, 시각 문장의 모듈화를 위하여 사용된다. 기호  는 부분적인 협업을 내포하는 객체 아이콘을 의미하며,  는 협업에 참여하는 사용자를 의미한다. 기호  ,  ,  , 는 각각 비동기, 복합동기, 부분동기, 완전동기 연산자를 의미한다. 그리고, 생성 규칙은 다음과 같다.

$p_1 : S ::$ = *Main* | *Sub*

$p_2 : Main ::=$  → *C* |  → *C* |  → *C* | *S1*

$p_3 : C ::$ =  | *C1*

$p_4 : S1 ::$ =  *OP* 

$p_5 : OP ::$ =  |  | 

$p_6 : C1 ::$ =  | 

$p_7 : Sub ::$ = → *SubC* | → *SubC* | → *C* | *S1*

$p_8 : SubC ::=$  | *C1*

규칙 집합 *R*은 $R=\{r_1, \cdots, r_7\}$이다. 여기서, 기호 *vs*는 단위 시각 문장, 기호 *VS*는 시각 문장의 집합, 함수 N은 집합에 대한 원소의 수를 반환하는 함수이다.

$r_1 : N(\{ex_s \mid ex_s \in EX, vs \in VS,$ 그리고 $vs = (ex_s\ op\ ex)\}) = 1.$

$r_2 : N(\{ex_e \mid ex_e \in EX, vs \in VS,$ 그리고 $vs = (ex$

$op\ ex_e)$)) = 1.

r_3 : 각 사용자 아이콘은 단지 한 명의 사용자와 연관된다.

r_4 : 시각 문장은 비동기 또는 복합 연산자에 의한 순환적 협업 관계를 가지지 않는다.

r_5 : 두 객체 아이콘 사이에는 단지 하나의 연산자만 존재할 수 있다.

r_6 : 단위 시각 문장의 객체 아이콘은 비동기 연산자나 복합 연산자에 연결되어야 한다.

r_7 : 시각 문장 VS는 연결되지 않은 객체 아이콘을 포함하지 않는다.

문법 VG는 다자간 다양한 협업 관계를 표현하고 협업에서 사용자의 참여 및 탈퇴를 효율적으로 표현한다. (그림 1)은 제안한 시각언어로부터 생성되는 시각 문장의 예를 나타낸다. 표현된 시각 문장의 의미는 회사에서 부서간 회의를 위한 협업을 표현한다. 여기서, 각 사용자는 화이트보드와 카메라를 사용하여 협업을 수행한다고 가정한다.

(그림 1)에서 시각 문장의 수행 의미는 다음과 같다.

(1) 기획팀장, 영업팀장, 생산팀장이 그룹 회의를 수행한다.

(2) 총무팀, 생산팀, 연구팀별로 각각 그룹회의를 수행한다.

(3) 각 팀의 회의가 모두 종료된 후, 부사장과 기획 팀장이 그룹 회의를 수행한다.

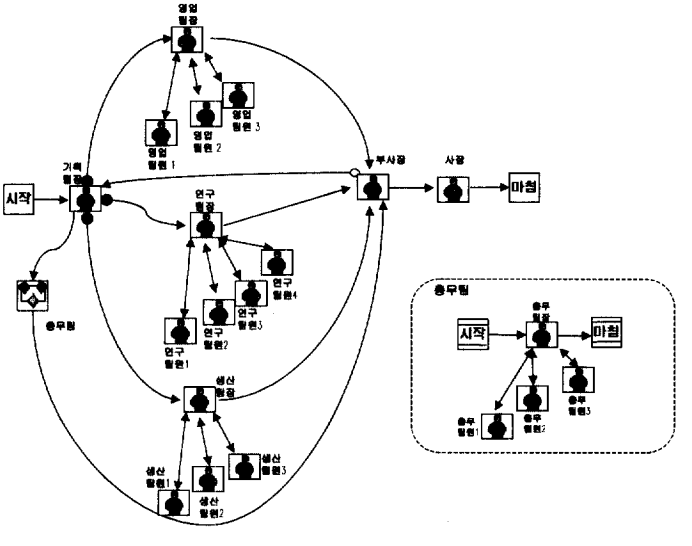
(4) 부사장과 기획팀장의 회의가 종료된 후 그 결과 메시지가 사장에게 전송된다.

(5) 모든 그룹회의가 종료된다.

3 시각언어의 함수적 속성

제안한 시각언어는 연산자와 객체 아이콘을 조합함으로써 협업에서 사용자간 다양한 협업 관계를 지원한다. 연산자에 따른 함수적 속성은 다음과 같다.

비동기 연산자 op_a의 연산 도메인과 의미 함수는 식 (1)과 같다. 여기서, 집합 $Interaction_a$는 협업 관계의 집합으로 원소 $async$는 사용자간 비동기 상호작용을 의미하며, 원소 $false$는 상호작용의 단절을 의미한다. 원시 함수 ***combine***은 두 객체 아이콘의 이미지와 비동기 연산자의 이미지를 결합한 이미지를 반환 값으로 가진다. 수행환경 Env와 Env'는 시각 문장을 생성할 때, 객체 아이콘에 연관된 사용자, 사용자의 역할, 사용자 상태 값이 정의된다. 원시 함수 ***isnonstrict***는 입력된 수식이 출력 값을 생성하면, 함수 ***true***를 반환 값으로 가지고, 그렇지 않으면 함수 ***false***를 반환한다. 그리고, i_{opa}는 비동기 연산자의 이미지, 함수 ***in***, ***out***은 정의 2-2의 함수 IN과 OUT, 기호 $\mapsto$는 변수의 바인딩, Oi는 객체 아이콘의 집합, U는 사용자 집합, S는 상태 집합, A는 행위의 집합을 의미한다. 그리고, 변수



(그림 1) 그룹 회의를 표현하는 시각 문장의 예

u, u'는 사용자, 변수 s, s'는 상태 값, 변수 a, a'는 행위를 나타내며, $u \neq u'$이고 $x, y \in EX$이다.

$$
\begin{aligned}
&\boldsymbol{op_a} : Oi \times Oi \rightarrow Interaction_a, \qquad (1)\\
&\boldsymbol{op_{ai}} : Oi \times Oi \rightarrow Image,\\
&\boldsymbol{op_{am}} : Oi \times Oi \rightarrow ((IM^+ \times S) \times (IM^+ \times S) \rightarrow\\
&\quad (((ID \times Env_1) \times (ID \times Env_2)) \rightarrow ((OM^+ \times\\
&\quad S) \times (OM^+ \times S)) \rightarrow Interaction_a\ ),\\
&\quad Env_1 = \{U \mapsto u,\ S \mapsto s,\ A \mapsto a\},\\
&\quad Env_2 = \{U \mapsto u',\ S \mapsto s',\ A \mapsto a'\},\\
&\quad Interaction_a = \{async.,\ false\},\\
&\quad \boldsymbol{combine} : Image \times Image \times Image \rightarrow Image,\\
&\quad \boldsymbol{isnonstrict} : expression \rightarrow \{\boldsymbol{true}, \boldsymbol{false}\},\\
&\boldsymbol{op_a} = \lambda x\ y.\ ((\boldsymbol{op_{ai}}\ x\ y)\ (\boldsymbol{op_{am}}\ x\ y)),\\
&\boldsymbol{op_{ai}} = \lambda x\ y.\ \boldsymbol{combine}\ x\ y\ i_{opa},\\
&\boldsymbol{op_{am}} = \lambda x\ y.\ ((\boldsymbol{isnonstrict}(\boldsymbol{out}\ (\boldsymbol{in}\ x)\ u\ a)) \rightarrow (\boldsymbol{out}\\
&\quad (\boldsymbol{in}\ y)\ u'\ a') \mid false).
\end{aligned}
$$

시각 문장 (ex_1 $\boldsymbol{op_a}$ ex_2)의 의미 연산은 식 (2)와 같다. 여기서, ex_1, $ex_2 \in EX$, $im \in IM$, $true \mapsto s_r$이다.

$$
\begin{aligned}
&\boldsymbol{op_a}\ ex_1\ ex_2 \qquad (2)\\
&= (\lambda x\ y.\ ((\boldsymbol{op_{ai}}\ x\ y)\ (\boldsymbol{op_{am}}\ x\ y)))\ ex_1\ ex_2\\
&= ((\boldsymbol{combine}\ ex_{1i}\ ey_{2i}\ i_{opa})\ ((\boldsymbol{isnonstrict}(((\boldsymbol{manipulate}\\
&\quad ex_1.im\ ex_1.u\ ex_1.a)\ s_r)) \rightarrow (\boldsymbol{out}\ (\boldsymbol{in}\ ex_{2m})\ ex_2.u\ ex_2.a)\\
&\quad \mid false))\\
&= ((\boldsymbol{combine}\ ex_{1i}\ ey_{2i}\ i_{opa})\ ((\boldsymbol{manipulate}\ ex_2.im\ ex_2.u\\
&\quad ex_2.a)\ s_r)).
\end{aligned}
$$

여기서, 함수 ***isnonstrict***는 연산 결과로서 ***true***를 반환 값으로 가질 경우이며, 함수 ***manipulate***는 사용자가 데이터를 처리하여 뷰(화면)를 변화시키는 출력 메시지를 생성하는 원시 함수이다.

완전동기 연산자 $\boldsymbol{op_s}$의 연산 도메인과 의미 함수는 식 (1), 식 (2)와 유사하다. 원시 함수 ***combine***은 단지 연산자의 이미지를 비동기 연산자의 이미지 대신에 완전동기 연산자 이미지를 사용한다. 의미 함수 $\boldsymbol{op_{sm}}$에서 함수 내에 포함된 연산들의 연산 순서는 λ수식의 이거(eager) 연산 순서로서 가장 안쪽에 내포된 함수부터 연산을 수행한다. 식 (3)에서 $\boldsymbol{op_{sm}}$과 $\boldsymbol{op_{sm_aux}}$에 내포된 조건 수식은 두 객체 아이콘의 사용자가 모두 협업에 참여함으로써, 조건의 결과가 참이면 협업은 계속 수행된다. 즉, 두 사용자가 동기적 협업을 수행한다. 그리고, 집합 $Interaction_s$의 원소 sync.는 사용자간 동기적 협업을 의미한다. 여기서, 함수 ***and***는 수식 $\lambda x\ y.\ (x \rightarrow y \mid false)$이며, $u \neq u'$이고 $x, y \in EX$이다.

$$
\begin{aligned}
&\boldsymbol{op_s} : Oi \times Oi \rightarrow Interaction_s, \qquad (3)\\
&\boldsymbol{op_{si}} : Oi \times Oi \rightarrow Image,\\
&\boldsymbol{op_{sm}} : Oi \times Oi \rightarrow ((IM^+ \times S) \times (IM^+ \times S) \rightarrow (((ID \times Env_1)\\
&\quad \times (ID \times Env_2)) \rightarrow ((OM^+ \times S) \times (OM^+ \times S))\\
&\quad \rightarrow Interaction_s\ ),\\
&\quad Env_1 = \{U \mapsto u,\ S \mapsto s,\ A \mapsto a\},\\
&\quad Env_2 = \{U \mapsto u',\ S \mapsto s',\ A \mapsto a'\},\\
&\quad Interaction_s = \{sync.,\ true,\ false\},\\
&\boldsymbol{op_{sm\_aux}} = Oi \times Oi \rightarrow (((IM^+ \times S) \times (IM^+ \times S))\\
&\quad \rightarrow ((((ID \times Env_1) \times (ID \times Env_2))\\
&\quad \rightarrow ((OM^+ \times S) \times (OM^+ \times S)))\\
&\quad \rightarrow Interaction_s)),\\
&\boldsymbol{op_s} = \lambda x\ y.\ ((\boldsymbol{op_{si}}\ x\ y)\ (\boldsymbol{op_{sm}}\ x\ y)),\\
&\boldsymbol{op_{si}} = \lambda x\ y.\ \boldsymbol{combine}\ x\ y\ i_{ops},\\
&\boldsymbol{op_{sm}} = \lambda x\ y.((\boldsymbol{and}\ \boldsymbol{isnonestrict}(\boldsymbol{out}(\boldsymbol{in}\ x)\ u\ a)\\
&\quad \boldsymbol{isnonestrict}(\boldsymbol{out}\ (\boldsymbol{in}\ y)\ u'\ a'))\\
&\quad \rightarrow (\boldsymbol{op_{sm\_aux}}\ x\ y) \mid false),\\
&\boldsymbol{op_{sm\_aux}} = \lambda xy.((\boldsymbol{and}\ \boldsymbol{isnonestrict}(\boldsymbol{out}(\boldsymbol{in}\ x)\ u\ a)\\
&\quad \boldsymbol{isnonestrict}(\boldsymbol{out}\ (\boldsymbol{in}\ y)\ u'\ a'))\\
&\quad \rightarrow \boldsymbol{op_{sm\_aux}}\ x\ y) \mid true).
\end{aligned}
$$

함수 $\boldsymbol{op_{sm}}$의 반환 값이 *false*인 경우는 사용자간 협업이 중단된 경우를 의미한다. 함수 $\boldsymbol{op_{sm_aux}}$은 함수 $\boldsymbol{op_{sm}}$의 보조 함수로서 사용자간 동기 협업의 수행을 지원하며, 사용자간에 협업이 중단되면, 연산 결과 값으로 *true* 값을 반환한다. 시각 문장 (ex_1 $\boldsymbol{op_s}$ ex_2)의 의미 연산은 식 (4)와 같다. 단, $x, y \in EX$이고, $true \mapsto s_r$이다.

$$
\begin{aligned}
&\boldsymbol{op_s}\ ex_1\ ex_2 \qquad (4)\\
&= (\lambda x\ y.\ ((\boldsymbol{op_{si}}\ x\ y)\ (\boldsymbol{op_{sm}}\ x\ y)))\ ex_1\ ex_2\\
&= ((\boldsymbol{combine}\ ex_{1i}\ ey_{2i}\ i_{ops})\ ((\boldsymbol{and}\ \boldsymbol{isnonestrict}((\\
&\quad \boldsymbol{manipulate}\ ex_1.im\ ex_1.u\ ex_1.a)\ s_r)\ \boldsymbol{isnonestrict}\\
&\quad ((\boldsymbol{manipulate}\ ex_2.im\ ex_2.u\ ex_2.a)\ s_r)) \rightarrow (\boldsymbol{op_{sm\_aux}}\\
&\quad ex_{1m}\ ex_{2m}) \mid false)\\
&= ((\boldsymbol{combine}\ ex_{1i}\ ey_{2i}\ i_{ops})\ ((\boldsymbol{and}\ \boldsymbol{isnonestrict}((\\
&\quad \boldsymbol{manipulate}\ ex_1.im\ ex_1.u\ ex_1.a)\ s_r)\\
&\quad \boldsymbol{isnonestrict}((\boldsymbol{manipulate}\ ex_2.im\ ex_2.u\ ex_2.a)\ s_r))\\
&\quad \rightarrow (\boldsymbol{op_{sm\_aux}}\ ex_{1m}\ ex_{2m}) \mid true)
\end{aligned}
$$

여기서, 함수 ***isnonestrict***는 연산 결과로서 함수 ***true***를 생성하면, 함수 op_{sm_aux}의 재귀적 호출에 의하여 객체 아이콘 ex_1과 ex_2의 사용자간 동기적 협업을 지원한다. 그리고, 두 사용중 한 사용자가 협업에서 탈퇴하면, 동기적 협업은 중단된다.

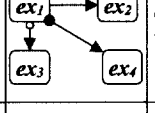
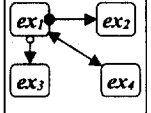
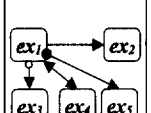
부분동기 연산자 op_p는 완전동기 연산자에서 파생된 연산자로, 두 객체 아이콘에 연관된 한 사용자가 협업을 수행함으로써 동기적 협업을 지원한다. 시각 문장 (ex_1 op_p ex_2)의 부분 동기 연산은 단지 객체 아이콘 ex_1의 사용자가 협업에 참여할 경우에 객체 아이콘 ex_1과 ex_2의 사용자는 서로간에 동기적 협업을 수행한다.

복합동기 연산자 op_c는 비동기 연산자와 완전동기 연산자가 함께 결합된 연산자로써, 복합동기 연산은 먼저 비동기 연산이 수행된 후에 완전동기 연산을 수행한다. 시각 문장 (ex_1 op_c ex_2)의 의미 연산은 수식 (5)와 같다. 복합동기 연산이 종료한 후에는 객체 아이콘 ex_2에 연결된 비동기 연산자나 복합동기 연산자의 메시지 흐름(화살표의 머리) 방향에 따라 다른 객체 아이콘의 사용자와 협업을 활성화시킨다.

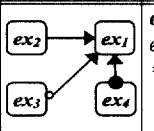
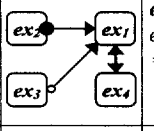
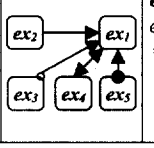
$$
\begin{aligned}
&op_c\ ex_1\ ex_2 \qquad (5)\\
&= \lambda xy.((\textbf{\textit{isnonstrict}}(op_{am}\ x\ y)\\
&\quad \rightarrow (op_{pm}\ x\ y) \mid false)\ ex_1\ ex_2\\
&= ((\textbf{\textit{isnonstrict}}(op_{am}\ ex_1\ ey_2)\\
&\quad \rightarrow (op_{pm}\ ex_1\ ey_2) \mid false)
\end{aligned}
$$

<표 1>은 생성되는 시각 문장의 의미 연산들을 나타낸다. 여기서, 원시 함수 ***execute***는 시각 문장의 수행을 시작하는 함수이다.

〈표 1〉 시각 문장의 의미 연산

	시각 문장	의미 연산
vs_1		***execute***[[($ex_1 \rightarrow ex_2$, $ex_1 \circ\!\!\rightarrow ex_3$, $ex_1 \bullet\!\!\rightarrow ex_4$)]] = ((***and*** ($op_p$ ex_1 ex_3)(op_p ex_1 ex_4)) →(***and***(op_a ex_1 ex_2)(op_a ex_1 ex_4)) | *false*)
vs_2		***execute***[[($ex_1 \bullet\!\!\rightarrow ex_2$, $ex_1 \circ\!\!\rightarrow ex_3$, $ex_1 \leftrightarrow ex_4$)]] = ((***and*** (***and*** (op_p ex_1 ex_2) (op_p ex_1 ex_3)) (op_p ex_1 ex_3)) → (op_a ex_1 ex_2) | *false*)
vs_3		***execute***[[($ex_1 \rightarrow ex_2$, $ex_1 \circ\!\!\rightarrow ex_3$, $ex_1 \leftrightarrow ex_4$, $ex_1 \bullet\!\!\rightarrow ex_5$)]] = ((***and*** (***and*** (op_p ex_1 ex_3) (op_s ex_1 ex_4)) (op_p ex_1 ex_5)) → (***and*** (op_a ex_1 ex_2) (op_a ex_1 ex_3)) | *false*)

〈표 1〉 시각 문장의 의미 연산 (계속)

	시각 문장	의미 연산
vs_4		***execute***[[($ex_2 \rightarrow ex_1$, $ex_3 \circ\!\!\rightarrow ex_1$, $ex_4 \bullet\!\!\rightarrow ex_1$)]] = (***and*** (***and*** (((***op***$_p$ ex_4 ex_1)) →(***op***$_a$ ex_4 ex_1) | *false*) (op_a ex_2 ex_1)) (op_p ex_3 ex_1))
vs_5		***execute***[[($ex_2 \bullet\!\!\rightarrow ex_1$, $ex_3 \circ\!\!\rightarrow ex_1$, $ex_4 \leftrightarrow ex_1$)]] = (***and*** (($op_p$ ex_2 ex_1) →(op_a ex_2 ex_1) | *false*) → (op_p ex_4 ex_1)) (op_p ex_3 ex_1))
vs_6		***execute***[[($ex_2 \rightarrow ex_1$, $ex_3 \circ\!\!\rightarrow ex_1$, $ex_4 \leftrightarrow ex_1$, $ex_5 \bullet\!\!\rightarrow ex_1$)]] = (***and*** (***and*** ((op_p ex_5 ex_1) → (op_a ex_5 ex_1) | *false*)(op_a ex_2 ex_1)) → (op_s ex_4 ex_1) | *false*) (op_p ex_3 ex_1))

제안한 시각언어는 비동기, 복합동기, 부분동기, 완전동기 연산자와 객체 아이콘을 조합함으로써 시각 문장을 생성한다. 생성된 시각 문장의 의미 연산은 동기 및 비동기 연산자의 함수적 관계에 의하여 동기 및 비동기 협업을 수행하여 사용자들의 참여 및 탈퇴를 동적으로 지원한다. 즉, 동기 연산에 의하여 동시에 다자간 협업을 지원하고, 비동기 연산은 순차적인 협업을 지원하면서 사용자의 참여 및 탈퇴와 동기 협업의 생성 및 소멸을 지원한다.

제안한 시각언어에서 생성하는 시각 문장은 비동기 연산에 의하여 시작 아이콘으로부터 마침 아이콘까지 선형적으로 연결되는 부분 시각 문장을 적어도 하나 이상 포함한다. 그러므로, 부분 시각 문장 vs가 $vs = \{(ex_1\ iop_a\ ex_2), (ex_2\ iop_a\ ex_3), (ex_3\ iop_a\ ex_4), \cdots, (ex_{n-1}\ iop_a\ ex_n)\}$, $ex_i \in EX$, i=1, ⋯, n이고, 함수 $ST_{interaction}()$와 $ET_{interaction}()$는 객체 아이콘을 입력으로 하며 객체 아이콘의 사용자가 수행하는 협업 시작 및 종료 시간을 반환한다면, 다음과 같은 식이 성립한다. 즉, $ET_{interaction}(ex_1) \le ST_{interaction}(ex_2) \le ET_{interaction}(ex_2) \le ST_{interaction}(ex_3) \le ET_{interaction}(ex_3) \le \cdots \le ET_{interaction}(ex_{n-1}) \le ST_{interaction}(ex_n)$ 이다. 그러므로, 시각 문장 vs는 순차적인 시간 속성을 가짐으로써, 시각 문장 vs의 의미 연산에 따라 사용자들은 비동기 협업을 수행한다. 또한, 제안하는 시각 언어는 동기 연산에 의하여 생성되는 부분 시각 문장을 포함할 수 있다. 만약, 부분 시각 문장 vs 가 $vs' = \{(ex_1\ iop\ ex_2), (ex_1\ iop\ ex_3), (ex_1\ iop\ ex_4), \cdots, (ex_1\ iop\ ex_n)\}$, $ex_i \in EX$,

$iop \in \{iop_s, iop_p\}$, $i=1, \cdots, n$ 이고, β가 작은 상수이면, 다음과 같은 식이 성립한다. 즉, $|ST_{interaction}(ex_1) - ST_{interaction}(ex_j)| < \beta$, $j=1, \cdots, n$ 이다. 시각 문장 vs'는 동시적인 시간 속성을 가짐으로써, 시각 문장의 vs'의 의미 연산에 의하여 사용자들은 특정 시간 동안에 서로간의 상호작용에 따라 동기 협업을 수행한다. 그러므로, 제안한 시각언어는 이러한 동기 및 비동기 연산의 수행에 의하여 기존의 협업 지원 시각언어들보다 더욱 다양하고 효율적인 협업을 지원한다.

4. 인터프리터

제안한 시각언어가 생성하는 시각 문장은 <표 1>과 같이 각 연산자의 함수적 속성들의 결합으로 협업을 지원한다. 시각 문장의 수행은 사용자간 상호작용 시점에 따라 활성화되는 시각 문장들만 연산되면서 시각 문장은 점진적으로 협업의 수행을 지원한다. 즉, 함수적 속성을 가지는 시각 문장의 수행은 다음과 같은 단계로 이루어진다. (1) 생성된 시각 문장을 연산자에 따라 비동기 연산 및 동기 연산 관계를 가진 부분 시각 문장으로 분류한다. (2) 각 부분 시각 문장은 의미 분석에 따라 개념 그래프로 변환한다. (3) 개념 그래프의 각 노드는 협업을 위한 실행 모듈로 맵핑된다. (4) 개념 그래프의 각 노드는 연산자의 함수적 연산에 따라 실행 모듈을 수행한다. 이러한 단계는 시각 문장에 포함된 부분 시각문장들을 모두 수행할 때까지 반복된다. 여기에서, 실행 모듈은 실제 작업을 수행할 어플리케이션이다.

제안한 시각언어는 기존 시각언어가 지원하기 어려운 분산 환경에서 자원 관리 및 협업의 수행을 효율적으로 표현한다. 특히, 사용자간 동기적 협업 관계를 효율적으로 표현한다. 제안한 시각언어는 다자간 그룹 회의를 위한 시스템[12]에 적용되었으며, 구현 환경은 *Solaris* 2.5의 *OPEN WINDOW* 환경에서 *OSF/Motif*와 *C++*언어를 사용하여 구현하였다.

<표 2>는 기존 협업 지원 시각언어와 제안한 시각언어와의 특성을 비교한 표이다. 비교 대상은 J. C. Grundy의 *EVPL*[6], K. D. Swenson의 *VPL*[9], C. Castroianni의 *Scarabaeus*[2], K. Zheng의 *Meteor*$_2$[11], K. Takeda의 *AP*[10], S. Kim의 *COVIL*[8]을 비교 분석하였다.

〈표 2〉 기존 시각언어와의 비교

시각언어 / 특성	*EVPL*	*VPL*	*Scarabaeus*	*METEOR*$_2$	*AP*	*COVIL*	제안한 시각언어
시간 속성	지 원	지 원	지 원	지 원	지 원	지 원 안 됨	지 원
위치 속성	지 원	지 원	지 원	지 원	지 원	지 원	지 원
비동기 협업	지 원	지 원	지 원	지 원	지 원	지 원 안 됨	지 원
동기 협업	지 원 어려움	지 원 어려움	지 원 어려움	지 원 어려움	지 원 어려움	지 원 안 됨	지 원
사용자간 협업 관계	1 : 1	1 : 1	1 : 1	1 : 1	1 : 1	지 원 안 됨	1 : N

(N : 사용자 수, N > 0)

<표 2>에서와 같이 제안한 시각언어는 사용자간 협업 관계를 1 : 1관계에서 1 : N의 관계로 확장함으로써, 동기적 협업 관계를 효율적으로 표현하고 다양한 협업 관계를 지원한다.

5. 결 론

기존 대부분의 시각언어는 시각 문장의 확장성과 객체간 다중 관계의 표현, 객체의 시간 속성 부여가 가능함으로써, 멀티미디어 객체의 처리와 다양한 의미의 표현을 지원한다. 그러나, 대부분의 시각언어는 단일 사용자 환경을 지원하고, 협업을 지원하는 시각언어들은 비동기 관계만 지원함으로 사용자간 동기 협업을 지원하기 어렵다.

제안한 시각언어가 생성하는 시각 문장은 객체 아이콘과 동기 연산자의 조합으로 구성된다. 객체 아이콘은 협업을 수행하는 사용자를 의미하며, 동기 연산자는 사용자간 상호작용 시점에 따른 협업 관계를 기반으로 구성되었다. 제안한 시각언어의 특징은 다음과 같다. (1) 시각 문장은 지리적으로 분리된 다자간 협업을 지원한다. (2) 시각 문장은 사용자 중심으로 기술되어 사용자간 명확한 협업 관계를 표현하고 지원한다. (3) 동기 연산자와 객체 아이콘의 조합으로 사용자간 협업 관계의 동적 변화를 지원한다. 즉, 완전동기 연산과 비동기 연산에 의하여 사용자의 협업 참여 및 탈퇴를 효율적으로 기술한다. (4) 생성되는 시각 문장은 복합 작업 아이콘의 사용으로, 부분 시각 문장의 재사용과 시각 문장의 확장성을 제공하고 작업 공간의 효율적 사용과 모듈화를 지원한다.

제안한 시각언어의 문제점은 객체 아이콘과 아이콘 연산자가 작업 공간에 많이 나열됨으로써, 설계자가 시각 문장을 이해하기 어렵다. 또한, 다자간 순환적 협

업 관계를 기술할 수 없다. 그러므로, 시각 문장의 가독성 향상을 위한 객체 아이콘 이미지의 세분화 및 분류화에 대한 연구와 순환적 협업의 지원에 대한 연구가 앞으로의 연구 방향이다.

참 고 문 헌

[1] P. Bottoni, S. Chang, M. F. Costabile, S. Levialdi, and P. Musio, "On the Specification of Dynamic Visual Languages," Proc. of the 14th IEEE Symposium on Visual Languages, 1998.

[2] C. Castroianni, Development of a Workflow Design Tool for the Scarabaeus Project, Masters thesis, Dept. of Computer Science, Univ. of Twente, 1995.

[3] S. Chang, "Dynamic Visual Languages," Proc. of the IEEE Symposium on Visual Languages, pp.308-315, 1996.

[4] S. Chang, Principles of Visual Programming Systems, Prentice-Hall, 1990.

[5] S. Chang, "Extending Visual Languages for Multimedia," IEEE Multimedia Magazine, Vol.3, No.3, pp.18-26, 1996.

[6] 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, pp.324-325, 1996.

[7] N. Hari Narayanan and R. Hubscher, "Visual Language Theory : Towards a Human -Computer Interaction Perspective," in Visual Language Theory, K. Marriott and B. Meyer (eds.), Springer-Verlag, 1998.

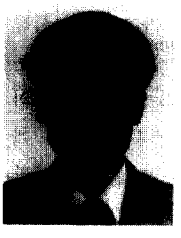
[8] S. Kim and M. Jin, "Collaborative Visual Languages," Proc. of the IASTED Int. Conf. : Artificial Intelligence and Soft Computing, pp.168-171, 1997.

[9] K. D. Swenson, "A Visual Language to Describe Collaborative Work," IEEE Int. Symposium for Visual Languages, pp.298-303, 1993.

[10] K. Takeda, M. Inaba, and K. Sugihara, "User Interface and Agent Prototyping for Flexible Working," IEEE Multimedia Vol.3, No.2, pp.40-50, 1996.

[11] K. Zheng, Designing Workflow Processes in METEOR$_2$ Workflow Management System, Master's thesis, Dept. of Computer Science, Univ. of Georgia, 1997.

[12] 김상욱, 김정미, 김태호, 이정훈, 배유석, "HITE : 객체 지향 멀티미디어 그룹웨어", 정보과학회논문지 (B), 제24권, 제12호, pp.1513-1521, 1997.



김 경 덕

e-mail : kdkim@mail.uiduk.ac.kr

1989년 경북대학교 자연과학대 (이학사)

1991년 경북대학교 컴퓨터공학과 (공학석사)

1999년 경북대학교 컴퓨터공학과 (공학박사)

1991년~1996년 (주)훼스트시스템 연구소 연구원

2000년~현재 위덕대학교 컴퓨터공학과 전임강사

관심분야 : 프로그래밍언어, 시각언어, 멀티미디어 프레젠테이션, 인터넷 컴퓨팅 등