객체지향을 기반으로한 추상화 정보의 시각화 시스템에 대한 연구

김 행 곤[†] · 한 은 주[††] · 정 연 기[†††]

요 약

소프트웨어 산업이 발달됨에 따라, 텍스트 위주의 정보보다는 시각적 정보의 요구가 점차적으로 증가하게 되었다. 이로 인해, 실세계에 존재하는 다양한 원시 정보를 사용자가 원하는 시각적인 설계 표현으로 나타내기 위해 자동화된 툴이 필요하게 되었다.

본 논문에서는 추상 정보를 의미 분석을 통해 파싱을 하고, 시각화 구조의 매핑을 통해 시각적 언어를 추출하는 방법론 및 툴에 대해 논한다. 이 툴은 정규 규칙을 사용하여 관계적 구조로 표현하고, 이를 시각화 구조로 매핑하여 시각 정보를 제공한다. 원시코드의 추상 정보를 시각 정보로 변환하는 제안 툴인 VOLS(Visual Object Layout System)는 문서를 시각화하여 사용자에게 이해성을 향상시키며, 기존 원시코드의 유지보수 측면에서 사용자에게 도움을 제공한다.

A Study on the Visual System of Object-Oriented Based on Abstract Information

Haeng-Kon Kim[†] · Eun-Ju Han[††] · Youn-Ki Chung[†††]

ABSTRACT

As software industry progresses, the necessity of visual information have increased more than text-oriented information. So, automatic tools are required to satisfy a user's desire for visual design representation of various source information in the real-world.

In this paper, we discuss the methodology and tools for parsing abstract information through semantic analysis and extracting visual information through visual mapping. Namely, as to abstract informations are represented as relational structure and then mapped into visual structure using regular rule, user can obtain visual information. We suggest VOLS(Visual Object Layout System) to transform a abstract information to visual information. It can improve user understandability and assist a maintenance for existing source code.

1. 서 론

소프트웨어 산업이 확산되고 시각적 정보에 대한 요구가 증가함에 따라 사용자와 컴퓨터간의 상호 대화는 자연스럽고 쉽게 이루어져야 한다. 또한, 시스템의 능력을 최대로 이용하는 방법과 사용자가 프로그램을 쉽게 사용할 수 있는 방법에 많은 관심이 집중되고 있다. 다양한 시각적인 자원을 이용하여 사용자에게 인터페이스를 제공해 주며, 이런 인터페이스를

† 종신회원:대구효성가톨릭대학교 컴퓨터공학과 부교수
†† 준 회 원:대구효성가톨릭대학교 대학원 전산통계학과
††† 정 회 원:경일대학교 전자계산학과 부교수
논문접수:1996년 1월 15일, 심사완료:1997년 8월 26일

개발하기 위해 요구되는 정보는 텍스트 위주의 정보보다는 그래픽 위주의 정보가 소프트웨어 구조를 쉽게 이해시키고, 편의성을 제공하게 되며, 수천 단어의 정보보다도 가치가 있다. 그러므로 인간은 문장을 자세히 조사하고 이해하는 것보다는 그림 형식으로 표현된 정보를 통해 인식하는 것이 빠르다.

또한, 이런 인터페이스를 통한 시각화 전달은 인간과 컴퓨터 사이에 없어서는 안될 HCI(Human Computer Interface)에 기반한것이며, 현재 소프트웨어 기술을 통해 새로운 연구 영역으로 자리잡히고 있다. 인간의 시각적 정보에 대한 이해는 텍스트 형식보다 우수함으로 그래픽을 사용한 표현을 선호하게 되었다. 컴퓨터를 사용하여 정보를 얻는 경우, 단순 작업의 경우라 할지라도 초보자들은 많은 노력과 시간을 낭비하게 되므로 정보를 쉽게 얻을 수 있는 시각적 프로그래밍 환경 지원에 대한 연구가 이루어지고 있다. 또한, 시각적 표현은 보다 많은 의미를 함축적인 표현 단위로 인간에게 전달할 수 있으며, 기억을 도와주고, 많은 흥미를 유발할 수 있다. 또한, 시각화를 위한 자료들이 단지 보여준다는 시각적인 것뿐만 아니라 그 자료들을 정보 저장소에 저장하여 또 다른 시각적인 정보로 나타낼 수 있다면 이를 재설계, 재구성[1, 2]함으로 새로운 정보로 표현할 수 있다.

본 논문에서는 추상 정보를 사용자가 요구하는 시각 정보로 표현하기 위한 방법론 및 도구에 대해 연구한다. 우선, 실세계에 존재하는 다양한 추상 정보를 시각 정보로 표현하기 위해 기술서로 스캐너 기술 화일(SDF:Scanner Description File)과 문법 기술 화일(GDF:Grammar Description File)을 작성한다. 이 정보는 분석기에서 스캐닝(scanning)과 파싱(parsing)을 통해 의미 분석을 하고, 중간 코드 생성과 코드 최적화를 거쳐 사용자가 요구하는 실행 가능한 정보를 얻게 된다. 또한, 실행 가능한 정보에 대한 적절한 컴파일러를 통해 원시정보를 시각정보로 표현할 수 있다.

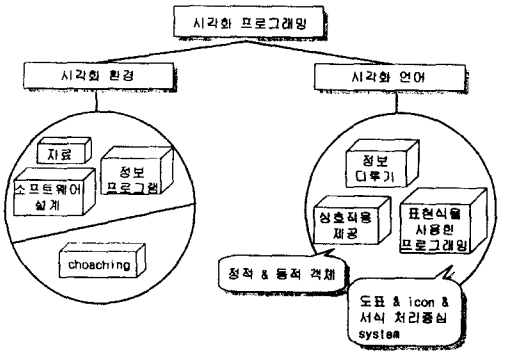
따라서, 의미 분석과 코드생성을 하는 컴파일러와 전체 실행환경이되는 부분을 연결하는 VOLS(Visual Object Layout System)를 구현하였다. 또한, 툴상에서 사용자가 정의해야될 스캐너 기술 화일과 문법 기술 화일을 문서화하므로 사용자에게 이해성을 향상시키며, 유지 보수 측면에서도 도움을 줄 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 일반적인 시각화 프로그래밍 환경에 대해 설명한다. 그리고 3장에서는 VOLS를 설계하고 적용예를 제시한다. 4장에서는 시스템의 구현에 대해서 살펴본다. 마지막으로 결론 및 향후 연구 방향을 제시함으로 맺는다.

2. 관련 연구

2.1 시각화 프로그래밍

사용자로 하여금 시각적 형태로 프로그램을 기술해 주는 시스템으로 시각화 프로그램은 시각적 정보를 다양한 형태로 지원하며 텍스트를 기반으로 한다. 또한, 그래픽 표현식을 사용하여 시각적 언어가 의미를 갖는 표현으로 나타낸다. 시각화 프로그래밍은 광범위하며, 그 범주는 이미 정의된 바 있다. 이 정의에 의하면 다음(그림 1)에서 설명하듯 두 가지 서로 다른 관점을 가진 시각화 프로그래밍의 범주를 나타낼 수 있다.



(그림 1) 시각화 프로그래밍의 범주
(Fig. 1) Category of visual programming

시각화 환경은 프로그램 개발에 관련되어 관심사는 주로 시스템 개발 툴의 시각화에 있으며 유지 보수와 디버그 검출 또한 중요시되고 있다. 사용자와 컴퓨터간의 상호작용을 지원하며 "보여준다는 것"에 주된 의미를 둔다.

또한, 시각화 언어는 정보처리, 프로그램에 의해 수행된 상호작용의 형태, 그리고 표현식을 사용한 언어로 나타낼 수 있다. 프로그래머는 시각적 프로그래밍 언어를 사용하고, 사용자는 이 언어를 프로그램 하여 사용자가 원하는 정보로 표현할 수 있다. 이 정보는

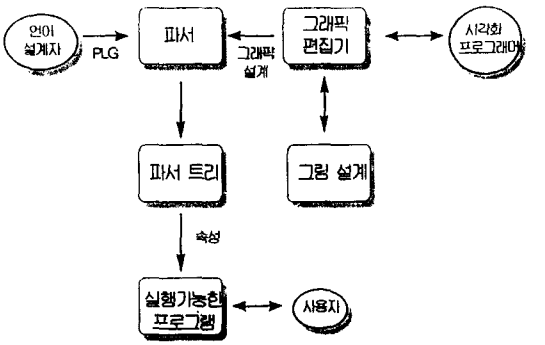
정적 행위와 동적 행위로 구분하며[3, 4], 이를 조합함으로서 완전한 시각화가 될 수 있다. 시각적 프로그래밍 언어에서 프로그램은 시각적 실행을 포함한 하나의 그림으로써 구성한다. 또한 시각적 프로그래밍은 컴퓨터에게 "무엇을 할 것인가"를 알려주는 역할에 의미를 둔다[5]. 이것은 시각적 언어를 사용하며, 목적에 따라 시각 정보를 다루는 언어, 시각적 상호작용을 지원하는 언어, 시각 프로그래밍 언어로 구분할 수 있다.

2.2 시각화 언어 구문의 명세

시각화 프로그래밍 언어는 문자뿐만 아니라 시각화 요소로부터 형식화된 그림들로 구성된 프로그램 언어이다[6]. 대부분의 연구는 시각화 언어 개발에 초점을 두며, 생성된 시각화 프로그래밍에 대한 인터페이스를 중요시한다. 그 이유는 시각화 구문을 기술하는데 있어 정규화 모델과 명세를 이용하는 툴의 부족을 들 수 있다. 전형적인 프로그래밍 언어에서의 문맥자유 문법(CFG:Context Free Grammar)과 같은 정규화 모델은 구문론 명세 메카니즘을 제공하며, Yacc과 같은 툴로서 파서의 생성을 쉽게 한다.

따라서, 시각화 프로그래밍에 대한 언어를 개발하는 것이 주요 문제가 된다. 이것은 그림 설계 문법(PLG:Picture Layout Grammar)을 기반으로 시각화 언어 구문으로 기술하거나, 프로그래밍 언어로 정의한다. 또한 문법의 새로운 모델을 기반으로 속성을 가진 객체의 집합을 만들어 낸다. 시각화 언어 설계자는 그림 설계 문법을 작성함으로써 시각화 언어를 정의할 수 있다. 또한 시각화 프로그래머는 그래픽 편집기를 사용하여 시각화 프로그래밍을 생성한다.

시각화 프로그래밍 언어를 사용하는 프로그램을 통해 시각화 프로그래밍 환경은 시각화 프로그램의 생성을 제공하며, 구문 명세를 기반으로 하는 시각화 프로그래밍 환경을 구성하게 된다. 시각화 언어의 구문을 기술하기 위한 모델로 그림 설계 문법을 기술하고, 파서가 그림으로부터 시각화 프로그램으로 복구시키기 위해 알고리즘을 필요로 한다. 시각화 언어를 위한 파서를 구현하되 구문은 그림 설계 문법에 의하여 기술되며, 파서 형태는 시각화 프로그래밍 환경을 기본으로 한다. 다음(그림 2)은 파서를 기반으로 한 시각화 프로그래밍 시스템이다.



(그림 2) 시각화 프로그래밍 시스템에서의 공간적 파서
(Fig. 2) A visual programming system based on spatial parsing

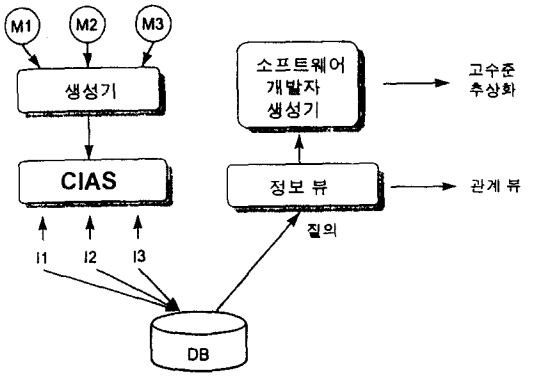
2.3 기존 시각화 툴

프로그램은 다양한 구성 요소와 관계를 수반하여 전반적인 프로그램 구조를 개념화하기가 어렵기 때문에 정보를 추상화함으로 표현되어진다. 기존의 절차적 언어에서 프로그램 구성 요소는 자료와 요소, 자료와 형 그리고 서브 프로그램으로 구체화할 수 있다. 이 구성 요소 사이에서의 관계를 "프로그램 종속상태"라고 한다. 이로 인한 프로그램의 이해와 소프트웨어 유지 보수를 위한 툴들로 CIAS(C Information Abstract System)[7]와 VIFOR(Visual Interactive FORtran)[8], 그리고 CARE(Computer Aided ReEngineering)[9, 10]와 같은 많은 연구가 진행되고 있다.

CARE(Computer Aided ReEngineering)의 경우, 기존의 프로그램은 여러 개의 모듈로 구분되고 최종적인 원시 코드는 CIAS를 통해 추출한 정보를 프로그램 데이터 베이스에 저장해 두고 필요에 따라 객체, 속성, 관계를 추출한다. 또한 소프트웨어 개발자는 이를 더욱 세분화하여 그래픽 관점, 서브 시스템 추출, 사용 불가능한 코드의 제거, 바인딩등을 분석함으로써 결합력을 측정할 수 있다.

CIAS는 원시 코드를 저장하고, 정보를 추출하며, 검색 과정을 일련의 과정으로 각각 분리함으로 융통성을 가질 수 있으며, 명확한 개념적 모델을 도식화함으로 기본적인 객체와 그들간의 관계를 시각화한다. 또한, 소프트웨어 정보 베이스와 문서 데이터 베이스를 분리하고 소프트웨어 모듈의 분류화로 효율적인 시스템을 형성한다. 다음(그림 3)은 CIAS의 구

성도이다.



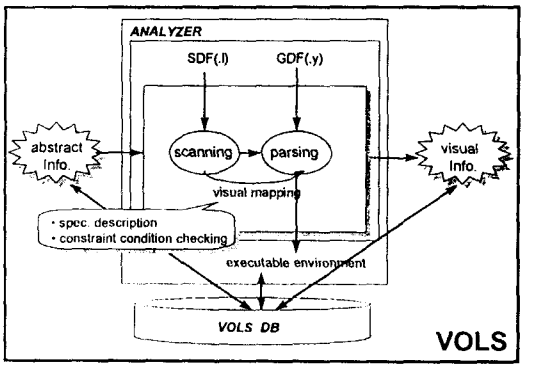
(그림 3) CIAS 구성도
(Fig. 3) CIAS system configuration

3. 시각화 객체 설계 시스템 : VOLS

시각화 정보는 그 자체로 인간에게 쉽게 그리고 다양하게 정보를 전달해 준다. 실세계에 존재하는 추상화 정보를 그림으로 표현하는 것은 인간과 대화하는데 있어 중요한 역할을 한다. 인간은 그림을 통해 그 구성 요소를 쉽게 인식하고, 요소들 사이의 관계를 빨리 인지할 수 있기 때문에 문장을 자세히 조사하고, 이해하는 것보다는 그림의 내용을 인식하는 것이 더 빠르다. 현재 고도의 성능을 가진 컴퓨터 기종들의 기술이 급속히 진보하는 가운데 시각화를 위한 인간과 컴퓨터 상호작용(HCI : Human Computer Interface)에 대한 관심이 높아지고 있다. 사용자 인터페이스는 광범위한 그래픽 형태로 인해 다양한 종류의 정보가 시각화되기를 요구한다[11]. 좋은 그림을 표현하는 것은 단순히 정보 시스템에서 자료를 다루는 것이라고는 하지만 표현하기 위한 목적 때문에 그래픽 표현 프로그램으로 구현하는데는 비용이 많이 든다.

따라서, 본 논문에서는 실세계에 존재하는 다양한 추상화된 정보를 사용자가 원하는 시각적인 설계 표현으로 나타내기 위해 분석기에서 스캐닝(scanning)과 파싱(parsing)을 하므로 의미 분석을 하고 이는 중간 코드 생성과 코드 최적화를 거쳐 사용자가 요구하는 실행 가능한 정보를 얻어 가능한 컴파일러를 통해 실행하므로 입력에 따른 출력의 정보가 나오게 된다.

내부적으로는 각 응용 영역의 문법에 의해 표현된 정보를 정제된 추상화 객체와 관계로 표현하는데, 이는 시각화 매핑을 기반으로 나타낼 수 있다. 매핑 관계를 통해 그래픽 정보로 변형함으로서 원시 정보가 시각 정보로 표현하기 위한 일련의 중간 과정을 제시할 수 있다. 이것은 그림을 설계하기 위한 것으로 매핑을 통한 시각적 구조는 제한 조건을 해결하고, 그 결과 최종적인 시각 정보가 생성된다. 이 최종적인 시각 정보는 시각화 도구인 VOLS(Visual Object Layout System)를 통해 제공된다. 다음(그림 4)은 VOLS의 구성도를 나타낸다.



(그림 4) VOLS 구성도
(Fig. 4) VOLS system configuration

3.1 시스템 구성

하나의 문장을 스캐닝하고 파싱하여 시각화 정보를 추출하되, 문장의 집합은 의미적 관계의 집합으로 번역된다. 또한, 주어진 언어에 대한 파서를 얻기 위해서는 파서 생성기를 사용하여 자연 언어를 분석하거나, 시각 정보로 번역하는 특별한 파서기가 필요하다[12].

따라서, 본 논문에서는 응용 프로그래머와 사용자가 가진 다양한 원시자료를 분석기를 통해 원시 정보와 그 의미론을 분석하여 정제된 정보로 나타낸다. 이때 원시 정보에서 문장의 집합은 의미론 관계의 집합으로 변형한다. 또한, 한 문장은 그래픽 정보 내에서 직접적으로 변형하기보다는 주어진 문법 구조로 접근함으로서 그래픽 구조를 쉽게 다룰 수 있게 된다. 원시 정보에서 직접적으로 그래픽 정보를 변형하는 방법을 고려해 보면, 시각 정보는 주어진 자료에

대한 문맥 자유 문법(CFG:Context Free Grammar)의 생성과 관계가 있는 설계 규칙에 따라 만들어진다. 즉, 원시 정보를 시각정보로 표현하기 위한 SDF (Scanner Description File)와 GDF(Grammar Description File)를 기술하여 구동하면 실행 가능한 환경을 제공하게 되며, 그 중간과정에 매핑 정보가 생성된다.

3.1.1 시각화 매핑(Visual mapping)

추상 정보를 시각 정보로 표현하기 위해서는 중간 과정이 필요한데, 본 논문에서는 의미 분석과 시각화 매핑을 제시하였다. 즉, 분석기를 통해 정제된 정보인 의미 분석 표현이 시각화 언어 추출 표현으로 매핑된다. 그래픽 정보에 대한 자료의 집합은 정보 저장소에 저장되며, 최종적으로 사용자가 원하는 정보와 일치하는 것이 저장소를 통해 생성된다. 이는 추상 정보에서 그래픽 정보를 매핑하는 역할을 한다. 이것은 사용자에 의하여 제어되고 설계된다. 매핑의 기호 명확성을 위하여 개념적 모델을 정의할 수 있는데, 추상화 객체들 사이에서 추상화 관계로서 추상 정보와 그래픽 객체들간에 그래픽적 관계로의 그래픽 정보를 중점적으로 다룬다. 다음 〈표 1〉과 같이 의미 분석과 시각화 언어 추출, 그리고 그들간의 매핑에 대한 내용을 정의하며, 텍스트 위주의 추상 정보를 그래픽 정보로 나타내기 위하여 그들간에는 중간 매개체 표현인 그림 설계 과정을 제시한다. 추상 정보는 응용 프로그램머나 사용자들이 가지고 있으며 프로그램 집합을 통해 표현되어진다. 또한, 관계 구조 표현은 표현식에 의한 것이며, 추상화 객체와 관계에 대한 표현이다. 원문 표현과는 다르게 다양한 정보 표현으로 변형된다. 시각화 구조 표현은 그림에서의 구성 요소 사이에서 관계되는 그래픽적 객체와 관계에 의하여 표현된다. 의미 분석은 사용자 정의 매핑 자료에 의하여 시각화 언어를 추출하도록 매핑된다. 그래픽 표현은 동적인 그림 표현으로 사용자 관점이며, 시각화 언어 추출의 명세서로부터 일반화된다.

이상에서 정의된 것에 따라 시각화 매핑은 의미 분석을 반복적으로 적용하는데, 본 논문에서는 명세기술을 사용자가 이해하기 쉬운 프롤로그로 표현하였다. 일반적으로 의미 분석은 의미적 구조이며, 이것을 기술한 프롤로그는 의미 분석과 시각화 언어 추출을 표현하는데 적당하다. 추상화 객체와 관계는 프롤로그 술어의 항목으로 표현되며, 그래픽 객체와 관계는 특별한 술어로써 정의한다. 추상 정보의 한 문장을 의미적 표현으로 분석하고, 그 의미론을 매핑하여 최종 언어로 표현하며, 이와 일치하는 표현을 일반화시킨다. 시각화 번역 과정 역시 위의 단계를 거친다. 이것을 시각화 매핑이라 하고 의미 분석을 시각화 언어 추출로 매핑 하는 것이다.

〈표 1〉 관계적 구조와 시각화 구조간의 매핑
〈Table 1〉 Mapping between relational structure and visual structure

분 류	정 의
관계적 구조 (AO, AR)쌍	AO = 추상화 초기 객체(AP) ∪ 추상화 객체 AR = 추상화 객체들 간의 관계의 집합
단일 매핑	v:(AP, AR)→(GP, GR)로 매핑 IF α:AP→GP β:AR→GR THEN v((AP, AR)) = (α(AP), β(AR))
다중 매핑	v:(AO, AR)→(GO, GR)로 매핑 IF α':AO→GO (여기서 α'(x):if x∈AP then α(x) elseif x∈AO-AP then v(x) and α:단일 매핑) β:AR→GR THEN v((AO, AR)) = (α'(AO), β(AR))
시각화 구조 (GO, GR)쌍	GO = 추상화 그래픽 초기 객체(GP) ∪ 추상화 그래픽 객체 GR = 추상화 그래픽 객체들 간의 집합

3.1.2 명세 기술(Specification description)

그림 설계 단계는 의미 분석을 통해 시각화 매핑되고, 시각화 언어 추출은 실제 그림으로 변환된다. 이 시각화 언어 추출 표현으로부터 그래픽 설계를 결정하는 문제를 해결하기 위해 시각 설계에 관한 연구가 필요하다.

일반적인 그래픽 시스템은 사용자 또는 응용 프로그래머가 원하는 위치, 변환, 회전, 그리고 크기 등에 그래픽 초기 위치에서 시각화를 위한 그림이 생성된다. 시각화 매핑 표현 과정에서 사용자들은 국소적 또는 구성 성분상의 제한들보다는 오히려 전역적 설계 방식을 중점적으로 한다. 상위 또는 하위 부분에

제한된 시스템의 문제들이 사용자들뿐만 아니라 시각화 시스템에서도 해결 가능하게 한다. 제한 조건은 정확하게 만족되어야 하는 비융통성 부분과 정확하지는 않지만 근접하게 만족되기를 요구하는 융통성 제한 조건들로 분류한다. 융통성 제한 조건은 일반적으로 최소제곱법에 의해 해결된다. 사용자들은 비융통성 제한 조건들에 대해 중요한 설계 규칙들과 융통성 제한 조건들에 대해 중요하지 않은 규칙들에 관련됨으로써 복잡한 설계를 쉽게 다룬다. 또한, 상위 또는 하위 부분에 제한된 시스템의 문제를 처리하기 위하여 시각화 계층을 제안한다.

또한, 이 시각화 계층 메카니즘에 의해 나타낸 계층적 구조는 시각화 언어 추출의 개념적 모델로 표현된다. 시스템은 상향식 방식의 그림 계층을 변환함으로써 계층적으로 제한 조건들을 해결할 수 있다.

〈표 2〉 명세 정보
〈Table 2〉 Specification information

	내 용
그래픽 객체	BOX(term,width,height,mode) DIAMOND(term,width,height,mode) CIRCLE(term,width,height,mode) ELLIPSE(term,width,height,mode) POINT(term)
그래픽 관계	X(Y)_RELATION(t1,t2,ref1,ref2,gap,mode) X(Y)_ORDER([t1,t2,...],x(y)gap,mode) X(Y)_AVERAGE(t1,[t2,...],mode) HORIZONTAL([t1,t2,...],mode) VERTICAL([t1,t2,...],mode) CONTAIN(t1,t2,gap,mode) HIDE(t1,t2)
저수준 & 고수준 관계	LABEL(term,label,mode) CONNECT(t1,t2,from,to,mode) ARROW(t1,t2,from,to,mode) CONNECTWITHLABEL(t1,t2,from,to,mode,label) ARROWWITHLABEL(t1,t2,from,to,mode,label). HORIZONTALLISTING([t1,...),gap,mode) VERTICALLISTING([t1,...],gap,mode). ABOVE([t1,...],[t2,...],gap,mode) BELOW([t1,...],[t2,...],gap,mode) LEFTOF([t1,...],[t2,...],gap,mode) RIGHTOF([t1,...],[t2,...],gap,mode) BETWEEN([t1,...],[t2,...],gap,mode) MULTI_CONNECT([t1,...],[t2,...],from,to,mode).

시각적 표현을 나타내기 위해 프롤로그로를 이용하며, 사용자의 작업은 특별한 술어를 사용함으로서 그들 사이의 그래픽 객체와 그래픽 관계를 생성하기 위한 것이다. 또한 이에 따른 시각 표현 명세 기술은 그래픽 객체, 그래픽 관계 그리고 저수준관계등으로 구성된다. 다음 〈표 2〉는 각각의 명세에 대한 정보들이다.

3.1.3 제한 조건 검사(Constraint condition cheaking)

매핑 과정을 통해 얻은 정보로 사용자는 명세서를 작성할 수 있는데, 사용자가 필요로 하는 정보로 제공되는 명세에는 필요한 제한 조건이 있다. 현재, 제한 조건을 만족하는 기술로는 지역 전파, 완화법, 그래프 변경, 단순한 등식으로 해결하는 등의 기술이 알려져 있다. 제한 조건을 만족하는 것이 실질적으로 어렵기 때문에 명세서 기술을 필요로 하게 된다[13, 14]. 또한, 그림 계층에 대한 계층적 제한을 해결하는 메카니즘은 그림 계층의 변형에 의한다. 즉, 서브 그림이 제한 조건을 만족한 후 그 넓이와 높이를 계산한다. 서브 그림의 크기는 상위의 서브 그림이 해결된 제한 조건일 때에만 해결된다. 이런 방법에서 서브 그림은 잎에서부터 루트까지 과정을 거친다. 본 장에서는 융통성, 비융통성 제한을 어떻게 사용하는가와 서브 그림 구조에 대해서 설명한다.

우선, 융통성 제한은 제한이 충돌하는 것을 해결하기 위해 융통성 제한으로 표현되어진 융통성 그래픽 관계를 소개한다. 융통성 관계는 비융통성과 혼합하여 사용한다. 융통성 제한의 충돌은 정확히 비융통성 제한을 만족하는 최소자승법에 의하여 만족하며, 그림을 생성하기 위해 아주 유용하다.

(1) 단일 문제 해결

선형조건을 해결하는 것으로, 자료 분석을 위해 사용된다. 다음과 같이 해결한다. $x_1, x_2, \ldots, x_n$은 변수, $f_1, f_2, \ldots, f_1, \ldots, f_m$은 그들간의 선형 등식이다.

$$f_j(x_1, x_2, \ldots, x_n) = a_{j1}x_1 + \cdots + a_{jn}x_n - b_j = 0 (1 \le j \le m) \tag{3.1}$$

이 등식은 일반적으로, $f = Ax - b$로써 행렬 형태로

표현되며, 그 내용을 각각 다음과 같이 정의한다.

$$f=\begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix},\ A=\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \vdots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix},\ x=\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix},\ b=\begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

또한, 자승합의 함수는 다음과 같이 나타낸다.

$$E=f_1^2+f_2^2+\cdots+f_m^2 \tag{3.2}$$

목적은 E가 최소가 되도록 $x_1, x_2, \ldots, x_n$을 계산하는 것이다. 따라서, 최소자승법은 다음 조건을 해결함으로 계산한다.

$$\frac{\partial E}{\partial x_1}=\frac{\partial E}{\partial x_2}=\cdots=\frac{\partial E}{\partial x_n}=0 \tag{3.3}$$

x_i에 의한 부분적인 유도는 다음과 같이 (식 3.2)으로부터 계산된다.

$$\frac{\partial E}{\partial x_i}=\sum_{j=1}^{m} 2 f_i \frac{\partial f_j}{\partial x_i} \tag{3.4}$$

(식 3.3)은 $\Delta E=If=0$와 같이 행렬 형태로써 표현되며, 각각의 내용을 다음과 같이 정의한다.

$$\Delta E=\begin{pmatrix} \frac{\partial E}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial x_n} \end{pmatrix},\ J=\begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_1} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_1} \end{pmatrix},\ f=\begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix}$$

그 결과 $JAx=Jb$이 된다. 결과적으로 식은 다음과 같은 형태를 가진다.

$${}^tAAx={}^tAb \tag{3.5}$$

최소자승법은 (식 3.5)를 해결함으로 얻을 수 있으며, 앞에서 정의한 $AX=b$에 의해 유도되었다.

(2) 혼합 문제 해결

융통성과 비융통성의 두 가지 형태를 가지는 것으로 제한 조건을 만족하는 혼합 문제를 계산하기 위해 최소자승법을 확장한 것이다. $g_1, g_2, \ldots, g_p$는 비융통성 제한 조건이며, $f_1, f_2, \ldots, f_q$는 융통성 제한 조건이라 하자. 비융통성 제한 조건의 등식은 다음과 같이 표현된다.

$$\begin{pmatrix} g_1 \\ \vdots \\ g_p \end{pmatrix}=B\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}-c=0 \tag{3.6}$$

변수 $x_1, x_2, \ldots, x_r$은 다음과 같이 $x_{r+1}, \ldots, x_n$의 선형 조합으로 표현된다.

$$\begin{pmatrix} x_1 \\ \vdots \\ x_r \end{pmatrix}=M\begin{pmatrix} x_{r+1} \\ \vdots \\ x_n \end{pmatrix}-b=0 \tag{3.7}$$

(식 3.7)에서 등식 $f_1, f_2, \ldots, f_q$에서 $x_1, x_2, \ldots, x_r$을 제거된다. 이 과정은 가우스 소거법에 의해 나타낸다. 결과적으로, 융통성 제한 조건의 등식은 다음과 같이 표현된다.

$$\begin{pmatrix} f_1 \\ \vdots \\ f_q \end{pmatrix}=A\begin{pmatrix} x_{r+1} \\ \vdots \\ x_n \end{pmatrix}-b=0 \tag{3.8}$$

이것은 $x_{r+1}, x_2, \ldots, x_n$를 계산한 후 (식 3.7)에 의하여 $x_1, x_2, \ldots, x_r$의 값을 얻을 수 있다. 이것은 확대된 형태로, 융통성 제한 조건의 최소 자승을 만족하는 것과 비융통성 제한 조건을 만족할 수 있다.

(3) 계층 메카니즘 문제 해결

서브 그림이 제한 조건을 해결한 후 넓이와 높이를 계산한다. 서브 그림의 크기는 상위의 서브 그림이 해결된 제한 조건일 때 제공된다. 이런 방법에서 서브 그림은 잎에서부터 루트까지 과정을 거친다. 그래픽 관계의 집합은 단지 서브 그림에서 그래픽 객체의 관계 위치를 기술하며, 두 가지 이상의 제한 조건은 일반적으로 객체의 x와 y좌표를 결정하도록 요구한다. 그런 까닭에 다음과 같은 두 가지 제한이 첨가된다.

$$(o_1.cx+o_2.cx+\cdots+o_n.cx)/n=center\_x$$
$$(o_1.cy+o_2.cy+\cdots+o_n.cy)/n=center\_y$$

o_1, o_2, ..., o_n은 서브 그림에서 객체이며, (*center_x, center_y*)는 화면 영역의 좌표로서 서브 그림의 중앙에서 모든 객체의 평균점에 위치하는 것이다. 우선, 각 제한 조건은 정규 형태로 전환한다. 그림 계층에서 해결되는 제한 조건의 과정 전체는 다음과 같은 형태의 프로그래밍 언어로 요약된다. 즉, 다음(그림 5, 6)은 각각 하향식과 상향식 단계의 메카니즘이다.

```
translate(subpicture:sp)
{
    translate( sp_n ) ;
        .
        .
    translate( sp_2 ) ;
    translate( sp_1 ) ;
}
```

(그림 5) 하향식 계층 메카니즘
(Fig. 5) Top-down hirerarchy mechanism

```
hierarchy mechanism(subpicture:sp)
{
hierarchy( sp_1 ) ;
hierarchy( sp_2 ) ;
    .
hierarchy( sp_n ) ;
if(
  (o_1.cx + o_2.cx + ... + o_n.cx)/n = center_x
  (o_1.cy + o_2.cy + ... + o_n.cy)/n = center_y
      constraints condition)
   then
        max_i o_i.rx - min_i o_i.lx;
        max_i o_i.ty - min_i o_i.by;
   else
        break;
}
```

(그림 6) 상향식 계층 메카니즘
(Fig. 6) Bottom-up hierarchy mechanism

3.2 적용 예

본 예제는 객체지향에서의 여러 패러다임중 상속성과 클래스간의 관계를 시각화하기 위해 입력 문장을 다음(그림 7)과 같이 나타내었다.

```
A is the superclass of B,C,D
B is the superclss of E,F
C,D are the inheritance G
```

(그림 7) 추상 정보
(Fig. 7) Abstract information

나타내고자 하는 표현은 내부적으로 다음(그림 8)과 같은 의미 분석으로 번역된다.

```
superclass_of([A],B,C,D):-
        semantics([be,_,ag([A]|_)]),
        comp([A],relation(of(B,C,D|_),singular)).
superclass_of([B],E,F):-
        semantics([be,_,ag([B]|_)]),
        comp([A],relation(of(E,F|_),singular)).
inheritance_of([C,D],G):-
        semantics([be,_,ag([C,D]|_)]),
        comp([A],relation(of(_|G),singular)).
Group(A,B,E):-
        superclass(A,B), superclass(B,E).
Group(A,B,F):-
        superclass(A,B), superclass(B,F).
Group(A,C,G):-
        superclass(A,B), inheritance(C,G).
Group(A,D,G):-
        superclass(A,B), inheritance(D,G).
```

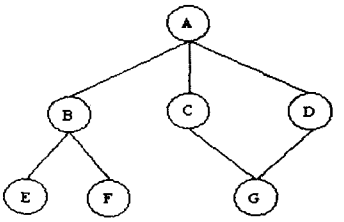
(그림 8) 의미 분석의 결과
(Fig. 8) Result of semantic analysis

또한, 이것은 명세 기술을 통해 시각화 언어 추출을 할 수 있다. 다음(그림 9)와 같이 시각화 매핑을 사용하여 그 관계를 시각화하였다.

```
objectmap:-
    object(X),
    circlewithlabel(X, 40,20,X,[invisible]),
    fail.
relationmap:-
    Group(A,B,C,D,E,F,G),
    point([a]),
    oblique([A],B,C,D,20,[pliable]),
    oblique([B],E,F,20,[pliable]),
    oblique([C],G),([D],G),20,[pliable]),
connect([A],B,C,D),([B],E,F),([C],G),([D],G),20,[thicked,solid]),
    oblique_map([A],B,E),([A],B,F):-
    oblique_relation([A],B,E)):-
            Group(A,B),
            Group(A,E).
    oblique_relation([A],B,F)):-
            Group(A,B),
            Group(A,F).
    oblique_map([A],C,G),([A],D,G):-
    oblique_relation([A],C,G)):-
            Group(A,C),
            Group(A,G).
    oblique_relation([A],D,G)):-
            Group(A,D),
            Group(A,G).
```

(그림 9) 시각화 매핑의 결과
(Fig. 9) Result of visual mapping

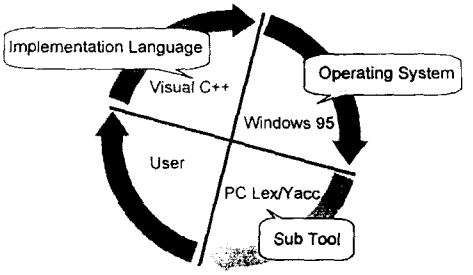
시각화 매핑을 사용하여 그 관계를 시각화한 정보를 가지고 시각화 도구를 통해 다음(그림 10)과 같은 시각 정보를 얻을 수 있다.



(그림 10) 시각 정보
(Fig. 10) Visual information

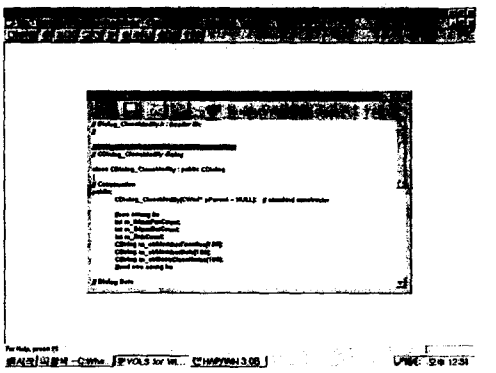
4. 시각화 객체 설계 시스템 구현 : VOLS

이미 이전 장에서는 본 논문에서 제시하고자 하는 최종정보의 시각화 표현에 접근하고자 적용예를 들어 살펴보았다. 시각화 객체 설계 시스템인 VOLS (Visual Object Layout System)는 내부적으로는 의미분석을 가능하게 하는 Abraxas사의 PC LEX/YACC를 사용하여 SDF와 GDF를 작성함으로 스캐닝과 파싱을 거쳐 사용자에게 실행 가능한 정보를 준다. 또한 외부적으로는 Windows 95환경에서 Visual 4.0을 이용하여 사용자 인터페이스 측면에서 내부적인 매핑후 이를 호출하여 표현해주는 역할을 함으로 사용자에게 이해성을 향상시키게끔 하였다. 전체적인 본 논문의 실행환경을 (그림 11)에 제시하였다.



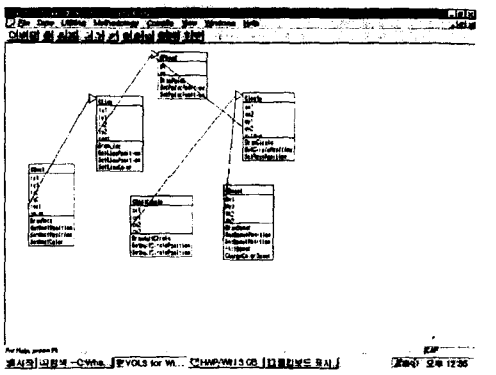
(그림 11) 구현 환경
(Fig. 11) Implementation environment

작업 환경의 구성은 다음(그림 12)와 같이 "File", "Draw", "Utilities", "Methodology", "Compiler", "View", "Windows", "Help"기능이 있다. 또한, 각각의 문서화된 자료는 "Utilities" 메뉴에서 서브메뉴인 "Reserver Header"와 "Reverser Directory"를 클릭함으로 header 혹은 directory단위로 문서를 편집할 수 있다.



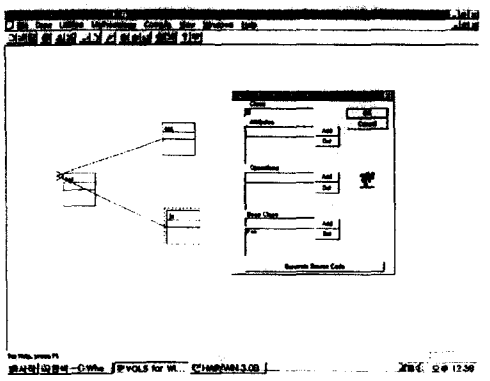
(그림 12) VOLS 실행 예(I)
(Fig. 12) Example of VOLS result(I)

문서화된 자료는 다음(그림 13)와 같이 사용자에게 도움을 주는 시각정보를 표현할 수 있는데 이것은 내부적으로 "Compiler" 메뉴에서 LEX/YACC을 호출하여 실행된다. 본 논문에서는 객체 모델링 방법중 Rumbaugh가 제안한 OMT표기법을 구현하였다.



(그림 13) VOLS 실행 예(II)
(Fig. 13) Example of VOLS result(II)

또한, 다음(그림 14)와 같이 제시된 서브메뉴를 통해 사용자가 직접 시각화 정보를 작성할 수도 있다.



(그림 14) VOLS 실행 예(Ⅲ)
(Fig. 14) Example of VOLS result(Ⅲ)

5. 결론 및 향후 연구 방향

최근 시각 정보에 대한 요구가 증가함에 따라 텍스트 위주의 정보보다는 시각 정보에 초점을 두고 있다. 이것은 사용자와 컴퓨터간의 상호 대화(HCI : Human Computer Interface)가 자연스럽고 쉽게 이루어져야 하며, 어떻게 하면 사용자가 이 프로그램을 쉽게 사용할 수 있겠는가 하는 것이다. 또한, 다양한 시각적인 자원을 이용하여 사용자에게 편이성을 제공해 주고, 그 자원들을 통해 쉽게 수행할 수 있는 사용자를 중심으로 한 인터페이스 부분에 관한 연구가 진행되고 있다. 이런 시각 정보는 초보자에게도 소프트웨어 구조를 쉽게 이해시킬 뿐만 아니라, 시각적 표현은 보다 많은 의미를 함축적인 표현 단위로 전달할 수 있으며, 기억을 도와주고, 많은 흥미를 유발할 수 있다[15].

따라서, 본 논문에서는 실세계에 존재하는 다양한 추상 정보를 시각 정보로 표현하기 위해 기술서를 작성한다. 이 정보는 분석기에서 스캐닝과 파싱을 통해 의미 분석을 하고, 중간 코드 생성과 코드 최적화를 거쳐 사용자가 요구하는 실행 가능한 정보를 얻게 된다. 또한, 실행 가능한 정보에 대한 적절한 컴파일러를 통해 추상정보를 시각정보로 표현하였다.

즉, 의미 분석과 코드생성을 하는 컴파일러와 전체 실행환경이되는 부분을 연결하는 VOLS(Visual Object Layout System)를 구현하였다. 또한, 툴상에서 사용자가 정의해야될 기술서를 문서화함으로 사용자에게 이해성을 향상시키며, 유지 보수 측면에서도 도움을 줄 수 있다.

향후 연구 방향은 사용자가 요구하는 다양한 시각화 정보를 위한 자동화된 툴의 개발과 CASE(Computer-Aided Software Engineering)툴로의 접목, 또한 그래픽적 정보를 문서 정보로의 재공학등이 있다.

참 고 문 헌

[1] Marc E, "Visual Knowledge Engineering", IEEE Transactions on Software Engineering, Vol. 16, No. 10, pp. 1164-1184, 1990.

[2] Jingwen Chang, "Improving the Software Reusability in Object-Oriented Programming", ACM-SIGSOFT SOFTWARE ENGINEERING, 1993.

[3] Wim De Pauw, "Visualizing the Behavior of Object-Oriented Systems", OOPSLA '93, pp. 326-337, 1993.

[4] Daniel Tkach, Object Technology In Application Development, Benjamin/Cunning Pub. Inc., 1994.

[5] 김행곤, 한은주, "추상화 객체의 시각화를 통한 재구성에 관한 연구", 한국정보처리학회 학술발표 논문집, 제22권, 1호, pp. 715-718, 1995.

[6] S. K. Chang, "Visual Language : A Tutorial and Survey", IEEE Software, Vol. 4, No. 1, pp. 29-39, 1987.

[7] Yhi-Farn Chen, "The C Information Abstraction System", IEEE Transactions on Software Engineering, Vol. 16, No. 10, pp. 325-336, 1990.

[8] Vaclar rajlich, "VIFOR : A Tool for Software Maintenance", SOFTWARE-PRACTICE AND EXPERIENCE, Vol. 20, No. 1, pp. 67-77, 1990.

[9] Mullery, "CORE : A Method for Controlled Requirements Expression" Proc. 4th. Int. Conf. Software Eng, pp. 126-135, 1979.

[10] Panagiotis K. Linos, "Visualizing Program Dependencies : An Experimental Study", SOFTWARE-PRACTICE AND EXPERIENCE, Vol.

24, pp. 387-403. 1994.

[11] Shi-Kuo Chang, Principles of visual programming system, Prentice Hall International Editions, 1990.

[12] Tomihisa kamada, Visualization Abstract Objects and Relations, Series in Computer Science, Vol. 5, 1989.

[13] SHI-KUO CHANG, "A Visual Language Compile for Information Retrieval by Visual Reasoning", IEEE Transactions on Software Engineering, Vol. 16, No. 10, pp. 1136-1149, 1990.

[14] Microsoft press, Microsoft Visual C++ 5.0, 1997.

[15] Pedro A. Szekely and Brad A. Myers, "A User interface Toolkit Based on Graphical Objects and Constraints", OOPSLA'88 Proc, pp. 25-30, 1988.

한 은 주

1994년 경북산업대학교 전자계산학과 졸업(공학사)

1996년 대구효성가톨릭대학교 대학원 전산통계학과 전자계산학과 전공(이학석사)

현재 대구효성가톨릭대학교 대학원 전산통계학과 전자계산학과 전공 박사과정 재학

관심분야: 객체지향, 사용자 인터페이스, 소프트웨어 재공학, CASE

김 행 곤

1985년 중앙대학교 전자계산학과 졸업(학사)

1987년 중앙대학교 대학원 전자계산학과 졸업(이학석사)

1991년 중앙대학교 대학원 전자계산학과 졸업(공학박사)

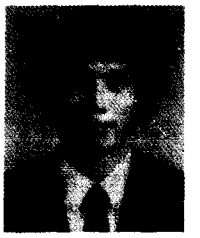
1978년~1979년 미 항공우주국 객원 연구원

1987년~1990년 한국전기통신공사 전임연구원

1988년~1989년 AT&T 객원 연구원

1990년~현재 대구효성가톨릭대학교 컴퓨터공학과 부교수

관심분야: 객체지향시스템 설계, 사용자 인터페이스, 소프트웨어 재공학, 유지보수 자동화툴, CASE



정 연 기

1982년 영남대학교 공과대학 전자공학과 졸업(공학사)

1984년 영남대학교 대학원 전자공학과 전자계산기 전공(공학석사)

1996년 영남대학교 대학원 전자공학과 전자계산기 전공(공학박사)

1985년~1990년 상지전문대학 전산정보처리과 조교수

1990년~현재 경일대학교 공대 전자계산학과 부교수

관심분야: 고속통신망 프로토콜, 멀티미디어 통신, 컴퓨터 구조, 객체지향멀티미디어