論 文

교육용 한글 저작도구 SMAT 설계및 구현

정회원 김 용 성* 정회원 심 재 홍**

Design and Implementation of Educational Korean Authoring Tool SMAT

Yong Sung KIM*, Jae Hong SIM** *Regular Members*

要 約 본 논문은 IBM 16/32 bit PC XT/AT의 MS-DOS 환경하에서 프로그래밍 전문 지식이 없어도 간편한 조작으로 양질의 교육용 소프트웨어를 생성할 수 있는 한글 저작 도구(Korean Authoring Tool)인 SMAT(Simple Manupulation Authoring Tool)을 설계하고 구현하였다.

ABSTRACT This paper is to design and implement the Korean authoring tool SMAT which is able to generate a high quality instruction software with simple manipulation without a special programming knowledge in surrounding IBM 16/32 bit pc XT/AT.

I. 서 론

전통적인 교수방법으로는 학습자가 교사로부터 배운 학습 과제를 반복 연습하거나 심화 보충할 필요가 있는데도 한 사람의 교사가 많은 학생들을 상대로 한 수업이기 때문에 학습자 개인별로 학습 내용의 교정, 심화, 보충 등 학습과정과 인지 과정이 실현되기가 매우 어려운 실정이다.[1X4]

이러한 대안으로써 컴퓨터의 다양한 교육 공학적 잠재력인 처리의 고속성, 정확성, 재생성, 용량성, 변인통제의 다양성 및 에니메이션과 시뮬레이션 등을 이용하면 학습자의 학습 능력과 학습 속도에 알맞는 개인차에 따른 개별화 학습이 가능한 교육적 환경을 조성할 수 있다.[1X9]

교육환경에 컴퓨터를 도입하여 학습내용을 전달하고 평가의 도구로 사용하는 것을 CAI (Computer Assisted Instruction)라고 한다.

CAI는 컴퓨터의 교육적 잠재력이 발견된 이후, 1950년대 Skinner에 의해서 교육전용 컴퓨터인 티칭머신(Teaching Machine)이 출현한 이래

*全北大學校 電算統計學科
Dept. of Computer Science & Statistics
Chonbuk Nat'l Univ.
**光云大學校
Kwang Woon Univ.
論文番號 : 91-7 (接受1990. 11. 10)

보다 효율적이고 과학적으로 교육환경에 적용할 수 있는 여건이 조성되었으며, 근래에 와서 컴퓨터 기술의 발달로 인공지능을 도입한 컴퓨터 시스템과 교육 전문가 시스템 등의 연구가 진행되고 있으며, 광디스크, 오디오, 비디오 등의 다양한 매체를 이용한 교육 시스템도 개발, 보급되고 있다. [5][6]

일반적으로 CAI 시스템은 교육내용과 교육전략에 필요한 코스웨어(Courseware)를 만드는 저작자(Author) 중심의 저작 단계와 이를 이용하는 학습자 중심의 운영단계로 나눌 수 있는데, CAI 시스템의 성능은 저작된 코스웨어의 질에 따라서 생산성이 좌우된다.[3][10]

따라서, 바람직한 CAI 시스템을 구축하기 위해서는 저작과 운영을 효율적으로 하는 방법이 요구되는데, 이러한 기능을 수행하는 도구를 저작도구(Authoring Tool)라고 한다.

저작도구가 일반화되지 않은 초창기에는 고급 프로그래밍언어에 숙련된 전문 프로그래머가 아니고서는 CAI프로그램을 작성할 수 없었으며, 1시간분의 CAI 프로그램을 저작하는데 500 1000시간이 소요되어 개발에 따른 엄청난 경제적 부담과 학습과정에서 코스웨어 설계자에 의해서 이루어져야 하는 많은 의사결정이 프로그래머에 의해서 처리되므로써 양질의 CAI 프로그램을 얻지 못하였다. [3] [4][8][11]

또한, 이러한 고급 프로그래밍 언어는 다양한 폰트의 출력이나 그래픽 처리시 많은 제한점과 문제점을 내포하고 있기 때문에 학습 효과를 높이기 위한 양질의 코스웨어를 저작할 수 없는 단조로운 학습 효과만을 얻을 수 있었다.

따라서, 이러한 고급언어에 의한 코스웨어 저작의 불편을 해소하기 위해서 1960년경에 범용 컴퓨터인 IBM 650에 의해서 Coursewrite의 첫 번째 버전(Version)이 개발됨으로써 대중화 시대를 열게 되었다.

그후, Coursewrite 개발 시기와 거의 같은 때에 개발된 CATO (Compiler for Automatic Teaching Operation) 언어의 장점을 살려 개선된 TUTOR 언어가 PLATO 시스템에서 구현되어 사용되고 있다.

이러한 저작언어들은, 고급언어를 사용한 것보다, 사용의 편의성, 저작의 다양성 등으로 인하여, 1시간분의 CAI 프로그램을 작성하는데 소요되는 시간이 대략 300 400시간 정도로 단축되었으나 다양한 학습 효과를 구사하기 위한 로직의 설계가 용이하지 못하며, 사용되는 명령어가 400개 이상이나 되어 저작 활동에 어려움을 주고 있다는 단점도 있다. [1][9][14][19][16][18]

그후, 1972년에 미니 컴퓨터의 출현과 함께 개발된 TICCIT(Time shared Interactive Computer Controlled Instructional Television)의 APT, WICAT의 WISE 등은 메뉴 운영 방식의 저작 언어로서, 1시간분의 CAI 프로그램을 개발하는데 50-150시간이 소요되고 코스웨어 설계시 학습설계 가이드를 제공하는 기능을 구비함으로써, 학습 전략 선택시 시간과 노력이 경감되었으나 일반 다목적으로 사용할 수 없는, 시스템에 종속된 명시적 저작 언어라는 것이다. [16][18][19]

이러한 메뉴 중심 방식은 생산성에는 많은 효과를 가져왔으나, 정형화된 학습모델에 맞추어 메뉴를 발생시키므로 코스웨어 저작시 저자의 의도가 제한되고, 많은 메뉴로 인하여 사용상의 난이도가 증가될 수 있는 문제가 발생된다.

그러나, 이러한 저작 도구들은 국내에서는 아직 개발되어 있지 않고, 현재 사용 가능한 저작 도구들도 모두 외국에서 개발된 마이크로 컴퓨터 이상에서 활용 가능한 것으로 한글 모듈을 접속해야 사용가능한 시스템 종속적인 저작도구들 뿐이고 이들 또한 국내 교육환경에 알맞는 이론이나 방법을 연구 개발한 것이 아니고 저작한 나라의 교육환경에 알맞게 만들었기 때문에 국내에서는 큰 실효성을 거두지 못하였다.

또한 현재 국내에서 개발활용되고 있는 CAI 프로그램은 범용 프로그래밍 언어로 개발되고 있어 인력의 제한성, 시간적, 경제적 측면에서 많은 문제점을 내포하고 있다[3][4][5].

이러한 문제점을 해결하기 위한 측면과 90년대 이후 컴퓨터의 보급에 따른 활용방안 측면에서 CAI 프로그램 작성을 누구나 참여할 수 있는

교육적 환경 조성이 그 어느 때 보다도 매우 절실하게 필요성이 강조되고 있다.

따라서, 본 연구는 기존 저작도구들의 구조와 기능을 분석,검토하여 우리나라 교육 환경에 알맞는 Prototype을 설계하여, 컴퓨터의 깊은 지식이 없는 저자들도 쉬운 키보드 조작으로 CAI 프로그램을 생성할 수 있는 프레임 중심이고 메뉴 중심인 한글 저작도구 SMAT을 구현하여 활용할 수 있는 교육 환경을 제공하고자 한다.

Ⅱ. 저작 도구의 역사적 배경 및 특징

2.1 배경

코스웨어의 대표적인 유형으로는, 훈련과 학습(Drill and Practice), 모의실험(Simulation), 개인교수(Tutorial), 게임(Game), 문제해결(Program Solving)형 등이 있다.

또한, 코스웨어에 필요한 자원들을 준비하는 사람을 코스웨어 저자(Author) 또는 저자라고 한다.

CAI 프로그램을 개발하는데 필요한 Tool로는 범용 프로그래밍 언어와 저작언어로 구분해 볼 수 있다. 전자는, 컴퓨터를 다양하고 폭 넓게 응용하는데 사용하는 반면에, 후자는 컴퓨터를 기초로 교육자료를 준비하는 사람을 위해 특별히 설계된 언어이다.

어떠한 유형의 언어라도 CAI 프로그램을 개발하는데 사용할 수 있지만, 공통적으로 코스웨어 모듈의 수행, 검사, 생성 등을 할 수 있어야 하고, 또한 저자의 교육목적을 반영하고 학습자에 의해 성취해야 할 학습 목표를 달성할 수 있어야 한다.

CAI 프로그램의 상당 부분이 10개 정도의 범용 프로그래밍 언어로 주로 사용되고 있으며, 특히 많이 사용되고 있는 언어로는 APL, BASIC, FORTRAN, COBOL 등이다. [11]

그후, 각종 소프트웨어의 발달로 인하여, 개인용 컴퓨터나 마이크로 컴퓨터의 출현으로 EBASIC, LOGO, SMALLTALK 등으로 교육용 CAI 프로그램이 쉽게 개발될 수 있는 요건이 조성되었으나 기대한 만큼 큰 효과는 없었다.

기존 프로그래밍 언어의 장점과 특징을 살려서 최초로 개발된 저작언어가 1960년경에 IBM650에서 개발한 TIP(Translator for Interactive Program)이다. 그후 TIP을 개선하여 IBM650에서 Course Writer의 첫 번째 버전(Version)이 개발됨으로써 대중화 시대가 열리기 시작하였다. [15][16][17]

Course Writer 개발 시점에서 CATO 언어로 PLATO을 구현하여 교육적 로직의 일부분이 자료의 특수 함수키들에 의해서 정의되어 활용하였다.

그후, CATO의 장점을 살려 프레임 중심으로 개선된 TUTOR 언어가 PLATO 시스템에서 구현되어 현재까지 사용되고 있다.

또한, 마이크로 컴퓨터의 출현으로 TICCIT의 APT, WICAT의 WISE 등의 저작 언어가 개발되었으나 이들 모두는 일반 다목적으로 사용할 수 없는 명시적 저작언어이다.

따라서, 이러한 문제점을 해결하기 위해서 시스템에 독립적으로 사용할 수 있는 저작언어의 연구가 진행되어 PLANIT (Programed Language for Interactive Teaching)이 개발되어 Q-32 컴퓨터에, 최초로 구현된 후 IBM360, UNIVAC1108, CDC6400, PDP11, VAX 등에서 이용되었다. [16][17][18][19]

2.2 환경

전통적인 교육 방법과 CAI 환경에 관해서 알아보면 그림 2.1과 같다. [14]

전통적인 교수법은 그림 2.1에서 제시된 바와 같이, 학습자와 교사간에만 양방향의 직접 채널이 구성되어 학습자간의 대화로써 수업을 진행하게 된다.

교사와 학습자간의 상호 대화 정도의 관계는, 직접 피드백 채널을 통해서 정보를 전달하는 데이타 흐름 함수를 K6, 교육 채널을 전달하는 함수를 K2로 표현하면, I(T:S)=K6/K2로 나타

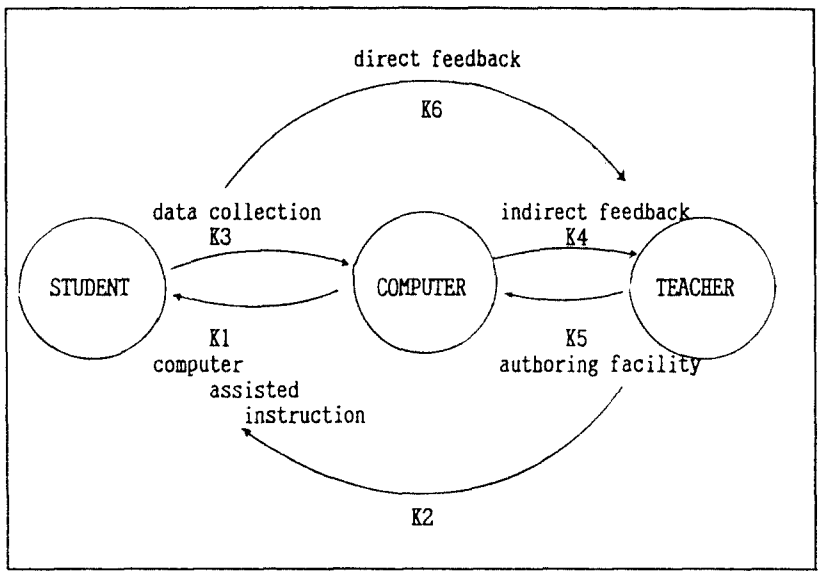


그림 2.1 전통적 교수법과 CAI 환경

낼 수 있다.

함수 K6, K2는 정보의 양, 유형, 가치, 방향, 방식 및 시간 등의 매개변수에 의해서 결정된다.

컴퓨터를 이용한 교수법인 학습자, 컴퓨터, 교사와의 상호 대화 정도의 관계는 I(S)=I(C:S) +I(T:S)=K3/K1+K6/K2로 나타낼 수 있으므로 교사에 의한 직접 채널 뿐만 아니라 컴퓨터와 학습자와의 채널도 구성되어 있다.

여기서, 저작 설비는 저작 언어와 저작 시스템을 의미하며, 학습자의 상호 대화의 정도는 학습자와 컴퓨터, 학습자와 교사간의 상호 관계를 합한 것과 같기 때문에 교육의 생산성을 높일 수 있다. 또한, 교사나 교과 개발자가 저작설비를 이용하여 CAI 프로그램을 작성하기 때문에 프로그래밍에 관한 전문적 지식이 없이도 손쉽게 CAI 프로그램을 작성할 수 있다. 즉, 저작 설비는 교사나 교과 개발자가 가르칠 내용과 교육적 논리와 전략 등을 명시하면, 명시된 내용에 따라 순차적으로, 교육용 프로그램이 작성되는 프로그램 생성기 (Program Generator)와 같기 때문에 컴퓨터 전문 지식이 없는 저자나 교과 개발자도 쉽게 CAI 프로그램을 작성할 수 있는 환경을 제공해 준다. 저작 언어의 형태에 따라 생산성이 크게 좌우되는 이러한 저작 도구는, 초기의 절차적 (Procedural) 프로그래밍 언어에서 출발하였으나 현재는 비절차적 (Nonprocedural)이며 자연어 처리도 가능하게 되었다. 프로그래밍 언어의 관점에서 저작 도구의 위치는 그림 2.2에 나타나 있다.[12]

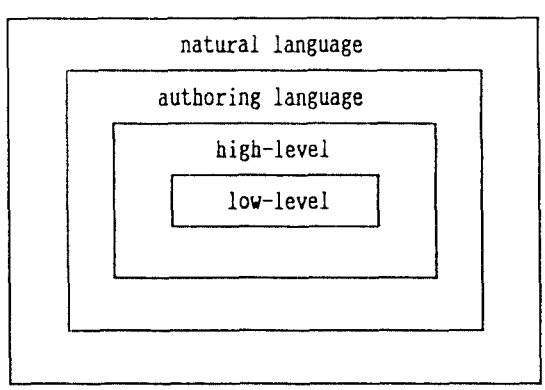


그림 2.2 Authoring Language의 위치

2.3 저작 언어의 학습 모형

저작 언어의 개발 모형은 학습자의 반응에 독립적인 선형 모형, 학습자의 반응에 적응하는 분기 모형, 훈련과 연습과 특징을 갖는 문제 해결 모형, 자연어 처리 등의 인공지능에 기초를 둔 대화식 모형으로 나누어 볼 수 있다.

선형 모형은 Skinner가 제시한 것으로 두가지 중요한 요소는 피드백(Feedback)과 개인차이다. 이 모형은 정확한 답이 있을 때만 피드백(Feedback)이 고려되기 때문에 학습자에 따라 적합한 속도로 학습 자료를 학습할 수 있다는 점이다.[6][7][10][15]

분기식 모형에서는 선형 모형에서 발생될 수 있는 변화를 학생의 반응에 따라 학습을 달리하는 방식이다.

즉, 학생의 응답에 맞는 경우는 다음 과제로 제어가 옮겨지며, 틀린 경우는 적당한 설명 또는 암시를 주어 응답을 옳도록 유도하며, 응답의 회수에 따라 여러가지 교육 방법론적으로 분기가 실행된 후, 문제를 해결토록 하는 방식이다. 생성 학습 모형은 교수 자료를 준비하는데 저작자의 일을 용이하게 하기 위한 실질적인 혹은 충족시키고자 학생에 맞는 문제와 답을 생성할 수 있어 효과적이다.

또한, 문제의 해답, 진단 등의 자료를 생성할 수 있는 프로그램을 보관하고 있으며 학습중 자동적으로 이를 처리한다.

이 방법의 장점은 교수 자료의 작성을 부분적으로 공급하므로 데이타 양역을 감소시킬 수 있으며, 학생이 어떠한 성취도에 도달하는데 필요한 많은 문제를 제공해 주고 난이도를 조정할 수도 있다.

단점으로는 매개 변수화할 수 없는 경우 문제처리가 어렵기 때문에 표준문제 처리로만 끝내게 된다.

이 방식은 Drill and Practice의 문제 학습 모형에 주로 사용된다.[15][17] 문제 해결 모형은 학습 자체가 개인화 되고 강한 학습 동기를 유발하여 학생 스스로 오류를 검색 및 수정할 능력을 배양하고 학습을 즐길 수 있다는 장점이 있는 반면에 다음과 같은 조건들이 충족되어야만 한다.

1) 사용된 프로그래밍 언어는 고급 언어로 충분한 심계가 되어 있어야만 되고,

2) 컴퓨터 환경은 사용자 위주로 구성되어야만 하고,

3) 학습자가 손쉽게 작성하고 흥미를 가질 수 있도록 각종장치가 마련되어야 한다.

외와 같은 모형은 컴퓨터가 모든 질문을 묻고 학습자가 항상 가능한 답의 집합으로부터 하나의 답을 선택하는 제한된 대화의 기준을 두고 있으나, 실제 교육에서 필요한 것은 일정한 규칙에 맞는 대화뿐만 아니라 학습 자료를 가지고 있는 내용의 추론적 기능과 학습자의 성취도에 따른 시스템 제어 기능을 추가함으로써 완전한 학습 모형이 될 것이다.

2.5. 저작 도구의 모델별 특성 분석

저작 언어나 저작 시스템은 명령문의 구성방법과 CAI 프로그램의 구조, 소프트웨어의 생산 방법 등에 따라서 크게 4가지 유형으로 나누어 볼 수 있는데 이는 CAI 저작을 어떠한 형식으로 작성하는가에 따라서 구분된다.[5][6]

첫째, 명령문 중심(Statement Oriented)형은 기존의 프로그래밍 언어를 이용하여 작성하는 방식으로, 기존 프로그래밍 언어로 CAI 프로그램을 작성하는 것보다는 생산적이나 프로그래밍 언어를 아는 사람만이 사용이 가능하며 대표적인 저작 언어로는 PILOT의 BASIC, STAF의 FORTRAN 등이다.

둘째, FORM Driven 형은 사전에 일정한 교육 절차에 따른 프레임을 구성하여 CAI 프로그램 작성시 FORM의 범주내에서 활용할 수 있도록 하는 방식으로 저자에게 유용성과 응용의 한계성을 갖게하는 것으로 TICCIT가 대표적이다.

세째, MENU 중심형은 다양한 메뉴를 계층화 시키고, 저자로 하여금 메뉴를 선택해 가면서 CAI 프로그램을 작성하는 방식으로, 복잡한 교육자료를 CAI 프로그램 생성을 감소화하는 장점이 있으나 많은 메뉴로 인하여 저작상 혼란

을 주는경향이 있는 것으로 WICAT 시스템의 WISE가 대표적이다.

네째, 프레임 중심형은 프레임 단위로 저작을 하기 때문에 소스웨어 작성이 용이하나 스크립 유형과 통신 연결이 표준화가 되지 않으며 작성상에 어려움이 있다. 대표적인 언어는 MICROTEXT 이다.

따라서, CAI 프로그램을 보다 쉽게 개발하기 위해 저작도구 설계상 고려할 점을 살펴보면 다음과 같다.

첫째, 저자 관점에서는 사용하기 쉬워야 하고, 손쉽게 저작할 수 있는 각종설비(에디팅, 스크립, 데이타, 폰트, 한글)가 갖추어져야 하며, 교육전략및 학습방법이 다양해야 한다.

둘째, 학생관점에서는 개인차에 따른 학습이 가능해야 하며, 흥미와 동기 유발, 이해력 증진 등의 교육과정 및 교육절차가 필요하다.

세째, 저자의 생산성을 증진시키고 교육성과를 향상시켜야 한다.

네째, 저작 Tool을 사용하는 사람이 알고 있는 언어를 고려해야 한다.

다섯째, 저작 Tool을 사용하는 사람의 교육환경을 고려해야 한다.

여섯째, 다른 기종에서도 쉽게 활용할 수 있도록 해야 한다.

일곱째, 새로운 매체나 새로운 교육전략에 맞추어 확장이 가능해야 한다.

여덟째, 다른 다양한 매체(audio, video, slide 등) 및 다른 terminal과 상호 통신이 가능해야 한다.

Ⅲ. 한글 저작도구 SMAT 설계

본 장에서는 2장에서 분석한 저작 도구의 특성을 고려하여 컴퓨터 전문 지식이 없는 저자들이 손쉬운 키보드 조작으로 저작할 수 있는 SMAT 를 정의하기 위해서 BNF(Backus Naur Form)을 이용하였다.

3.1 SMAT의 구성요소와 기능

제2장에서 이미 고찰한 저작도구의 기능과 특성을 고려하여 설계한 저작도구의 prototype의 구성요소와 기능을 나타내면 그림 3.1과 같으

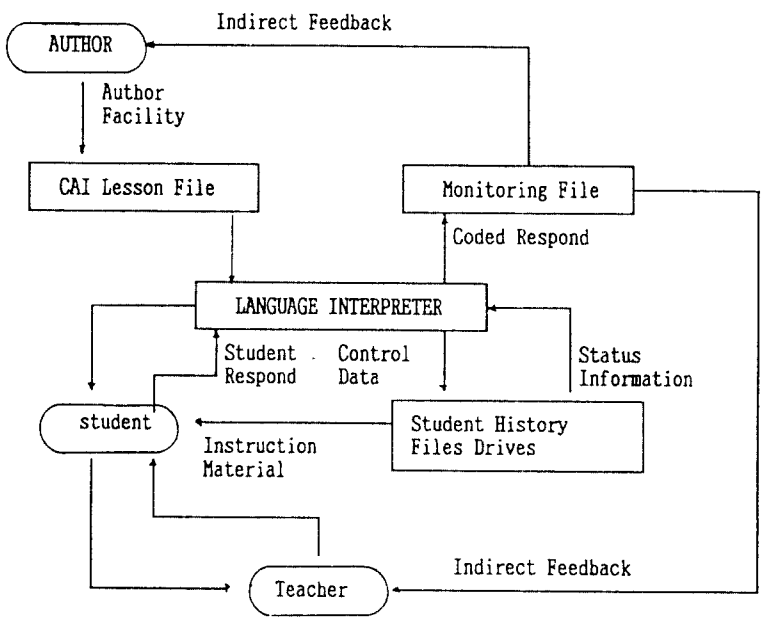


그림 3.1 저작 언어의 성분과 구세

며, 이는 MICROTEXT의 frame 중심, COA와 WISE의 Menu 중심 등의 장점들을 도출하여 한국 교육환경에 맞게 3계층 menu와 frame 별로 저작운영 할 수 있도록 하였다.

CAI Lesson File은 저작된 코스웨어를 화면단위인 Frame 단위로 Disk에 Linked list 형태로 기억하고, 저작실비는 Command에 없는 그래픽 명령을 확장하여 저자가 보다 쉽게 그래픽을 할 수 있도록 환경을 조성한 그래픽 에디터, 문자나 특수 문자 등을 생성하는 Font Editor의 기능과 음향효과를 활용하기 위한 음향, 멜로디 생성 및 활용할 수 있는 Sound Editor 로 구성한다.

또한, 학습 진행 과정을 제어 및 관리를 용이하게 하는 로직 에디터와 영상을 보다 동적으로 표현할 수 있도록 하는 Animation Editor로 구성되어 있다. 모니터링 파일은 저작과정을 도와주는 메뉴 생성기, 레슨 파일을 관리하는 레슨 파일 핸들러, 학습 진행의 조건, 분기를 하는 로직 핸들러, 자료의 입.출력을 관리하는 파일 핸들러로 구성된다.

CAI 인터프리터는 저자에 의해서 작성된 CAI Lesson File을 명령어 단위로 번역하여 프레임 단위로 실행하며 학습자로부터의 응답에 대한 처리 실차 등도 처리하여 준다.

학습자 파일은 학습자의 학습 성적, 개인차, 능력등의 정보를 보관하여 학습 진행을 제어하기 위한 자료를 기억하여 필요할 때 제공해 준다.

3.2 SMAT 언어의 정의

3.2.1 어휘적 요소

(1) 문자 집합과 프로그램 작성 양식

SMAT는 Ada와 같이 ASCII 128자를 문자 집합으로 하고 코드의 증가순으로 대소 서열을 정하되 문자형은 한문자의 문자열로 간주한다.

SMAT에서의 문자 집합은 다음과 같다.

· 영문자 — 대, 소문자 52자.
· 숫 자 — 0 9
· 특수부자 —27자 (space 포함)

=〈 〉, : ; ?( ) # { } [ ] ₩ * + — / ! $ " '

· 한글문자 —자수문자, 특수생성문자등
· 한글 —

한글에 관한 내역은 3.3에서 상세히 논하며, SMAT의 프로그램 작성양식은 메뉴 형식에 따른 명령어 선택으로 자료부만 지정된 입력 장소에서만 키인하도록 한다.

(2) 명칭

SMAT는 예약어 주의를 채택하였으며 예약어는 각종 메뉴명, 명령 등으로 제한되며 작성규칙은 다음과 같다.

〈명칭〉:=〈첫 문자〉〈둘째 문자 이후〉
〈첫 문자〉:=〈한글 문자〉 | 〈영문자〉
〈둘째 이후 문자〉:=〈첫 문자〉 〈10진 숫자〉 〈_〉

명칭의 길이의 유효범위는 15자로 한다.

3.2.2 프로그램 요소

(1) 변수와 영역 규칙

SMAT에서 변수의 작성규칙은 한글을 사용할 수 있다는 점외에는 기존 프로그래밍 언어와 유사하며, 식별 가능한 길이는 15자로 제한한다.

변수를 명명할 때 변수명 만으로 정의하면 전역 변수 (global variable)로 프로그램 전역에서 참조 가능하며 생존 기간은 레슨 화일 단위의 실행 기간이다.

또한, 프레임 변수는 해당 프레임만 참조 가능한 것으로 변수명 + $로 정의하며, 매개변수는 사용하지 않고 전역 변수로 대치하여 활용하며 명명 규칙은 정적영역 규칙(static scope rule)을 채택한다.

(2) 식과 연산자

SMAT에서 수식은 문자열로 표현하는 것과 일반 프로그래밍 언어에서 사용하는 수식의 표현을 사용할 수 있기 때문에, 우선 순위와 결합성은 후자인 경우만 적용한다. 즉, 교육용에서 사용하는 수식의 대부분은 주어진 수식의 계산절차가 교안에 포함되었기 때문에 우선 순위와 결합성을 고려하지 않는다.

그러나, 학습모형의 게임이나 시뮬레이션 유형인 경우는 우선 순위와 결합성을 부여하는데 이는 사용하는 명령어에 따라서 구별하여 활용한다.

(3) 문

SMAT에서 문은 Text, Graphic, Animation, Sound, Logic 등으로 크게 분류할 수 있으며, 문의 선택은 메뉴로 지정하며, 오퍼랜드만 입력하므로 한개의 문이 종료되고, 문과 문사이는 콤마(,) 이고, 프레임의 분리자는 return 키로 대신한다.

3.2.3 프로그램의 구조

(1) 프레임

SMAT의 프로그램 단위는 단일 화면(Screen)인 프레임(Frame)이며 이는 C언어의 수나 Beta 언어의 패턴과 유사하며 그 구조는 다음과 같다.

〈프레임〉:=〈프레임-번호〉〈명령어들〉〈리턴키〉
〈명령어들〉:=〈명령어〉+
〈명령어〉:=〈명령코드〉〈자료부〉
〈명령코드〉:=〈커서 이동〉
〈커서 이동〉:=〈⬅〉 | | 〈➡〉
〈자료부〉:=〈문자열〉

(2) 프로그램의 구성

SMAT의 프로그램의 구조는 화일 내역과 프레임들로 구성되며 그 구조는 다음과 같다.

〈프로그램〉:=〈화일.내역〉〈프레임〉+
〈화일내역〉:=〈화일명〉〈프레임수〉〈드라이브명〉〈작성일〉〈작성자〉〈레슨링크〉
〈레슨링크〉:=〈프레임 번호〉
〈프레임번호〉:=〈숫자〉

여기서 레슨 링크는 후속 연계 단원의 링크로, 선수 학습인 경우는 본 학습 링크, 본 학습인 경우는 후속 단원 학습의 레슨 화일에 링크 포인터가 기억되어 있다.

또한 SMAT는 병렬처리는 허용하지 않기 때문에 한 실행 순서만을 유치하며 분기는 프레임 단위로 제어가 이전된다.

(3) 문

i) 구문 구조와 문의 종류

SMAT는 D구조를 구현하여 복합문(Compound statement)이나 블럭(Block)은 없으며, 문의 형태는 다음과 같으며, 문의 선택은 메뉴로 하며, 해당 명령에 대한 필요한 자료만 입력하면 된다.

〈문〉:=〈레이블〉〈명령코드〉〈자료부〉
〈레이블〉:=〈프레임 번호〉 | 〈로직 레이블〉
〈로직레이블〉:=〈|〉〈명칭〉〈|〉

SMAT에는 텍스트문, 에니메이션, 그래픽, 로직, 사운드 등이 있다.

ii) 텍스트문

텍스트문은 화면에 문자나 폰트를 입.출력하는 명령어들로 종류와 구조는 다음과 같다.

〈텍스트〉:=〈TAB〉 | 〈WRITE〉 | 〈FONT〉
〈TAB〉:=〈한글 간격〉〈영문 간격〉〈행 간격〉.
〈WRITE〉:=〈X시작점〉〈Y시작점〉〈행바꿈위치〉〈문자 크기〉〈출력문자〉
〈FONT〉:=〈폰트크기〉 | 〈폰트 생성〉

여기서 TAB는 입.출력 위치지정, WRITE는 출력 형식 지정, FONT는 폰트 문자 선택과 생성하는 명령을 의미한다.

(4) 그래픽

SAMT의 그래픽 명령어는 점, 선, 원, 박스 등을 작성할 때 사용하는 명령으로 기존 프로그래밍 언어의 문법과 유사한 종류로 구조일부를 나타내면 다음과 같다.

〈그래픽〉:=〈LINE〉 |〈-LINE〉 |〈DLINE〉 |
〈-DLINE〉|〈BOX〉 |〈BOXFILL〉
|〈BOXTILE〉 |〈CIRCLE〉|〈CIRCL
EFILL〉 |〈CILCULAR〉 | 〈PAINT〉
|〈VERTICAL〉 |〈HORIZON〉 |
〈SPRITE〉 | 〈-SPRITE〉 |〈SHAPE〉
|〈SHAPER〉

〈LINE〉:=〈출발X좌표〉〈출발Y좌표〉〈도착X좌표〉〈도착Y좌표〉〈선분색〉
〈BOXTILE〉:=〈상한대각선X좌표〉〈상한대각선Y좌표〉〈하한대각선X좌표〉

〈하한대각선Y좌표〉〈박스 윤곽 선색〉〈칠모양〉

(5) 에니메이션

SMAT의 에니메이션 명령어는 그림의 복사, 이동, 생성 등을 처리하는 명령으로 종류와 구조는 다음과 같다.

〈에니메이션〉:=〈MOVE〉|〈COPY〉|〈ANIMATOR〉

〈MOVE〉:=〈상한X좌표〉〈상한Y좌표〉〈하한X좌표〉〈하한Y좌표〉〈도착X좌표〉〈도착Y좌표〉〈이동 방향〉〈이동 간격〉

〈COPY〉:=〈상한X좌표〉〈상한Y좌표〉〈하한X좌표〉〈하한Y좌표〉〈복사X좌표〉〈복사Y좌표〉〈복사모양〉

〈복사모양〉:=〈P〉|〈XOR〉〈OR〉|〈AND〉|〈R〉

〈P〉:=〈원래모양〉

〈R〉:=〈원래반전〉

〈Animator〉:=〈패턴보기〉|〈패턴만들기〉

(6) 로직

로직은 프로그램의 실행 순서를 제어하는 명령어로 형식과 종류는 다음과 같다.

〈로직〉:=〈JUMP〉〈DELAY〉|〈ANSWER〉|〈CLS〉〈SUBROUTINE〉〈KEY CONTROL〉|〈KEYIN〉|〈SPEED〉|〈ERROR〉

〈JUMP〉:=〈프레임번호〉|〈레이블〉

〈DELAY〉:=〈지연 시간 초〉

〈ANSWER〉:=〈입력키〉〈정답 분기레이블〉〈오답시 분기레이블〉〈정답 판정기준〉〈정답수〉〈정답+〉

〈입력키〉:=〈모든문자〉|〈영문자 및 숫자〉|〈숫자〉

〈정답 판정기준〉:=〈숫자 비교〉|〈문자 비교〉

〈CLS〉:=〈화면 지움 형태〉

〈화면 지움형태〉:=〈B〉|〈W〉

〈SUBROUTINE〉:=〈시작 프레임 번호〉〈종료 프레임 번호〉

〈keycontrol〉:=〈멈춤키〉

〈멈춤키〉:=〈ASCII 코드〉

(7) 사운드

사운드는 음악, 멜로디 등을 저장 및 생성하는 명령으로 종류는 다음과 같다.

〈SOUND〉:=〈Sounder〉〈Sound Name〉

〈Sound Name〉E〉:=〈제목〉〈음향-멜로디〉

〈제목〉:=〈Code〉

〈Code〉:=〈문자인〉〈숫자〉

〈음향-멜로디〉:=〈문자열〉

〈Sounder〉:=〈Sound Maker〉|〈Copy-Melody〉

3.3 Monitoring file

Monitioring file은 lesson file을 쉽고 편리하게 작성할 수 있도록 저자에게 저작 환경을 제공해 주는 것으로 저작운영 메뉴 구조는 계층적 트리 구조를 갖고 있다.

그 구조를 나타내 보면 다음과 같다.

〈저작〉:=〈편집〉|〈로직〉|〈실행〉|〈화일〉|〈에니터〉|〈종료〉

〈편집〉:=〈입력〉|〈검색〉〈삽입〉|〈삭제〉|〈메뉴〉

〈로직〉:=〈내용〉|〈순서〉〈모델〉|〈저작〉|〈메뉴〉

〈실행〉:=〈화면〉|〈단원〉|〈연속〉|〈메뉴〉

〈화일〉:=〈읽기〉|〈저장〉|〈삭제〉|〈이름변경〉|〈목록〉|〈메뉴〉

〈에니터〉:=〈문자〉|〈그림〉|〈움직임〉|〈음향〉〈교수〉|〈메뉴〉

저작 메뉴는 프로그램 편집, 로직내용 검사, 프로그램 실행, 화일내역 변경, 및 그림, 문자 등을 제어할 수 있도록 메뉴를 제공해 준다.

편집 메뉴에서 입력은 프로그램 명령을 선택하여 프레임 단위로 해당 명령어에 대한 자료를 입력할 수 있도록 메뉴를 제공해 주고, 검색, 삽입, 삭제는 프레임 별로 작업을 제어할 수 있도록 한다.

로직은 현재까지 작성된 프로그램의 내역을 화면단위로 나타내 주는 것이 〈내용〉이고, 〈순서〉는 프레임 단위로 로직단위로 화면에 나타내고, 〈모델〉은 교수 및 학습모형을 제시하는 것을

의미한다.

〈실행〉은 화면 단위, 단위별, 전체 화일 전체 등 선택하여 실행할 수 있도록 메뉴가 나타나며, 〈화일〉은 화일별로 읽기, 저장, 삭제, 화일명 변경, 화일 디렉토리를 볼 수 있도록 제공된 메뉴가 제공된다.

〈에디터〉의 기능중 〈문자〉는 폰트선택 및 폰트 작성, 〈그림〉은 그림 데이타 선택 및 그림 데이타 작성, 〈움직임〉은 그림 데이타 이동, 〈음향〉은 음악 멜로디 선택 및 작성, 〈교수〉는 학습전략과 로직 등의 저작상 필요한 작업을 메뉴로 선택되며 같은 계층에 메뉴 선택은 커서키 →, ←로 다음 레벨의 메뉴는 리턴키로 결정된다.

Ⅳ. 한글 저작도구(SMAT) 구현

4.1 화면 구조(Screen Organization)

화면은 저자가 자료나 도형등 입력시 입력위치나 출력위치중 알아 볼 수 있는 입력 가상위치 화면, 저자가 저작할 저자 견지에서의 화면, 학습자가 학습할 화면으로 구성되며 각 화면과 각 구성요소 간에 관계를 나타내면 그림 4.1과 같다.

즉, 입력 가상위치 화면은 자료나 도형 등 입력시 수시로 화면의 입력위치를 알아볼 수 있도록 한 화면이고, 학습자 견지에서 화면은 작성된 레슨 화일의 학습내용을 한 화면당 출력되어 저자의 저작과정을 용이하게 하여 준다.

또한 저자 견지에서 화면 내의(1)은 메인 윈도우로 제1레벨과 제2레벨의 메뉴가 ICON식으로 선택할 수 있도록 되어있고, (2)는 HVW(Help View Window)로 만들어진 폰트나 도형 등을 선택및 참고하기 위한 곳이고, (3)은 명령문 형식과 내역등을 나타내어 저자로 하여금 필요한 명령을 커서키로 선택하여 명령어 형식을 참조하

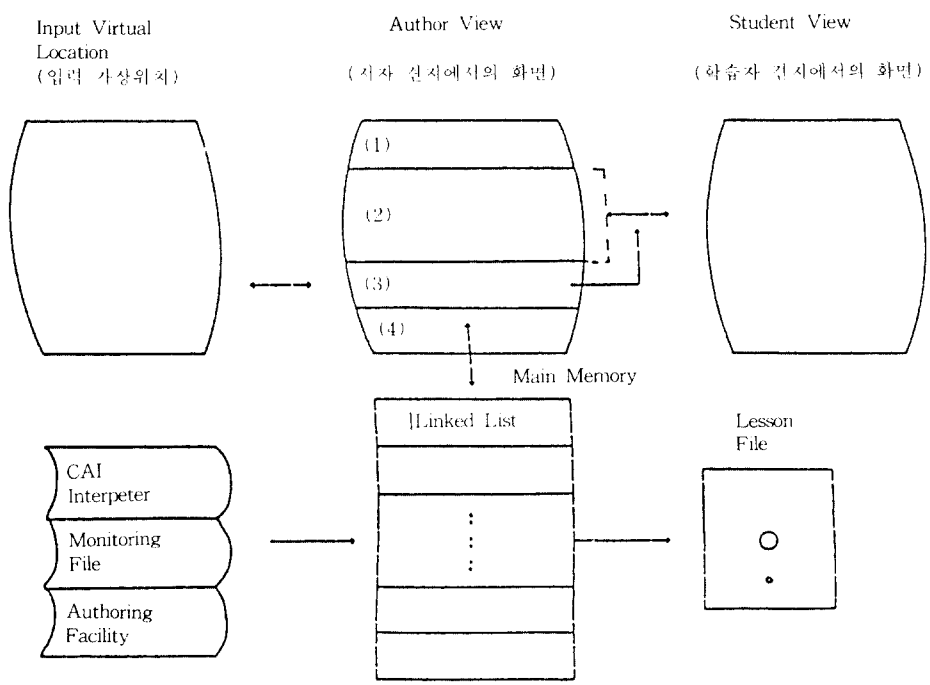


그림 4.1 화면의 구성

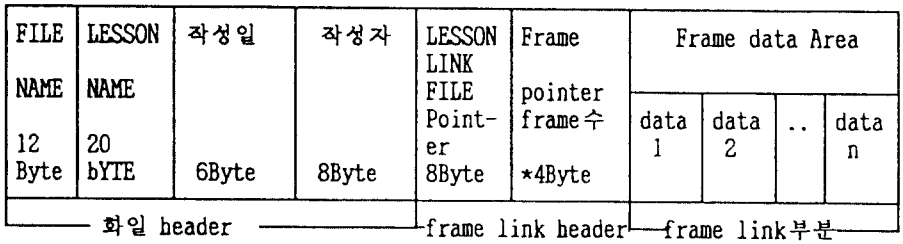


그림 4.2 레슨 화일의 물리적 구조

여, (4)의 영역에 필요한 자료부를 지인하여 해당 명령어를 작성한다.

작성된 명령의 실행 사항은 입력 가상위치 화면으로 도형이나 문자의 출력위치와 내용등을 수시로 검색하므로 올바르게 입력되었는지를 확인할 수 있다.

4.2 레슨 화일의 자료구조

레슨 화일은 학습 자료가 프로그래밍되어 있는 것으로서 한 개 이상의 프레임으로 구성될 때, 프레임은 하나의 화면 내용을 갖는다.

한 화면은 가로로 한글 40자(영문 80자), 세로는 25라인이며 그래픽이 될 수 있다. 프레임 구분은 프레임 번호로 정해지며, 저자가 직접 에디팅하거나 메뉴조작으로 자동 세팅할 수 있다.

레슨 화일의 물리적 구조를 나타내면 그림 4.2과 같다.

SMAT가 사용되는 주시 화일 헤더 부분이 나타나, 화일명, 레슨명, 작성일, 작성자, 선수학

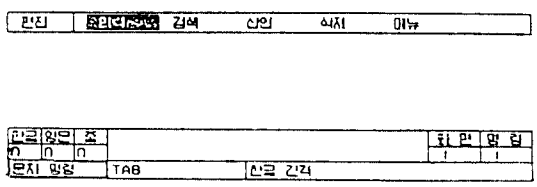


그림 4.3 입력화면 형식

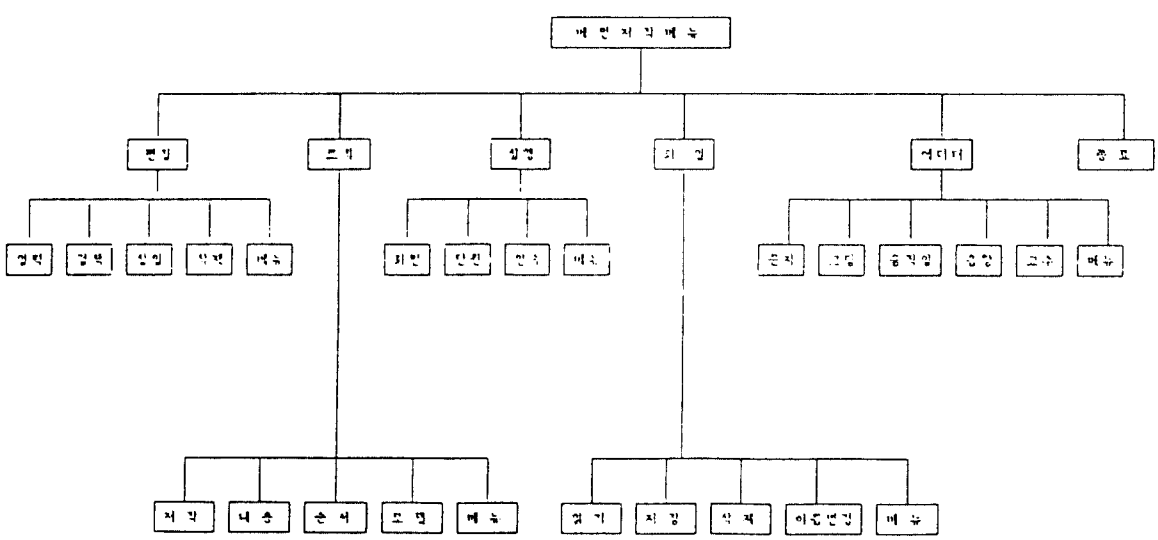


그림 4.5 저작 메뉴 계층

습 레슨명, 연계학습 레슨명을 입력하며 입력화면 형식은 그림 4.3와 같다.
frame 헤더 포인터는 프레임의 시작 데이타 영역을 지칭하는 포인터로 프레임 내 세개의 링크 영역을 두어 학습 로직에 따라 명령문 내 직접 전.후의 프레임 데이타 영역을 지칭하여 연계된 수업을 할 수 있도록 하였다.

4.3 모니터링 파일

모니터일 파일을 레슨 파일은 쉽고 편리하게 작성할 수 있도록 저자 환경을 제공해 주는 것으로 메뉴의 계층적 트리 구조를 갖으며, 이를 도시하면 그림 4.4과 같다.

4.4 에디터의 구성

SMAT의 에디터에는 텍스트, 그래픽, 에니메이션, 로직 등이 있다.

(1) 텍스트 에디터

WRITE 문에 의해서 화면에 출력되는 문자의 모양과 크기 등을 변경하고, 키보드에 없는 문자를 저자가 만들어 일반문자와 혼합하여 사용하도록 하며 알고리즘은 다음과 같다.

```
Procedure TextEditor (cursor-key);
// 문자 모양과 크기 지정및 새로운 문자를 생성 //
1. if location-cursor-key=Exchange-char-set then
   // 모양이 다른 3벌 한글, 영문중 하나 선택 //
     begin
     if location-cursor-key=한글 then select-한글-패턴(number);
                                  else select-영문-패턴(number);

     goto3 :
     end;
2. if location-cursor-key=Exchange-pattern then
   // font의 모양과 크기 변경 및 새로운 문자 생성 //
     begin
     if location-cursor-key=Alphabet-set then
        begin
        set-alpha-char(string);
        // 영문자 세트 지정 //
        set-modified-char(char-set);
        // 수정 문자 지정 //
        display-screen-char(char-set);
        // 에디터 화면 출력 //
        exchange-shape-char(char-set);
        // 모양 변경 //
        save-char-set(number);
        // 문자 저장 //
     goto 3;
     end;
     end;
  else
```

```
if location cursor key=special char then
  begin
    display font(num, font);
    // 저장된 특수 문자 font display //
    while select font num do
      assign font size(x num, y num);
      // font 크기 지정 //
      display screen font(font);
      // 에디터 화면에 출력 //
      design screen font(font);
      // 폰트 설계및 작성 //
      save fout (num, font); //
      새로운 font 저장 //
      end;
3. end
```

(2) 그래픽 에디터

간단한 모형은 그래픽 명령에 의해 쉽게 사용될 수 있으나, 복잡한 그림을 명령어 만으로 그리기는 매우 어려운 작업을 하여야 한다.

따라서 그래픽 에디터에서는 이러한 복잡한 그림들을 쉽게 그릴 수 있도록 화면에서 직접 그려 가면서 에디팅할 수 있도록 기능을 제공하며 알고리즘은 다음과 같다.

```
Procedure GraphicEditor (Cursor key);
1. if location cursor key=file main then
        // graphic data file 관리 //
        begin
        if location cursor key=read then load (file no);
          else if location cursor key=save then save(file no)
          go to 4  else delete(file no);
        end;
2. if location cursor key=file conv then
        // file conversion //
        begin
        select editor (cursor key); //
        사용할 editor 선정 //
        select conversion file (file no); //
        conversion 할 file 선정 //
        convert file (file no, file);
        // conversion file //
        save convert file (file no, file);
```

```
        / / 저장 / /
        goto 4;
        end;

3. if location-cursor-key=draw-graphic then
        / / 도형 작성 / /
        Begin
        select-pattern edited(number);
        / / pattern에는 선굵기 지정, 칼라 무늬
        draw-pattern;
        점, 선, 사각형, 원, 축소, 확대점단위
        save-pattern(num, pattern-date);
        / / 복사, 회전 페인트의 기능이 있다. / /
        end;

4. end.
```

(3) 에니메이션 에디터

CAI 프로그램 작성시, Animation은 기술적으로 매우 어렵고 작성 시간도 많이 요구된다. 따라서, 여러 형태의 Animation 처리 방법을 내장시켜 저자가 선택하여 사용할 수 있도록 하며 알고리즘은 다음과 같다.

```
Procedure AnimationEditor (cursor-key);
/ / 동적 영상을 표현하기 위한 것 / /
1. if location-cursor-key=file-main then
        / / animation data-file 관리 / /
        begin
        if location-cursor-key=load then (file-no);
        else if location-cursor-key=save then save(file-no)
            else delete(file-no);
        goto 5;
        end;

2. if location-cursor-key=repeate then
        / / 반복 data file 처리 / /
        begin
        set-file(no, date);
        / / repeat할 자료선정 / /
        set-repeat-range(x1,x2,y1,y2);
        / / 반복범위지정 / /
        process-simulation-screen;
```

```
        save-animation-data(file no, data);
        / / 저장 / /
        goto 5;
        end;

3. if location-cursor-key=Drawing then
        / / animation 제작 / /
        begin
        display-screen(num, data);
        / / 그림 데이터 display / /
        select-graphic-data(num);
        / / 그림 선택 / /
        move-location-data(x1, y1);
        / / 그림 이동 / /
        delay-time(num);
        / / 지연시간 지정 / /
        update-graphic-data(num, data);
        / / 저장 / /

4. simulate-graphic-data(num); end;
5. end.
```

(4) 로직 에디터

프로그램 수행의 흐름을 결정하는 logic 부분은, 교육 전략적인 면을 고려하여 CAI 프로그램에서 빈번하게 사용되는 logic 패턴들을 수용하므로서, 저자가 교육적인 전문 이론을 모른다 하여도 선형, 분기, 반복모형을 기본으로 저자가 이를 변형 또는 혼합할 수 있는 저자 환경을 제공하여 학습자의 개인차에 따라 level 순서를 결정할 수 있도록 설계한다.

또한, 효과가 높은 학습 방법을 선택하여 사용할 수 있도록 하며 알고리즘은 다음과 같다.

```
Procedure LogicEditor (cursor-key);
/ / 학습 전략과 키제어 / /
1. if location-cursor=key-design then
     / / key 설계 / /
     begin if function-interrupt(num) or function-control(num) then
        begin
        set-process-frame(num);
        / / 처리 frame-no 지정 / /
        set-key(char);
        / / key character 선정 / /
```

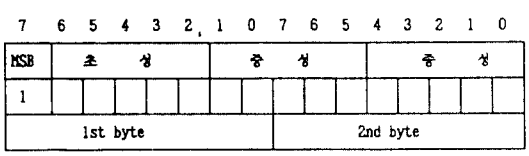
```
            display set-key-shape(num);
            / / 출력모양지정 / /
            goto 3;
        end:

2. if location-cursor-key=study-logic then
            / / 학습 전략 선택 / /
            begin
            select help-answer(num, string);
            / / 도움말 지정 / /
            select-logic-pattrn-spec(num, string);
            / / logic 패턴 선정 / /
            display-logic-pattern-spec(num, string);
            / / logic 패턴 출력 / /
            goto 3;
            end:
                  else
            begin
              display-authoring-pattern(num, string)
              / / 시작 패턴 선택 / /
              select-authoring-pattern(num);
            end:

3. end.
```

4.5 저작한글

저작 한글 코드 체계는 조합형 2바이트 코드이다. 이것은 음절 하나가 2바이트 코드 값을 가지는 것을 의미한다.

2 바이트 중 첫 바이트는 MSB가 1로 고정되고 차례로 초성, 중성, 종성이 각각 5비트 씩 할당된다.

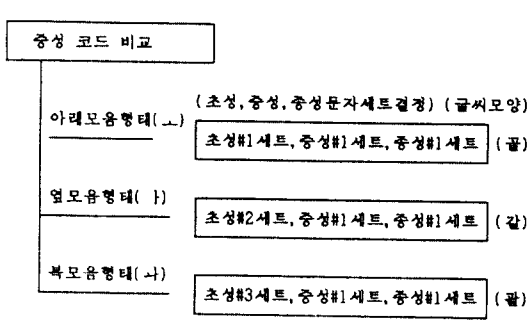
한글 자모의 비트 구성은 부록에 게재되어 있다.



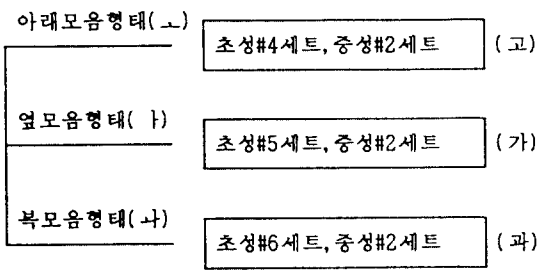
(1) 한글 모양 만들기 알고리즘

한글 자모의 세트는 초성 6세트, 중성 2세트, 종성 1세트로 구성되어 있으며 글모양 형성의 알고리즘은 블럭도로 도시하면 다음과 같다.

i) 종성이 있는 경우:



ii) 종성이 없는 경우:



(2) 한글 display 알고리즘

한글 Display는 해당 한글 코드를 읽고 그에 대응하는 코드로 분류한 후, 문자를 세팅하여 처리하며 처리과정을 나타내면 그림 4.6과 같다.

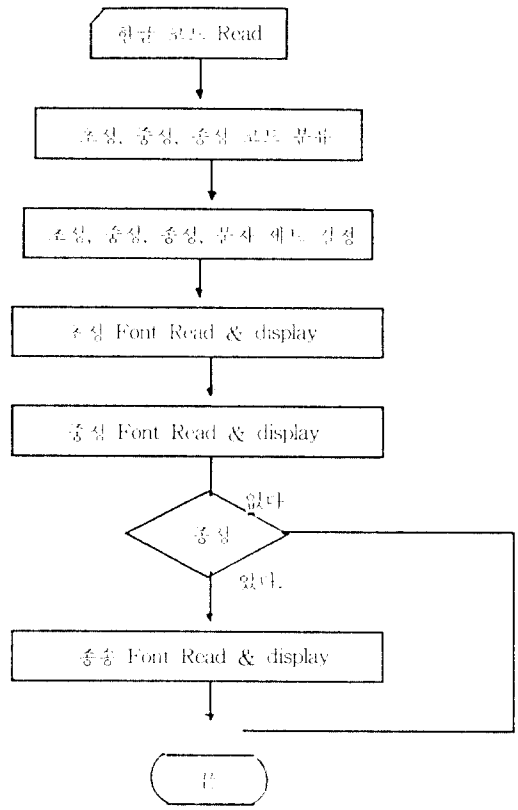


그림 4.6 한글 display algorithm flow

4.6 인터 프리터 구성과 내역

저자가 입력한 명령어는 일단 커버퍼에 저장되고 Operand의 입력이 끝나면 프로그램 메모리 영역을 줄이기 위하여 명령어 동작 버퍼와 비교하여 토큰으로 변환한다.

변환된 토큰은 실제 프로그램 영역에 Linked List 형식을 갖추면서 저장된다.

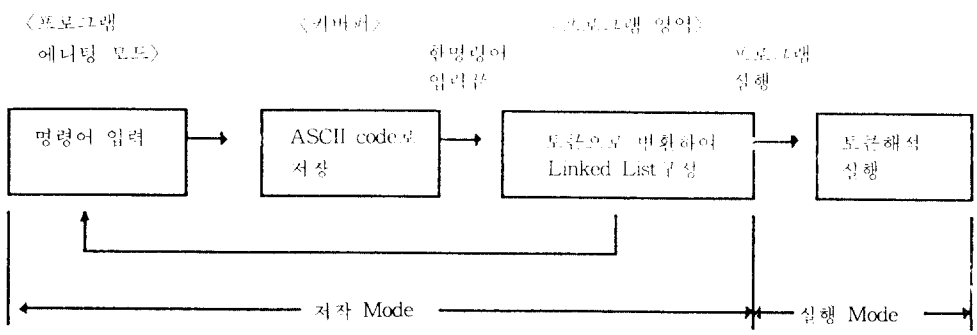


그림 4.7 인터프리터 처리 과정

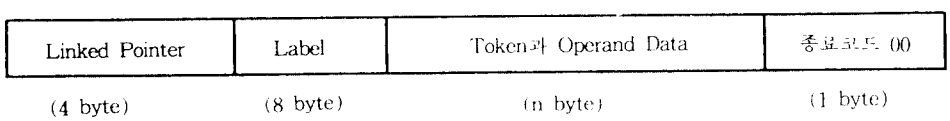


그림 4.8 Lesson File의 물리적 구조

이와같이 하여 저자의 입력이 모두 끝나고 프로그램을 실행시키면 토큰의 내용을 차례로 해석하면서 수행한다.

토큰의 처리는 토큰을 처리해주기 위한 기계어 서브루틴(명령어 처리루틴)을 토큰 점프 테이블에서 알아내어 그곳으로 가서 해당되는 처리를 한다.

프로그램 영역에 Linked List 형식으로 저장된 것이 Lesson File이며, 다음과 같은 구조를 갖는다.

인터프리터의 논리적 구조는 부록에 제시되어 있다.

V. 결 론

본 연구에서는 국내외의 저작 도구에 관한 문헌과 연구 논문을 중심으로, 각 모델의 특성을 비교 분석 검토하여 저작 도구의 설계 방향을 학국 교육 여건과 환경에 적합하도록, 메뉴 운용 방식을 좀 더 개선하여 학습 화면을 프레임 단위로 연계 리스트화하여 다양한 학습 전략을 모색할 수 있도록 방안을 조성하였다.

또한, 실제 사용하는 명령문 수도 고급 수준의 명령문 수 보다 적고, 저자가 저작 활동시에 사용하는 명령문의 사용 방법이 스크린 상에 나타나 용이한 저작 활동을 할 수 있으며, 프로그래밍의 기법을 잘 모르더라도 간단한 명령어와 약간의 교육 절차와 방법을 명시하면 쉽게 CAI 프로그램을 구현할 수 있도록 하였다.

문제점으로는 첫째, 주요 프로그램들이 BASIC으로 구현되었기 때문에 저작시 속도가 늦으나, 시뮬레이션을 통해 확정된 기능과 에디터들을 어셈블리 언어로 구현 시키므로 해결될 수 있으며, 둘째는 저작 설비로 축적된 기초 데이타인 그림이나 도형 및 특수 문자 폰트 등의 부족으로 저작환경에 다소 어려움이 있으나, 기초 데이타를 만들어 넣는 경우에는 보다 용이하게 저작 활동을 할 수 있을 것이다.

끝으로 칼라 그래픽의 지원과 다양한 주변 장치를 제어하지 못하고 있으나, 향후에는 완성하여 IBM-XT 호환 기종과 주변 기기(마우스, 조이스틱)을 활용할 수 있어 그 활용 범위가 넓어질 것이다.

참 고 문 헌

1. 김용성, 장옥배, 심재홍, "CAI 개발을 위한 저작도구에 관한 연구", 1989.6., 정보과학회지, 제7권 3호 p17-27.
2. 이태욱, "FORTH 언어를 이용한 저자용 컴퓨터 언어 개발에 관한 연구", 1989.6., 정보과학회지, 제7권 3호 p66
3. 김문규, "컴퓨터 학습용 소프웨어 개발 현황과 발전 방향", 1989.6., 정보과학회지, 제7권 3호, p38.
4. 고대곤, "한글 저작 시스템의 설계와 특성 연구", 1989.8., 연세대학교 대학원 학위 논문, p8-11.
5. 장옥배, 김용성, "ICAI 모델 프로그램에 관한 연구", 1988.4., 정보과학회지 발표 논문집, p105-118.
6. 김용성, 장옥배, "저작 도구 설계 방안", 1989. 10., 추계 정보과학회지 심포지움.
7. P.G. Barker and R. singh, "Author Languages for computer-Based Learning", British Jouranl of Education Technology, Vol 13, 1982, pp.167-194.
8. Peterson, JWM and sessions, AG, "A transportable authoring system", Computers and Education, Vol 2, No. 4, 1978, pp331-334.
9. Chung Taech-hee, "The study and Development of CAI model program", RK85-11, 1985, pp 9-20.
10. KLALA Vancso-polacsek, "Automated construction of interactive learning programs in modula-2", comput. Educ. vol 12, 1988 No 4, pp 507-511.
11. Greg Kearsley, "Artifical Intelligence and Instruction", Addition wesley, 1987, pp 323-328.
12. M.D. Leiblum, "A Model for Describing CAL Authoring systems applied to TAIGA", Comput. Educ. Vol 12, No 1, 1988, up 141-149.
13. Philip G Barker, "MUMEDALA-an approach to multimedia authoring" Vol 15, No 1, 1984, pp 5-13.
14. M. Darid Merrill, "Prescription for an authoring system", Joural of computer-Based Education, Vol 14, No 1, 1987, pp 1-10.
15. P.G. Barker and R. singh, "A practical Introduction

to authoring for computer assisted Instruction, PART 1 : IPS", British Journal of Education Technology, Vol 14, No 2, 1983, pp 26-45.

16. P.G. Barker and R. singh, "A practical Introduction to authoringfor computer assisted Instruction, PART 2 : PILOT", British Journal of Education Technology, Vol 14, No 1, 1983, pp 174-200
17. P.G. Barker and R. singh, "A practical Introduction to authoring for computer assisted Instruction, PART 3 : MICROTEXT", British Journal of Education Technology, Vol 15, No 2, 1984, pp 82-106.
18. P.G. Barker and R. singh, "A practical Introduction to authoring for computers assisted Instruction, PART 4 : STAF", British Journal of Education Technology, Vol 16, No 2, 1985, pp 115-134.
19. Greg Kearsley, "Authoring Systems in computer Based Education", ACM, Vol 25, No 7, 1982, pp 429-437.
20. Isreael Pressman and Brace Rosenbloom, "CAI : Cost and Its Role", Journal of Educational Technology Systems, Vol. 12(3), 1981-84, pp 183-208.
21. Authorware Inc., "Course of Action Reference Manual".
22. Control Data Corporation, "PLATO USER's GUIDE".
23. WICAT Systems Inc., "WISE Reference Manual".
24. Locatic, C. and Carr, V, "Systems for Authoring Computer Based Instruction", Technical Report No. LHNCBC 85-1, National Institute of Health, Bethesda, HD 1985.
25. Michie, D, "Expert Systems", The computer Journal, 23(4), pp. 369-376, 1980.
26. Philip Baker, "Author Language for CAI", Elsevier Science Publishing co., Inc., 1987.
27. Alty, J. L, "Path algebras: a useful CAI/CAL analysis technique", Computers and Education, 8 (1), pp.5-13, 1984.
28. Baker, P. G. and Yeates, H, "Preparing, obtaining and evaluation courseware materials", pp.50-111, (Chapter 2), in Introducing Computer Assisted Learning, Interactive Systems Research Group, April 1983.
30. Conlon, T, APPLE PILOT, Prentice-Hall, in press.

김 용 성(Yong Sung KIM) 정회원
1978년 : 고려대학교 수학과 졸업
1986년 8월 : 성균대학교 대학원 전자계산학과 박사과정 수료.
1978년 1월~1980년 9월 : 삼성그룹 사회조성실 전산팀 근무.
1986년~현재 : 전북대학교 전산통계학과 전임강사로 재직 현재 교무과장 겸무중.
관심분야 : 고대로 알고리즘, 소프트웨어공학, 교육공학.

심 재 홍(Jae Hong SIM) 정회원
1967년 : 서울대학교 수학과 졸업.
1960년 : 고려대학교 대학원(이학석사).
1981년~1988년 : 성균대학교 대학원 (이학박사).
1984년~1986년 : 정보사회의 부학장 역임.
현재 : 성균대학교 교수로 재직중.
관심분야 : 고대로알고리즘, 고대로스, 수치해석.

[부 록] 한글 저작 도구

1. 편집키

Programming 작업이 편리하도록 다음과 같은 편집 Key가 있다.

- Ins : 문자 삽입
- Del : 문자 삭제
- BS : 앞문자 삭제
- Home : Frame 1 출력
- End : 마지막 Frame 출력
- PgUp : 현 Frame의 앞 Frame 출력
- PgDn : 현 Frame의 다음 Frame 출력
- Tab : OP code 또는 Operand 단위로 커서 이동
- Enter : 다음 Line 첫 문자로 커서 이동
- Esc : Program Editing Mode Menu로 복귀
- F1 : 현 Frame Print
- F2 : 전체 Frame Print
- F7 : Font Table 출력
- F8 : 칼라 모양 Table 출력
- F9 : 한글 출력
- F10 : 영문 출력

2. 한글 자모의 비트 구성

	비트 구성	초 성	중 성	종 성
0	00000			
1	00001			
2	00010	ㄱ		ㄱ
3	00011	ㄲ	ㅏ	ㄲ
4	00100	ㄴ	ㅐ	ㄳ
5	00101	ㄷ	ㅑ	ㄴ
6	00110	ㄸ	ㅒ	ㄵ
7	00111	ㄹ	ㅓ	ㄶ
8	01000	ㅁ		ㄷ
9	01001	ㅂ		ㄹ
A	01010	ㅃ	ㅔ	ㄺ
B	01011	ㅅ	ㅕ	ㄻ
C	01100	ㅆ	ㅖ	ㄼ
D	01101	ㅇ	ㅗ	ㄽ
E	01110	ㅈ	ㅘ	ㄾ
F	01111	ㅉ	ㅙ	ㄿ
10	10000	ㅊ		ㅀ
11	10001	ㅋ		ㅁ
12	10010	ㅌ	ㅚ	
13	10011	ㅍ	ㅛ	ㅂ
14	10100	ㅎ	ㅜ	ㅄ
15	10101	ㄲ	ㅝ	ㅅ
16	10110		ㅞ	ㅆ
17	10111		ㅟ	ㅇ
18	11000			ㅈ
19	11001			ㅊ
1A	11010		ㅠ	ㅋ
1B	11011		ㅡ	ㅌ
1C	11100		ㅢ	ㅍ
1D	11101		ㅣ	ㅎ
1E	11110			
1F	11111			

3. 인터프리터 논리적 구조