멀티미디어 저작을 위한 스크립트 인터프리터의 설계 및 구현

차 현 성† · 한 광 록††

요 약

저작도구에서 스크립트는 기본적인 이벤트에 대한 정의 뿐만 아니라 객체에 대한 특별한 행동을 표현하고 저작도구 전체를 동적으로 변화시킬 수 있는 특성을 가져야 한다. 본 논문에서는 스크립트에 의해서 다양한 멀티미디어 객체의 속성 및 이벤트를 정의하고 실행하기 위한 스크립트 인터프리터를 설계 및 구현하였다. 본 인터프리터에서 스크립트 소스의 분석과정과 실행 테이블 관리 방법 및 객체에 스크립트 정보 저장 방법, 그리고 사용자가 발생시키는 이벤트에 반응하여 객체가 소유하고 있는 스크립트 정보를 실행하는 과정에 대하여 논한다.

The Design and Implementation of Script Interpreter for Multimedia Authoring

HyunSung Cha† · KwangRok Han††

ABSTRACT

The script of multimedia authoring tool must be able to define the basic events, represent the special function of an object and change the entire system dynamically.

In this paper, we design and implement a script interpreter that defines the properties and events of various multimedia objects and executes them, and describe the analysis of script source, the management of execution table, and the method for storing the script information to the objects and executing it according to the user's events.

1. 서 론

1990년대 이후 컴퓨터 하드웨어 기술의 급속한 발전과 더불어 CD-ROM등과 같은 대용량 저장 매체가 등장하여 많은 처리 시간을 요구했던 멀티미디어 자원을 실시간으로 처리할 수 있게 되었다. 이러한 영향으로 멀티미디어에 대한 요구 및 관심이 증가하게 되었다. 초기 멀티미디어 저작물의 형태는 단순히 멀티미디어 자원을 배열하는 수준에 그치고 있다. 현재 멀티미디어 저작이 교육, 의료 및 사업 등 이용 분야가 확대되고 있는 현실 속에서 많은 한계를 드러내게 되었다.[9]

이러한 한계를 극복하기 위해 멀티미디어 저작도구에 대한 많은 연구가 국내외에서 진행되어 왔다. 현재 국내에서도 많은 소프트웨어 개발자들이 저작도구를 개발하였지만 단순히 멀티미디어 저작물을 나열하거나 간단한 스크립트 기능을 구현하여 기본적인 행위만을 지정하는 수준에 그치고 있다. 저작도구에서의 스크립트의 기능은 기본적인 이벤트에 대한 정의뿐만 아니라 객체에 대한 특별한 행동을 구현하고 저작도구 전체를 동

†준 회 원 : 호서대학교 컴퓨터공학과
††종신회원 : 호서대학교 컴퓨터공학과
논문접수 : 1997년 6월 13일, 심사완료 : 1998년 2월 13일

적으로 변화시킬 수 있는 특성을 가진다. 따라서 본 논문에서는 객체에 관한 세밀한 연산을 위하여 저작도구에 제공되는 객체를 C++의 클래스와 유사한 형태로 설계하여 그 속성을 제어하도록 하였으며 각 객체를 스크립토 언어에서 자료형이 되도록 설계하고 참조 연산자를 사용하여 각 객체의 속성을 제어함으로서 보다 구조적이고 객체 지향적인 프로그래밍 환경을 개발하였다. 또한 저작도구가 실행되는 시점을 시스템이나 사용자에 의한 메시지 발생으로 하여 객체마다 정의되어 있는 이벤트에 대한 스크립트와 비교하여 실행한다. 이러한 저작구조는 주로 교육용 타이틀, 종합 안내 시스템인 키오스크(KIOSK), 게임, 데이타베이스 전위시스템, 오락, 그리고 교육과 오락을 결합한 에듀테인먼트(edutainment)등의 응용 프로그램 개발이 용이하다.

본 논문의 구성은 다음과 같다. 2장에서는 저작도구의 개요에 대해서 논하고, 3장에서는 객체 표현 모듈과 이벤트의 발생과 처리에 대하여 논한다. 4장에서는 저작도구에 대한 개략도와 문서 표현 구조에 대해서 설명하며, 4장에서는 스크립트 인터프리터의 구성과 구문분석기, 어휘 분석기에 대하여 논하고 5장에서 스크립트를 실행하는 실행기에 대하여 논한다. 6장에서 설계 및 구현된 저작도구를 예제를 통하여 실행되는 과정을 기술하고 7장에서 결론 및 향후 계획에 대하여 서술한다.

2. 저작도구 개요

2.1 저작도구의 요구 기능

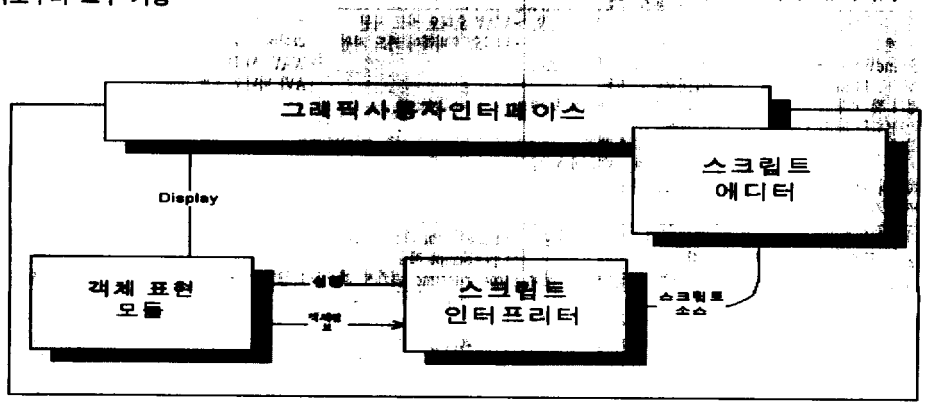
저작도구에서 요구되는 기능을 분류하면 다음과 같다.[1][12]

① 각 미디어 자체(intramedia)나 미디어 장치간(intermedia) 연결 기능
② 사용자 입력에 관한 미디어 제어기능
③ 사용자 입력에 관한 미디어 재생기능
④ 미디어 파일의 동시 실행 기능

사용자 입력에 관한 미디어 제어 및 재생 기능은 사용자가 임의로 정의한 스크립트에 와한 처리와 저작도구에 정의되어 있는 모든 이벤트에 해당되는 기본적인 행위에 의하여 처리된다.[6] 전자의 경우 사용자가 스크립트 에디터를 호출함으로써 해당 미디어에 관한 기능을 정의할 수 있고, 후자의 경우 저작도구에 기본적으로 정의된 기능을 수행하는 것이다. 미디어 파일의 동시 실행 기능은 저작도구에서 가장 중요시되어야 할 기능이라 할 수 있다. 동시 실행 중에 동기화가 어긋나게 되면 음성과 모양이 일치하지 않는 어색한 상황이 발생되게 된다. 따라서 저작도구를 구현하기 위해서는 위와 같은 미디어에 관한 기본적인 네 가지 기능을 통합한 멀티미디어의 다양한 기능이 요구된다.

2.2 전체 시스템 구성

본 저작 도구는 별도의 윈도우에서 개별 미디어 편집기를 수행시켜 내용을 편집한 후 편집된 멀티미디어 자원을 통합하여 보여 줄 수 있으며 저작도구는 (그림 1)과 같이 세 부분으로 구분되어 설계되었다. 첫 번째 그래픽 사용자 인터페이스는 저작 단계에서 객체를 배치



(그림 9) 저작도구의 전체 구성도
(Fig. 1) The whole construction diagram of authoring system

하고 실행시 저작물을 보여주거나 작성된 저작물을 수정한다. 두 번째 스크립트 인터프리터는 사용자에 의하여 입력된 스크립트 소스(Script Source)를 입력으로 어휘 분석 및 구문 분석하여 실행 테이블을 생성하고 생성된 실행 테이블을 해당하는 객체에 저장한다. 세 번째 객체 표현 모듈은 객체에 대한 속성을 실제 코드로 가지고 있으며 스크립트에서 정의한 객체에 대한 연산을 수행하고 결과를 GUI에 출력하는 역할을 한다.[5][10]

2.3 기존 저작도구 연구

본 저작도구는 책/페이지 방식의 저작 과정을 제공한다. 이러한 저작 방법은 기존의 저작도구인 멀티미디어 툴북과 유사하다. 또한 부분적으로 아이콘에 의한 흐름도 방식을 채택하고 있어서 기존의 저작도구에 대한 여러 가지 특징을 최대한으로 수용하였다고 할 수 있다. 또한 현재 연구가 진행중인 다중 사용자에 의한 공동 저작 환경을 추가함으로서 다른 저작도구와의 차별화 하는 것을 목표로 하고 있다. 다음 (표 1)은 기존의 국내, 외 저작도구들을 각 기능별로 분류 및 비교한 것이다.

3. 객체 표현 모듈 및 이벤트

3.1 객체 표현 모듈 설계

저작도구의 객체 표현 구조는 (그림 2)와 같은 계층 구조를 갖는다. 최상위 객체인 북(Book)은 하나 이상의 객체를 포함한다. 각 객체의 종류는 다음과 같다.[6][7][10]

〈표 6〉 저작도구의 비교
〈Table 1〉 Comparison of a Authoring Tool

	Authorware	IconAuthor	ToolBook	Artware	TargetSystem
사용환경	윈도우즈,Mac	윈도우즈	윈도우즈	윈도우즈	윈도우즈
저작방식	아이콘 흐름도	아이콘 흐름도	책 / 페이지	아이콘 흐름도	책 / 페이지
디스플레이 (텍스트 및 그래픽)	· 기본적인 워드프로세서 기능 · 선,원,사각형등 그림툴 제공 · 36가지 필 패턴 제공 · BMP, PCX, DIB 등 그래픽 파일지원	· 전체 화면이나 부분 화면에서 편집 가능 · 3차원적인 입체 및 그림자의 특수효과 가능 · WMF,PCX,RLE,BMP등 파일지원	· HotWord 기능으로 HyperText의 제작이 쉬움 · 선,원,사각형 등 툴 팔렛트 지원 · OpenScript 제공 · BMP,DIB,WMF파일 지원	· 색상, 글꼴 변환 · 벡터 그림(선, 원, 사각형 등)지원 · BMP파일 지원	· 텍스트 색상, 글꼴 및 다양한 효과 지원 · 벡터 그림(선, 원, 사각형 등)지원 · 다양한 효과 (지움 등) 지원 · BMP, DIB, GIF파일 지원
에니메이션	· 목적지, 경로 등 정해진 방향으로 이동하거나 경로, 속도를 조절 · 셀과 비트맵 에니메이션 생성	· Wipe, Zoom, Fade등 10여 가지의 특수효과 지원 · 마우스를 이용한 경로지정 가능 · Overlay와 3차원적인 에니메이션 효과 가능	· 스크립트 언어를 통한 에니메이션 제어 · Authorware, IconAuthor 보다 미흡한편	· FLC, FLI 에니메이션 파일 지원	· FLC, FLI에니메이션 파일 지원 · 스크립트에 의한 에니메이션 (비트맵, 객체, 텍스트 등) 지원
오디오 및 비디오 처리	· 다양한 속도와 디지털 오디오 재생 기능 · SoundWave 편집기 제공 · AIFF, PCM오디오 지원 · 정지 및 동화상 비디오 지원 · 전체/일부 화면 출력 · 화면 번호에 의해 속도 조절 가능 · 하나의 채널에 2개의 오디오 연결 가능	· 디지탈 오디오 파일 저장, 재생 가능 · MCI를 통한 CD, WAV, MIDI오디오 지원 · 편집이 쉬운 VCR리모트 기능 지원 · 전체/일부 화면 출력 · 이미지 포착, 저장,압축 및 크기 조정 가능	· WAV 오디오 카드 지원 · 비디오 오버레이 카드 지원	· 윈도우즈의 사운드 카드 드라이버 사용 · WAV, MIDI등의 MCI 지원 · AVI 비디오 포맷 지원	· 윈도우즈의 사운드 카드 드라이버 이용 · WAV, MIDI등의 MCI 지원 · AVI 비디오 포맷 지원 · 윈도우즈의 MCI 지원
데이터 처리	· 다양한 토픽 구사가능 · 응용 프로그램 호출, 복귀 가능 · CMI기능으로 사용자의 학습 수준과 결과 판단 가능	· dBase III,IV의 dbf 파일과 호환 · 데이터의 로드, 수정, 저장 가능 · 사용자 입력 내용 분석 가능 · 다양한 Bookmark기능	· dBase의 dbf 파일과 호환 · OpenScript 제공 · free runtime 패키지 제공으로 산출물의 복사, 배부가 용이 · Password로 보안 기능 제공	· 자체 개발의 스크립트 언어 지원 · 다양한 변수 및 내장 함수 지원	· 스크립트 언어 지원 : 자체 인터프리터 개발 · 다양한 개발 유형 지원(반복연습형, 개인교수형, 게임형, 시뮬레이션형등) · 다양한 CMI기능 구현 가능 · 다중 저작자의 접근에 관한 관리 기능 · 공동 저작에 필요한 데이터 관리 기능 · 다중 사용자 인터페이스 지원

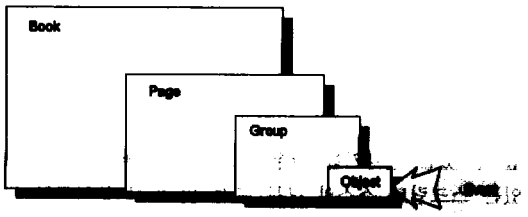
① 페이지(Pages) : 페이지 객체는 다른 객체들을 실제로 표현 할 수 있는 곳이며 다수의 페이지 객체는 북을 구성한다.

② 벡터 그래픽 객체(Vector Graphics Objects) : 다각형(Polygon), 선(Line), 베지어 곡선(Bezier Curve), 타원(Ellipse), 다각선(PolyLine), 아크(Arc)로 구성된 객체.

③ 윈도우즈 컨트롤 객체(Windows Control Objects) : 버튼(Button), 콤보 박스(Combo-Box), 텍스트(Text), 리스트 박스(ListBox), 체크 박스(CheckBox), 라디오버 튼(Radio Button)으로 구성된 객체.

④ 그룹 객체(Group Objects) : 다수의 객체를 하나의 객체로 통합한 객체.

⑤ 멀티미디어 객체(Multimedia Objects) : 웨이브 사운드(Wave Sound), 미디 사운드(Midi Sound), CD 오디오, 디지털 비디오(Digital Video), 비트맵 그래픽(Bitmap Graphic), 에니메이션(Animation)으로 구성된 객체.



(그림 10) 객체 구성도
(Fig. 2) Object construction diagram

3.2 이벤트 처리

사용자 또는 시스템에서 발생한 이벤트는 객체 내의 이벤트 핸들러(Event Handler)에 의해 처리지로 전환되어 이벤트가 발생한 객체에서부터 까지 전달되어진다. 즉 어떤 객체에 이벤트가 메시지가 발생하고 그것에 해당하는 스크 객체를 찾기 위해 이벤트가 발생한 위치로부터 상위 객체를 찾아간다.[10] 최상위 객체 즉, 북 객체에서도 해당되는 이벤트에 대한 스크립트가 존재하지 않을 경우 저작 도구에서 기본적으로 정의한 기본 스크립트를 실행한다. 기본 스크립트는 라이브러리 형태로 북이 생성될 때 번역되어 테이블 형태로 저작도구에 존재하게 된다.

4. 스크립트 인터프리터

4.1 스크립트의 기능 정의

텍스트 기반에서의 객체 연산은 단순히 라인 단위의 스크립트 명령 형태나 간단한 형태의 연산만을 지원하는 스크립트로 충분했지만 현재 다양한 멀티미디어 객체들을 실시간으로 제어 할 수 있는 시점에서 구조적 객체 연산이 요구되고 있고 사칙연산 및 기본의 프로그래밍 언어가 가지고 있는 기능을 필요로 한다. 본 저작도구에서는 스크립트에 다음과 같은 기능을 설계하고 구현하여 이러한 구조적 프로그래밍 환경을 구현하였다. 멀티미디어 저작도구를 위한 스크립트의 기능은 크게 세 가지로 구분된다. 첫 번째 기능은 타 프로그래밍 언어(BASIC, PASCAL, C등)가 기본적으로 가지고 있는 기능을 가져야 하며, 두 번째로 가져야 할 기능으로는 멀티미디어 데이터를 조작할 수 있는 기능이다. 마지막으로 세 번째는 기존의 작성된 스크립트 소스를 라이브러리 파일 형태로 저장하여 같은 작업을 수행하는 다른 객체에 대하여 기존 코드를 재사용 할 수 있는 기능이다. 본 저작도구의 스크립트는 하나의 객체와 참조 연산자로서 그 속성을 제어하는 객체 지향적 형태를 제공한다. 이러한 형태는 기존의 C++의 클래스와 유사한 개념으로 객체에 대한 행동 정의를 간편히 할 수 있고, 객체를 효율적으로 관리 할 수 있다는 장점이 있다. 또한 사용자 정의화 객체 및 그 속성을 추가함으로서 객체에 대한 폭넓은 확장을 가능하게 하였다.

(1) 프로그래밍 언어의 기본 기능

일반적인 프로그래밍 언어는 연산의 수행에 필요한 순차 실행문, 주어진 조건을 판단할 수 있는 판단문, 명령어의 반복 수행을 위한 반복문 등 세 가지 기본 구조를 갖는다. 또한 대부분의 프로그래밍 언어는 내장된 데이터 타입(문자형, 숫자형, 배열형 등)을 가지며, 세 종류의 명령문으로서 프로그래머가 원하는 결과를 얻기 위한 연산을 수행한다. 따라서, 저작도구를 위한 스크립트 언어는 기본적으로 이러한 기능을 포함해야 한다.

(2) 멀티미디어 데이터 조작 기능

멀티미디어 데이터는 텍스트, 비트맵 이미지 등의 정적인 데이터와 사운드, 비디오 등의 동적인 데이터로 구분될 수 있다. 멀티미디어 저작을 위하여 저작도구는 동적인 데이터를 조작하기 위한 것과 더불어 정적인 데

이터를 조작 할 수 있는 명령들을 제공하여야 한다. 다음 스크립트 소스는 버튼에 다운 이벤트가 발생되었을 때 사전에 생성된 비디오 객체를 재생하기 위한 명령들이다.

```
Event btn1.LButtonDown()
begin
 mcatMCI(open, AVI, Video1);
 mcatMCI(play, AVI, Video1);
end.
```

위의 명령어들은 비디오를 보여주기 위한 가장 기본적인 기능이지만 이 외에도 동적인 멀티미디어 데이터의 특성을 고려한 여러 가지 속성 제어 명령군들이 존재한다. 또한, 본 스크립트 언어는 텍스트의 색상이나 글꼴 변경, 비트맵 이미지의 조작(회전, 역상)등, 정적인 데이터를 조작하기 위한 명령을 제공한다. 다음 스크립트 소스는 생성된 백터 그래픽 객체의 색상과 속성을 변경하기 위한 명령들이다.

```
Event btn1.LButtonDown();
var
 Rectangle rt1;
begin
 mRect1.Color := GetRGB(255, 255, 0);
 mRect1.Name := "그림";
end.
```

(3) 코드 재사용 기능

본 스크립트는 코드의 재사용을 위해서 텍스트 형태의 스크립트 라이브러리(text based Script library)를 제공한다. 이러한 라이브러리는 사용자가 직접 작성하는 함수(user define function)의 재사용을 위한 것이다. 각 객체에서 작성된 사용자 정의 함수는 『Procedure』라는 키워드를 사용하여 선언되며 처음 분석된 후 저작도구의 최상위 객체인 북 객체에 일괄적으로 저장된다. 또한 사용자는 북이 제거될 때까지 어떤 객체에서든지 정의된 사용자 함수를 호출 할 수 있다. 라이브러리 파일은 이러한 사용자 함수들을 사전에 텍스트 형태로 작성하여 저장한 파일로서 『import』라는 키워드를 사용하여 이벤트에 반응하는 스크립트를 가진 객체에 가장 먼저 선언한다.

다음은 사용자 함수 사용 예이다.

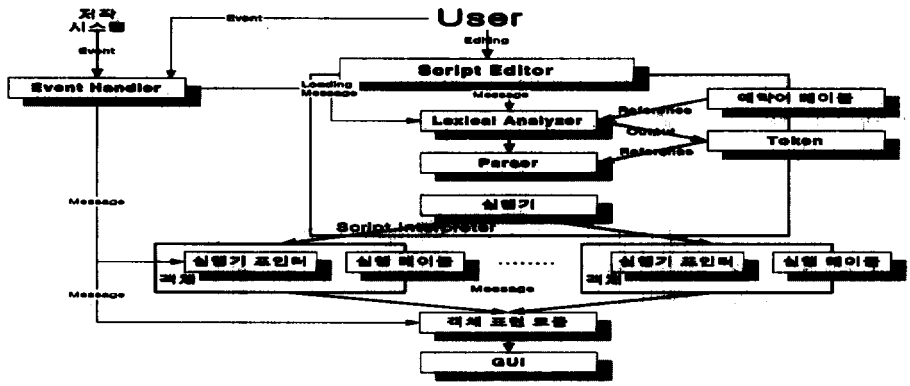
```
import "Create_Button.ssf"
Event btn1.LButtonDown();
 var
  Rectangle rt1;
  Rectangle rt2;
begin
 rt1.Color := rt2.Color;
end.
```

4.2 스크립트 인터프리터의 설계

스크립트 인터프리터(script interpreter)는 사용자에 의하여 입력되어진 스크립트 소스를 입력으로 각 토큰을 분리하는 어휘 분석기(lexical analyzer), 생성된 토큰을 입력으로 문법을 검색하고 실행시 사용되어질 테이블을 생성하는 구문 분석기(syntactic analyzer), 실행 테이블을 입력으로 객체에 대한 행위를 실행하는 실행기, 스크립트의 분석 과정 및 실행 과정에서 발생하는 오류에 대한 처리를 담당하는 오류 처리기로 구성된다.[1][3][8] (그림 3)은 사용자, 객체 표현 모듈과 스크립트 인터프리터간의 동작 관계를 나타낸 것이다. 모든 객체에는 스크립트 소스를 저장할 장소와 실행 테이블을 저장하고 실행테이블을 자원으로 실행할 수 있는 실행기를 내장하고 있다. 객체가 소유하고 있는 스크립트 소스의 구문 분석 및 어휘 분석되는 경우는 두 가지로 구성될 수 있다. 첫 번째는 사용자가 특정 객체를 선택하여 스크립트 편집기(Script Editor)를 호출 한 후 스크립트를 부여하여 스크립트 에디터에 있는 스크립트 분석에 대한 메뉴를 선택하는 경우이다. 이 경우 스크립트 에디터는 구문 분석에 분석 메시지를 보내게 되고 최종적으로 객체에 실행 테이블을 생성하고 저장하게 된다. 두 번째는 기존에 생성되어진 페이지가 적재 될 때 이벤트 처리기는 페이지 내의 모든 객체를 대상으로 적재된 메시지를 보내게 된다. 메시지를 받은 모든 객체는 자신의 스크립트 소스의 유무를 검색하여 스크립트가 있을 경우 구문, 어휘 분석을 수행하여 앞의 경우와 마찬가지로 객체에 실행 테이블을 생성 및 저장한다.

4.2 어휘 분석기의 설계 및 구현

스크립트 에디터를 통하여 입력된 스크립트 소스는



(그림 11) 스크립트 인터프리터의 동작도
(Fig. 3) The operation diagram of script interpreter

예약어 테이블과의 비교를 수행함으로써 각 속성에 해당하는 토큰을 분리하여 테이블 형태로 저장되어진다. (그림 4)는 어휘 분석과정을 도식화한 것이다.[1] 어휘 분석과정에서 보다 편리한 변수 관리를 위하여 변수명에 대한 한글 처리를 첨가하였다. 토큰의 속성은 다음과 같다.

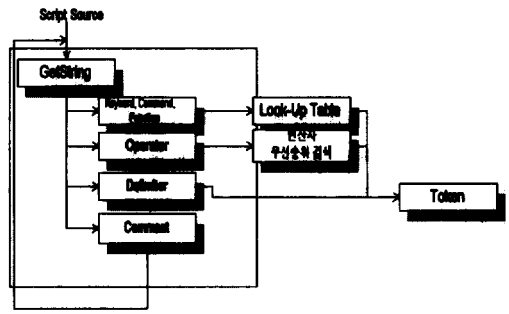
① 기본 속성
- 토큰 입력 순서, 토큰 스트링 길이, 토큰 스트링
- 선언된 토큰 타입 정보(연산자, 키워드, 함수 등)
- 오류 발생시 스크립트 에디터 상에서의 토큰의 위치에 대한 정보.

② 연산자 속성
- 연산자 우선 순위

③ 객체 속성
- 객체 타입 정보 (3.1 절에서 명시한 객체 타입)

이와 같은 토큰에 대한 클래스를 다음과 같이 정의한다. [10]

```
// Token Table
class token : public CObject
{
public :
  int token_index;                  // token 입력 순서.
  int strln;                        // token string 길이
  CString ch_string;                // token string
  CString INFO_token;               // 선언된 token 타입.
  CString INFO_DECL;                // 토큰 정의 타입 정보.
  int INFO_OB;                      // Operator priority 또는 Object 여부 표시.
  int INFO_Object;                  // Object 타입에 관한 정보.
  // error 발생시 highlight를 위한 속성 값
  int line_index;                   // y value
  int index;                        // x value
}
```



(그림 12) 어휘 분석 과정
(Fig. 4) The process of Lexical Analysis

4.3 구문 분석기의 설계 및 구현

본 논문에서 구현한 구문 분석기는 프리딕티브 파서(Predictive Parser) 알고리즘으로 구현한다.[1][8]

입력 : Grammar G.

출력 : 실행 테이블

Method

(1) 각 문법에 있는 각 생성식 A -> α에 대하여 과정 2, 3을 실행한다.

(2) FIRST(α)에 속해 있는 각 터미널 a에 대해서, A -> α를 테이블 M[A, α]에 넣는다.

(3) 만일 FIRST(α)가 ε을 포함하고 있으면 FALLOW(A)의 터미널인 b에 대해서 A를 테이블에 넣는다. 만일 FIRST(α)가 ε이고 FOLLOW(A)가 $(End of Token)일 경우 A->$\alpha$를 테이블 M[A, $] 에 넣는다.

(4) 정의되지 않은 테이블 M의 목록을 오류로 만든다.

본 저작도구는 각 이벤트가 수행하는 기본 행위에 대한 스크립트와 전역적으로 사용되어지는 상수형 선언 자료 등을 기본적으로 가지고 있다. 이러한 기본적인 스크립트는 저작도구가 처음 실행될 때 분석과정을 거쳐 실행 테이블을 생성한다. 또한 사용자가 직접 구현할 수 있는 함수(User Define Function)의 재사용을 위하여 스크립트를 별도의 파일에 저장할 수 있게 하였다. 사용자 정의 함수로 구성된 파일은 "import"라는 키워드를 사용하여 사용자 함수를 호출하는 스크립트의 처음에 포함되어진다.

4.4 실행 테이블 관리

본 시스템의 모든 객체들은 실행 테이블을 가지고 있다. 이러한 실행테이블은 사용자가 특정 객체에 스크립트를 부여하여 어휘 및 구문 분석을 수행한 결과로 생성되거나 페이지가 처음 적재될 때 페이지 내의 모든 객체들의 스크립트를 분석함으로써 생성되어진다. 이렇게 생성된 테이블은 실행시 실행기에 의해서 참조되어 이벤트에 해당하는 스크립트를 실행한다. 구문 분석과정에서 생성되어진 테이블들은 각각 저장되는 장소에 따라서 전역 테이블과 지역 테이블로 구성되어 진다.

(1) 전역 테이블

전역 테이블은 북에 저장되어 있으며 스크립트가 사용자 정의 함수이거나 전역 변수 및 상수에 해당되는 경우이며 그 종류는 다음과 같다.

① 사용자 함수 테이블 : 예약된 함수를 제외한 사용자가 임의로 정의한 모든 함수를 저장한다.

변수, 연산자, 상수토큰에 대한 배열 및 사용자 함수 이름과 상위 객체의 이름으로 구성되어져 있으며 "Procedure" 키워드로 함수를 선언한다.

② 사용자 함수 정보 테이블 : 선언된 사용자 함수에 대한 여러 가지 정보를 저장한다. 함수에 대한 인자, 인자 개수, 반환값 저장 장소, 반환형으로 구성되어져 있다.

③ 전역 변수, 전역 상수 테이블 : 북의 모든 영역에서 참조될 수 있는 형태의 변수로 전역상수의 경우 정의된 이후 새로운 값을 재정의 하지 못한다. 변수에 대한 값과 타입등으로 구성되어져 있으며, "Procedure" 또는 "Event" 키워드 앞부분에 선언한다.

전역으로 정의되는 테이블들은 직접적으로 실행되지 못한다. 단지 참조 가능한 라이브러리 파일 형태로 지역 테이블에서 호출되어야만 실행이 가능하다. 다음은 각 전역 테이블에 대한 실제 코드를 나타내었다.

```
. 사용자 함수 테이블
class CGlobalTable : public CObject
{
public :

        CObArray ident_table;   // Identifier table
        CObArray op_table;      // Operator table
        CObArray token_table;
        CObArray const_table;

        CString TBL_name;
        CString BookPageName;
        CString Msg_name;
}
```

```
. 상수 테이블
// 상수 테이블에 대한 정의...
class CConstTable : public CObject
{
public :
        double mConstValue;
        CString mConstName;
        int mType;
}
```

```
. 사용자 함수 정보 테이블
class UserFuncINFO  : public CObject
                      // 사용자 함수에 대한 정보
{
public :
        int ParaCount;// parameter Count...
        CString FuncName;// 함수 이름...
        int RtnType;// 반환값 타입...
        void operator = (UserFuncINFO *);
        DataStorage mRtnValue;
}
```

(2) 지역 테이블

모든 객체들은 이벤트에 따른 독립적인 지역 테이블을 가지고 있으며 사용자가 "Event" 키워드로 선언함으로써 정의된다. 지역 테이블은 선언되어진 이벤트가 발생할 경우 직접 실행될 수 있으며 전역 테이블에 저장된 사용자 함수를 호출하여 실행할 수 있다.

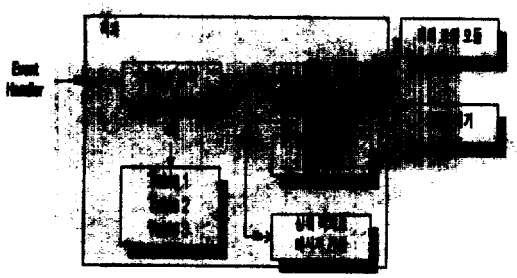
```
. 사용자 함수 테이블
class CGlobalTable : public CObject
{
public :

        CObArray ident_table;   // 변수 테이블
        CObArray op_table;      // 연산자 테이블
        CObArray token_table;
        CObArray const_table;

        CString TBL_name;
        CString BookPageName;
        CString Msg_name;
}
```

```
. 연산자 테이블
class COp_table : public CObject
{
public:
        token* OP_left;// 왼쪽 토큰의 인덱스...
        token* OP_string;
      token* OP_right;//오른쪽토큰의 인덱스.....
        int OP_flag;
}
```

```
. 변수 테이블
class CId_table : public CObject
{
 public:
  token* Id_name;          // Identifier...
  token* Id_definition;    // Identifier 선언 부...
  int Id_value;            // Identifier table의 인덱스...
  int INFO_type;           //객체 형태
  DataStorage mDataSt;     // 변수저장 장소...
  BOOL mColorFlag;         // Line Color, Brush Color...
  int  mTextFlag;          // Text 종류를 구별
  BOOL mParaFlag;          // Parameter flag 구별...
  BOOL mArray;             // 배열...
  int ArrayFlag;           // 배열의 type.
  int ArrayMax;            // 배열의 크기...
  int ArrayPos;            // 배열의 포지션...
}
```

5. 실행기의 구현

본 논문의 저작도구는 두 가지 모드로 구성되어져 있다. 첫 번째 모드는 저작 모드로 사용자가 객체들을 임의로 생성 및 이동 할 수 있고 객체에 스크립트를 할당 할 수 있다.(6) 다른 한가지는 실행 모드로 객체에 대한 수정 및 스크립트 입력이 금지되며 이벤트의 발생과 이에 대응하는 스크립트의 실행으로 객체에 변화를 줄 수 있다. 각 스크립트의 실행기는 저작은 실행 모드에서 객체가 스크립트에 해당하는 메시지를 받을 경우이다. 해당 메시지는 사용자 및 시스템에 의하여 발생한 이벤트를 이벤트 처리기에 의하여 해당 메시지로 맵핑(mapping)되며 이벤트 처리기는 실행기와 어휘 분석기(페이지가 처음 적재될 경우) 및 객체 표현 모듈로 메시지를 보낸다. 메시지를 받은 객체는 자신이 소유하고 있는 스크립트를 검색하여 해당 스크립트인 경우 실행기를 호출하고 해당되지 않는 경우는 자신의 상위 객체로 이벤트를 옮겨 준다. 스크립트의 실행은 이벤트가 발생된 객체의 지역 테이블을 검색하여 연산자 테이블을 순차적으로 처리함으로써 실행하게 되며 연산자의 종류에 따라 계층연산, 조건 연산, 사칙 연산, 함수 연산 등 각 부분으로 분리되어진다. (그림 5)는 이벤트 처리기로부터 메시지를 받아서 스크립트를 실행하는 과정을 나타낸다.



(그림 13) 스크립트 실행 과정
(Fig. 5) The process of script execution

6. 실 행

본 [illegible] 스크립트 소스를 분석하여 실행하는 [illegible]. 다음 스크립트 소스는 마우스 [illegible] 객체가 [illegible]을 경우 발생한 버튼 [illegible]는 이름을 주고 버튼 사용자에 의해서 [illegible] 「Cap[illegible]ton.mif」라는 파일을 임포트해서 실행 즉, 버튼(Button) 객체를 그려주는 명령을 하는 스크립트 소스를 나타낸다.

본 저작도구의 초기화면은 (그림 6)과 같다. 처음 툴바 또는 메뉴상의 객체를 선택한 후 GUI에 마우스로 객체를 생성해 주는 작업을 한다. (그림 7)는 두 개의

```
. File name : CreateButton.ssf
Procedure Create_Object(rect mRegion) : BOOL;
var
  PushButton    btn1;
begin
  btn1.Position := mRegion;
  btn1.Color := GetRGB(255, 0, 255); /* 기본 함수*/
  if (btn1.Show = TRUE) then
    bnt1.Show := FALSE
  else
    btn1.Show := TRUE;
  Create_Object := TRUE;
end.
```

```
. File name : CallButton.ssf
import "CreateButton.ssf";
Event LButtonDown(int nFlag, point pt);
var
  rect mRegion;
  BOOL mTF;
begin
  mRegion := GetRegion(100, 100, 300, 300);
  mTF := Create_Object(mRegion);
end.
```

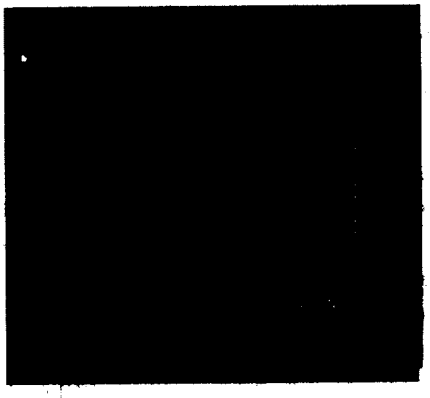
버튼객체와 비트맵 객체를 생성하고 첫 번째 버튼에 스크립트를 할당하는 과정을(그림 8) 나타낸다. (그림 9)는 저작물의 실행 상태를 나타낸다.

7. 결 론

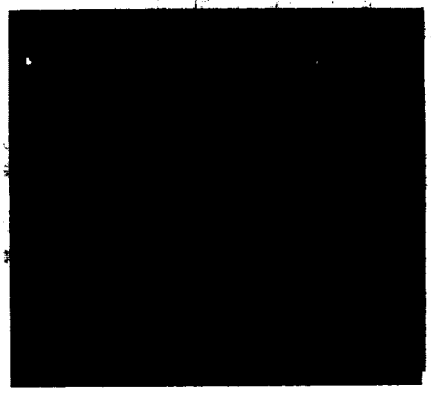
본 논문에서는 저작 도구에서 다양 메시지에 반응하는 스크립트에 대한 처리와 정적인 객체에 동적인 변화를 줄 수 있는 스크립트 언어와 인터프리터를 설계 및 구현하였다. 구현을 위한 환경으로는 객체 지향적인 모델을 제시하는 마이크로소프트사의 비주얼 C++를 사용하여 구현하였고 저작 환경을 위한 운영체제로는 윈도우즈 95를 사용하였다. 본 스크립트 언어는 표준 파스칼을 기본으로 객체를 조작하기 위한 부분을 첨가 시켰고 이벤트의 발생에 따른 함수만을 작성해 줌으로서

(그림 6) 저작도구
(Fig. 6) Authoring Tool

(그림 7) 객체 생성
(Fig. 7) Object Create



(그림 8) 스크립트 소스 편집
(Fig. 8) Editing of Script Source



(그림 9) 실행결과
(Fig. 9) The Result of Execution

각 객체에 대하여 객체 지향적인 실행을 할 수 있게 하였다. 또한 페이지 내의 모든 객체의 스크립트 소스에 대한 분석을 페이지가 적재될 때 수행하여 실행시 어휘, 구문 분석과정을 생략함으로서 실행시간을 단축 시켰다. 모든 객체에 스크립트 소스 및 실행 테이블을 저장하여 북 내에서 객체의 이동을 용이하게 하였다. 향후 연구 과제로는 초보적인 사용자라도 쉽게 사용할 수 있는 라인 단위의 실행기 설계 및 구현과 네트워크 환경에서 공동으로 저작할 수 있는 스크립트 언어에 대한 함수를 설계 및 구현할 예정이다.

참 고 문 헌

[1] Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, "Compilers Principles, Techniques and Tools", Addison-Wesley, pp181-195, 1988.

[2] 임은기, 권용래, "Structure-Oriented 프로그래밍 환경을 위한 Syntax-Directed 에디터의 구현", 한국정보과학회 가을 학술발표 논문집, Vol.15, No.2, pp115-118, 1988.

[3] Thomas W. Parson "Compiler Construction Programming", AP propessional, 1994.

[4] Richard N. Taylor, Nenad Medvidovic, Kenneth M. Anderson, E. James Whitehead Jr., "A Component-and Message_Based Architectual Style for GUI Software", IEEE Transactions on Software Engineering, Vol.22, No.6, 1996.

[5] Koji Takeda, Mitsuyuki Inaba, and kazuo Sugihara, "User Interface and Agent Prototyping for Flexible Workinhttp://www.nb.net/~lrc/page6.htmlg", IEEE Multimedia, pp40-50, 1996.

[6] 성미영, "분산 멀티미디어 환경에서의 동등 저장", 한국정보처리학회 멀티미디어 [illegible] 회 멀티미디어 산업기술 [illegible] -126, 1995.

[7] "OpenScript Reference Manual", Asymetrix, 1994.

[8] Christopher W. Fraser, David R. Hanson, " A Retargetable C Compiler : Design and Implementation", The Benjamin/Cummings, 1995.

[9] 차현성, 한광록, "멀티미디어 공[illegible]구 개발을 위한 사용자 인터페이스의 설계[illegible]회 춘계 학술발표 논문집[illegible] pp700-703, 1996.

[10] 이종인, 차현성, 한광록외, "스크[illegible]터 개발을 위한 이벤트 처리와 객체 [illegible] 구", 한국정보처리학회 춘계 학술발표 논문집, 제4권 1호, pp325-329, 1997.

[11] 양옥렬, 정영식, 이용주, "멀티미디어를 기반으로 하는 저작도구 툴북에서 객체 자동 변환을 이용한 자동 프리젠테이션 시스템 개발", 한국정보처리학회 논문지, 제4권, 제5호, pp1182-1195, 1997.

[12] Frans C. Heeman, Ivan Herman, Graham Reynolds, "Interaction objects the MADE Multimedia Environment", Multimedia/Hyp-ermedia in Open Distributed Environments, Proceeding of the Eurographics Symposium, Springer-Verlag Wein New York, pp. 264-277, 1994.

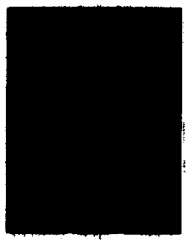
[13] M. E. Hodges, R. M. Sasnett, "Multimedia Computing, Case Studies from MIT Project Athena", Addison-Wesley, 1993.

차 현 성

[illegible] 2월 호서대학교 컴퓨터공[illegible] 졸업(공학사)

[illegible] 2월 호서대학교 대학원 컴퓨터공학과 졸업 (공학석사)

관심분야 : 멀티미디어 저작시스템, 데이터베이스, 정보검색 자연언어처리 등



한 광 록

1984년 2월 인하대학교 전자공학과 졸업

1986년 2월 인하대학교 대학원 전자공학과(공학석사)

1989년 8월 인하대학교 대학원 전자공학과(공학박사)

1991년 3월~1998년 현재 호서대학교 컴퓨터공학과 부교수

관심분야 : 자연언어처리, 정보검색, 멀티미디어 저작 시스템 등