추상적인 시각 프로그래밍 시스템

박영조°, +최종명, 유재우
숭실대학교 컴퓨터학과, +국립목포대학교 컴퓨터공학과
yjpark@ss.ssu.ac.kr, choijm@dreamwiz.com, cwyoo@comp.ssu.ac.kr

An Abstract Visual Programming System

Park Youngjo°, +Choi JongMyoung, Yoo Chaewoo
Computing School of Soongsil Univ.
+Computer Engineering of Mokpo National Univ.

요 약

현재 다양한 시각 프로그래밍 시스템과 다이어그램을 사용하는 모델링 도구가 증가하고 있다. 이러한 도구에서 사용되는 각 그래픽 요소들은 매우 다양하고, 동일한 형태라도 의미가 다르며, 각 그래픽 요소들이 연결될 수 있는 형태가 다르기 때문에 범용적인 그래픽 편집기를 개발하기 어렵다. 본 논문은 다양한 시각 프로그래밍 언어에 적용가능한 편집기를 개발하기 위해 시각적인 표현에 필요한 기본적인 기능을 제공하는 추상적 시각 프로그래밍 편집기와 각 시각적 프로그래밍 언어마다 의미하는 그래픽 요소를 생성하는 토큰편집기를 가진 추상적인 시각 프로그래밍 시스템을 제안한다. 사용자는 제안한 시스템을 이용하여 각 언어별로 알맞은 명시적 시각 프로그래밍 편집기의 생성이 가능하다. 또한, 모든 명시적 시각 프로그래밍 편집기는 플랫폼에 독립적인 형태로 생성이 가능하다.

1. 서론

현재 다양한 시각 프로그래밍 시스템과 다이어그램을 사용하는 모델링 도구가 개발되고 활용되어지고 있다. 텍스트 편집기에서는 모든 텍스트 기반 언어에 대해서 프로그래밍을 할 수 있는 반면에, 시각 프로그래밍 편집기에서는 각 언어별 특성에 맞는 편집기가 필요하다. 시각적으로 표현되는 그래픽 요소들의 다양함과 각 시각 프로그래밍 언어들에서 사용되는 요소들의 의미 차이는 범용적인 시각 프로그래밍 편집기의 개발을 어렵게 한다.

본 논문에서는 각 언어에 알맞은 그래픽 요소를 생성하기 위한 토큰편집기를 이용하여 기본적인 그래픽 요소와 추가적인 토큰의 생성이 가능하다. 또한, 추상적 시각 프로그래밍 편집기에서는 각 시각 프로그래밍 언어에서 공통으로 사용되는 기능과 각 그래픽 요소를 시각적으로 표현하기 위한 기능을 가진다.

추상적 시각 편집기의 기능들을 상속받고, 토큰편집기를 이용한 언어에 알맞은 토큰을 생성하고, 각 언어에 필요한 추가적인 기능을 구현한 명시적 시각 프로그래밍 편집기의 생성이 용이한 시스템을 제안한다.

본 논문의 2장에서는 시각 프로그래밍 언어와 편집기에 대해서 정의하고, 3장과 4장에서는 본 논문에서 제안한 시스템의 설계와 실험예제를 서술하며, 5장에서 본 논문의 결론과 향후에 연구할 방향에 대해서 논한다.

2. 시각 프로그래밍 언어와 편집기

시각 프로그래밍 언어는 사용자가 시각적인 요소를 사용해서 프로그래밍을 할 수 있는 언어를 지칭한다[1]. 많은 사람들이 그림 형태로 사물을 생각하고 기억하기 때문에 그래픽적인 방법으로 사물과 연관 지으며, 만들어진 생각의 주된 요소들처럼 상상하는데 그림들을 사용한다. 이 이유로 시각 프로그래밍 언어는 텍스트 기반 언어들에 비해서 생산성이 높고, 배우기 쉬운 장점을 가진다[2].

시각 프로그래밍 편집기는 시각 프로그램의 생성을 지원하도록 명시된 그래픽 편집기이다. 범용적인 텍스트 편집기가 모든 텍스트 기반의 프로그래밍 언어를 만족시키는 반면에, 시각 프로그래밍 언어는 각 언어들에 대해서 특화된 편집기가 요구된다.

범용적인 시각 프로그래밍 편집기 개발 시 문제점은 다음과 같다. 첫째, 시각적으로 나타나는 그래픽 원시 요소들의 집합만으로 범용적인 편집기에서 모든 시각 프로그래밍 언어에 대한 프로그래밍은 한계가 있다. 간단한 시각 프로그래밍은 가능할지라도 통합된 요소가 요구되는 경우는 표현이 불가능하다. 둘째, 편집기에서 제공하는 원시 요소들의 집합이 너무 포괄적이거나, 집합 내에 프로그램 안에는 포함되지 말아야 하는 그래픽 요소가 포함될 수 있다. 셋째, 일반적인 그래픽 편집기는 의미정보의 기술 및 활용방법을 지원하지 않는다.

범용적인 시각 프로그래밍 편집기의 장점은 매번 새로운 시각 프로그래밍 언어에 대해서 새로운 편집기가 요구되지 않는다. 이러한 형태의 예로는 Palette[3]가 있다. 이 편집기는 많은 다른 시각 프로그래밍 언어에서도 활용 가능한 시각 프로그래밍 편집기이다. 이 편집기의 특징은 시각 프로그래밍 편집기를 만들기 위해서 일반적인 그래픽 편집기를 생성하고 각 시각 프로그래밍 언어에 알맞도록 변화시킨다[3].

다른 형태로는 시각 구문지향 편집기 생성시스템[4]이 있다. 이 시스템은 시각 프로그래밍 언어에 대해서 그들의 명세에 알맞은 객체지향적이고, 구문지향적 편집기를 생성해주는 시스템이다. 이 시스템은 크게 두 부분으로 구성되며, 명세언어와 생성 시스템으로 나누어진다. 명세언어는 시각 프로그래밍 언어에 대해서 구문과 의미를 정의하는데 사용된다. 생성 시스템은 명세언어를 입력으

로 받아들여서 구문지향 편집기를 생성한다[4].

3. 설계

3.1. 시스템 구조

본 논문에서 제안한 시스템은 각 시각 프로그래밍 언어의 특성을 부여하기 위해서 특화된 토큰을 정의할 수 있는 토큰 편집기와 모든 시각 프로그래밍에서 공통적으로 사용되는 기능을 제공하는 추상적 시각 프로그래밍 편집기, 추상적 시각 프로그래밍 편집기를 상속받아 각 시각 프로그래밍 언어에 특화된 명시적 프로그래밍 편집기로 구성된다.

기본 그래픽 요소만으로 각 언어가 가진 특성들을 적절하게 표현하기 어렵다. 토큰 편집기에서는 기본 그래픽 요소들과 기본 그래픽 요소들이 결합한 새로운 그래픽 요소를 생성한다.

추상적 시각 프로그래밍 편집기는 각 언어에서 공통적으로 지원하는 기능을 제공한다. 예를 들어, 토큰 편집기에서 생성한 토큰을 화면에 보여주거나 이동 등의 편집기 기본동작을 지원하는 기능을 제공한다.

명시적 시각 프로그래밍 편집기는 추상적 시각 프로그래밍 편집기를 상속받아 토큰 편집기에서 생성한 토큰을 활용할 수 있도록 해주는 시각 프로그래밍 언어에 의존적인 편집기이다.

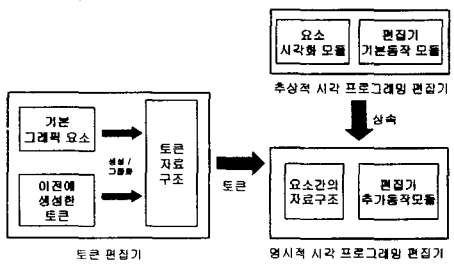

그림 1 전체 시스템 구조

본 논문에서 제안하는 편집기는 기본적으로 TokenEditor클래스에서 토큰을 생성하는 역할을 수행한다. Token클래스는 모든 그래픽 요소의 기본이 되는 클래스로써, 기본적인 그래픽 요소를 TokenEditor클래스에 제공하고 토큰편집기에 의해서 새로운 토큰을 생성할수 있는 기능을 제공한다.

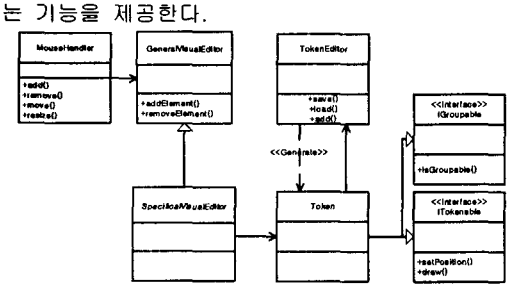

그림 2 클래스간의 관계

GeneralVisualEditor클래스에서는 MouseHandler클래스를 이용해서 토큰들의 제어를 위한 기능을 제공하고 제어되는 토큰들에 대한 정보를 자료구조로 내포하고 있다. SpecificalVisualEditor클래스에서는 GeneralVisualEditor클래스를 상속받아서 토큰들의 제어와 정보를 이용하며, 토큰편집기에서 생성된 토큰을 활용가능하게 한다.

3.2. 토큰 편집기 설계

토큰 편집기는 시각 프로그래밍 언어에서 사용되는 토큰들의 특성을 정의하고 시각적인 표현을 생성하는 편집기이다. 이 편집기는 기본 그래픽 요소와 이전에 생성했던 토큰들이 결합한 그룹을 이용하여 새로운 토큰의 생성이 가능하며, 새롭게 생성된 토큰과 다른 토큰과의 연결성을 나타내기 위해서 연결자가 사용된다.

연결자는 각 토큰에 연결 가능한 토큰 타입을 정의하게 된다. 이렇게 정의된 타입은 명시적 시각 편집기에서 다른 토큰들과 연결 관계를 표현 시에 타입 검사를 통해서 일치하는 타입들만이 연결 가능하다.

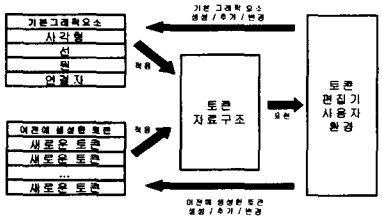

그림 3 토큰 편집기 구조

3.3. 추상적 시각 프로그래밍 편집기

추상적 시각 프로그래밍 편집기의 경우는 아래 그림 4에서 보는 바와 같이 일반적으로 시각 프로그래밍 편집기에서 요구되는 기본적인 모듈들을 내포하고 있다.

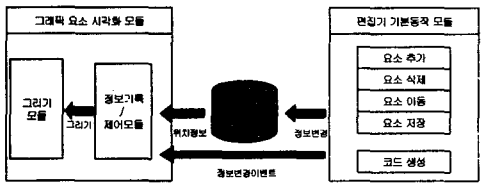

그림 4 추상적 시각 프로그래밍 편집기 구조

그래픽 요소 시각화 모듈은 편집기에서 사용되는 그래픽 요소들의 시각적 표현에 대한 관리와 방법들이 정의되어 있다. 이 모듈에 내포된 정보 기록 / 제어 모듈은 현재 각 요소가 가진 정보를 유지하며, 변경 시에 다시 그래픽 요소를 표현하도록 명령하는 역할을 수행한다.

그리기 모듈은 현재 그래픽 요소가 정확하게 화면에 그려질 수 있도록 하는 역할을 담당한다. 편집기 기본동작모듈은 이러한 그래픽 요소들에 대한 추가, 삭제, 이동, 저장 등에 대한 기능을 제공하며, 추가적으로 텍스트 기반의 언어에 대해서 코드를 생성하는 부분도 제공한다.

3.4 명시적 시각 프로그래밍 편집기

명시적 시각 프로그래밍 편집기는 실제로 추상적 시각 프로그래밍 편집기를 상속해서 구현한 편집기를 지칭

하며, 추상적 시각 프로그래밍 편집기의 모든 기능과 특성들을 상속받아서 사용된다. 또한, 추상적 시각 프로그래밍 편집기가 각 언어의 특성을 반영하지 못하는 단점을 극복하기 위해서 명시적인 시각 프로그래밍 편집기에서는 각 언어별 특성에 알맞은 토큰을 활용하고 이러한 토큰들을 기반으로 한 특화된 편집기를 생성한다. 각 토큰들 간의 관계와 추가적으로 필요한 동작 등을 서술할 수 있는 기능도 제공한다.

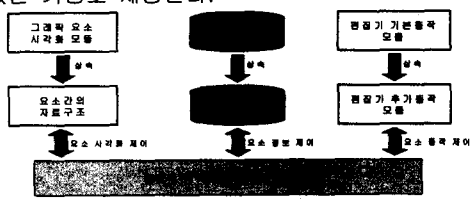

그림 5 명시적 시각 프로그래밍 편집기 구조

4. 실험

4.1. 실험예제

시작과 중지버튼을 가진 윈도우 프레임의 개발을 할 수 있는 편집기를 생성한다. 시작과 중지버튼을 생성하기위해 상위에 "버튼" 토큰을 상속받는다. "윈도우 프레임"은 "버튼"속성을 가진 토큰과 연결될 수 있다.

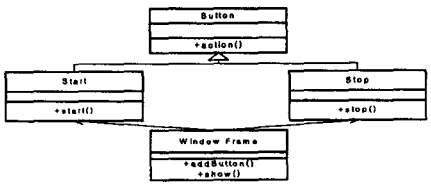

그림 6 실험 클래스 관계도

4.2. 토큰편집기

아래 그림 7에서 나타낸 바와 같이 토큰편집기는 각 언어에 알맞도록 사각형, 원, 선과 같은 기본적인 요소들을 이용해서 새로운 토큰을 생성한다. 연결자는 각 토큰에서 연결 가능한 토큰들의 타입을 설정하며, 명시적 시각 편집기에서 토큰 활용 시에 연결 가능성 여부를 판단하는데 사용된다.

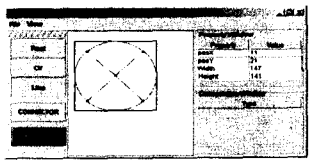

그림 7 토큰 편집기 실행화면

4.3. 명시적 시각 프로그래밍 편집기

명시적 시각 프로그래밍 편집기는 각 언어별 특성에 맞춘 편집기라고 정의한다. 명시적 시각 프로그래밍 편집기는 추상적 시각 프로그래밍 편집기를 상속받아서 토큰들의 동작 제어와 토큰의 관계 설정 등을 행한다.

아래 그림 8의 예제는 위의 그림 7을 이용해서 4가지 형태의 토큰인 "버튼", "중지", "시작", "프레임"을 생성하였다. 중지는 버튼을 상속받아서 중지버튼 형태를 의미하며, 시작도 중지와 유사한 시작버튼 형태를 나타낸다. 프레임은 시작과 중지버튼과 연관되어 실제로 윈도우 프레임에서 시작, 중지기능버튼이 추가됨을 의미한다.

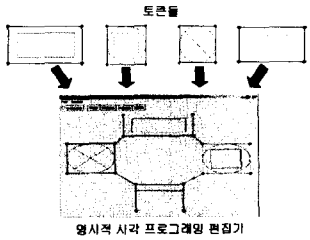

그림 8 명시적 시각 프로그래밍 편집기 실행화면

5. 결론 및 향후과제

현재 프로그래밍 언어들은 언어별 고유특성으로 인하여 편집기의 개발속도 저하를 가져오며, 특히 시각 프로그래밍 언어의 경우는 각 언어별로 사용되는 그래픽 요소들의 차이로 인하여 범용적인 시각 프로그래밍 편집기의 개발에 어려움을 가진다.

본 논문에서 제안한 시스템은 각 시각 프로그래밍 언어에 알맞은 시각 프로그래밍 편집기의 개발에 용이하도록 각 언어표현에 필요한 토큰 생성과 토큰을 활용이 가능하도록 해주는 기능을 제공한다. 토큰 생성 기능과 토큰 활용 기능은 각각 토큰편집기와 추상적 시각 프로그래밍 편집기가 담당한다. 사용자는 두 가지 편집기를 이용해서 명시적 시각 프로그래밍 편집기를 작성하며, 명시적 시각 프로그래밍 편집기는 명시된 시각 프로그래밍 언어를 활용이 가능하도록 하는 편집기이다.

본 논문에 제안한 시스템의 장점은 다음과 같다.

- 각 시각 프로그래밍 언어에 맞도록 토큰 생성과 토큰의 타입 검사를 통한 각 언어별 특성을 보장
- 명시적 시각 프로그래밍 편집기는 추상적 시각 프로그래밍 편집기를 상속받아서 기본적인 기능을 제공하고, 내부적으로 추가적인 기능을 제공할 수 있도록 하여 확장성을 증대
- 자바 기반의 구현언어로 인한 플랫폼 독립성을 보장

향후에는 명시적 시각 프로그래밍 편집기에서 다중상속이 가능하도록 하여 다양한 추상적 시각 프로그래밍 편집기에서 기능의 활용이 가능하도록 할 예정이다.

6. 참고문헌

[1] N. C. Shu, "Visual Programming: Perspective and approaches", IBM Systems Journal, Vol. 38, 1999.

[2] M. Boshernitsan, "Visual Programming Languages : A Survey", 1997.

[3] Eric J.Golin, "Palette: An Extensible Visual Editor", Proc. of ACM/SIGAPP symposium on Applied computing, 1992.

[4] Frahangiz Arefi, "Automatically Generating Visual Syntax-Directed Editors", Vol. 33, CACM, p349~360, 1990.