C# 코드 생성 기능의 설계 및 구현

이장우°, 김태균
부산외국어대학교 컴퓨터공학과
{jwlee°,ktg}@saejong.pufs.ac.kr

Design and Implementation of C# Code Generation

Jangwoo Lee°, Taegyun Kim
Dept. of Computer Engineering, Pusan University of Foreign Studies

요 약

본 논문에서는 기존에 구현된 객체지향 CASE 도구인 OODesigner에 C# 언어 코드 생성 기능을 추가로 설계하여 구현한 결과를 제시한다. 본 논문의 결과로 기존의 OODesigner에는 Java, 일반적인 C++, Visual C++ 코드 생성기능을 가지고 있었는데 C# 코드 생성을 지원할 수 있도록 기능이 향상되었다. .NET 프레임워크 하에서 동적 웹 페이지, 분산 응용 프로그램의 컴포넌트, 데이터베이스 엑세스용 컴포넌트 등을 개발하기 위한 C#의 사용이 확산되고 있는 상황으로 볼 때 본 논문에서 구현된 C# 코드 생성 기능의 구현은 C#을 이용한 소프트웨어 개발의 생산성 향상에 크게 이바지할 것으로 판단된다.

1. 서론

객체 지향 패러다임의 확산으로 인하여 소프트웨어 개발을 위한 시스템 분석 및 설계를 위해 UML(Unified Modeling Language)[1]의 이용이 보편화되고 아울러 UML을 지원하는 도구의 사용이 확산되고 있다. 시스템의 분석 및 설계 결과를 시각적으로 보여주는 UML 모델의 이용은 CASE 도구를 통한 코드 생성 기능과 함께 사용될 경우에 소프트웨어 생산성을 크게 향상시킬 수 있다. 현재 IT 분야의 산업계와 학계에서 많이 사용되고 있는 상업적인 CASE(Computer Aided Software Engineering) 도구인 Rational 사의 Rational Rose나 TogetherSoft 사의 Together와 같은 도구가 객체 지향 언어에 대한 코드 생성 기능을 지원하고 있다는 점은 코드 생성 기능이 보편화되고 있음을 보여준다.

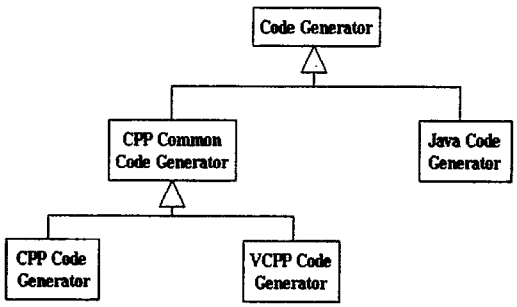
본 연구의 이전에 본 연구팀에 의해 개발된 UML CASE 도구인 OODesigner[2]는 UML에 속하는 각종 표기법인 class diagram, use case diagram, sequence diagram, collaboration diagram, state diagram, activity diagram, deployment diagram 등의 다이어그램을 작성할 수 있으며 C++, Visual C++, Java 언어에 대한 코드 생성 기능을 가지고 있다. 본 연구팀에 의해 개발된 OODesigner의 코드 생성 기능이 다른 상업적 도구의 코드 생성 기능에 비해 상대적으로 우수한 점은 두 가지이다. OODesigner 코드 생성 기능의 첫 번째 장점은 우선 도구 상에서 메소드의 바디(method body) 부분을 명세할 수 있다는 것이다. 즉 도구 상에서 메소드 바디를 명세하고 그 내용을 포함한 코드를 생성함으로써 단순히 코드 골격(code skeleton)만을 생성하는 것이 아니라 컴파일 후 실행 가능한 완벽한 프로그램을 생성할 수 있다. 본 도구의 두 번째 장점은 특정 언어에 의존적인 문법 구조를 처리할 수 있다는 것이다. 예를 들어 Visual C++의 경우 MESSAGE_MAP, INTERFACE_MAP 과 같이 특이한 문법 구조를 고려한 코드를 생성함으로써 코드 생성 후에 수작업으로 다시 코딩을 하지 않아도 된다.

본 논문은 이러한 기존 연구를 배경으로 본 도구에 C#[3,4] 언어에 대한 코드 생성 기능을 추가로 구현한 연구 결과를 제시한다. MS .NET의 발표 이후에 인터넷 프로그래밍과 컴포넌트 프로그래밍을 위해 C# 언어의 사용이 확산되고 있다. 따라서 본 연구의 성공적인 결과는 C# 언어를 위한 소프트웨어 개발 시에 생산성 향상에 크게 도움이 될 것이다. 본 논문의 2장에서는 시스템의 설계에 대하여 논하며 3장에서는 구현 결과를 제시하고 4장에서는 결론과 함께 차후 연구 방향에 대하여 기술한다.

2. 시스템 설계

본 장에서는 OODesigner의 C# 코드 생성 기능에 대한 설계 내용에 대해 기술한다. 서론에서 언급한 바와 같이 C# 코드 생성 기능은 기존의 C++, Visual

C++, Java 코드 생성 기능에 대하여 추가로 이루어졌기 때문에 본 장에서는 이전 설계 결과와 추가된 설계 결과를 순차적으로 설명한다. 기존의 OODesigner의 구현을 위해 약 200개의 클래스들이 구현되었으며 이들은 MVC(Model View Controller) 패러다임에 의해 설계된바 있다. OODesigner의 설계 문서 중에서 C# 코드 생성 기능 추가 이전에 코드 생성 기능과 관련하여 작성된 설계 결과는 (그림 1)과 같다.



(그림 1) 코드 생성 기능을 위한 클래스 설계

(그림 1)에 나타나있는 클래스들의 기능은 다음과 같다.

- CodeGenerator: 이 클래스는 CObject로부터 상속되며 OODesigner에서 제공되는 모든 코드 생성기의 상위 클래스로서 모든 코드 생성 기능의 구현 시에 공통적으로 필요한 데이터 멤버와 멤버 함수를 정의한다. 이 클래스 제공하는 중요한 공통 기능은 파일에 대한 엑세스, 에러 메시지 창에 대한 엑세스를 제공하는 것이다.

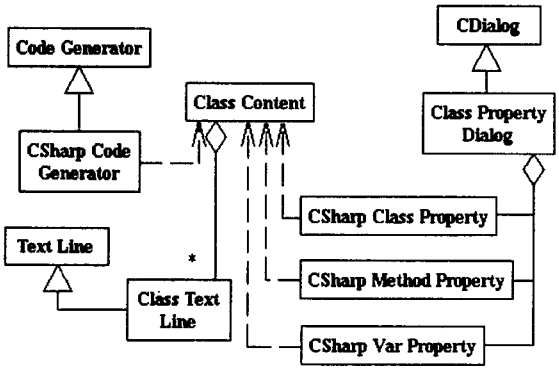
- CPPCommonCodeGenerator: 이 클래스는 CPP CodeGenerator 클래스와 VCPPCodeGenerator 클래스의 상위 클래스로써 C++ 와 Visual C++ 코드 생성 시에 공통적으로 수행할 기능을 제공한다. C++과 Visual C++ 코드 생성 시에는 헤더 파일과 소스 파일을 개별적으로 출력하며 출력되는 양식이 거의 비슷하다. 따라서 주석 문의 출력이라든지 멤버 함수 바디의 출력은 이 클래스에 의해 똑같이 처리된다.

- CPPCodeGenerator: 이 클래스는 CPPCommon-CodeGenerator 클래스로부터 상속되며 일반적인 C++ 코드 생성 기능에 사용된다. 이 클래스에서 이루어지는 출력은 대부분 CPPCommonCodeGenerator에 의하여 처리되므로 구현 함수의 내용은 CPPCommon-CodeGenerator에 의해 override되어 있다.

- VCPPCodeGenerator: 이 클래스는 CPPCommon Code 클래스로부터 상속되며 Visual C++ 코드 생성에 사용된다. 이 클래스가 제공하는 기능은 일반적인 C++ 코드 생성에 대하여 추가로 필요한 작업으로 예를들어 Message Map과 같은 각종 Map에 대한 처리나 클래스 정의에 포함되는 Visual C++에 의존적인 함수나 데이터의 출력에 대한 작업을 수행한다.

- JavaCodeGenerator: 이 클래스는 CodeGenerator로부터 상속되며 Java 코드 생성 기능에 사용된다. 자바 코드 생성 시에 출력은 C++ 경우와 달리 한 개의 파일에 출력되며 클래스 정의에 함수의 바디도 함께 포함되므로 이 클래스는 C++ Code Generator 계통의 클래스들과 별개로 설계되었다.

본 논문에서 구현된 C#코드 생성 기능을 구현하기 위해서 먼저 C# 과 Java, C++ 과의 차이점과 C#언어에서 새롭게 정의된 데이터 타입과 변경자(modifier)들을 조사하였다. C# 코드 생성 기능이 기존의 코드 생성 기능과 이질적인 면이 많이 있었음에도 불구하고 C# 코드 생성 기능 설계 과정에서 기존의 구현되었던 OODesigner의 코드 생성 기능 부분을 재사용 할 수 있었다. 기존에 개발된 코드 생성 기능 부분의 설계문서와 구현코드를 재사용 함으로써 도구 개발이 보다 생산성 있게 이루어 질 수 있었다. C# 코드 생성 기능을 설계하기 위해서 클래스 속성을 편집하기 위한 대화 상자와 관련된 클래스를 추가하고 C# 코드 생성기 클래스를 추가해야했다. (그림 2)는 C# 코드 생성 기능을 위한 클래스 설계 결과이다.



(그림 2) C# 코드 생성 기능을 위한 클래스 설계

(그림 2)의 C# 코드 생성 관련 클래스들의 역할은 다음과 같다.

- ClassContent: 이 클래스는 컨테이너 클래스로서 클래스에 대하여 명시된 모든 멤버들의 정보를 관리한다. 클래스 멤버 각각에 대한 개별적인 정보들은 ClassTextLine 클래스에 저장된다.

- ClassTextLine: 이 클래스는 다이어그램 상에 나타나는 하나의 행을 표시하기 위해 정의된 TextLine으로부터 상속되는 것으로 하나의 데이터 멤버나 멤버 함수 정보를 관리한다. 클래스 멤버의 속성을 명시하기 위한 대화상자에 포함된 컨트롤의 상태 값은 이 클래스의 멤버로 저장된다. 예를 들어 클래스 멤버의 변경자(modifier) 나 가시성(visibility)에 대한 값들이 속성 대화 상자를 통해 이 객체에 저장된다.

- ClassPropertyDialog: 이 클래스는 탭 컨트롤을 포함하는 대화 상자로서 클래스의 속성을 명시하기 위해 사용된다.

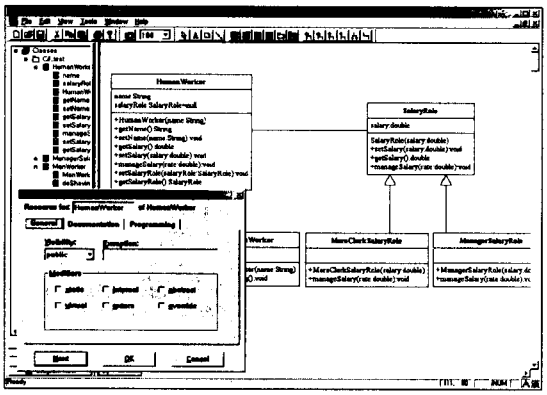
- CSharpClassProperty, CSharpMethodProperty,

CSharpVarProperty: 이 클래스들은 ClassProperty Dialog의 탭을 구성하는 컨트롤들로서 클래스나 멤버의 속성을 명시하기 위해 사용된다.

- CSharpCodeGenerator: CodeGenerator 클래스로부터 상속받으며, ClassContent 객체에 저장된 클래스 내용을 참조하여 C# 코드 생성 기능을 수행한다.

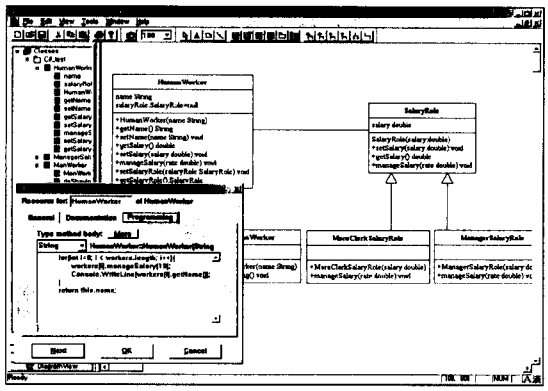
3. 구현 결과

본 장에서는 도구의 구현 결과를 실행 예를 중심으로 기술한다. OODesigner의 C# 코드 생성 기능을 위한 사용자 인터페이스는 기존에 존재하는 상업적인 CASE 도구와 유사하기 때문에 부연 설명없이 제시한다.



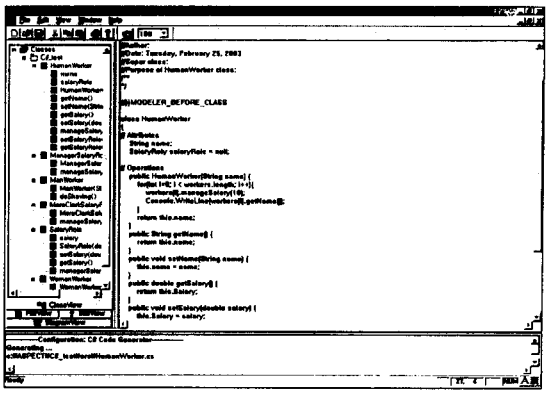
(그림 3) C# 멤버 함수 속성 편집 예

(그림 3)은 클래스 다이어그램에서 C# 멤버 함수의 속성을 편집하기 위한 대화상자의 사용 예를 보여주는 화면이다. 이 탭 컨트롤에서 명시된 C# 멤버 함수의 가시성과 변경자들은 코드 생성 시에 그대로 반영된다.



(그림 4) 메소드 바디 부분 코딩 화면

서론에서 언급한 바와 같이 OODesigner에서는 멤버 함수의 바디를 도구 상에서 코딩할 수 있다. (그림 4)는 C# 멤버 함수의 속성 편집 대화상자에서 코딩을 수행할 수 있는 탭 컨트롤의 사용 예를 보여준다. 이 부분에서 명시된 코딩한 내용들은 코드 생성시 메소드 바디 부분에 출력된다.



(그림 5) C# 코드 생성의 예

(그림 5)는 OODesigner의 코드 생성 기능에 의해 출력된 C# 프로그램이다. 화면에서와 같이 출력된 결과에는 C# 코드 골격뿐 아니라 함수 내용도 포함되어 있다.

4. 결론

본 논문에서는 기존에 구현된 객체 지향 CASE 도구인 OODesigner에 C# 코드 생성 기능을 추가로 구현한 결과를 제시하였다. 본 연구의 이전에 OODesigner는 UML에 속하는 각종 표기법을 위한 다이어그램을 작성할 수 있으며 C++, Visual C++, Java 언어에 대한 코드 생성 기능을 가지고 있었다. MS .NET의 발표 이후에 인터넷 프로그래밍과 컴포넌트 프로그래밍을 위해 C# 언어의 사용이 확산되고 있는 점에서 볼 때 본 연구의 성공적인 결과는 C# 언어기반의 웹서비스 단위생성 시에 생산성 향상에 크게 도움이 될 것이다.

현재 구현 결과물에 존재하는 문제점은 시스템의 안정성에 관한 것이다. 차후의 계속적인 연구를 통하여 시스템의 신뢰성이 확보될 수 있도록 보완하며 본 도구가 좀더 실용적인 CASE 도구가 될 수 있도록 지속적인 노력을 투입할 계획이다.

참고문헌

[1] Martin Fowler, *UML Distilled: Applying the Standard Object Modeling Language*, Addison-Wesley, 1997

[2] Taegyun Kim, Gyusang Shin, "Restructuring OODesigner: A CASE Tool for OMT," Proc. of 20th ICSE, pp. 449-451, April, 1998.

[3] 배재현, *PROFESSIONAL C#*, 정보문화사, 2002.

[4] Microsoft 저, C#나라 역, *Microsoft VISUAL C# .NET LANGUAGE REFERENCE*, 정보문화사, 2002.