효과적인 소스코드 수정 관리를 위한 에디터 기능 제안

성다혜° Scott Uk-Jin Lee
한양대학교 컴퓨터공학과
skyholic03@hanyang.ac.kr, scottlee@hanyang.ac.kr

Editor Feature to Effectively Manage Source Code Modification

Da-Hae Sung° Scott Uk-Jin Lee
Department of Computer Science and Engineering, Hanyang University

요 약

프로그램 소스코드의 수정은 프로그래밍을 배우는 학생부터 중대형 프로젝트를 제작하는 개발자에 이르기 까지 매우 일상적으로 수행해야 하는 작업이다. 또한 소스코드의 수정은 주로 중복 제거, 의도를 파악하기 어려운 함수 이름의 변경, 복잡하고 이해하기 힘든 코드의 가독성 강화가 필요할 때, 오류로 인해 프로그램이 정상동작하지 않을 때, 또는 기능을 보완하거나 추가할 때 등 많은 경우에 이루어진다. 따라서 일상적으로 빈번하게 수행되는 소스코드 수정을 간단하고, 편리하게 수행할 수 있도록 돕는 에디터의 추가 기능을 상세설명과 필요성 검증, 사용자 요구도 파악을 통해 제안하고자 기능에 대한 특성과 성질을 기술하며, 이 기능의 예상 성능과, 예상 사용자들의 반응을 비교, 측정, 분석해본다.

1. 서 론

실행 성능의 안정성과 무결성을 만족하는 하나의 프로그램을 완성하기 위해서는 프로그램의 크기와는 무관하게 빈번한 소스코드의 수정 과정이 수반된다 [1,2]. 프로그램의 반복적인 수정 과정 중에서 소스코드는 처음의 형태를 모두 잃어버릴 수 있으며, 기존에 존재하지 않던 코드가 추가될 수도 있다. 또한, 이 과정으로 인해 알 수 없는 오류가 발생할 가능성도 배제할 수 없다. 하지만 프로그래머의 지각능력만으로 수정중인 코드의 전 후를 모두 기억하고, 고려한다는 것은 불가능에 가까운 일이다. 때문에 코드의 수정 전과 후를 직관적으로 비교가능하게 하고, 필요하다면 다시 수정 전의 코드로 손쉽게 되돌릴 수 있는 기능적 장치가 필요하다.

하지만 현존하는 대부분 에디터의 기능들은 위의 요구조건을 완전히 부합시키지 못하고 있다. 대표적으로, 프로그램의 수정을 도와주는 프로그램인 GIT(DVCS기반 툴)은 웹기반으로 프로그램을 구현할 때의 공동 작업에 초점이 맞춰져 있는 프로그램이며, 레퍼지토리를 개인 PC에 설치하기 위한 과정이 필요하다 [3]. 그리고 프로그램이 동작하는 과정을 이해하고, 상세한 기능을 익혀야 하는 등 사용하기 까지 많은 과정을 거쳐야 하므로 간편하게 사용하기 어려운 단점이 있다. 또한 ClipX와 같은 클립보드 프로그램은 에디터와 독자적으로 구동되어 코드 작성 시 에디터와 ClipX 프로그램 모두를 실행시켜야 한다 [4]. 더불어 수정 전에 사용자 임의로 코드를 저장하는 기능만을 제공하므로 수정중인 코드의 수정과정을 직관적이며 즉각적으로 비교하는 데에 편의를 제공하지 못한다.

이에 따라 앞으로 살펴볼 내용에서는 간편하게 수정할 수 있고, 에디터의 부분 기능으로서 작동할 수 있으며, 에디터에 빌트인(built-in)되어있는 도구로서 수정 전후의 코드를 즉각적, 시각적으로 비교 가능한 기능을 제안하고 살펴보고자 한다.

2. 관련 연구

소스코드를 수정할 때를 대비하여 코드 작성 중이나 업데이트 시 이를 보다 효율적으로 저장하는 방법에 대한 연구들이 다수 진행되었다.

CodingTracker[5]는 기존의 DVC 코드 데이터 저장 방식을 향상시킬 수 있는 새로운 툴을 제안한 것이다. 제안한 툴은 DVC방식의 불완전성과 부정확성 그리고 코드의 수정이 다른 개발활동과 연관되지 못하는 점을 코드 저장범위 및 저장요소의 확대를 통해 개선하였다.

이처럼 기존의 연구는 코드의 수정된 데이터 저장 효율성을 향상시키기 위하여 진행되었다. 그러나 이와 같이 효율성을 높인 저장방식을 사용자들이 보다 직관적이고 원활하게 사용하는 데에 대한 연구나 제안은 미비한 상황이다. 따라서 본 논문에서는 소스코드 수정 시 사용자들의 편리성에 초점을 맞춘 수정 기능을 제안하고자 한다.

3. MSIG : Modification Sign 기능

MSIG (Modification Sign)는 별도의 설치 없이 사용가

능한 에디터의 빌트인 기능으로서 코드의 수정 과정 시에 사용할 수 있는 기능이다. 이 기능은 수정하고자 하는 범위를 지정하여 소스코드를 저장할 때마다 변경 과정들을 즉각적으로 저장하며, 저장된 코드는 에디터 내에서 지정된 범위의 첫 줄 옆에 아이콘으로서 존재한다. 따라서 사용자는 이 아이콘을 선택, 확대함으로서 수정 전의 코드를 즉각적으로 수정중인 코드와 비교할 수 있다. 또한 공통 범위에 중복적으로 다수의 과정 코드를 저장할 수 있도록 한다.

MSIG의 적용 범위는 최소로 하나의 토큰 이상 단위, 최대로는 기능 구현 시 MSIG 기능을 한번 작동시킬 때마다 사용할 수 있도록 정한 최대 저장 용량 이하로 설정하고자 한다. 그리고 프로그램 내에서 내부구조의 기능 단위(method, class, interface)사이에 걸쳐져 있거나, 지정된 범위가 마지막 지정 토큰의 모든 문자열을 포함하지 않는다고 하더라도 이를 허용한다.

MSIG기능의 정의와 적용 범위를 토대로 상세 기능을 살펴보고자 한다. 상세 기능의 종류는 아래와 같이 크게 Sign marking, Show marking, Turn back, Delete marking의 4가지로 구성된다.

1) *Sign marking*: 사용자가 선택하고 싶은 범위의 코드를 지정하고 저장하는 기능을 한다. 선택된 범위의 배경 색깔은 다른 코드의 배경과 차이를 두며, 저장된 코드는 비활성화 상태로 선택된 범위의 첫 토큰 옆에 표시된다.
2) *Show marking*: 평소 비활성화 상태 즉, 표식만을 남긴 상태에서 이를 클릭하게 되면 활성화 상태로 변환된다. 이 상태는 수정 중인 코드와 비교할 수 있도록 저장된 모든 코드가 수정 중인 코드의 옆에 줄을 맞추어 위치하게 되어 직관적인 비교를 가능하게 한다.
3) *Turn back*: 코드의 수정을 완료했을 때 수정된 부분에서 알 수 없는 컴파일 및 런타임 오류 발생 시, 임의로 지정해 놨던 시점의 코드로 돌아갈 수 있게 하는 기능이다.
4) *Delete marking*: 코드의 수정을 모두 완료 한 이후 프로그램이 정상동작 하는 경우, 혹은 이전에 저장한 코드가 더 이상 필요하지 않게 되었을 때 저장했던 코드를 삭제하는 기능이다.

4. MSIG기능의 편리성 검증

MSIG기능은 별도의 학습 없이 사용할 수 있고 조작이 간편하며, 자신이 수정하고 있는 코드에 대한 일련의 수정 과정을 동시에, 시각적으로 검토할 수 있는 편리성을 지닌다. 따라서 이와 같은 편리성을 검증하기 위하여 MSIG기능과 유사하며 일반적으로 많이 사용되는 복사·붙여넣기 기능과 Clipx의 저장기능을 이용한 수정을 비교를 해보고자 한다. 비교는 이 두 가지 기능을 사용하여 MSIG기능의 각 세부 기능에 상응하는 과정을 구현할 때의 복잡성을 살펴봄으로서 수행한다. 각 기능 및 프로그램의 동작 과정은 다음과 같다. (각 실행 과정의 시작점은 프로그램 에디터 창을 켜놓은 상태로 가정한다.)

- **복사·붙여넣기 기능**

첫 번째로 'Sign marking'과 상응하는 과정은 소스코드의 복사, 텍스트 편집기나 다른 코드에디터 창을 띄움, 화면에 코드 붙여넣기, 붙여넣은 내용을 담은 파일을 저장, 기존의 코드로 돌아오는 과정을 거친다. 두 번째로 'Show marking'은 임의로 저장된 파일을 표시한 후 보고자 하는 코드 부분을 검색함으로서 완수된다. 세 번째로 'Turn back'과정의 수행은 임의로 저장된 파일 표시하기, 'turn back'할 코드 부분을 복사하기, 프로그램 에디터 창을 띄움, 'turn back'될 코드 부분 삭제, 'turn back'될 자리에 복사한 코드 붙여넣기, 그리고 변경사항 저장으로 이루어진다. 네 번째로 'Delete marking'과 일치하는 과정은 임의로 저장된 파일이 존재하는 경로 탐색 및 파일 제거를 실행해야 한다.

- **Clipx 프로그램**

첫 번째의 과정인 'Sign marking'은 코드의 복사기능을 선택함으로서 이루어질 수 있다. 다음 'Show marking'기능은 popup창 띄움, 해당 코드 검색 후 클릭, 우측에 생성된 창 내부에 있는 코드를 확인하는 과정을 거쳐야 한다. 세 번째로 'Turn back'기능에 호응하는 결과를 만들기 위해서는 popup창에서 붙여넣기 할 코드('turn back'할 코드)를 Ctrl+Shift+V를 이용하여 선택 복사, 프로그램 에디터 창을 띄움, 'turn back'될 코드 부분 삭제, 'turn back'될 자리에 복사한 코드 붙여넣기, 변경사항 저장하기의 과정을 거쳐야 한다. 마지막으로 'Delete marking'기능은 popup창에서 리스트 삭제함으로서 이루어 질 수 있다.

각 기능을 이용하여 한 범위의 코드를 수정하기 시작해서 마칠 때 까지 사용자가 수행해야 하는 조작의 횟수는 위에서 설명한 실행과정들을 통해 알 수 있으며, 이를 기준으로 세 기능의 효율성을 비교할 수 있다.

표 1. 코드 수정 중 수행해야 하는 조작횟수

도구명 / 실행횟수	MSIG	복사 · 붙여넣기	Clipx
프로그램 설치 필요 유무	X	X	O
저장 가능한 코드 수	n	1	n
기능 창 이동 횟수의 합(횟수)	0	6	5
모든 기능 수행 시 버튼 클릭횟수(횟수)	7	22	8
MSIG표시 대비 창 이동 및 버튼클릭 실행 비율(%)		400%	185.7%

표1을 보고 판단 할 때, 우선 MSIG기능과 복사·붙여넣기 기능에서는 추가로 프로그램을 설치해야할 필요가 없었다. 이는 저장 공간의 필요를 최소화 시킬 수 있으며, 대다수의 프로그래머들이 별도의 검색 없이 간편하게 기능을 이용할 수 있음을 의미한다. 다음으로는 MSIG기능

대비 가장 좋은 성능을 보이는 Cilpx기능조차 이동 및 버튼클릭 실행 비율 비교에서 185.7%이상을 상회하는 결과를 보여준다. 이는 코드를 수정할 때에 창을 이동하며, 코드를 드래그(drag)해야 하는 등의 번거로움을 줄일 수 있으며, 창의 이동 없이 즉각적으로 변경 전 후의 코드를 비교하고, 수정 판단을 내림으로서 프로그래머의 직관적인 판단능력 향상에 도움을 줄 수 있음을 의미한다.

5. 예비 사용자들의 MSIG기능에 대한 요구 및 만족도

MSIG기능은 제작중인 프로그램 내에서 수정하고자 하는 코드 크기에 구애받지 않으며, 수정과정을 직관적으로 판단할 수 있는 상태에서 수정을 진행 할 수 있도록 도와주고, 원하는 결과를 얻을 때 까지 조작해야 하는 과정을 줄여 줌으로서 사용자의 편의를 도모하기 위해 제안되었다. 때문에 이 기능은 무엇보다도 사용자들이 느끼는 필요성과 효용성을 검증받아야 할 필요가 있다.

예비 사용자들에게 MSIG기능의 용이성과 필요성에 대해 설문조사를 진행하였으며, 이 때 예비 사용자들의 정의는 프로그램 에디터를 사용하는 사람으로 한정지었다. 설문조사는 100명을 대상으로 하였으며, 필요성과 용이성, 기존 기능 대비 편리성을 1(매우 그렇지 않다)~5(매우 그렇다)의 범위로 설정했다.

표 2. MSIG기능에 대한 예비 사용자들의 만족도 분포

점수(점) / 평가 기준(명)	1	2	3	4	5	평균점수
용이성	6	12	12	47	23	3.69
필요성	2	9	28	52	19	4.07
기존 기능 대비 편리성	5	4	27	34	30	3.80

표 2는 프로그램 에디터를 이용하는 사용자들의 MSIG기능에 대해 느끼는 용이성과 필요성, 기존 기능 대비 편리성을 조사한 결과이다. 첫 번째로 용이성 부분은 사용자들이 MSIG기능이 수정과정 중에서 도움이 되는 정도를 측정하고자 하였다. 그 결과, 평균 3.69점으로 측정되었으며, 이는 중간점수 이상의 점수로서 대부분의 사용자들이 코드를 보완, 수정할 때 MSIG기능이 향상된 수정 효과를 가져다 줄 것이라 예측한다고 해석된다. 다음으로 필요성의 정도는 MSIG기능을 구현 했을 때, 사용자들이 이 기능을 사용하게 될 빈도수를 측정하고자 의도한 것으로서, 평균 4.07점의 결과를 보였다. 따라서 사용자들은 매우 빈번하게 이 기능을 사용할 것이라고 예측했다는 것을 나타낸다고 볼 수 있다. 마지막으로 기존에 존재하는 기능 대비 편리성 부분에서도 유의미한 결과를 파악할 수 있다. 평균 3.80의 점수를 얻은 이 부분은 복사·붙여넣기, GIT, Clipx등의 기능을 이용할 때보다 더욱 사용하기 쉽고 편리하다는 것을 의미하고 이는 즉, 프로그램 소스코드의 수정 보완적인 측면에서 사용자 중심적으로 만들어진 유용한 기능이라는 점을 피력하는 결과라고 결론내릴 수 있다.

6. 결론 및 향후 연구

본 내용에서는 프로그램 에디터의 소스코드 수정관리 기능으로서 MSIG를 제안하였으며 이 기능의 유용성 및 적용 시 효과를 다각도로 측정하고, 사용자의 요구정도를 파악하여 유의미한 결론을 도출해 내었다. 공유작업과 코드 변화의 기록에 중점을 둔 GIT, 코드의 저장방식을 향상시키기 위한 CodingTracker 및 일반적인 저장, 수정기능인 복사·붙여넣기 기능 등 기존의 연구나 도구는 저장방식에 초점을 두었으며, 변화를 기록한다고 하더라도 직관적으로 수정 진행 중인 코드와 기록되어 있는 코드들의 비교가 어려운 점이 수정 과정에 장애가 되었다. MSIG기능은 Show marking, Turn back을 이용하여 코드의 비교와 수정이 원활하게 진행될 수 있도록 도움으로서 기존의 관련 기능 및 도구의 한계를 개선하였다고 볼 수 있다.

나아가 MSIG기능은 전문 프로그래머들의 코드 수정을 용이하게 하는 것은 물론, 자신이 작성한 코드 수정의 일련과정을 동시적으로 관찰 할 수 있기 때문에 프로그래밍 언어와 설계를 처음 배우고자 하는 학생들도 유용하게 사용할 수 있다. 따라서 개선할 부분과 향상된 능력에 대한 인지가 쉽게 이루어 질 수 있고 또한 이 과정을 교육자와 공유할 수 있으므로 교육자가 학생의 수준과 교육할 부분에 대하여 명확히 파악 할 수 있어, 교육 여건에 대한 개선도 기대할 수 있는 바이다.

향후 연구에서는 MSIG기능을 사용할 때에 편리성을 증대시킬 수 있는 방안을 모색할 것이며, MSIG기능을 여러 에디터의 부분 기능으로 적용시킬 수 있는 환경을 탐색하고 조성하고자 한다.

참 고 문 헌

[1] Roberts, Donald Bradley, and Ralph Johnson. Practical analysis for refactoring. University of Illinois at Urbana-Champaign, p.23, 1999.

[2] Tran, John B., et al. "Architectural repair of open source software." Program Comprehension, 2000. Proceedings. IWPC 2000. 8th International Workshop on. IEEE, 2000.

[3] Dabbish, Laura, et al. "Social coding in GitHub: transparency and collaboration in an open software repository." Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work. ACM, pp.2-4, 2012.

[4] Mulder, Bernard H. "Combined clipboard and file container." U.S. Patent No. D261,903. 17 Nov. 1981.

[5] Negara, Stas, et al. "Is it dangerous to use version control histories to study source code evolution?." ECOOP 2012 - Object-Oriented Programming. Springer Berlin Heidelberg, 2012. 79-103.