국방 소프트웨어 자원관리시스템의 소스 코드 품질 향상을 위한 소프트웨어 가시화 적용 구축 사례

박보경○, 문소영, 서채연, 김영철
홍익대학교 소프트웨어 공학 연구실
{park, msy, chyun, bob}@selab.hongik.ac.kr

A Constructive Practice of Software Visualization Mechanism for Code Quality Improvement on the Army's Software Resource Management System

Bo Kyung Park○, So-Young Moon, Chae Yun Seo, R. Young Chul Kim
SE Lab, Dept. of Computer Information Communication, Hongik University

요 약

기존 육군에서는 고품질의 소프트웨어 개발을 위한 품질 관리 방법 적용에 함축적인 어려움을 겪고 있었다. 현재는 꾸준한 자체 노력 진행으로 군 지원 관련 소프트웨어의 품질 향상 중이다. 본 논문의 제안하는 방안은 적절한 공개 소프트웨어로 개발 단계의 코드 가시화 구현이다. 이런 가시화 기법을 국방 소프트웨어 자원관리시스템에 적용하였다. 즉, 코드의 내부 구조 및 내부 품질 지표들을 가시화함으로써, 나쁜 코드 영역의 리팩토링을 통해 설계 품질과 코드를 개선시켰다. 이 방법을 통해, 고품질 소프트웨어 개발 및 유지보수성을 향상이 기대된다.

1. 서 론

현재, 육군에서 필요한 소프트웨어 개발 요구는 신규 개발과 기능 개선을 포함하여 지속적으로 증가하고 있다. 소프트웨어 중요성이 증가함에 따라, 다양한 소프트웨어 개발 요구사항 및 고품질의 소프트웨어가 요구되고 있다. 이에 육군에서는 고품질의 소프트웨어 개발을 위한 품질관리 방법을 적용하는데 많은 어려움을 겪고 있다. 육군에서 분석한 문제점으로는 1) 1인 1건의 개발 추진으로 개발자 성향에 따라 개발 패턴이 다르기 때문에 복잡도의 일관성이 부족하다. 2) 요구사항이 지속적으로 변경됨에 따라 요구 분석, 설계, 구현의 전 과정이 일치하지 않는다. 3) 개발 후 운용 지원 부서로 개발업무가 이관되기 때문에, 운용 유지 측면에서 다양한 개발 패턴을 유형별로 분석하고 조치하는데 많은 시간이 소요된다. 이러한 문제를 해결하기 위해서는 소프트웨어의 비가시성 문제 해결이 가능한 소프트웨어 가시화 방법이 필요하다. 소프트웨어의 가시화는 소프트웨어 개발의 전 과정을 파악할 수 있다. 그리고 소프트웨어 개발의 과정과 구조를 파악함으로써 전문적이지 않더라도 고품질을 위한 소프트웨어 개발과 품질 관리가 가능하다[1,2]. 본 논문은 소스 코드의 가시화(SW Visualization) 기법을 적용하여 국방 소프트웨어의 코드 내부 구조를 가시화하고, 결합도와 응집도 기반의 소프트웨어 품질을 개선하고자 한다. 이를 위해, 오픈 소스 도구들을 활용하였으며, 기존 코드의 시각화된 결과물을 도출할 수 있었다. 또한 결합도 측정 모듈과 품질 지표를 정의하고 리팩토링을 수행함으로써 소프트웨어 품질 개선이 가능하였다. 이 방법을 통해 고품질 소프트웨어 개발 및 유지보수성 향상을 기대할 수 있다. 본 논문의 구성은 다음과 같다. 2장에서는 소프트웨어 가시화 방법 및 소스 네비게이터에 대해 소개한다. 3장에서는 소스 코드 품질 향상을 위한 소프트웨어 가시화 적용 방법에 대해 설명한다. 마지막으로 4장은 결론 및 향후 연구를 언급한다.

2. 관련 연구

현재의 소프트웨어 개발은 개발 프로세스 및 품질 관리가 매우 중요하다. 기존의 소프트웨어 개발은 개발자나 설계의 부재로 많은 문제가 발생했으며, 전문화된 인력과 자본이 부족하다. 소프트웨어 가시화는 소프트웨어의 비가시성 문제를 해결할 수 있기 때문에, 소프트웨어 개발 전 과정을 파악할 수 있다. 또한 소프트웨어 구조 파악이 가능하기 때문에 전문적이지 않더라고 고품질의 소프트웨어 개발과 품질 관리가 가능하다[1]. 소프트웨어 아키텍처를 복원하는 Tool-Chain 프로세스는 소스 코드로부터 정보를 추출하여 의미 있는 단위로 구분하고, 이를 데이터베이스에 저장한다. 저장된 데이터를 기반으로 시각화 도구를 사용하여 그래프 형태로 소프트웨어 아키텍처를 나타낸다[1,2]. Tool-Chain은 코드 분석, 정보 저장 및 분류, 시각화의 서로 독립적인 기능을 수행한다.

3. 국방 소프트웨어의 소스 코드 품질 향상을 위한 소프트웨어 가시화

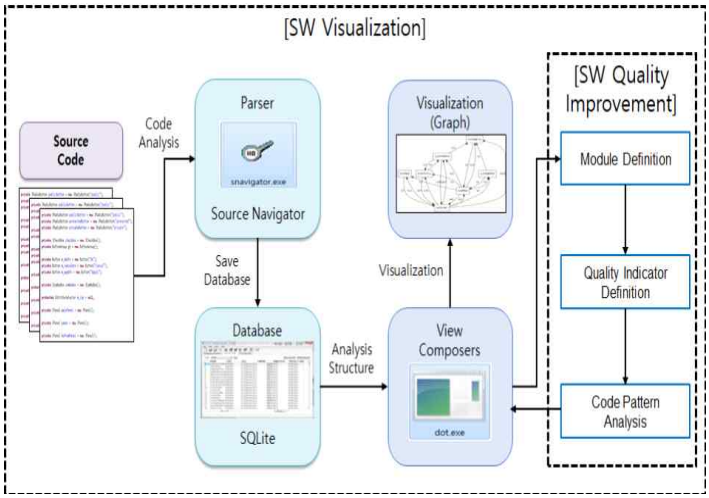

그림 1. 소프트웨어 가시화 방법[2]

그림 1은 소스 코드 품질 향상을 위한 소프트웨어 가시화 방법이다. 소프트웨어 가시화는 코드 분석, 분석된 정보 저장, 소스 내부 구조 분석, 가시화 단계로 구성된다. 구조 분석에서는 소프트웨어의 품질 개선을 위해 모듈 정의, 품질 지표 정의, 코드 패턴 분석을 수행한다.

코드 분석 단계는 파서를 통해 소스 코드를 분석한다. 파서는 공개 소프트웨어인 소스 네비게이터(Source Navigator)를 사용하였다. 소스 네비게이터는 소스 코드를 분석하여 전체 프로그램의 구조와 클래스, 함수, 메소드 등의 정보를 쉽게 확인할 수 있는 정적 분석도구이다[3]. 소스 코드에서 클래스, 기능 및 멤버들 간의 관계식별 및 호출 트리 확인이 가능하기 때문에 소스 코드의 전반적인 이해를 돕는다. 소스 네비게이터는 소스 코드의 경로를 입력하면, 클래스, 메소드, 변수 정보와 같은 소스코드의 정보를 파싱한다. 생성된 파일(SNDB)은 바이러리로 구성되며, DBdump를 통해 바이너리 데이터를 텍스트로 변환하여 필요한 정보를 추출한다[6]. 그림 2는 소스 네비게이터의 소스 코드 분석 과정이다.

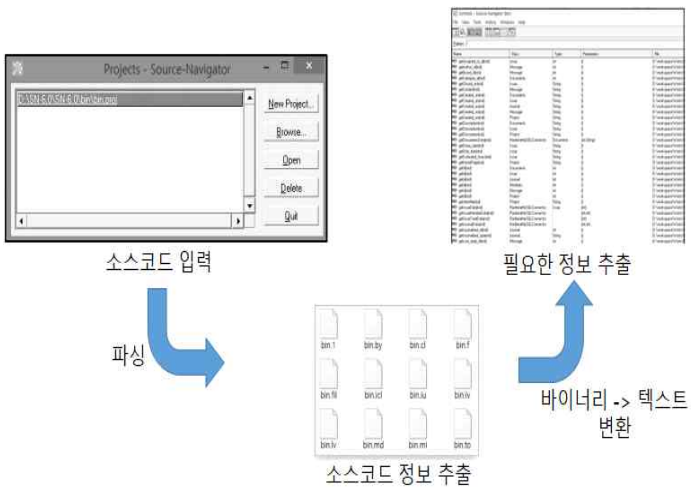

그림 2. 소스 네비게이터의 소스 코드 분석 과정

데이터베이스 저장 단계는 소스 네비게이터로 분석된 코드 정보를 데이터베이스에 저장한다. 데이터베이스는 공개 소프트웨어인 SQLite를 사용하였다. SQLite로 선정한 이유는 다른 데이터베이스 관리 시스템에 비해 비교적 가볍다. 또한 다른 DB에 비해 대규모 작업에는 적합하지 않지만, 중소 규모의 작업에는 적합하다. 소스 네비게이터로 추출된 파일(SNDB)은 바이너리로 구성되어 있기 때문에 DBdump 파일로 변환하여 데이터베이스에 저장한다. DBdump 파일을 이용하여, cl, mi, by 등의 파일 내용을 추출하고 분류한다.

구조 분석 단계는 소프트웨어 품질 측정을 위해 모듈 정의, 품질 지표 정의, 코드 패턴 분석을 수행한다. 즉, 결합도(Coupling)와 응집도(Cohesion)를 기반으로 소프트웨어의 품질을 측정한다. 또한 소프트웨어의 품질을 개선하기 위해서, 반복적인 리팩토링을 수행한다[7]. 품질 측정 방법은 다음과 같다.

1) 입력된 소스 코드에 적합한 모듈 단위를 정의한다. 본 논문에서는 Java 코드를 분석하기 때문에 객체를 추상화한 클래스(Class)를 모듈로 정의하였다.
2) 품질 지표 정의 단계에서는 결합도와 응집도를 기반으로 정량적인 측정 지표를 정의한다. 결합도를 사용하는 이유는 소프트웨어를 설계할 때, 모듈 간의 결합도를 최소화하고 응집도를 높이면 고품질의 소프트웨어 개발이 가능하기 때문이다. 따라서 이 단계에서는 결합도와 응집도를 정의하고, 측정지표를 정의한다. 결합도는 내용결합도로 갈수록 점수가 높아지며, 응집도는 기능적 응집도로 갈수록 점수가 높아진다.
3) 코드 패턴 분석에서는 코드 패턴을 검출한다. 각 결합도 및 응집도에 대한 코드 패턴을 정의한다. 정의된 코드들은 리팩토링을 수행할 때, 타겟 코드에서 검출하게 된다. 검출된 결과를 분석하여, 낮은 결합도 및 높은 응집도를 위해, 코드를 수정한다.

시각화 단계에서는 분석된 정보들을 기반으로 코드를 시각화한다. 코드의 시각화를 위해서는 이전 단계에서 분석된 모듈, 모듈간의 관계, 품질지표 정보들을 이용하여 그래프로 출력한다. 이 그래프는 공개 소프트웨어인 GraphViz를 사용하였다. 이 도구는 DOT 언어 스크립트로 지정된 그래프를 생성이 가능한 오픈 소스 도구 패키지이다.

4. 사례 연구

본 연구에서는 국방 소프트웨어 자원관리시스템에 소프트웨어 가시화 기법을 적용하였다. 그림 4는 소프트웨어 가시화 결과이다. 자원관리시스템의 가시화 결과는 군의 보안목적 상 공개할 수 없기 때문에 다른 사례로 대체하였다. 이 그래프는 자원관리시스템에 적용된 소프트웨어 가시화 기법으로 출력된 가시화 결과이다. 그림에서 네모는 클래스를 나타내며, 클래스와 클래스를 연결하는 화살표는 클래스 간의 결합도를 의미한다. 예를 들어 ① CClassView와 ② CClassEventHandler 간의 결합도 값은 36으로, 다른 결합도 수치보다 높다. 또한 ① CClassView와 ③ CClassDocument 간의 결합도 값은 28

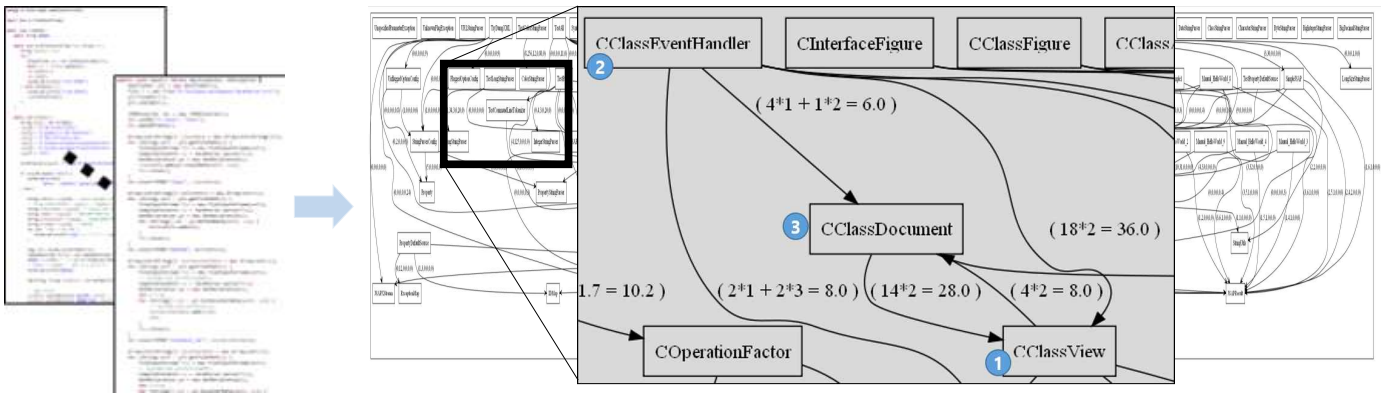

그림 4. 소프트웨어 가시화 결과

이다. 이는 많은 코드 수정이 필요함을 의미한다.

표 1은 기존에 구축된 시스템과 소프트웨어 가시화 구축을 통한 비교 결과이다. 기존 시스템에서는 산출물에 대한 검증이 제한되었으나 소프트웨어 가시화를 통해 역공학을 수행함으로써 검증이 가능하였다. 기존에 구축된 시스템에서는 소스 코드 가시화 체계가 구축되어 있지 않아 코드의 내부 구조 확인이 어려웠다. 개발된 시스템에서는 국방 소프트웨어 자원관리 시스템에 소프트웨어 가시화 기법을 적용하여 소프트웨어의 품질 개선이 가능하였다. 기존에는 개발일정을 수작업으로 관리하였으나, 소프트웨어 가시화 구축을 통해 개발일정의 체계적인 관리가 가능하였다. 기존 시스템에서는 개발자의 잘못된 코드 작성 습관 식별이 불가능하였으나, 소프트웨어 가시화 방법을 적용하여, 개발자의 Bad Smell을 식별하였다. 기존 시스템에서는 소프트웨어의 오류나 요구사항 변경이 발생했을 때, 소스 코드를 수정하였다. 하지만 개발된 시스템은 소스 코드의 가시화가 가능하기 때문에 복잡도 측정을 통한 코드의 품질 개선이 가능하다. 소프트웨어 가시화는 품질 지표를 선정하고 지표를 정량화하여 소스 코드의 복잡도를 측정한다. 따라서 측정된 복잡도를 이용하여, 기존 소스 코드의 개선이 가능하다.

표 1. 기존 시스템과 개발된 시스템의 비교 결과

항목	기존 시스템	개발된 시스템
산출물 검증	부분 가능	가능
소스 코드 가시화	불가	가능
개발일정 관리	수작업	자동화
개발자의 Bad Smell 식별	불가	가능
복잡도 측정	불가	가능
소스코드 품질 개선	미비	가능

5. 결 론

본 논문은 국방 소프트웨어의 소스 코드 품질 향상을 위해 소프트웨어 가시화 기법을 적용하였다. 기존에 구축된 시스템은 개발자의 성향에 따라 개발 패턴이 다르기 때문에 복잡도의 일관성이 부족했다. 또한 요구사항이 변경되거나 추가되면, 소프트웨어 개발 전 과정에서 불일치 문제가 발생하였다. 소프트웨어 가시화 기법을 적용함으로써, 기존 시스템의 문제점을 보완할 수 있었다. 소프트웨어 가시화는 코드의 정량적 분석이 가능하기 때문에 개발한 코드의 복잡도 문제를 확인할 수 있다. 이를 통해 개발자의 나쁜 습관을 개선하고, 올바른 소프트웨어 개발 및 유지보수성을 향상시킬 수 있을 것으로 기대한다. 향후에는 소프트웨어 프로세스 가시화 기법을 국방 소프트웨어에 적용하고자 한다.

ACKNOWLEDGEMENT

이 논문은 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (NRF-2017R1D1A3B03035421).

참 고 문 헌

[1] NIPA SW공학센터, “SW개발 품질관리 매뉴얼(SW Visualization)“, 2013. 12.

[2] 박보경, 권하은, 손현승, 김영수, 이상은, 김영철, “소프트웨어 가시화를 통한 품질 개선 사례 연구”, 한국정보과학회논문지, Vol.41, No.11, pp.935-942, 2014.11

[3] Source Navigator NG, http://sourcenav.sourceforge.net/

[4] https://docs.oracle.com/javase/tutorial/essential/regex/

[5] 신야 료마, 스즈키 유스케, 타카타 켄, “다양한 언어로 배우는 정규표현식”, 제이펍, 2016

[6] 박지훈, 장우성, 이진협, 손현승, 김영철, “확장된 나쁜 코딩 습관 식별 구현”, 한국정보과학회, pp.401-403, 2016.12

[7] 박지훈, 손현승, 박보경, 이진협, 김영철, “효율적인 리팩토링을 위한 조직 특성의 품질지표 기반 우선순위 선정 자동화”, 소프트웨어공학회, pp.175-178, 2017.02

한국정보과학회
KOREAN INSTITUTE OF INFORMATION SCIENTISTS AND ENGINEERS
http://www.kiise.or.kr

2017 Korea Software Congress

Date
2017. 12. 20(수)~22(금)

Venue
부산 벡스코 컨벤션홀

Theme
소프트웨어, 4차 산업혁명의 열쇠

주요행사

12.20

Keynote–Editor in Chief Hamido Fujita(Elsevier)
Invited Talk–허재혁 교수(KAIST)
배헌섭 사장(슈어소프트테크)
김상현 부사장(한국오라클)
특별세션–신진교수 최신연구소개
데이터베이스 및 데이터 마이닝 최신기술 워크샵
지능형 사물인터넷 기반 융합 서비스 기술 동향
머신러닝연구회 동계 워크샵
SW+City: Smart City Reloaded
국제표준 기반 오픈 데이터 유통 플랫폼 확장 기술 개발 워크샵
우수 국제학술회의 목록 갱신 토론회

12.21

Keynote–손상혁 교수(DGIST)
President Hironori Kasahara(2018 IEEE Computer Society)
Invited Talk–하정우 리더(네이버)
전진옥 사장(비트컴퓨터)
김태훈 본부장(KIAPS)
김윤재 과장(기상청)
Dr. Kohtaro Asai(Mitsubishi Electric Corporation)
특별세션–신진교수 최신연구소개
SW 구현/데모 경진대회(대학원생)
군집지능(SAAL)플랫폼 요구 분석 워크샵
인공지능 국가전략프로젝트 개방형 평가
비디오튜링테스트(VTT)를 위한 비디오 이해 딥러닝기술 워크샵

12.22

SW 구현/데모 경진대회(학부생)
학부생주니어논문경진대회

Sponsored by

골 드 SAMSUNG 삼성SDS

실 버 MANYCORESOFT IBM SK telecom ORACLE kt 한화시스템

브론즈 bto 부산관광공사 BUSAN TOURISM ORGANIZATION BIT 비트컴퓨터 Suresoft 싸인텔레콤 www.signtelecom.com woojoo telecom 웨이버스 WAVUS WINITECH hanscom SINCE 1991 주식회사 한스컴정보통신

세션	시간	분야	장소	좌장
21A2	10:30-12:30	전산교육시스템	101호	조정원(제주대)

21A2-1	걸음걸이 관리를 위한 웨어러블 시스템 개발 고란희·강윤수·조정원(제주대), 이경희, 오홍식
21A2-2	한국지리 백지도 학습 및 분석 SW 개발 양승헌 · 김지영 · 조정원(제주대), 이경희, 김태호
21A2-3	다 측면 코딩 자동 평가 및 피드백 시스템 박형준(동국대), 임창신(동국대), 박시현(동국대), 홍지영(동국대), 박미화(동국대)
21A2-4	로봇 저널리즘 기반 사용자 선호 맞춤형 스포츠 기사작성 프로그램 진민구· 양준호 · 조정원(제주대) · 이경희 · 오홍식
21A2-5	e-Learning 콘텐츠의 학습 형식이 학습자의 몰입도에 미치는 영향 유인환(동국대)

세션	시간	분야	장소	좌장
21B1B2	09:00-12:30	언어공학I	102호	옥철영(울산대)

21B1B2-1	음절 단위 태그 분포와 멀티 태스크 학습 기반 포인터 네트워크를 이용한 한국어 의존 구문 분석 안재현·고영중(동아대)
21B1B2-2	Dynamic Memory Network를 이용한 End-to-End 레스토랑 예약 대화 시스템 신창욱 · 차정원(창원대)
21B1B2-3	도메인의 통계정보를 이용한 감성분석 이정훈, 김민호, 권혁철(부산대)
21B1B2-4	주제 일치도 기반의 N-gram 문장 정보를 이용한 경제 분야 문장 분류 유홍연 · 고영중(동아대)
21B1B2-5	언어학적 패턴을 이용한 소셜 미디어 사용자의 우울증 증세 예측 송호윤, 박한철, 양원석, 박종철
21B1B2-6	질의응답에서 정답후보와 어휘정답유형 사이의 의미적 연관성 추론을 위한 지식베이스 구축 양성훈(부산외대), 정예지(부산외대), 류법모(부산외대)
21B1B2-7	개체명 인식 성능 향상을 위한 배깅 기반의 부트스트래핑 기법 정유진 · 김주애(서강대), 고영중(동아대), 서정연(서강대)
21B1B2-8	Attention 기반 한국어 의미역 결정 박광현(전북대), 나승훈(전북대)
21B1B2-9	전이 학습을 이용한 특정 도메인의 신경망 대화 모델 최수정, 신정완, 정이안, 박세영, 박성배

세션	시간	분야	장소	좌장
21D3D4	14:00-17:30	국방소프트웨어	104호	진현욱(건국대)

21D3D4-1	실질적 DIS-HLA 통합 사례 및 의미론적 접근 방법 김서향, 박준현, 김종권 (서울대)
21D3D4-2	고품질 소프트웨어를 위한 군인력 자원관리 개발 프로세스 가시화 구축 사례 장우성(홍익대), 손현승(모아소프트), R.Young Chul Kim(홍익대)
21D3D4-3	무인기 비행조종컴퓨터 OFP용 소프트웨어시험환경 개발 및 평가 한동건(국방과학연구소),김연균(국방과학연구소),허종수(국방과학연구소),윤형식(국방과학연구소)
21D3D4-4	북한 운영체제 붉은별(Red Star)의 보안 취약점 분석 박기훈, 강동수(국방대학교)
21D3D4-5	무기체계 내장형 소프트웨어 보안약점 식별 방법론 장희진, 김진국, 정승훈, 김희동, 김현숙
21D3D4-6	비인가 AP 탐지와 로그 분석을 이용한 실시간 공격 식별 시스템 개발 김도연(세종대),김두회(세종대),김용현(국방과학연구소),김동화(국방과학연구소),신동규(세종대),신동일(세종대)
21D3D4-7	기계 학습을 적용한 사이버 자산의 중요도 산출과 보호 방안 윤현수(세종대), 김용현 · 김동화(국방과학연구소), 신동규 · 신동일(세종대)
21D3D4-8	SystemC UART 시뮬레이터 구현 노정민(카이스트), 한욱현(카이스트), 이길호(카이스트), 신인식(카이스트)
21D3D4-9	국방 소프트웨어 자원관리시스템의 소스 코드 품질 향상을 위한 소프트웨어 가시화 적용 구축 사례 박보경, 문소영, 서채연, 김영철