한국정보과학회
KOREAN INSTITUTE OF INFORMATION SCIENTISTS AND ENGINEERS

제 22 권 제 1 호
Vol. 22 No. 1

SOFTWARE ENGINEERING SOCIETY

KIPS 소프트웨어공학소사이어티

2020

제 22 회 한국 소프트웨어공학 학술대회 논문집

Proceedings of the 22nd Korea Conference on Software Engineering (KCSE 2020)

- 일시: 2020년 2월 3일(월) ~ 2월 5일(수)
- 장소: 강원도 평창 한화리조트(휘닉스파크점)

주최: 한국정보과학회, 한국정보처리학회
주관: 한국정보과학회 소프트웨어공학 소사이어티
한국정보처리학회 소프트웨어공학연구회

후원: ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute, SOLUTIONLINK ㈜비트컴퓨터, ㈜모아소프트, ㈜이에스지, 한국소프트웨어기술진흥협회(KOSTA), T3Q㈜, ㈜다한테크, 슈어소프트테크㈜, ㈜온페이스, STA 테스팅컨설팅㈜, TTA 소프트웨어시험인증연구소, ㈜에스피아이디, 신뢰적지능형 CPS 연구단

2 월 4 일 (화)				
시 간	**행 사 내 용**			
	논문 발표 B			
	B1: SW 안전 좌장: 서영석(영남대) 장소: 세미나실 1	**B2: 스마트 시스템 I** 좌장: 이지현(전북대) 장소: 세미나실 2	**B3: 인공지능 II (AI for SE)** 좌장: 남재창(한동대) 장소: 세미나실 3	**B4: SW 분석** 좌장: 홍장의(충북대) 장소: 그랜드홀 2
11:10-12:30 (80 분)	소프트웨어 위험원 분석을 위한 상황 분류 기반의 조합을 통한 STPA 문맥표 최적화 [일반논문] 양현수, 권기현(경기대) 결함 트리와 마코프 모델을 이용한 안전 기능의 정량적 검증 **[최우수 일반논문]** Ngoc-Tung La, 김소연, 권기현(경기대) 기능 안전 표준 기반의 무기체계 소프트웨어 개발 및 관리 매뉴얼 문제점 분석 및 개선 방안 제시 **[우수 산업체 논문]** 김태현, 박다운, 백옥현(국방과학연구소) STPA 를 이용한 군집 운행 시스템의 안전성 분석 사례 연구 [단편논문] 김의섭, 유준범(건국대)	Effect-driven Dynamic Selection of Physical Media for Visual IoT Services using Reinforcement Learning **[초청논문 ICWS'19]** K. Baek, I-Y. Ko 데이터베이스 서버 그룹 관리를 위한 EMS [산업체 논문] 김효일(KT), 백종문(KAIST) 트리거-액션 기반 서비스 매쉬업의 신뢰도를 개선하기 위한 전제 상태 및 목표 상태 검증 방법 [단편논문] 김상훈, 고인영(KAIST) BERT 기반의 SNS 메시지 불안 분류기 및 사회적 불안 분포 시각화 시스템 [단편논문] 송지수, 최용석(한양대)	소프트웨어 결함 예측의 조선해양/해상운송 산업 적용 사례 연구 **[우수 일반논문]** 강종구(KAIST), 류덕산(전북대), 백종문(KAIST) 순환 인공 신경망을 활용한 코드 변경 추천 시스템의 학습 시간 단축 방법 연구 **[우수 일반논문]** 배병일, 강성원(KAIST), 이선아(경상대) 딥러닝 기반 버그 추적 기법의 OOV 단어 처리를 위한 형태 정보와 문맥 정보 통합 [단편논문] 김영경, 김미수, 이은석(성균관대) 이슈 보고서와 사용자 매뉴얼 간의 추적성 수립을 위한 머신러닝 기법들의 비교 [단편논문] 조희태, 박도제, 이선아(경상대)	Precise Learn-to-Rank Fault Localization using Dynamic and Static Features of Target Programs **[초청논문 TOSEM]** Y. Kim, S. Mun, S. Yoo, M. Kim 행위 모델 기반 RESTful 웹 어플리케이션 결함 위치 추정 **[우수 일반논문]** 장종인, 이낙원(KAIST), 류덕산(전북대), 백종문(KAIST) 조치 가능한 버그 예측을 위한 유사 패치 추천 **[우수 단편 논문]** 신지호, 남재창(한동대) Python 코드를 위한 정적 분석기 개발 [단편 논문] 홍제성, 박보경, 장우성, 이원영, 정세준, 김영철(홍익대)
12:30-13:40	**중식**			

	논문 발표 C			
	C1: 학부생 논문 I 좌장: 서영석(영남대) 장소: 세미나실 1	**C2: 학부생 논문 II** 좌장: 홍신(한동대) 장소: 세미나실 2	**C3: 블록체인** 좌장: 김순태(전북대) 장소: 세미나실 3	**C4: SW 테스팅 I** 좌장: 백종문(KAIST) 장소: 그랜드홀 2
13:40-15:40 (120 분)	CNN 모델 평가를 위한 이미지 데이터 증강 도구 개발 **[우수 학부생 논문]** 최영원, 이영우, 채흥석(부산대) 소프트웨어 소스 코드의 다중 카테고리 분류를 위한 딥러닝 기반 기법 김민하, 심규진, 이찬근(중앙대) YOLO 모델을 활용한 영상 내의 유해 정보 검출 및 필터링 시스템 개발 조성윤, 권용준, 김채록, 최지원, 김석호, 최나람, 김민석, 정순기(경북대) 텍스트 마이닝과 TF-IDF 유사도 가중치를 이용한 연관 아이템 발견 김민정, 김주창, 박성수, 신동훈(경기대), 정호일(대림대), 정경용(경기대) 문서 유형에 따른 분류를 이용한 마이닝 기반 토픽 추출 구병국, 최소영, 강지수, 백지원(경기대), 박찬홍(상지대), 정경용(경기대) AI Impact on Software Engineering [후원 산업체 발표] 김동영(한국 IBM)	블록체인 기반 CCTV 영상정보 무결성 지원 방안 강민구, 홍준기, 송성한, 김태영, 김순태(전북대) STEAM 교육과 게임을 도입한 EPL 기본 설계 **[우수 학부생 논문]** 이정희, 김민우, 조성휘, 김정아(가톨릭관동대) 도커시스템에서 사용 가능한 모니터링 도구 비교 분석 정채은, 박용범(단국대) 미들웨어 기반 빅데이터 시각화 시스템 아키텍처 설계 백재형, 송진범, 서상원, 남재창(한동대) AI 와 SE 를 활용한 수화 번역 프로젝트 이유렬, 김은진, 이혁진, 이선아(경상대) CCTV 를 이용한 실시간 폭행 및 난동행위 인식 시스템 개발 권용휘, 전승원, 배재빈, 김성윤, 김석호, 정순기(경북대)	이더리움 스마트 컨트랙트 상태 모니터링 시스템의 설계 및 구현 **[초청논문 JSES]** 홍준기, 김순태, 류덕산 실시간 인식 기반의 제품의 표시 정보 추출을 위한 모바일 어플리케이션 설계 및 구현 **[초청논문 JSES]** 민경식, 최지수, 이철훈, 정동주, 이병정 계약 관련 음성 녹취 정보의 신뢰성 있는 저장과 체계적 관리를 위한 블록체인 기반 시스템 구성 방식 [단편 논문] 김인근, 서중원, Alshiri Saad, 장민오, 이유정, 박수용(서강대) 그룹 리더 선출을 통한 비트코인 병렬 트랜잭션 처리 [단편 논문] 이도현, 송아람, 이선재, 박우람, 박수용(서강대) 블록체인 코드의 복잡도 측정을 위한 코드 가시화 [단편 논문] 안현식(홍익대), 박지훈, 김기두(TTA), 박보경, 조재형, Md Ibrahim Khalil(홍익대), 김동호(데이터메트릭스), 김영철(홍익대) 블록체인 기반 실시간 수하물 트레킹 시스템 [단편 논문] 이열국, 이동현, 박수용(서강대)	Concolic Testing for High Test Coverage and Reduced Human Effort in Automotive Industry **[초청논문 ICSE '19]** Y. Kim, D. Lee, J. Baek, M. Kim Javadoc 대상 테스트 요구사항 추출 기법의 정확도 평가 [단편 논문] 김지웅, 홍신(한동대) Enhancing Patch Validation in APR by Introducing Test Case Prioritization and Sampling [단편 논문] Y. Venugopal, Q-N. Phung, E. Lee(성균관대) 무인이동체 소프트웨어 시험 평가 기준 및 절차 연구 [단편 논문] 장정훈(모아소프트) DO-178C 기반의 항공 SW Safety 검증 프로세스 [후원 산업체 발표] 강유선(모아소프트)
15:40-16:00	**휴식**			
	기조강연 II 장소: 그랜드홀 2			사회: 이병정 대회장 (서울시립대)
16:00-16:50	인공지능과 인간의 공존, 그리고 포스트 휴머니즘 이중원 교수 (서울시립대, 한국철학회 차기회장)			
	기조강연 III 장소: 그랜드홀 2			사회: 김정아 대회장 (가톨릭관동대)
16:50-17:40	Build AI-Powered Applications through Data driven Innovation 최창남 부문장 (한국 오라클)			

블록체인 코드의 복잡도 측정을 위한 코드 가시화

안현식[1], 박지훈[2], 김기두[2*], 박보경[1], 조재형[1], Md Ibrahim Khalil[1], 김동호[3], 김영철[1*]
홍익대학교 소프트웨어공학연구실[1], 한국정보통신기술협회(TTA)[2],
데이터메트릭스(Datametrex)[3]
{ahn[1], park[1], cho[1], Ibrahim[1], bob[1*]}@selab.hongik.ac.kr

Code Visualization for Measuring Complexity of Blockchain Code

Hyun-sik Ahn[1], Jihoon Park[2], Kidu Kim[2*], Bokyung Park[1], Jae Hyeoung Cho[1], Md Ibrahim Khalil[1], Stephen D. Kim[3], R. YoungChul Kim[1*]
SE Lab[1], Hongik University
TTA[2], Datametrex[3]

요 약

현재 4차 산업 혁명 시대에 다양한 영역의 많은 소프트웨어에대해 고품질 고려가 미비하다. 특히 블록체인 관련 소프트웨어 품질이 중요한 이슈가 될 것이다. 기존의 소프트웨어 개발보다 더 많은 품질 요소(병렬, 동시화, 성능, 전력)가 필요하다. 또한 새롭게 등장한 Go 기반 블록체인 코드에 대한 분석기의 부재와 모듈 내 두 리턴 값으로 내부 코드의 복잡도가 매우 높다. 이를 해결하기 위해 Go기반 코드 정적분석기를 제안한다. 이를 기반으로 코드 내부 복잡도 추출 및 가시화하여 복잡도 개선 방법을 제안한다. 이런 코드 가시화 방법은 보다 쉽게 코드를 유지보수 할 수 있으며 이는 블록체인 시스템의 소프트웨어 고품질화가 가능하게 된다.

1. 서 론

최근 4차 산업혁명이 가장 큰 이슈로 떠오르면서 AI, 자율주행, 블록체인 등 새롭고 다양한 시스템들이 등장하고 있다. 그 중 수많은 블록체인 기반 플랫폼들이 개발되고 있지만, 블록체인 소프트웨어의 신뢰성, 가용성, 데이터 무결성 확보 등에 대한 검증은 누구도 하고 있지 않다. 이처럼 고품질 소프트웨어와 거리가 먼 구현 위주의 개발을 하게 되면 시스템 전체의 복잡도가 높아지고 유지보수가 어렵게 된다. 복잡도가 지나치게 높다면 전체적인 소프트웨어의 아키텍처를 파악해 복잡도를 줄이고 모듈간의 결합도를 줄여 모듈간의 독립성을 만들어야 한다. 이렇게 소프트웨어 공학적 접근을 통한 아키텍처가 가시화됨으로써 전체 시스템의 복잡도를 파악할 수 있고 모듈간의 결합도를 판단할 수 있다.

본 연구에서는 블록체인 코드의 정적 분석을 통해 복잡도, 결합도를 가시화하는 방법을 제안한다. 2장에서는 관련연구로 아키텍처 가시화와 결합도, Go AST Viewer에 대해 설명한다. 3장에서는 Go language로 구현된 소프트웨어 가시화 과정에 대해 설명하고 4장에서는 결합도 측정 방법에 대해서 설명한다. 5장에서는 실제 프로그램에 대한 적용사례를 보여준다. 6장에서는 결론 및 향후 연구에 대해서 언급한다.

2. 관련연구

2.1 아키텍처 가시화

소프트웨어 아키텍처 가시화는 소프트웨어를 개발하는데 가장 어려운 소프트웨어의 비가시성을 극복할 수 있다. 이를 통해 소프트웨어의 구조를 파악함으로써 비전문가도 고품질을 위한 소프트웨어 개발의 품질 관리가 가능하다[1,2]. 코드를 정적 분석하여 클래스, 변수, 함수, 등으로 구분 지어 데이터를 추출한다. 이 데이터들을 데이터베이스(SQLite)에 저장하고, 필요한 정보를 쿼리문으로 SELECT하고 이 데이터들로 Dot script를 작성한다[3,4]. 작성된 Dot script는 Graphviz에서 그래프로 그려진다.

2.2 Go AST Viewer

Go AST Viewer는 Go 코드를 Abstract Syntax Tree 구조로 가시화하여 보여주는 웹 기반의 오픈소스이다. 소스코드를 파싱하기 필요한 해당 문구의 AST Node 정보를 얻는데 사용된다. 해당 Node의 정보로 Go Parser 패키지 내의 inspect 함수의 모든 노드 깊이 우선탐색 과정을 통해 필요한 Node를 조건문으로 추출한다[5].

2.3 결합도

결합도는 모듈과 모듈 사이 연관관계 또는 상호의존관계를 의미한다. 재사용성 관점에서 바라봤을 때 결합도가 강할 경우, 모듈간의 의존성이 높아 변경 및 유지보수가 힘들어진다. 결합도가 약할 경우, 모듈간의 상호 의존성이 줄어들어 모듈간의 독립성이 높아지고, 독립성이 높으면 모듈간의 영향이 적어 유지보수하기 좋은 소프트웨어이다. 결합도는 Data, Stamp, Control, External, Common, Content 총 6개의 종류가 있다. Data 결합도일수록 모듈간 독립성이 높고, Content 결합도일수록 모듈간의 상호의존성이 높다.

3. Go language 가시화 과정

다음 그림 1은 Go Language 소프트웨어 가시화의 전 과정을 보여준다.

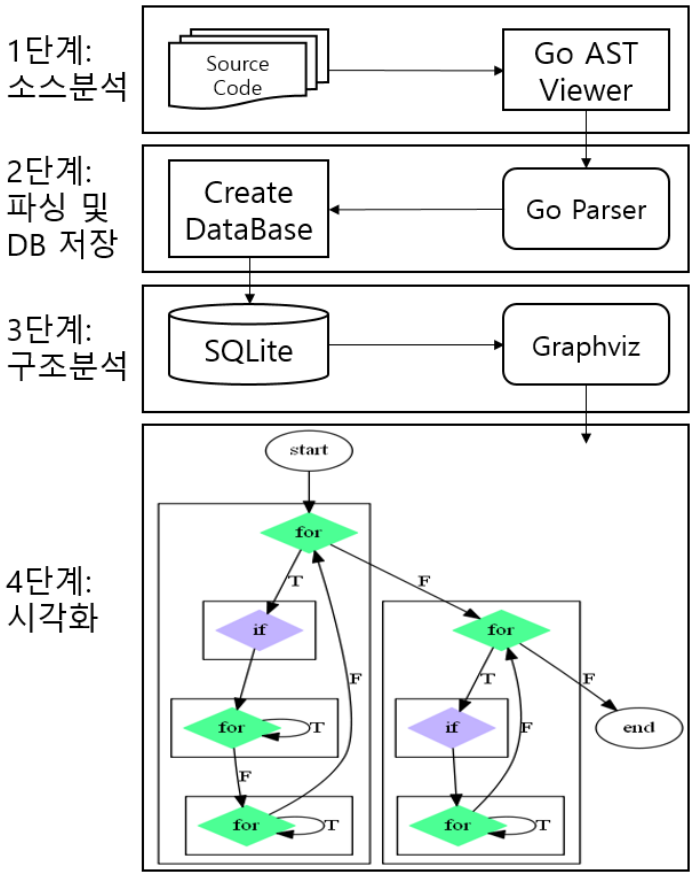

그림 1 Go Language 가시화 과정

소프트웨어 가시화를 위해서는 소스분석, 파싱 및 데이터베이스에 저장, 구조분석, 시각화 4가지 단계로 구성된다. 다음 그림 2는 Go AST Viewer를 사용한 파싱 과정이다. 그림 2의 ①처럼 Go 언어로 이루어진 코드를 Go AST Viewer에 입력하고 Parse 버튼을 누르면 각 문구의 AST Node 구조가 나온다. 그림2의 ②에서 확인 할 수 있듯이 ast.Ident 노드는 해당 코드의 packge name의 정보를 갖고 있는 노드인걸 알 수 있다. 이렇게 찾은 정보를 파싱 및 데이터베이스 저장 단계에서 그림 2의 ③처럼 Go Parser 내부의 inspect함수의 모든 노드 깊이 우선 탐색과정으로 조건문을 통해 추출한다. 이런 방식으로 결합도를 도출하기 위해 필요한 데이터를 저장할 수 있도록 정제과정을 거친다. 구조 분석 단계에서는 저장된 데이터들 중 가시화 과정에서 필요한 정보들을 쿼리문으로 SELECT하여 Dot Script를 작성한다. 시각화 단계에서는 작성된 스크립트를 통화여 이미지화 한다[6].

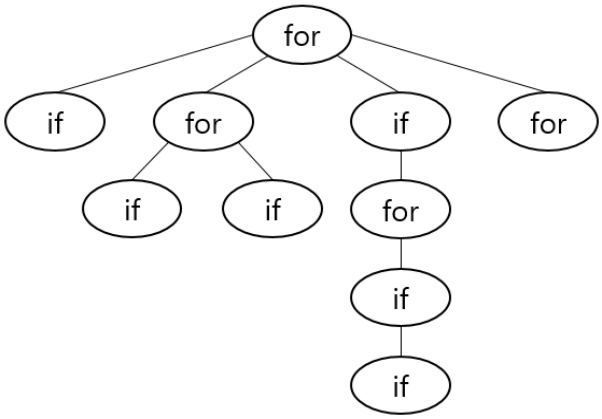

그림 3 Flow Chart를 그리기 위한 함수 내 구문 트리 구조

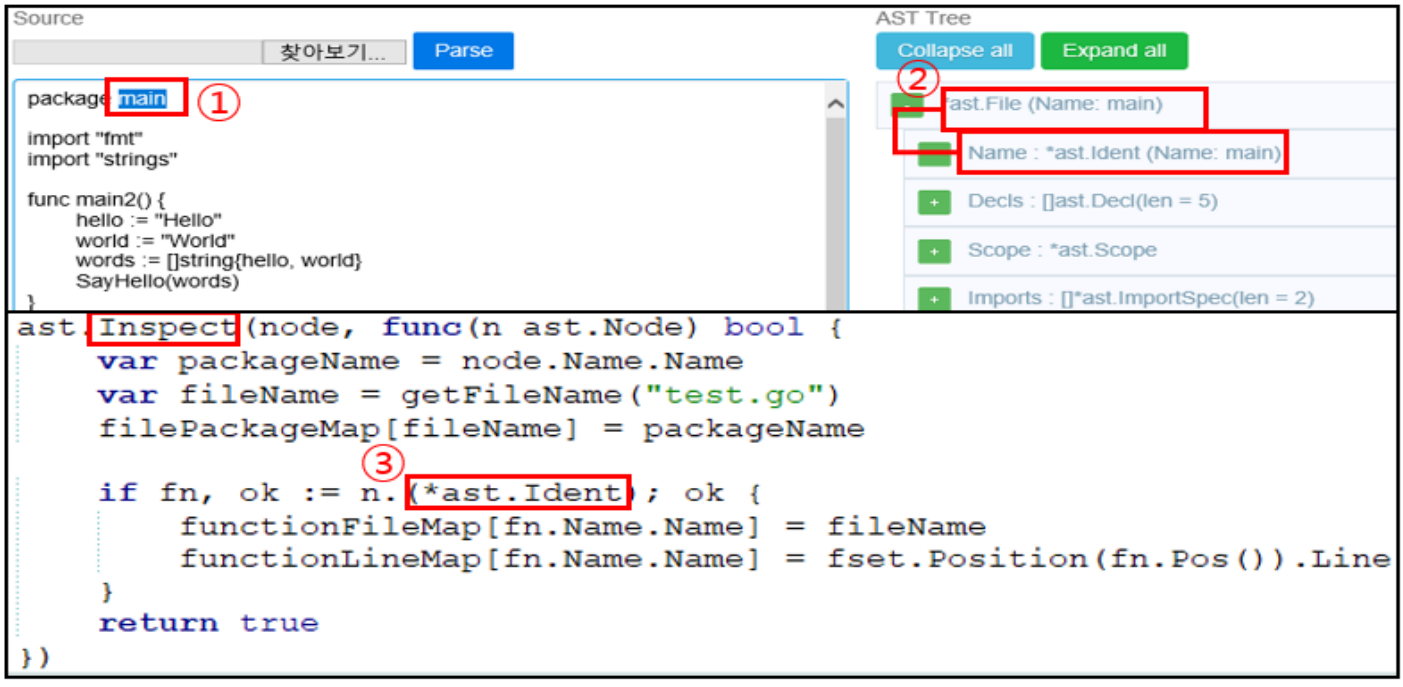

그림 2 Go AST Viewer를 사용한 파싱 과정

4. 결합도 측정 방법

Go 코드 내에서 함수가 호출할 때, 호출될 때 서로 주고받는 매개변수의 종류에 따라서 결합도를 측정할 수 있다. 아래의 그림 4는 Data Coupling에 대한 예제 코드와 Data Coupling 일 때 그려지는 그래프이다.

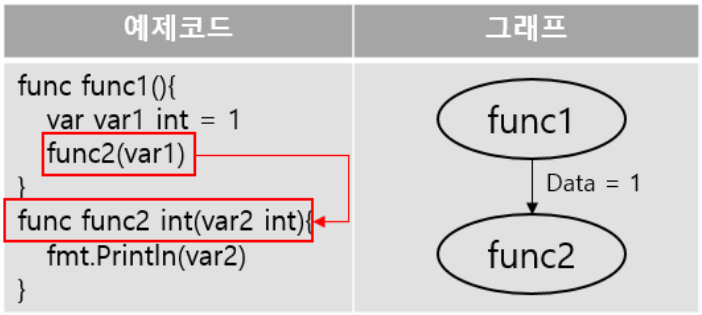

그림 4 Data Coupling 예제코드 및 그래프

아래의 그림 5는 Stamp Coupling에 대한 예제 코드와 그래프이다.

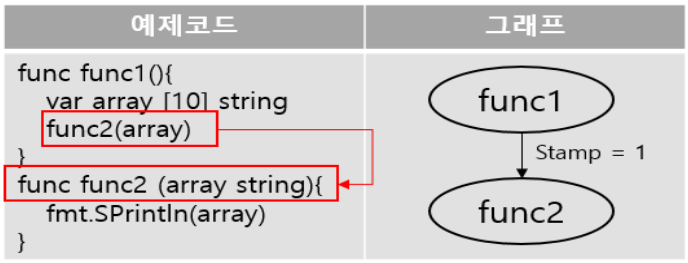

그림 5 Stamp Coupling 예제코드 및 그래프

아래의 그림 6은 Control Coupling에 대한 예제 코드와 그래프이다.

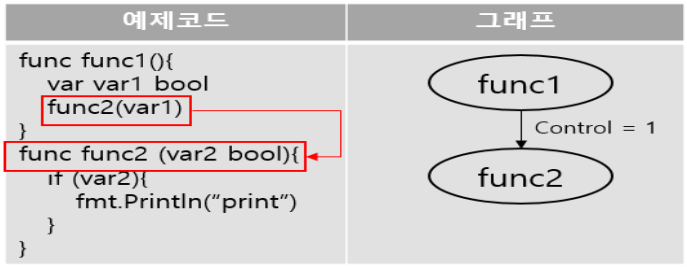

그림 6 Control Coupling 예제코드 및 그래프

아래의 그림 7은 External Coupling에 대한 예제 코드와 그래프이다.

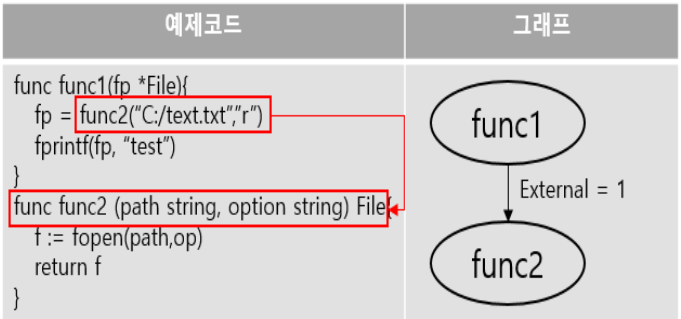

그림 7 External Coupling 예제코드 및 그래프

다음의 그림 8은 Common Coupling에 대한 예제코드와 그래프이다.

그림 8 Common Coupling 예제코드 및 그래프

다음의 그림 9는 Content Coupling에 대한 예제 코드와 그래프이다.

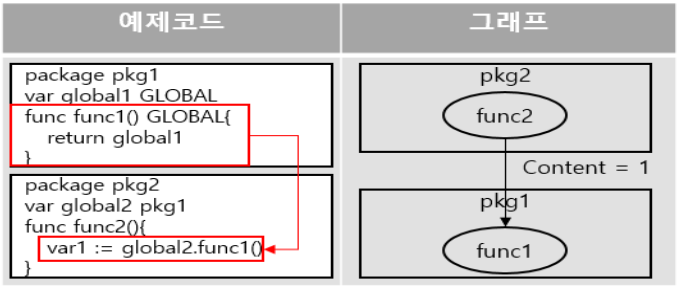

그림 9 Content Coupling 예제코드 및 그래프

예제코드에 pkg1, pkg2라는 패키지 2개가 있다. pkg1에는 GLOBAL 타입의 전역변수 global1과 func1 함수가 있고 func1은 global1을 반환한다. pkg2의 func2 함수는 var1이라는 GLOBAL 타입을 선언할 때 pkg1 패키지의 전역변수 global1을 직접 불러오지 않고 func1의 반환 값을 통해 가져온다. 이와 같이 직접 선언하지 않은 변수를 다른 객체, 함수의 반환 값 등을 통해 호출하는 경우 Content Coupling으로 인식한다. 그래프는 Common Coupling과 마찬가지로 서로 다른 패키지를 구분하기 위해 패키지 그림이 추가된다.

5. 적용사례

다음 그림 10은 코드 가시화의 적용사례로 실제 Go language를 가시화 해주는 코드 중 데이터베이스를 연결해주고 데이터를 저장할 테이블을 create해주는 Database.go와 이를 실행 시키고 결합도를 측정해주는 main.go 두 파일을 사용했다.

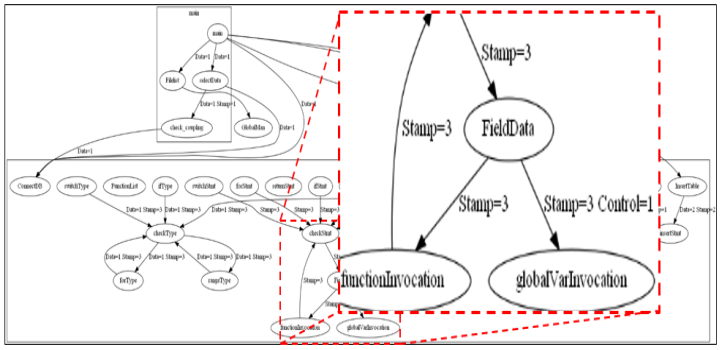

그림 10 리펙토링 전 결합도 가시화 그래프

가시화 그래프에서 확인할 수 있듯이 FieldData함수가 functionInvocation, globalVarInvocation함수를 호출할 때 각각 Stamp 결합도가 3번씩 발생한다. Stamp 결합도는 상대적으로 Data 결합도보다 높은 결합도이다. Stamp 결합도를 상대적으로 낮은 Data 결합도로 리펙토링하면 소프트웨어의 품질을 높일 수 있다. 하지만 롱 파라미터 리스트라는 나쁜 냄새를 유발할 수 있기 때문에 변경에 유의해야한다.

6. 결론 및 향후 연구

본 연구는 고품질 소프트웨어를 위해 Go 코드의 아키텍처 가시화를 통하여 모듈간의 결합도를 시각화하여 조사 분석한다. 이로 인해 어떤 모듈에서 높은 결합도가 존재하는지 알 수 있으며 이를 낮은 결합도로 리펙토링하여 결합도 수치를 감소시킨다. 이를 통해 평가 기준에 따라 해당 함수가 안정된 코드인지, 복잡한 코드인지를 확인한다. 현재 소프트웨어 고품질을 위한 지표로써 결합도 경우의 수를 가시화하여 전부는 아니어도 찾을 수 있었다. 향후 연구로 모든 가능한 경우의 수를 찾고, 응집도, Bad Smell등 다양한 지표들을 추가하여 연구하고, 다양한 사례에 적용하여 연구를 진행하고자 한다.

ACKNOWLEDGE

본 논문은 2019년도 산업통상자원부의 '창의산업융합 특성화 인재양성사업'(과제번호 N0000717)과 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2017R1D1A3B03035421)

참고 문헌

[1] NIPA SW Engineering Center "SW Development Quality Management Manual(SW Visualization)" 2013. 12.
[2] 박지훈, "정적 분석 기반 블록체인 코드 취약성의 자동 가시화 연구", 2018
[3] SQLite, https://www.sqlite.org/
[4] Graphviz, http://www.graphviz.org/
[5] Go AST Viewer, http://goast.yuroyoro.net
[6] 안현식, 박지훈, 박보경, 김영철, "Go language로 구성된 블록체인 코드를 측정하기 위한 효과적인 코드 정적분석기" 2019. 12.

Electronics & Telecommunications
Research Institute

ETRI의 최첨단 기술은 새로운 영역과 새로운 가치를 만들어내고 있습니다.
꿈꾸는 모든 것을 현실로 구현하는 기술, ETRI의 첨단 ICT기술은 세상을 더욱 풍요롭게 만들어 갑니다.