한국정보과학회
KOREAN INSTITUTE OF INFORMATION SCIENTISTS AND ENGINEERS

https://www.kiise.or.kr/conference/KSC/2022/

2022 한국소프트웨어종합학술대회
Korea Software Congress 2022

2022. 12. 20.(화)~23.(금)
라마다 프라자 제주호텔

"반도체 시대를 선도할 정보기술"

주요행사

12.20

[특별세션]
- 우수논문발표세션

[워크샵]
- 디지털 플랫폼 도시, 성과와 도전과제 워크샵
- 대용량 데이터 분석 최신 기술 및 데이터베이스 교육 워크샵
- 사물 협업 행동 지능 핵심기술 워크샵
- 전원 불안정 감내 자율 에너지 기반 컴퓨팅 워크샵
- GEdge Platform 커뮤니티 제5회 컨퍼런스
- 개방형 확장현실 플랫폼 기술개발 워크샵

12.21

[특별세션]
- Top Conference 세션
- 우수논문발표세션

[워크샵]
- 한국소프트웨어공학역사워크샵
- Kakao Tech Workshop
- 연구지원기관 사업설명회
- Emerging Technology Workshop in Computer Systems
- 부산대학교 인공지능융합연구센터 성과공유 워크샵
- 2022 AI 데이터 품질 개선 오픈랩 프로그램
- 2022년 예지형 시각지능 과제 워크샵
- MEC, 연합학습, 블록체인 융합 기술 워크샵
- 인하대학교 AI 확산 및 산학협력 전략 워크샵
- 엣지마이크로데이터센터 시스템SW 워크샵
- 차세대융합기술연구원 경기도자율주행센터 성과 발표회

12.22

[기조강연]
- 이재진 교수(서울대학교)
- 장우석 Master(삼성전자)

[초청강연]
- 김상연 교수(한국기술교육대학교)
- 전원석 책임연구원(현대자동차)
- Ph.D. Yuichi Nakamura(NEC Corp.)
- 이상원 교수(성균관대학교)
- 김성진 이사(카카오)
- 노강호 상무(삼성전자)

[특별세션]
- 박사포럼
- Top Conference 세션
- 우수논문발표세션

[워크샵]
- 코드 분석과 오류 마이닝이 결합된 SW 오류 자동 수정 기술 개발
- 융합 서비스를 위한 인공지능 & 모바일 기술 활용 워크샵
- 강원권 SW중심대학 협력워크샵
- 빅데이터 엣지클라우드 서비스 워크샵

12.23

[특별세션]
- Top Conference 세션
- 우수논문발표세션

[워크샵]
- 초거대 그래프의 지능적 고속 처리를 위한 그래프 DBMS 기술 개발 워크샵

후원

DIAMOND (사)제주컨벤션뷰로

PLATINUM kakao HYUNDAI

GOLD DINNO IT

SILVER MIND ONE kt LG히다찌 NIA 한국지능정보사회진흥원

BRONZE 쌍용정보통신 turnitin 올포랜드 CUBRID grepp. SK broadband LG CNS OSSTEM IMPLANT cadence

역공학 기반 비용 예측의 소프트웨어 비용 검증 메카니즘

박찬솔 O* 장민철* 문소영* 김영수** 김영철*

*홍익대학교 소프트웨어융합학과 **정보통신산업진흥원

c2193102@g.hongik.ac.kr, garaczi@g.hongik.ac.kr, whit2@hongik.ac.kr, ysgold@nipa.kr, bob@hongik.ac.kr

Software Cost Validation Mechanism of Cost Estimation based on Reverse Engineering

Chansol Park O* Minchul Jang* Soyoung Moon* Young s. Kim** R. Young Chul Kim*

*Dept. of Software and Communications Engineering, Hongik University **NIPA

요 약

현재 소프트웨어 프로젝트의 규모를 측정하여 비용을 예측하기 위해서는 기능점수 값(Function Point Value)을 산출해야 한다. 그리고 기능점수 산출을 위해서는 기능점수 전문가가 자연어로 작성된 요구사항을 분석하고 유사 사업들을 제시해야 한다. 그러나 민간/공공/국가 사업 진행 시 소프트웨어 비용 예측을 통한 사업 비용 지불이 타당한지에 대한 의문이 제기되고 있다. 따라서 기능점수를 통해 예측한 비용에 대한 검증을 통해 실제 완성된 소프트웨어 개발 비용과의 차이를 줄일 필요가 있다. 이러한 문제를 해결하기 위해 소스 코드로부터 기능점수를 추출 자동화 메커니즘을 제안한다. 이를 위해 정적 분석 기반 코드 내부 정보를 3차 정규화 DB 테이블 구조에 저장하고, ILF, ELF, EI, EO, EQ 추출을 질의어(Query)화 한다. 이를 도구화 함으로서 소프트웨어 프로젝트 비용 예측에 대한 검증을 기대한다. 또한 추후 연구를 통해 코드 복잡도를 보정 계수에 적용하려 한다.

1. 서 론

2020년 소프트웨어 진흥법이 개정됨에 따라 소프트웨어를 발주함에 있어 설계와 개발을 분리하여 발주해야 한다. 이에 따라 소프트웨어를 개발하기 위한 비용을 산정하기 위해 보편적으로 사용되고 있는 기능점수의 중요성이 더욱 커졌다[1]. 하지만 기능점수는 사용자 관점에서 작성된 요구 사항에 근거하기 때문에 개발 후 결과물과의 괴리가 발생할 수 있다[2]. 또한 지금까지는 사업 발주 시 측정된 기능점수를 개발이 끝난 후 검증할 수 있는 방법이 제시되지 않았다. 이러한 문제점을 해결하기 위해 역공학 기반 코드 가시화 기법을 통한 기능점수 추출 메커니즘을 제안하고, 기능점수 예측에 대한 검증을 기대한다.

2. 관련 연구

2.1. 기능점수 기반 정교한 비용 예측 추출을 위한 요구사항 스펙 구조화

그림 1은 구조화된 요구사항 스펙으로 기능점수를 산정하여 기능점수에 대한 예측을 정교화 하는 연구이다[3]. 우선 비즈니스 규칙 및 제약사항을 정의하는 Constraints 층, 수행하는 활동에 대한 구체적인 활동을 정의하는 Processes 층, Process들의 내부 기능을 정의하는 Functions 층 그리고 Function들에서 사용하는 데이터를 정의하는 data층까지 총 4개의 층으로 이루어진 클로즈 아키텍처 기반의 요구사항 명세 구조 템플릿을 통해 요구사항을 정의하였다. 이후 기능과 데이터를 식별하고 보정 계수를 계산하여 기능점수를 예측한다.

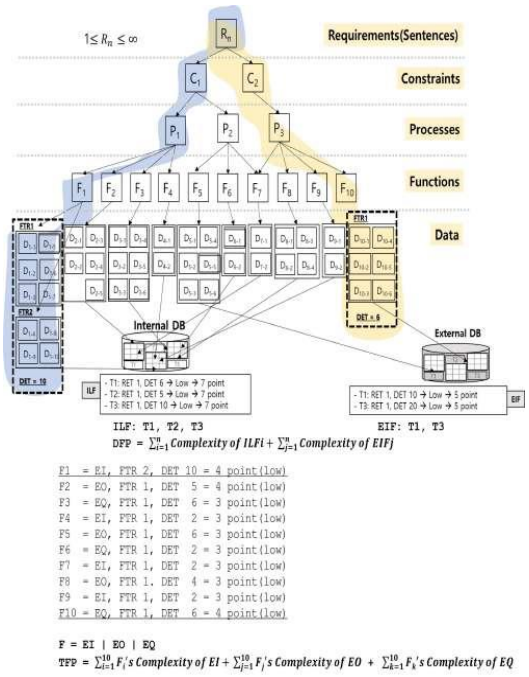

그림 1 요구사항 스펙 구조화 기반 기능점수 예측

2.2. Open Source 기반 툴 체인화를 통한 코드 정적 분석 연구

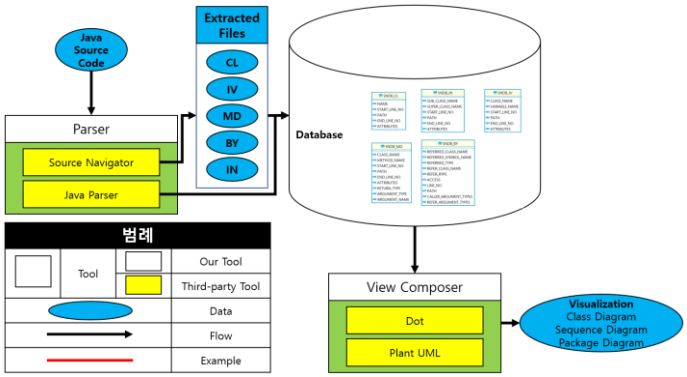

그림 2 기존 코드 정적 분석 툴 체인 구조

해당 연구에서는 그림 2과 같은 툴 체인을 이용하여, 소스코드를 분석 후, 정보를 추출하여 Database(DB)에 저장한다. 이후 DB에 저장된 데이터를 통해 소스코드의 복잡도를 계산하고, 설계를 추출하여 이를 가시화한다 [4].

해당 연구에서 사용한 DB 구조는 그림 2와 같이 Source Navigator 도구를 통해 소스 코드의 클래스, 메서드, 필드, 클래스간 상속, 함수 호출 정보를 담고 있는 파일을 생성하고, 이를 각각의 테이블로 저장한다. 이때 각각의 클래스, 메서드, 필드는 이름을 통해 구분되며, 특정 조건을 만족하는 정보를 추출하기 위해서는 테이블들을 이름을 통해 JOIN한 후 검색한다. 이러한 방법을 통해 데이터를 추출하는 경우 클래스, 메서드, 필드의 이름이 중복되는 경우가 있으므로 JOIN 연산 시 잘못된 데이터가 추출될 수 있다는 치명적인 문제가 있다. 또한 Source Navigator 도구가 분석할 수 있는 정보의 범위가 제한되어, 소스코드를 분석하는데 있어 한계가 있다는 문제점도 있다.

3. 소프트웨어 가시화 기반 비용 예측의 소프트웨어 비용 검증 메커니즘

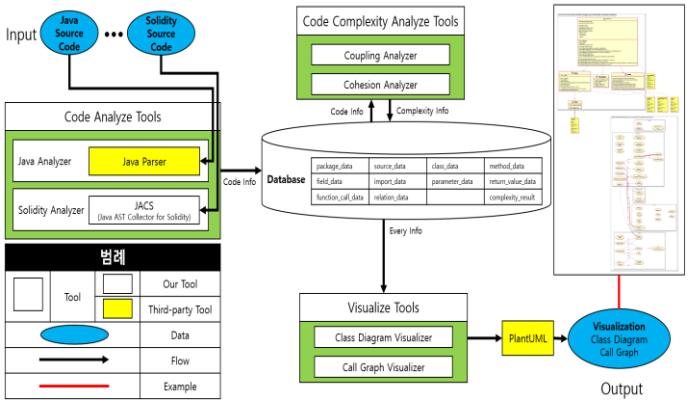

그림 3 개선된 코드 정적 분석 툴 체인 구조

그림 4는 소프트웨어에 대한 자세한 정보를 저장하고, 기존의 문제점을 해결하기 위해 개선한 Toolchain이다. 우선 AST를 통해 코드에서 정보를 추출하여, 코드에 대한 전반적인 정보를 추출할 수 있게 되었다. 또한 DB를 중심으로 코드 분석, 복잡도 분석, 가시화 모듈로 나누어 재사용성을 높였다. 이를 통해 코드 분석 도구만 교체하면 나머지 모듈을 그대로 이용해 다른 언어에 대해 분석이 가능하다.

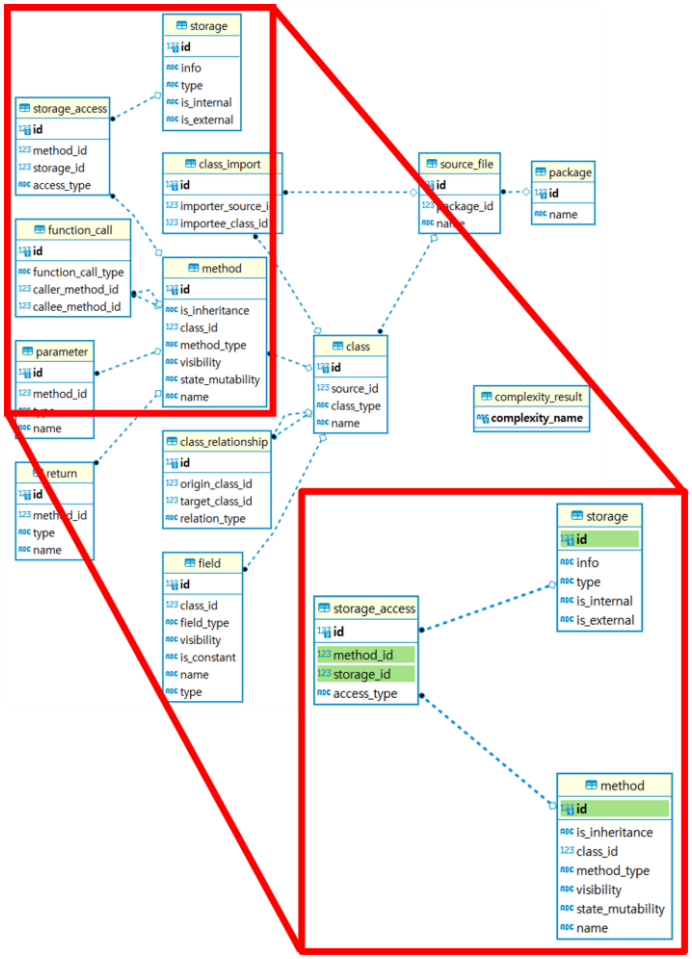

그림 4 3차 정규화 된 DB 테이블 구조

소스코드로부터 추출한 정보를 저장하기 위해 개선한 DB의 구조는 그림 5와 같다. 기존 DB 구조가 가지고 있는 구조적 문제를 해결하기 위해 DB를 3차 정규화 했다. 이를 통해 데이터 저장 및 검색의 효율을 높이고, 잘못된 데이터 검색을 방지해 툴체인의 정확도를 높였다.

본 논문에서는 개선된 툴체인과 정규화 된 DB를 통해 소스 코드와 관련된 저장소들을 식별하고, 함수와 저장소 간의 관계를 분석하였다. 기능점수를 분석하기 위해 식별한 저장소(File system 혹은 DB)를 저장하기 위해 storage 테이블을 설계했으며, 식별한 저장소와 함수와의 관계를 저장하기 위해 storage_access 테이블을 설계했다.

storage 테이블은 분석 과정에서 부여된 고유 ID와 3개의 속성을 가진다. 우선 info는 식별된 저장소의 정보를 말한다. File system의 경우 경로, DB의 경우 DB 시스템의 종류 및 주소를 저장하며, 변수를 통해 접근하는 경우와 같이 접근할 수 없는 경우 해당 변수명이 입력된다. 다음으로 type의 경우 식별된 저장소의 종류가 입력된다. 데이터의 종류가 File

System인 경우 File이 Database인 경우 DB가 입력된다. 마지막으로 is_internal 속성의 경우 애플리케이션 경계 내부의 저장소인지 아닌 지가 입력되며, is_external 속성의 경우 애플리케이션 경계 외부의 저장소인지 아닌 지가 입력된다.

storage_access 테이블의 경우 마찬가지로 분석 과정에서 부여된 고유 ID와 3개의 속성을 가진다. 저장소에 접근하는 함수의 고유 ID를 method_id 속성으로, 해당 함수가 접근하는 저장소의 고유 ID를 storage_id 속성으로 갖는다. 또한 access_type 속성은 해당 함수에서 저장소에 접근 방식에 따라 input, output, query 등을 값으로 갖는다.

DB로부터 기능점수를 추출하는 질의는 다음과 같다. ILF의 개수는 표 1과 같이 storage 테이블에서 is_internal 속성이 'true'인 행의 수와 같다.

표 1 ILF의 개수 질의 문

```
SELECT COUNT(id)
FROM storage
WHERE is_internal ='true';
```

EIF의 개수는 표 2와 같이 storage 테이블에서 is_external 속성이 'true'인 행의 수와 같다.

표 2 EIF의 개수 질의 문

```
SELECT COUNT(id)
FROM storage
WHERE is_external ='true';
```

EI는 표 3과 같이 storage 테이블의 is_internal 속성이 'true'이고, storage_access 테이블의 access_type 속성이 'input'인 행의 수와 같다.

표 3 EI의 개수 질의 문

```
SELECT COUNT(storage_access.id)
FROM method, storage_access, storage
WHERE method.id = stage_access.method_id
AND storage.id = starage_access.storage_id
AND storage.is_internal ='true'
AND storage_access.access_type = 'input';
```

EO는 storage 테이블의 is_external 속성이 'true'이고, storage_access 테이블의 access_type 속성이 'output'인 행의 수와 같다.

표 4 EO의 개수 질의 문

```
SELECT COUNT(storage_access.id)
FROM method, storage_access, storage
WHERE method.id = stage_access.method_id
AND storage.id = starage_access.storage_id
AND storage.is_external ='true'
AND storage_access.access_type = 'output';
```

EQ는 storage 테이블의 is_external 속성이 'true'이고, storage_access 테이블의 access_type 속성이 'query'인 행의 수와 같다.

표 5 EQ의 개수 질의 문

```
SELECT COUNT(storage_access.id)
FROM method, storage_access, storage
WHERE method.id = stage_access.method_id
AND storage.id = starage_access.storage_id
AND storage.is_external ='true'
AND storage_access.access_type = query;
```

이러한 쿼리를 통해 추출한 EIF와 ILF 및 EI, EO, EQ 각각의 개수 구한 후, 표 6과 같이 간이법 기능점수 가중치[1]에 유형별로 각각 곱한 후 더한다. 추출한 기능점수를 요구사항을 통해 예측한 기능점수와 비교하여 검증할 수 있다.

표 6 간이법 기능점수 계산 공식

FP = 4.0*i + 5.2*o + 3.9*q + 5.4*p + 7.5*f

where:

i : Number of External Inputs
o: Number of External Outputs
q: Number of External Queries
p: Number of External Interface Files
q: Number of Internal Logical Files

4. 결론 및 추후 연구

본 논문을 통해 우리는 소프트웨어 역공학을 통한 소스코드로부터 데이터 및 트랜잭션 기능점수를 식별하여 검증할 수 있는 메커니즘을 제안하였다. 이러한 접근으로 소프트웨어 개발 비용 예측의 정확도 향상을 통해 부정확한 비용 예측에 따른 예산 누수를 막을 수 있을 것을 기대한다. 추후 기능점수의 보정계수에 코드 복잡도를 접목하여 정통법 기능점수 산출 공식을 통한 기능점수 검증 연구를 진행할 예정이다.

ACKNOWLEDGEMENT

이 논문은 교육부 및 한국연구재단의 4 단계 두뇌한국 21 사업의 지원(F21YY8102068)과 2022 년도 정부(교육부)의 재원으로 한국연구재단의 지원(No. 2021R1I1A305040711, No. 2021R1I1A1A01044060)을 받아 수행된 연구임.

참고문헌

[1] 한국소프트웨어산업협회, "SW사업대가산정 가이드(2022년 2차 개정판)", 2022.

[2] 정인용, et al. "소프트웨어 규모 산정을 위한 개선된 기능점수 측정 모델". 인터넷정보학회논문지, vol. 10, 115-126, 2009.

[3] 문소영, et al. "기능점수 기반 정교한 비용 예측 추출을 위한 요구사항 스펙 구조화" 한국스마트미디어학회 2022년 종합학술대회, 대전:한남대학교.

[4] 강건희, et al. "Open Source 기반 툴 체인화를 통한 코드 정적 분석 연구." 정보과학회 컴퓨팅의 실제 논문지, 21.2, 148-153, 2015.