제21권 제2호 2015년 2월

ISSN 2383-6318(Print)
ISSN 2383-6326(Online)

정보과학회 컴퓨팅의 실제 논문지

KIISE Transactions on Computing Practices

VOLUME 21, NUMBER 2, FEBRUARY 2015

대용량 경로데이터 분류에 기반한 경험적 최선 경로 추천 ..... 이계형, 조영훈, 이태호, 박희민 101

단편 논문

정보이론 관점에서 본 서울시 지역구간의 미세먼지 영향력 재조명 ..... 이재구, 이태훈, 윤성로 109

소프트웨어 제품과 프로세스 관점에서 국제표준과 비교를 통한 ..... 윤형진, 최진영 115
테스팅 프론티어 역량평가 모델 개선 방안

Preference Difference Metric을 이용한 아이템 분류방식의 추천알고리즘 ..... 박찬수, 황태규, 홍정화, 김성권 121

대용량 플래시 저장장치에서 신뢰성 향상을 위한 무작위 기반 정적 마모 평준화 기법 ..... 최길모, 김세욱, 최종무 126

타임드 오토마타 모델 기반 테스팅 기법 분석 및 사례 연구 ..... 김한석, 지은경, 배두환 132

Boyer-Moore 알고리즘을 위한 GPU상에서의 병렬 최적화 ..... 정요상, 쟌느앗-프엉, 이명호 138
남덕윤, 김직수, 황순욱

도메인 질의응답 시스템 ..... 윤승현, 임은희, 김덕호 144

Open Source 기반 툴 체인화를 통한 코드 정적 분석 연구 ..... 강건희, 김영철, 이근상 148
김영수, 박용범, 손현승

일상생활 계획을 위한 스마트폰-사용자 상호작용 기반 지속 발전 가능한 ..... 이범진, 김지섭, 류제환 154
사용자 맞춤 위치-시간-행동 추론 방법 허민오, 김주석, 장병탁

프롬프트 레이블링을 이용한 적응형 음성기반 감정인식 프레임워크 ..... 방재훈, 이승룡 160

한국정보과학회
KOREAN INSTITUTE OF INFORMATION SCIENTISTS AND ENGINEERS

ISSN 2383-6318(Print) / ISSN 2383-6326(Online)
KIISE Transactions on Computing Practices, Vol. 21, No. 2, pp. 148-153, 2015. 2
http://dx.doi.org/10.5626/KTCP.2015.21.2.148

Open Source 기반 툴 체인화를 통한 코드 정적 분석 연구

(A Practical Study on Code Static Analysis through Open Source based Tool Chains)

강 건 희 [†] (Geon-Hee Kang)
김 영 철 [††] (R. Young Chul Kim)
이 근 상 [†††] (Geun Sang Yi)
김 영 수 [††††] (Young Soo Kim)
박 용 범 [†††††] (Yong. B. Park)
손 현 승 [†] (Hyun Seung Son)

요 약 국내의 소프트웨어 산업계는 고품질 소프트웨어를 위해, 개발/테스트 프로세스, 성숙도 측정 등에 초점을 두고 있다. 그러나 실제 중소기업의 산업현장에서는 코드중심으로 개발되고 있다. 그리고 대부분의 기존 레거시 시스템은 설계의 부재 그리고 코드 패칭으로 코드 내부의 복잡도가 매우 높은 현실이다. 이를 해결하고자, 코드의 가시화(visualization)를 적용하였다. 이 가시화는 모듈간의 복잡도를 줄이려는 목적을 가지고 있다. 이를 위해 기존 공개 도구로 툴 체인 구성 방법을 제안한다. 제안한 방법은 NIPA의 SW Visualization 기법을 적용·확장하였다. 또한 코드 가시화내의 품질지표 중에 결합도 요소 중의 나쁜 지표에 대한 리팩토링 시도이다. 결과적으로 레가시 코드에 대해 역 공학 기법(from programming via model to architecture)적용과 이를 통한 소프트웨어 고품질화이다.

키워드: 소프트웨어 가시화, 리펙토링, 역공학, 결합도, 툴 체인

Abstract In our domestic software industries, it is focused on such a high quality development/testing process, maturity measurement, and so on. But the real industrial fields are still working on a code-centric development. Most of the existing legacy systems did not keep the design and highly increased the code complexity with more patching of the original codes. To solve this problem, we adopt a code visualization technique which is important to reduce the code complexity among modules. To do this, we suggest a tool chaining method based on the existing open source software tools, which extends NIPA's Software Visualization techniques applied to procedural languages. In addition, it should be refactored to fix bad couplings of the quality measurement indicators within the code visualization. As a result, we can apply reverse engineering to the legacy code, that is, from programming via model to architecture, and then make high quality software with this approach.

Keywords: SW visualization, refactoring, reverse engineering, coupling, tool-chain

· 이 논문은 2014년도 정부(미래창조과학부)의 재원으로 한국연구재단-차세대정보·컴퓨팅기술개발사업(No. 2012M3C4A7033348)과 2014년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2013R1A1A2011601)
· 이 논문은 2014 한국컴퓨터종합학술대회에서 '절차식 언어 기반의 코드 정적 분석을 위한 툴 체인 사례 연구'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 홍익대학교 컴퓨터정보통신 SE.Lab(Hongik Univ.)
kang@selab.hongik.ac.kr
hson@live.co.kr
(Corresponding author임)
†† 정 회 원 : 홍익대학교 컴퓨터정보통신 SE.Lab 교수
bob@hongik.ac.kr
††† 정 회 원 : (재)전북테크노파크
yi@jbtp.or.kr
†††† 종신회원 : NIPA 소프트웨어공학센터
ysgold@nipa.kr
††††† 종신회원 : 단국대학교 컴퓨터과학과 교수
ybpark@dankook.ac.kr

논문접수 : 2014년 9월 15일
(Received 15 September 2014)
논문수정 : 2014년 11월 5일
(Revised 5 November 2014)
심사완료 : 2014년 11월 19일
(Accepted 19 November 2014)

정보과학회 컴퓨팅의 실제 논문지 제21권 제2호(2015. 2)

1. 서 론

정보통신산업진흥원(NIPA)의 '2013 SW공학백서[1]'에 따르면 프로세스·품질인력·공학기술을 종합한 국내 기업의 소프트웨어공학 품질수준이 100점 만점에 64.8점으로 최하위 등급인 '열등'을 간신히 벗어났다. 국내 대기업의 소프트웨어 개발은 SW공학적으로 접근하여 개발을 하고 있지만 중소기업은 SW공학적용을 위한 전문인력과 개발시간, 자본 등의 부족으로 인해서 코드중심의 개발을 하고 있다.

신규로 개발하는 프로그램은 시작부터 소프트웨어공학을 적용해 고품질화 하지만 기존의 소프트웨어는 그렇지 못한 경우가 많다. 그러므로 이러한 현재 상황을 극복하고 기존 소프트웨어의 고품질화를 위해서는 역공학으로 소프트웨어의 가시화가 필요하다.

소프트웨어 가시화는 몇 가지 오픈소스도구(Source Navigator[2], DOT 스크립트[3], SQLite[4])를 묶은 툴 체인으로 구성해야한다. 그 이유는 각 도구가 가지는 기능이 정해져 있어 하나의 도구로는 우리가 원하는 소프트웨어 가시화를 수행할 수 없기 때문이다. 또한 코드를 분석하여 함수의 콜 그래프를 그려주는 SN(Source Navigator)가 이미 있지만 우리가 원하는 품질속성은 표현하지 못한다. 그러나 SN이 코드를 파싱한 정보를 DB를 통해 제공하기 때문에 우리는 파서를 개발하지 않고 SNDB(Source Navigator DataBase)정보를 활용한다. 그리고 DB정보를 저장할 수 있는 SQLite와 그래프를 그릴 수 있는 DOT 스크립트를 사용한다.

본 논문에서는 소스코드의 가시화를 위해서 SNDB의 정보를 텍스트화 하여 SQLite로 저장한 뒤 품질속성 추출을 위한 쿼리문을 통해 DOT 스크립트로 그래프를 표현하는 과정을 보여준다. 이를 통해 품질속성의 정량화로 품질지표를 정의하고 레가시 코드를 리펙토링 하여 고품질화 하는 것이 본 논문의 목표이다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구로 NIPA 소프트웨어 공학센터에서 제안한 S/W 가시화(Visualization)와 역 공학을 설명하고, 3장에서는 정적 분석을 위한 툴 체인 구성을 설명을 언급한다. 4장에서는 SN를 이용한 품질지표 검출을 설명하고, 5장에서는 툴 체인에 대한 적용사례를 보여준다. 6장에서는 결론 및 향후 연구를 언급한다.

2. 관련연구

2.1 S/W 가시화(Visualization)

현재 성공적인 소프트웨어 개발 및 관리를 위해 소프트웨어 개발, 프로세스, 테스트 자동화, 그리고 품질인증 등은 필수적이다. 그러나 개인적인 소견이지만, 실제 국내 소프트웨어 현장멘토링 경험으로 볼 때, 현실적인 해법은 "NIPA의 소프트웨어 Visualization"이 벤처/중소/중견 IT업체에 적합한 방법일 수 있다. 이는 소프트웨어의 역 공학을 통한 시각화와 문서화이다.

첫째, 시각화는 소프트웨어 개발의 가장 어려운 점인 소프트웨어 비가시성을 극복하고 전체 소프트웨어 개발 과정을 파악하여 품질관리를 수행하는 방안이다. 둘째, 문서화는 기업 혹은 단체의 개발 노하우 관리 및 내부 인력간의 업무 이해도 향상과 특정 상황에서 외부와의 의사소통을 위한 방안이다.

Christian Collberg[5]는 자바 컴파일 이후 생성되는 클래스파일의 바이트코드 정보를 분석하여 control-flow graphs, call graphs, inheritance graphs를 그리는 방법을 제안하였다. 이 방법은 추출된 그래프로 시간적인 관점의 SW 변화를 감지 할 수 있지만 시각적으로 볼 수 있는 정보가 많지 않아 정보를 파악이 어렵다. Michele Lanza[6]는 다양한 메트릭(클래스, 특징)을 통해 SW를 측정한다. 이 방법은 각 요소에 대한 정보를 볼 수 있지만 SW의 아키텍쳐를 확인 할 수 없고 프로그램의 흐름을 파악하기 어렵다.

본 연구는 기존연구와 다르게 코드의 복잡도(결합도, 응집도)를 가시화 및 리펙토링하는 방법이다. 결과적으로 소스코드와 개발 프로세스를 관리 하는 것이 목적이며, 시각화와 문서화로 소프트웨어 개발 품질을 향상한다.

2.2 역 공학(Reverse Engineering)

순 공학 프로세스가 요구사항 분석, 설계, 구현, 테스트, 유지보수의 순서를 따른다면 역 공학 프로세스는 반대로 구현된 프로그램에서 설계문서를 설계문서에서 요구사항을 추출한다[7]. 그림 1은 역 공학 프로세스를 설명이다.

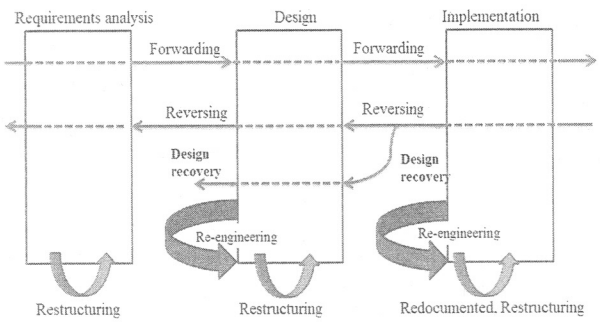


그림 1 역공학 프로세스

Fig. 1 Reverse engineering process

본 논문에서는 구현단계에서 설계단계로 가는 역공학으로, 구현단계에서의 설계 회복과, 소스코드의 재문서화와 재구조화가 목표이다.

3. 코드 정적 분석의 툴 체인 구성

코드 정적 분석을 위해서는 프로그램의 소스코드를 분석하는 파서, 그 정보를 DB화 그리고 그래프화 까지를 모두 구현하기는 쉽지 않다. 그러므로 우리는 기존의 오픈소스(SN(Source Navigator), DOT 스크립트, SQLite)를 이용한 툴 체인[8]을 구성하였고 본 논문에서는 기존 툴 체인을 개선한다.

코드 정적분석은 그림 2와 같이 SNDB 파일 추출→소프트웨어의 정보 추출→DB저장→품질지표 적용→가시화 총 5단계를 거치게 된다.

- 1단계 : 소스코드로 부터 그래프를 생성하기 위해서는 1단계로 SN으로부터 SNDB 파일을 추출한다. SNDB 파일에는 프로그램 코드에 전반적인 정보(함수의 정보, 지역변수, 전역변수, 파라미터 등)이 포함 되어있다.
- 2단계 : SN에서 추출한 모든 SNDB(.cl,. con,. e, fu, .gv, .iv 등)는 바이너리 파일이기 때문에 읽을 수 있는 정보로 변환해야한다. 이 단계에서는 SN에서 제공하는 DBdump.exe를 이용하여 내부에서 바이너리 파일을 텍스트로 변환한다.
- 3단계 : CreateDB 프로그램은 DBdump.exe를 이용하여 추출된 텍스트 정보를 SQLite 데이터베이스의 각 테이블에 저장한다.
- 4단계 : 품질지표 적용단계는 결합도의 추출을 위해 DB 테이블로 부터 정보를 뽑아오기 위한 쿼리문을 작성하고 정량적인 점수를 측정하는 단계이다.
- 5단계 : 가시화 단계에서는 GenerateDotContents 프로그램을 사용하여 쿼리문으로 추출된 정보로 부터 코드에 대한 아키텍처와 정량화된 품질지표 수치를 DOT 스크립트로 생성하고 graphViz 도구를 통해서 그래프로 그린다.

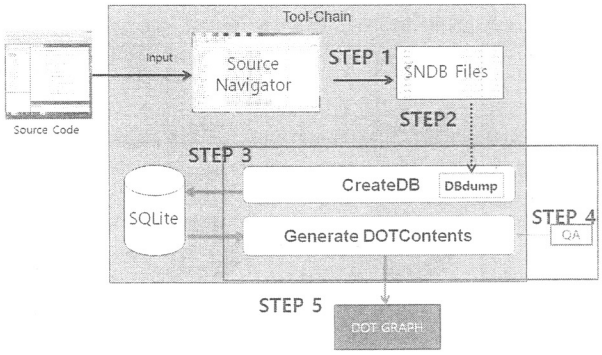


그림 2 툴 체인 구성도

Fig. 2 Tool-chain configuration diagram

4. 품질지표 검출-결합도

소프트웨어공학적으로 고품질의 소프트웨어를 만들려면, 결합도를 최소화와 응집도를 최대화해야 한다[9]. 본 논문에서는 결합도와 응집도 중에서 모듈 간 결합도를 품질지표로 정의하고 이를 가시화한다.

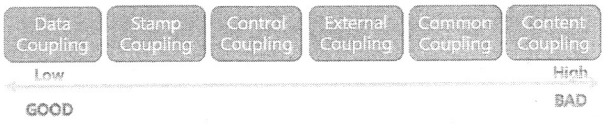


그림 3 결합도

Fig. 3 Coupling

그림 3은 강~약의 결합도의 표현이다. 결합도는 프로그램에서 모듈간의 상호 의존 혹은 연관관계를 의미하며 높을수록 모듈간의 의존성이 강해, 변경 및 유지보수 그리고 모듈의 재사용성에 나쁜 영향을 끼치게 된다. 결합도는 자료, 스탬프, 제어, 외부, 공유, 내용 총 6가지의 결합도가 있는데 자료에서 내용으로 갈수록 모듈간의 상호의존성이 높아져 결합도가 높아진다. 아래에서는 각 결합도별로 정의와 정의에 대해서 데이터베이스 상에서 어떻게 추출하는지 설명한다.

4.1 자료 결합도(Data Coupling) 및 스탬프결합도 (Stamp Coupling)

자료 결합도란 모듈 간의 인터페이스가 자료 요소로만 구성될 때의 결합도이다. 어떤 모듈이 다른 모듈을 호출하면서 매개변수를 기본 데이터형으로 넘겨주고 받는 경우를 말한다. 그리고 스탬프 결합도란 모듈간의 인터페이스로 배열이나 레코드 등의 자료구조가 전달될 때를 의미한다.

그림 4는 자료 및 스탬프 결합도 추출 쿼리문이다. ref_argument_types에 호출을 당하는 모듈의 파라미터가 기본 자료형(int, float, boolean, char 등)일 때 자료결합도로 판단한다. 그리고 기본 자료형 이외의 배열 혹은 자료형이 포함되어있을 때 스탬프 결합도로 판단한다.

```
query = "select class, symbol_name, "
+ "ref_class, ref_symbol, ref_argument_types, "
+ "filename, ref_type "
+ "from RefersTo "
+ "where  ref_type <> 'ma' and ref_type <> 'e' "
+ "and ref_type <> 'ec' and class <> '#' "
+ "and ref_argument_types <> ''";
```

그림 4 자료와 스탬프 결합도 추출 쿼리문

Fig. 4 Extracting query for Data and Stamp coupling

4.2 제어 결합도(Control Coupling)

제어 결합도란 한 모듈에서 다른 모듈로 논리적인 흐름을 제어하는데, 사용하는 제어 요소(function code, switch, tag, flag)가 전달 될 때의 결합도이다. 대부분의 조건식에 들어가는 논리 값을 함수로 만들었을 때 제어 결합도가 발생한다. 그러므로 호출되는 모듈의 반

환형이 논리 값으로 전달되는 경우 제어 결합도이다. 그림 5는 제어 결합도를 추출하기 위한 쿼리문으로, 호출되는 모듈에서 반환형 정보를 받아 논리 값인지 확인한다.

예를 들어 모듈 간의 관계정보를 받아온 뒤 호출을 당하는 모듈의 이름을 통하여 쿼리를 실행할 때 ret_type이 논리형(boolean)이면 제어 결합도로 판단한다.

```
query = "select class,name,ret_type "
+ "from fucntionImplementations where name='"
+ Callee + "' and class='" + Callee_class + "'";
```

그림 5 제어 결합도 검출 쿼리문

Fig. 5 Extracting query for Control Coupling

4.3 외부 결합도(External Coupling)

외부 결합도는 어떤 모듈이 외부로부터 선언된 데이터(변수)를 다른 모듈에서 참조할 때 발생하는 결합도이다.

```
query = "select class, symbol_name, ref_class, "
+ "ref_symbol, caller_argument_types, filename,
ref_type "
+ "from RefersTo "
+ "where  ref_type <> 'ma' and ref_type <> 'e' "
+ "and ref_type <> 'ec' and ref_type <> 'lv' "
+ "and ref_type <> 'gv' and class <> '#'";
```

(a) Relation information extraction queries

```
query = "select count(name) "
+ "from Classes "
+ "where name = '" +Callee_class +
"'";
```

(b) Determined query for external coupling

그림 6 외부 결합도 검출 쿼리문

Fig. 6 Extracting query for External Coupling

그래서 참조 되는 모듈의 이름에 대한 정보가 존재하지 않을 때를 외부 결합도라고 한다. 한 개의 쿼리문으로는 외부 결합도를 검출하기 어렵기 때문에 외부 결합도를 추출하는 쿼리문은 그림 6의 (a), (b)와 같이 2번 작성한다.

그림 6(a)의 쿼리문은 모듈과 모듈간의 관계정보를 받아온다. 그림 6(b)는 외부 결합도를 판단하기 위한 쿼리문이다. 외부의 것인지 알아보기 위해서 카운트를 세어서 카운트가 0인 즉, 존재 하지 않으면 외부 결합도로 검출한다.

4.4 공유 결합도(Common Coupling)

공유 결합도는 공유되는 공통 데이터 영역을 여러 모듈을 사용할 때의 결합도이다. 이 결합도는 공유되는 공통 데이터 영역을 사용하는 모든 모듈에 영향을 미쳐 독립성을 약하게 만든다. SNDB에서 모듈의 공유여부는 확인이 어려워, 전역변수로 추출한다.

```
query = "Select class, symbol_name, "
+ "ref_class, ref_symbol "
+ "from RefersTo where ref_type = 'gv'";
```

그림 7 공유 결합도 검출 쿼리문

Fig. 7 Extracting query for Common coupling

그림 7은 공유 결합도 추출하기 위한 쿼리문이다. 호출을 당하는 모듈의 타입이 'gv' 전역변수인 것을 찾아 검출한다.

4.5 내용 결합도(Content Coupling)

내용 결합도는 한 모듈이 다른 모듈의 내부 기능 및 그 내부 자료를 직접 참조하거나 수정할 때의 결합도이다. 그림 8은 내용 결합도를 검출하기 위한 쿼리문이다. 모듈 간 관계정보에서의 호출되는 모듈이 읽혀지거나 수정이 되는 모든 정보를 불러와 추출한다.

```
query = "select R.class, R.symbol_name, R.ref_class, "
+ "R.ref_symbol, R.filename, R.ref_type, R.position "
+ "from RefersTo as R LEFT JOIN "
+ "MethodImplementations as M "
+ "on R.ref_symbol = M.name "
+ "where R.ref_type = 'ud' and R.class <> '#' "
+ "and M.name IS NULL and R.ref_argument_types = '' "
+ "and R.access <> 'p' order by R.class, "
+ "R.symbol_name, R.position";
```

그림 8 내용 결합도 추출 쿼리문

Fig. 8 Extracting query for content coupling

5. 사례연구

본 논문은 그림 9와 같이 직접 개발된 6족 로봇 시뮬레이터를 적용 사례로 한다[10,11]. 6족 로봇 시뮬레이터는 다관절 로봇의 동작제어를 도구 내의 컨트롤러로 쉽게 할 수 있다. 그리고 실제로 로봇이 존재하지 않더라도 도구 내의 시뮬레이션을 통하여 실제 로봇의 동작을 수행 가능하다. 즉, 시뮬레이션 로봇이 동작할 환경을 구축한

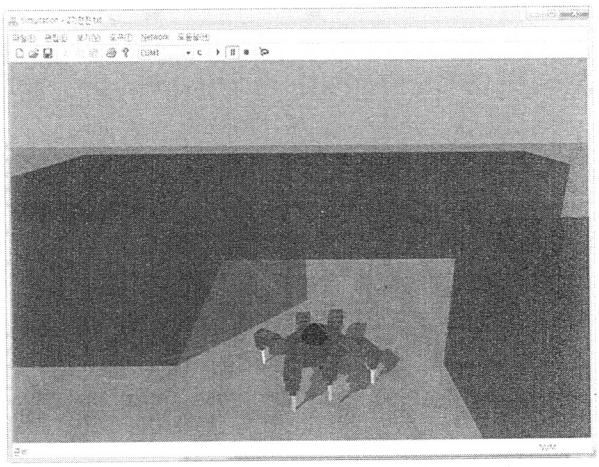


그림 9 6족로봇 시뮬레이션 도구

Fig. 9 Simulation tool of six-legged robot

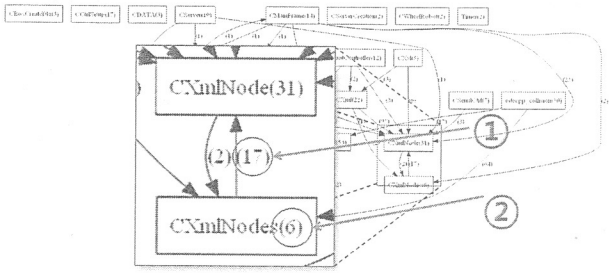


그림 10 육족로봇 시뮬레이터의 코드 구조도 1
Fig. 10 Six-legged robot simulator's code architecture 1

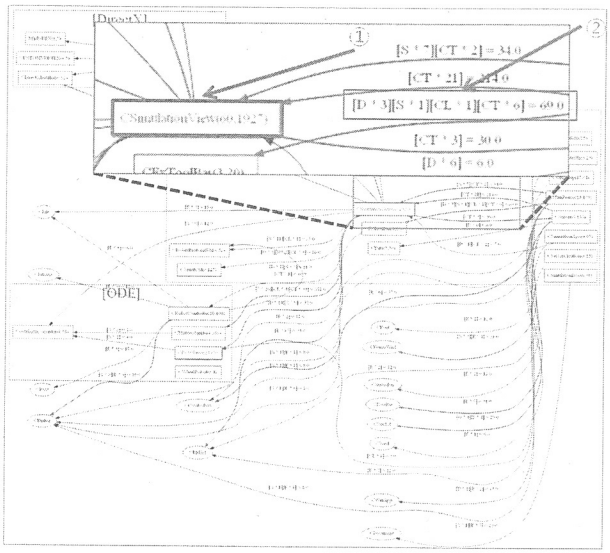


그림 11 육족로봇 시뮬레이터의 코드 구조도 2
Fig. 11 Six-legged robot simulator's code architecture 2

후 다관절 로봇의 모션을 입력하면 해당 환경에서 동작을 한다. 또한 모델링 도구인 HiMEM[12]과 TCP/IP 네트워크로 연결이 가능하다. 이를 통해, Pre-Testing(선 테스팅) 방법을 제안했다[13].

그림 10은 기존 툴 체인을 사용해서 얻어낸 6족 로봇 시뮬레이터 코드의 구조도이다. 이 구조도는 모듈의 정의가 되지 않았기 때문에 툴 체인 내부에 임의의 모듈로 정의해 출력한 이미지이다. 이미지의 모듈 간의 숫자는 모듈간의 콜 개수와 메소드 개수이다.

그림 10의 ①은 CXmlNodes에서 CXmlNode 모듈의 의존도(호출횟수)를 표현한 것이다. 그림 11의 ②는 CXmlNodes이 메소드의 개수를 표현한 숫자이다. 그림 12는 결합도를 정의하고 적용한 구조도이다.

그림 11의 ①의 2가지 숫자는 모듈의 라인수와 모듈의 함수의 개수를 의미한다. 원래는 응집도를 표시 하는데 응집도를 적용하지 못해 라인수와 함수의 개수로서 모듈의 크기를 대체하였다. 그리고 그림 11의 ②는 모듈간의 결합도를 표현하는 수치인데, 각 모듈 간의 결합도를 약자(D: Data, S: Stamp, CL: Control, E: External, CN: Common, CT: Content)로 표현하고 각각의 지표에 맞춰 점수가 표현된다. 그리고 일정이상의 점수 이상이 되면 빨간색으로 표시되어 소프트웨어 공학적으로 품질이 좋지 않은 파트를 구분할 수 있다. 현재 모든 결합도를 한 번에 표시할 수 있고 각각 표시도 가능하다. 가시화를 통해 코드의 결합도 척도를 알 수 있어 높은 결합도는 낮은 결합도로 리펙토링이 가능하다.

6. 결론 및 향후 연구

현재 국내의 소프트웨어 학계는 고품질 소프트웨어를 위해, 개발/테스트 프로세스, 성숙도 측정 등에 초점을 두고 있으나, 현실적으로 산업계의 고품질 문제를 해결못하는 실정이다. 이런 방향은 개발자 관점에서 개발이외의 추가업무가 된다. 그리고 특히 기존 레가시 시스템을 고품질화하기 위한 대안이 될 수 없다. 이를 해결하고자, 우리가 제안한 툴 체인을 통해서 임의의 모듈을 정의하고, 소프트웨어의 구조기반의 코드 복잡도, 모듈간의 관계빈도를 숫자화, 더 나아가 모듈도 정의와 결합도의 품질지표로 코드의 품질을 수치화를 통한 소프트웨어 가시화를 하였다. 이를 통해 나쁜 습관의 개발자들도 스스로 리펙토링하여 코드 복잡도를 낮추는 일이 가능하다. 향후 연구에서는 결합도의 대한 더 많은 경우의 수를 추가하고 모듈의 라인수로 수치화 하는 것만 아니라 응집도에 대한 정의를 하여 응집도를 수치화 할 예정이다. 그리고 타깃의 소스 프로그래밍으로부터 설계 그리고 아키텍쳐 추출과 가시화로 역공학의 기틀을 만들 것이다. 이를 통해 기존의 레가시 시스템의 고품질화에 방향을 제시하고자 한다.

References

[1] NIPA SW Engineering Center, SOFTWARE ENGINEERING WHITE BOOK : KOREA 2013, NIPA, 2013.

[2] SN group. Source Navigator User Guide [Online]. Available: http://sourcenav.sourceforge.net/

[3] GV team. GraphViz User's Guide [Online]. Available: http://www.graphviz.org/Documentation.php

[4] SQLite. About SQLite [Online]. Available: http://www.sqlite.org/about.html

[5] C. Collberg, S. Kobourov, J. Nagra, J. Pitts, K. Wampler, "A System for Graph-Based Visualization of the Evolution of Software," *Proc. of The SoftVis '03 ACM symposium on Software visualization*, pp. 77-86, 2003.

[6] M. Lanza, "The Evolution Matrix: Recovering Software Evolution using Software Visualization Techniques," *Proc. of The IWPSE '01 4th International Workshop on Principles of Software Evolution*, pp. 37-42, 2001.

[7] B. Bruegge, A .Dutoit, Object-Oriented Software Engineering Using UML, Patterns and Java Third Edition, 3rd Ed., Pearson, 2010.
[8] Geon-Hee Kang, Keun Sang Yi, Dong Ho Kim, Jun Sun Hwang, Young Soo Kim, Young B. Park, R. Young Chul Kim, "A Practical Study on Tool Chain for Code Static Analysis on Procedural Language," *Proc. of The KIISE KCC2014*, pp. 559-561, 2014. (in Korean)
[9] Roger S. Pressman, Software Engineering: A Practitioner's Approach, 7th Ed., Pressman, 2009.
[10] Hyun Seung Son, Woo Yeol Kim, Robert Young Chul Kim, "Semi-Automatic Software Development based on MDD for Heterogeneous Multi-Joint Robots," *Proc. of The International Symposium on Control and Automation*, pp. 93-98, 2008. (in China)
[11] Woo Yeol Kim, Hyun Seung Son, R. Young Chul Kim, C. R. Carlson, "MDD based CASE Tool for Modeling Heterogeneous Multi-Jointed Robots," *Proc. of The IEEE Computer Society CSIE 2009*, Vol. 7, pp. 775-779, 2009. (in USA)
[12] Hongik SELab, Hongik MDA based Embedded Software Development Methodology (HiMEM) [Online]. Available: http://selab.hongik.ac.kr
[13] Jae Soo Kim, embedded System For Small heterogeneous Moving Object based on Modeling & Simulation, Doctoral dissertation, Hongik University Graduate, 2009.

강 건 희

2014년 홍익대학교 컴퓨터정보통신공학과(학사). 2014년~현재 홍익대학교 소프트웨어공학 석사과정. 관심분야는 소프트웨어 공학, 역공학, SW Visualization, 소프트웨어 리펙토링

김 영 철

2000년 Illinois Institute of Technology (IIT)(공학박사). 2000년~2001년 LG산전 중앙연구소 Embedded system 부장 2001년~현재 홍익대학교 컴퓨터정보통신 교수. 관심분야는 테스트 성숙도 모델(TMM), 모델 기반 테스팅, 메타모델, 비즈니스 프로세스 모델

이 근 상

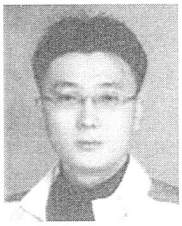
2003년~2005년 유비젠(주) 대표이사. 2008년~현재 (재)전북테크노파크 산업자원지원팀장. 2014년~현재 홍익대학교 소프트웨어공학 박사과정. 관심분야는 소프트웨어 모델링, 임베디드 소프트웨어 자동화 도구, SW Visualization, 메타모델 설계 및 모델변환, 모델변환, 모바일에이전트

김 영 수

1994년 광운대학교 전자계산학과(석사) 2010년~현재 정보통신산업진흥원 SW 공학센터, 정보처리기술사(시스템 응용), 현장 멘토링 총괄, SW Visualization 품질 멘토링. 관심분야는 임베디드 소프트웨어공학, 역공학/재공학 기법 연구

박 용 범

1991년 N.Y. Polytechnic University Science & Engneering(Ph.D). 1993년~현재 단국대학교 컴퓨터 과학과 교수. 관심분야는 Intelligent Software Engineering, Security Software



손 현 승

2007년 홍익대학교 컴퓨터정보통신(학사) 2009년 홍익대학교 일반대학원 소프트웨어공학(석사). 2009년~현재 홍익대학교 일반대학원 소프트웨어공학 박사과정. 관심분야는 임베디드 소프트웨어 자동화 도구 개발, 임베디드 RTOS 개발, 메타모델 설계 및 모델 변환, 모델 검증 기법 연구

KIISE Transactions on Computing Practices

VOLUME 21, NUMBER 2, FEBRUARY 2015

Recommendation of Best Empirical Route Based on Classification of Large Trajectory Data ..... Kye Hyung Lee, Yung Hoon Jo, Tea Ho Lee, Heemin Park 101

Short Papers

A Reexamination on the Influence of Fine-particle between Districts in Seoul from the Perspective of Information Theory ..... Jaekoo Lee, Taehoon Lee, Sungroh Yoon 109

Proposal : Improvement of Testing Frontier Capability Assessment Model through Comparing International Standards in Software Product and Software Testing Process Perspective ..... Hyung-jin Yoon, Jin-Young Choi 115

Recommendation Algorithm by Item Classification Using Preference Difference Metric ..... Chan-soo Park, Taegyu Hwang, Junghwa Hong, Sung Kwon Kim 121

Randomness based Static Wear-Leveling for Enhancing Reliability in Large-scale Flash-based Storage ..... Kilmo Choi, Sewoog Kim, Jongmoo Choi 126

Analysis of Timed Automata Model-based Testing Approaches and Case Study ..... Hanseok Kim, Eunkyoung Jee, Doo-Hwan Bae 132

Parallelization and Performance Optimization of the Boyer-Moore Algorithm on GPU ..... Yosang Jeong, Nhat-Phuong Tran, Myungho Lee, Dukyun Nam, Jik-Soo Kim, Soonwook Hwang 138

Domain Question Answering System ..... Seunghyun Yoon, Eunhee Rhim, Deokho Kim 144

A Practical Study on Code Static Analysis through Open Source based Tool Chains ..... Geon-Hee Kang, R. Young Chul Kim, Geun Sang Yi, Young Soo Kim, Yong. B. Park, Hyun Seung Son 148

Smartphone-User Interactive based Self Developing Place-Time-Activity Coupled Prediction Method for Daily Routine Planning System ..... Beom-Jin Lee, Jiseob Kim, Je-Hwan Ryu, Min-Oh Heo, Joo-Seuk Kim, Byoung-Tak Zhang 154

Adaptive Speech Emotion Recognition Framework Using Prompted Labeling Technique ..... Jae Hun Bang, Sungyoung Lee 160

KOREAN INSTITUTE OF INFORMATION SCIENTISTS AND ENGINEERS