Go 언어 기반 블록체인 코드의 품질 검증을 위한 효율적인 정적분석기 개발

안현식*, 박지훈**, 박보경*, 김영철***
*홍익대학교 소프트웨어공학연구실
**한국정보통신기술협회(TTA)
***홍익대학교 소프트웨어융합학과
e-mail:(*ahn, *park, ***bob)@selab.ac.kr, **jh91082@tta.or.kr

Constructing Effective Code Analyzer to Measure the Quality of Blockchain Code based on Go Language

Hyun-sik An*, Jihoon Park**, Bokyung Park*, R. Young-chul Kim***
*, ***SE Lab, Dept. of software and Communications Engineering, Hongik University
**Telecommunications Technology Association

요 약

현재 4차 산업 혁명과 가상화폐에 대한 전 세계적인 관심으로 블록체인 시스템이 급부상하고 있다. 현재 구현중심인 국내외 블록체인 시장에서 무수히 많은 블록체인 기반 플랫폼들이 등장과 함께 오류가 발생하고 있다. 하지만 블록체인 시스템의 신뢰성, 확장성, 안정성 등에 대한 검증은 누구도 하고 있지 않다. 이런 문제 해결을 위해 Go language로 구성된 블록체인 코드를 분석할 수 있는 정적분석기를 통한 품질 가시화 방법을 제안한다. 이를 통하여 Blockchain Code의 내부 복잡도를 식별하고자한다. 즉, 코드 내부를 가시화하고 개발자가 보다 쉽게 코드를 유지보수 할 수 있으며 블록체인 시스템의 소프트웨어 공학적인 고품질화가 가능하다.

1. 서론

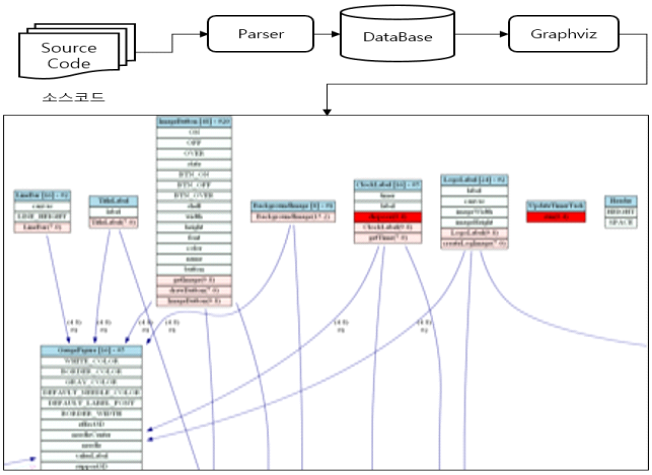
프로젝트 개발에 소프트웨어 공학적인 방법을 적용 하여 소프트웨어의 품질을 높여야 한다는 점은 모두가 인지하고 있다. 하지만 어떤 방법으로 품질을 향상시켜야하는지는 고려하지 않고 있다. 최근 4차 산업혁명으로 새롭게 비즈니스 시장에 나온 블록체인도 마찬가지다. 현재 구현중심인 블록체인 국내외 시장은 수많은 블록체인 기반 플랫폼들이 개발되고 있지만, 블록체인 시스템의 신뢰성, 확장성, 안정성 등에 대한 중요한 기술 검증은 누구도 하고 있지 않다. 이런 기술들을 검증하기 위해 소프트웨어공학적 접근을 통한 높은 품질의 블록체인을 구현해야한다. 높은 품질의 소프트웨어를 만들기 위해선 소프트웨어 품질 측정 및 개선 기술이 필요하다.

이러한 문제를 해결하기 위해 본 연구에서, 블록체인 코드의 정적 분석을 통한 소프트웨어 내부 구조 가시화 방법을 제안한다. Go language로 구현된 블록체인 시스템을 오픈 소스인 Go Parser를 사용해 내부 복잡도를 식별하고 소프트웨어 품질 측정을 위한 결합도를 가시화한다. 2장에서는 관련연구로 아키텍처 가시화와 Go AST Viewer, Go Parser에 대해 설명한다. 3장에서는 Go language로 구현된 블록체인 코드의 가시화 방법에 대해 자세히 설명한다. 마지막으로 4장에서는 결론 및 향후 연구에 대해 언급한다.

2. 관련연구

2.1 아키텍처 가시화

아키텍처 가시화는 전체 소프트웨어 시스템을 정적분석하여 구조를 가시화한다[1]. 우선 소스 코드의 정보를 추출할 수 있는 정적 분석기가 필요하다. 정적 분석기는 소스 코드의 정보를 클래스, 변수, 구조체, 메서드 등으로 구분지어 데이터를 추출한다. 이 데이터들 중 소프트웨어 내부 구조를 가시화 하는데 필요한 정보들을 데이터베이스(SQLite)에 저장한다[2]. 필요한 정보를 쿼리문을 사용하여 SELECT하고 추출된 데이터들을 통해 Dotscript를 생성한다. 생성된 Dotscript로 Graphviz에서 가시화 그래프를 생성할 수 있다. [그림 1]은 아키텍처 가시화의 구성도이다.



(그림 1) 아키텍처 가시화 도구

2.2 Go AST Viewer

Go AST Viewer은 Go 코드를 Abstract Syntax Tree 구조로 가시화하여 보여주는 웹 기반의 오픈소스다[3]. Go Parser로 파싱을 할 때 해당 문구의 AST Node 정보가 있어야 파싱이 가능하다. [그림 2]는 소스코드의 return문구를 AST구조에서 ReturnStmt의 노드 형태로 찾는 것을 보여주는 그림이다. 왼쪽의 입력창에 파일을 업로드하거나 소스코드를 입력하면 오른쪽에 AST구조가 나오는 형태이다.

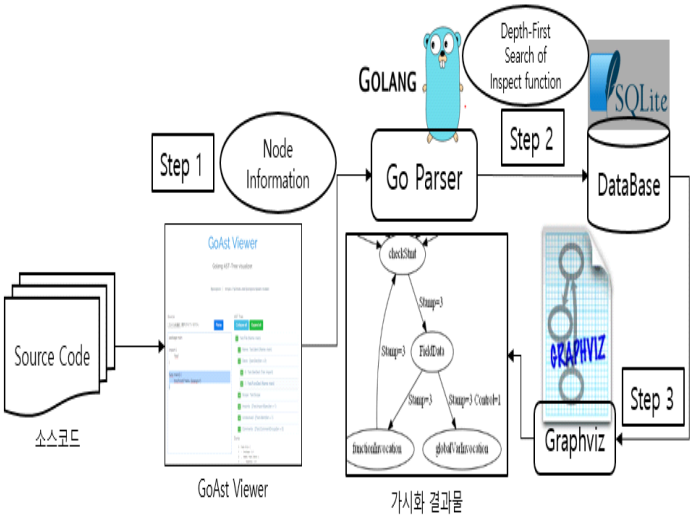
(그림 2) Go AST Viewer 사용화면

2.3 Go Parser

Go Parser는 Go language를 Abstract Syntax Tree 구조로 파싱할 수 있는 오픈 소스 패키지 및 라이브러리이다[4]. 코드 내에서 패키지를 import해서 사용한다. Go language에서 특정 코드가 절차지향 코드인지 아닌지 판별하는데 도움을 받을 수 있다. Go AST Viewer에서 찾은 Node의 정보를 통해 Inspect 함수의 모든 노드 깊이우선 탐색 과정을 통해 원하는 Node를 조건문으로 파싱할 수 있다. 파싱된 데이터는 데이터베이스에 저장된다.

3. Go lang으로 구현된 블록체인 코드 가시화

[그림 3]은 Go Language로 구성된 블록체인 코드를 정적분석하기 위한 프로세스이다. Step 1에서 소스코드를 Go AST Viewer에 입력하여 Node 정보를 파악한다. Step 2에서 파악한 Node정보를 Go Parser를 사용해 Inspect함수의 깊이 우선 탐색을 통해 데이터베이스에 저장한다. Step 3에서 저장된 데이터들을 쿼리문으로 SELECT하여 DotScript를 작성한다.



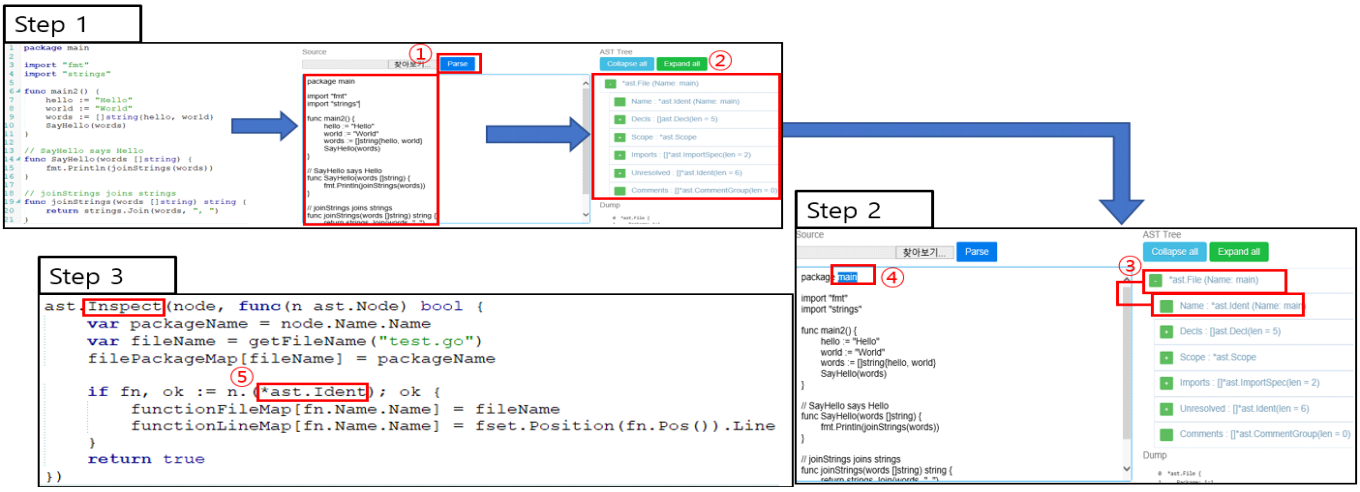
(그림 3) Go Language로 구성된 블록체인 코드를 정적분석하기 위한 프로세스

- Go Parser를 사용한 트리 구조 분석

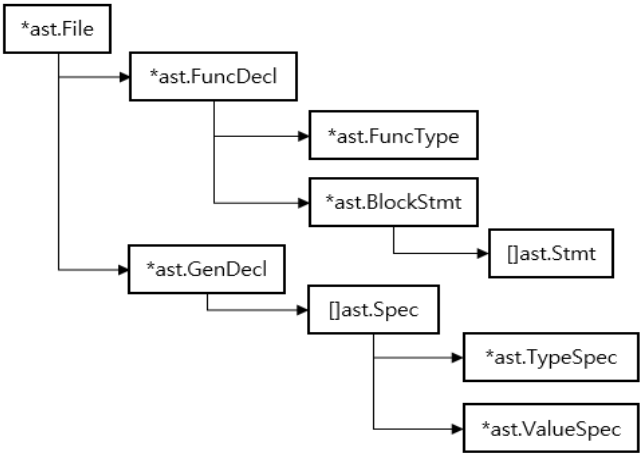
Go AST Viewer로 코드를 AST구조로 가시화하여 분석하고 Go Parser로 파싱을 한다. 아래 [그림 4]은 Go Parser로 파싱하는 과정이다.

먼저 [그림 4]에서 보여준 것처럼 아래와 같다.

- Step 1의 ①에서 보이는 것과 같이 소스코드를 Go AST Viewer에 입력하고 파싱하면 ②처럼AST구조가 나오게 된다.
- Step 2에서 ③을 클릭하면 ④에서 해당 Node가 드래그상태로 맵핑(mapping)되어 나타난다.
- Step3에서, 이렇게 찾은 Node를 Step 3의 ⑤에서 Inspect함수의 모든 노드 깊이우선 탐색을 통해 Step 2의 ③에서 찾은 Ident노드를 if문으로 추출이 가능하다. 이런 방식으로 결합도를 도출하기 위해 필요한 데이터를 데이터베이스에 저장할 수 있도록 정제과정을 거친다. 다음 [그림 5]는 Go Parser로 분석한 Go 코드의 구성도이다.



(그림 4) Go Parser 파싱 흐름도



(그림 5) Go Parser로 분석한 Go 코드 구성도

Go 코드는 *ast.File이 최상위 노드이다. 하위 노드는 *ast.FuncDecl과 *ast.GenDecl로 구분할 수 있다. 각각 함수의 정보, ast.Spec 배열로 구성되어있다. FuncType에는 함수의 타입, BlockStmt에는 함수의 Body 즉, 내용에 대한 정보로 구성되어있다. 함수의 Body는 여러 개의 Statement 배열로 구성되어있다. ast.Spec 배열에는 TypeSpec, ValueSpec으로 구성되어있고 각각 변수의 이름, 타입, 값으로 구성되어있다. 각각 파싱된 데이터를 데이터베이스에 저장한다.

결국에 정적분석 적용사례는 다음의 [그림 6]는 실제 코드를 정적분석하여 함수마다 어떤 결합도가 존재하는지 가시화 결과물이다.

사용한 소스코드는 데이터베이스를 연결하는 Database.go와 이를 실행하는 main.go 두 파일을 사용했다. 가시화 결과물에서 확인할 수 있듯이 FieldData 함수가 GlobalVarInvocation 함수를 호출할 때 Stamp Coupling이 3번, Control Coupling이 1번 발생한다. 이를 통해 재사용성 측면에서 상대적으로 좋지 않은 Control Coupling을 Data Coupling이나 Stamp Coupling으로 변경하여 소프트웨어의 품질을 높일 수 있다.

4. 결론 및 향후연구

최근 4차 산업혁명과 가상화폐에 대한 전 세계적인 관심으로 블록체인 시스템에 대한 관심도 증가하고 있다. 블록체인 시스템은 탈중앙화로 구성되어 데이터 위변조에 대해 완전무결한 시스템으로 착각하는 문제가 발생한다. 블록체인의 분산장부 시스템으로 인해 여러 명의 승인을 통한 높은 보안성을 자랑하지만 특정 한 명에게 부하가 생겨 승인하는데 지체된다면 블록체인 시스템에 문제가 생긴다.

본 연구에서 이러한 문제를 해결하기 위해서 Go lang을 정적분석하여 코드에서 소프트웨어공학적인 방법으로 코드의 결합도를 추출하는 방법을 제안했다. 이로 인해 어떤 코드에 안 좋은 결합도가 존재하는지 알 수 있고 보다 높은 품질의 소프트웨어를 만들 수 있다.

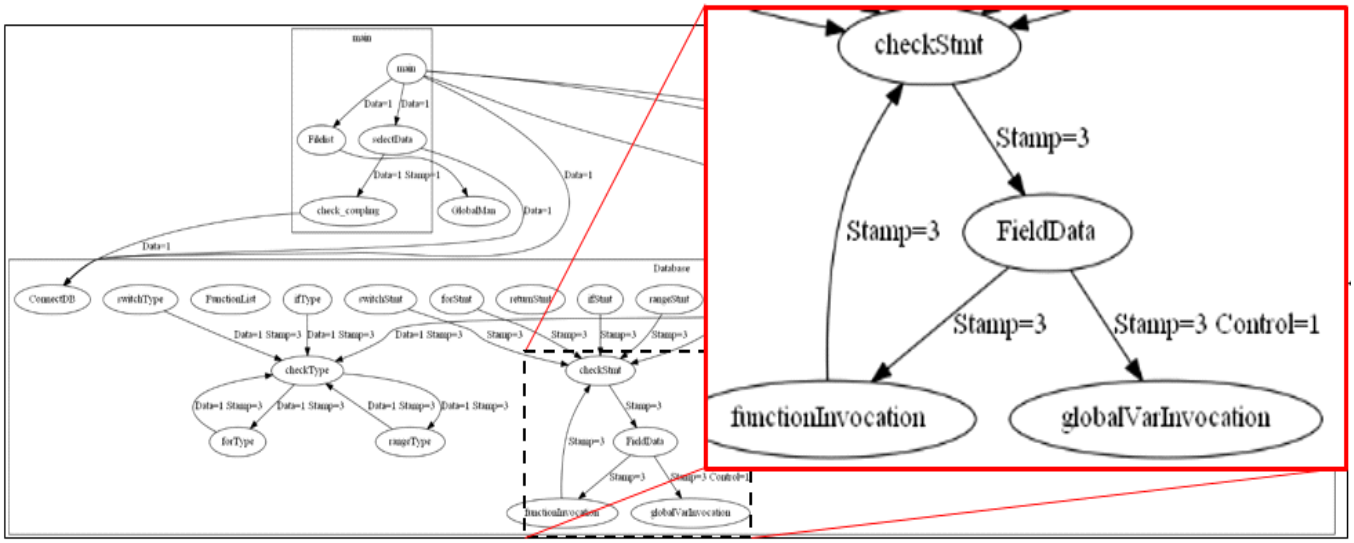
지금은 소프트웨어 고품질을 위한 지표로써 결합도 한 가지만 제시한 상태이다. 향후 연구로 결합도 뿐만 아니라 응집도, 복잡도, Bad Smell 등 소프트웨어의 품질을 떨어트릴 수 있는 지표들을 추가하면서 더 정확하고 세밀한 정적분석을 통해 블록체인 시스템의 소프트웨어 공학적인 고품질화 연구를 진행할 예정이다.

ACKNOWLEDGE

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

참고문헌

[1] 박지훈, "정적 분석 기반 블록체인 코드 취약성의 자동 가시화 연구", 2018
[2] SQLite, https://www.sqlite.org/
[3] Go AST Viewer, http://goast.yuroyoro.net/
[4] Go Parser, https://golang.org/pkg/go/parser
[5] Graphviz, http://www.graphviz.org/



(그림 6) 실제 적용사례 가시화 결과물