오픈소스 소프트웨어 구성요소 탐지와 모듈성 측정 연구

황동훈 [1], 우승훈 [2]
[1] 고려대학교 SW AI 융합대학원 석사과정
[2] 고려대학교 정보대학 컴퓨터학과 교수

dhhwang1@korea.ac.kr, seunghoonwoo@korea.ac.kr

A Study on Open Source Software Component Detection and Modularity Measure

Dong-Hun Hwang[1], Seunghoon Woo[2]
[1]Graduate School of SW AI Convergence, Korea University
[2]Dept. of Computer Science and Engineering
College of Informatics, Korea University

요 약

최근 소프트웨어(Software, S/W) 제품 개발에 오픈소스 소프트웨어(Open Source Software, OSS)의 활용이 증가하고있다. 그러나 활용된 오픈소스의 구조, 구성 오픈소스의 중복 여부 및 사용된 오픈소스의 내부 연관관계를 파악하지 못하여 새롭게 작성된 S/W 제품의 안정성, 보안성, 유지보수성에 중대한 영향을 미칠 수 있다. 본 연구에서는 OSS 의 구성을 파악하고 연관관계를 밝혀내어 안정적인 S/W 제품을 만들기 위한 모듈성(Modularity) 분석 기법을 제안한다.

인기 OSS 인 MongoDB 를 대상으로 한 분석 결과, 버전이 올라갈수록 모듈화 수준은 점진적으로 개선되었으며, 제안한 모듈성 분석 기법이 이러한 변화를 효과적으로 포착함을 확인하였다.

다만 외부 의존성과 중복 코드는 여전히 구조적 안정성을 저해하는 요인으로 작용하고 있어, 향후 안정성과 유지보수성을 높이기 위해서는 의존성 관리와 중복 코드 제거가 필요함을 시사한다.

1. 서론

최근 S/W 개발은 비용과 시간을 절감을 위해 OSS 를 소스 그대로 가져다 사용하거나 라이브러리(Library)를 가져다 사용하는 등 다양한 방식으로 활용하여 작업을 하고 있다. 이러한 방법은 개발 비용 절감과 신속한 서비스 제공이라는 장점을 제공하지만, 동시에 복잡한 의존성(dependency) 문제를 야기한다. 특히 OSS 프로젝트 내부에 또 다른 OSS 가 중복하여 포함되는 경우가 있음에도 불구하고, 이러한 구조적 관계를 명확히 인지하지 못한 채 사용하는 사례가 빈번하다. 이는 S/W 의 안정성, 보안성, 유지보수성에 중대한 영향을 미칠 수 있다.

모듈성은 네트워크 이론에서 사용되는 개념으로, 노드(node)와 엣지(edge)의 연결성을 기반으로 모듈 간 응집도(Cohesion)와 결합도(Coupling)를 수치화하는 기법이다.

본 연구는 이러한 문제를 해결하기 위해 OSS 구성요소를 식별하고 구성요소 간의 관계를 체계적으로 파악하고, 이를 기반으로 한 모듈성 분석 방법론을 제안한다.

2. 연구방법 및 분석

본 논문에서는 OSS 의 버전별 OSS 프로젝트의 구조적 변화와 모듈간 연관성을 도출하였다.

방향성 네트워크 모듈화 공식에 기반하여 OSS 간의 종속 관계를 분석하고, 버전별로 변화하는 모듈성 지표(Q 값)를 추적하였다.

OSS 간의 종속 관계 확인을 위하여 OSS 식별 및 구성요소 관계의 네트워크 그래프화하였고, 도출된 그래프의 모듈성 측정은 방향성 네트워크 모듈성지표를 확장하여 적용하였다.

S/W 버전별 소스 내부 구성요소와 그 연결성을 분석을 위하여 MongoDB 버전 5.0.0, 6.0.0, 7.0.0, 8.0.0 을 이용하여 분석하였다. 해당 S/W 는 NoSQL 의 한 종류로서 대표적인 OSS 중 하나이다.

3. OSS 구성요소 탐지 (SCA: Software Composition Analysis)

OSS 프로젝트의 구성요소를 탐지하기 위해 CNEPS[1]를 이용하였다. CNEPS 는 C/C++ 기반 오픈소스 프로젝트에서, 정확한 의존 관계 분석을 가능케 하는 모듈 단위 기반 탐지 기법으로 주요특징으로는 다음과 같다.

(1) **모듈 단위 정의:** 각 모듈은 헤더 파일과 해당 헤더를 포함하는 소스 파일들의 집합으로 정의되며, 분석의 기본 단위로 활용
(2) **중복 OSS 탐지:** 동일한 오픈소스가 여러 경로에서 중복 포함된 경우 이를 식별하여 구조 분석의 정확성을 확보
(3) **의존성 그래프 구축:** 모듈 간 참조 관계를 기반으로 방향성 그래프를 구성하여 전체 OSS 의존성 구조를 표현

이러한 과정은 단순 파일이나 심볼 수준의 의존성 분석을 뛰어넘어, OSS 수준의 연계 구조를 정확히 파악할 수 있는 모듈성 분석의 기반 데이터를 제공한다.

4. 모듈성지표 계산

OSS 내부 모듈 간 응집도와 결합도를 측정하기 위해 Leicht & Newman(2008)[2]에서 제안한 방향성 네트워크 모듈성지표를 확장하여 적용하였다. 모듈성지표(Q)는 다음과 같이 정의된다.

$$Q = \frac{1}{m}\sum_{u,v}\left[w_{ij}\cdot A_{uv} - \frac{k_u^{out}k_v^{in}}{m}\right]\delta(c_u,c_v)$$

(그림 1) 모듈화 지표(Q)

여기서 A_{uv}는 노드 u에서 v로의 연결 여부 k_u^{out}, k_v^{in}은 각 노드의 출입 차수, m은 전체 엣지 수, $\delta(c_u,c_v)$는 두 노드가 동일 모듈에 속하는 경우 1, 그렇지 않으면 0 이다.

본 연구에서는 다음의 추가 가중치를 적용하여 OSS 구조적 특성을 반영하였다.

$$w_{ij} = \begin{cases} 1 = & 내부\text{-}내부\ 연결\ (일반\ OSS\ 내부\ 의존) \\ w_{dup} = & 중복\ OSS\ 노드\ 간\ 연결 \\ w_{ext-int} = & 외부\ OSS \leftrightarrow 내부\ OSS\ 연결 \\ w_{ext-ext} = & 외부\ OSS\ 간\ 연결 \end{cases}$$

(그림 2) 가중치 목록

중복 노드 가중치(w_{dup}): 동일 기능을 수행하는 복제 코드에 대해 기여도를 감소시켰다.

외부 의존 엣지 가중치($w_{ext-int}$, $w_{ext-ext}$): 외부 OSS 간 연결 또는 외부와 내부 간 연결이 과도할 경우 가중치를 부여하여 모듈 경계 왜곡을 보정하였다.

모듈성지표(Q)는 네트워크가 얼마나 뚜렷한 모듈(커뮤니티) 구조를 가지는지를 나타내는 값으로 이론적으로 -1.0 에서 1.0 사이의 범위를 가지며, 값의 범위와 의미는 다음과 같다.

(1) **Q > 0.5 (높은 모듈화)**: 연결 없는 개별 모듈의 집합으로, 노드들이 모듈별로 뚜렷하게 구분되어 있으며, 모듈 내부에서만 강하게 연결됨
(2) **0 < Q ≤ 0.5 (부분적 모듈화)**: 모듈 내부 연결성도 강하지만, 일부 노드는 다른 모듈과도 연결됨으로 여전히 모듈 구조는 존재하나, 외부 의존성이 점차 증가하는 상황
(3) **Q ≤ 0 (모듈 간 의존성 우세)**: 모듈 내부보다 모듈 간 연결이 더 많아 모듈 경계가 불분명하고, 구조적 안정성이 약화된 상태
(4) **Q ≤ -0.5 (매우 낮은 모듈화)**: 모듈 간 의존성이 복잡하게 얽혀, 사실상 모듈 경계가 의미 없음

따라서 Q 값이 높을수록 내부 구조가 모듈간 의존성이 없는 개별 모듈화된 집합임을 의미하며, 낮을수록 모듈 간 의존성이 크고 구조적 안정성이 떨어짐을 시사한다.

5. 연구대상 S/W 버전별 분석 결과

연구대상 S/W 의 버전 5.0.0 부터 8.0.0 까지 4 개 주요 배포 버전에 대해 모듈성지표(Q)를 산출하였다.

버전	노드	엣지	모듈성(Q)
5.0.0	26	55	-0.0677
6.0.0	36	74	-0.0686
7.0.0	50	95	-0.0537
8.0.0	45	58	-0.0362

<표 1> 버전별 Q 지표

연구대상 S/W 에서는 OSS 특성상 모듈간 의존성이 많아 전체적으로 음수로 표시 되었으나, 버전간 비교해 보면, 버전이 개정 될수록 OSS 구조가 점차 개선되어 Q 값도 개선되고 있음을 알 수 있다(표 1).

CNEPS 로 분석된 내부 노드와 엣지 연결상태를 도식화 해보면 Q 값이 가장 낮은 5.0.0 버전 모듈 구조는 그림 3 와 같다. 상대적으로 Q 값이 높은 8.0.0 버전

의 모듈 구조는 더 복잡해졌으나 모듈성이 더욱 뚜렷해 짐을 알 수 있다(그림 4).

6. 실험 결과의 시사점

연구대상 S/W 의 버전별 모듈성지표(Q)값을 추적한 결과, 시간이 지남에 따라 점진적으로 구조가 개선되는 흐름을 확인할 수 있었다.

예를 들어, 2013 년에 발표된 5.0.0 버전의 Q 값은 -0.0677 로 매우 낮았다.

이는 해당 시기에 모듈 간 의존성이 복잡하고, 중복 OSS 가 다수 존재하여 구조적 응집도가 떨어졌음을 의미한다.

반면, 2022 년에 발표된 8.0.0 버전의 Q 값은 -0.0362 로 개선되었다.

이는 중복 코드 제거와 외부 의존성 관리가 강화되면서 모듈화 수준이 이전보다 높아졌기 때문이다.

비록 여전히 음수 영역에 머물러 있어 완전한 모듈화에는 이르지 못했지만, 장기간의 프로젝트 진화를 통해 OSS 의 구조가 점진적으로 안정화되는 경향을 보여준다.

따라서 Q 값은 OSS 의 진화 과정에서 구조적 품질을 추적하는 유효한 지표라 할 수 있다.

7. 결론

본 연구는 OSS 프로젝트의 내부 구성요소와 연결성을 네트워크 관점에서 분석하고, 모듈성지표를 활용하여 구조적 품질을 정량화 하였다. 연구대상 S/W 사례 분석을 통해 버전별 구조적 진화를 확인했으며, 다음과 같은 결론을 얻었다.

사례분석에서 확인했듯이 연구대상 S/W 버전의 개정에 따라 모듈화가 점진적으로 개선되지만, 외부 의존성과 중복 코드는 여전히 구조적 안정성을 저해하는 요소로 작용한다.

모듈성지표는 OSS 구조의 모듈성을 정량적으로 측정하는 데 유용하였다.

향후 연구에서는 다양한 OSS 를 대상으로 한 모듈성 분석과, AI 기반 의존성 탐지 및 모듈 구조 최적화 기법 개발이 필요하다.

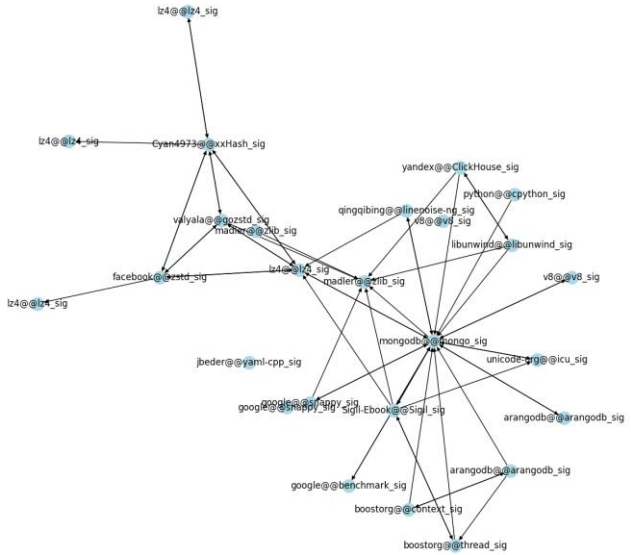
참고문헌

[1] Y.J Na, S.H Woo, J.M Lee, H.J Lee, “CNEPS: A Precise Approach for Examining Dependencies among Third-Party C/C++ Open-Source Components”, ICSE '24, 2024.

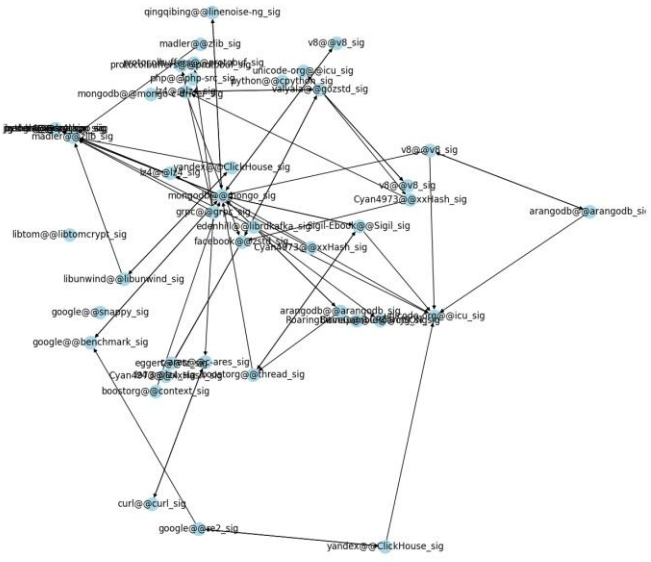
[2] E. A. Leicht, M. E. J. Newman, “Community structure in directed networks” Physical Review Letters, vol. 100, no. 11, pp. 118703, 2008.

[3] M. S. Zanetti, F. Schweitzer, “A Network Perspective on Software Modularity,” ARCS Workshops 2012, pp. 175-186

[4] M. Newman, “Networks: An Introduction,” Oxford, Oxford University Press, 2010.



(그림 3) 5.0.0 버전 모듈 구조도



(그림 4) 8.0.0 버전 모듈 구조도