소스코드 기반 소프트웨어 품질 측정 방법 및 도구 개발

지도교수 : 채흥석

팀 명 : 졸업참쉽조

201624567 전민기

201724492 성민우

201524593 천동혁

목차

1. 요구조건 및 변경사항

1.1. 요구조건

소스코드 기반 소프트웨어의 품질을 측정하는 방법을 조사한 후 측정 결과를 알려주는 도구를 개발한다.

- MISRA-C, SEI CERT, ISO/IEC 5055:2021를 활용하여 품질 요구사항과 코딩규칙 매핑
- 품질 지수 측정 계산법 개발
- 품질 측정 결과를 시각화하는 기능 구현

1.2. 변경사항

사용 표준 추가 (ISO/IEC 5055:2021)

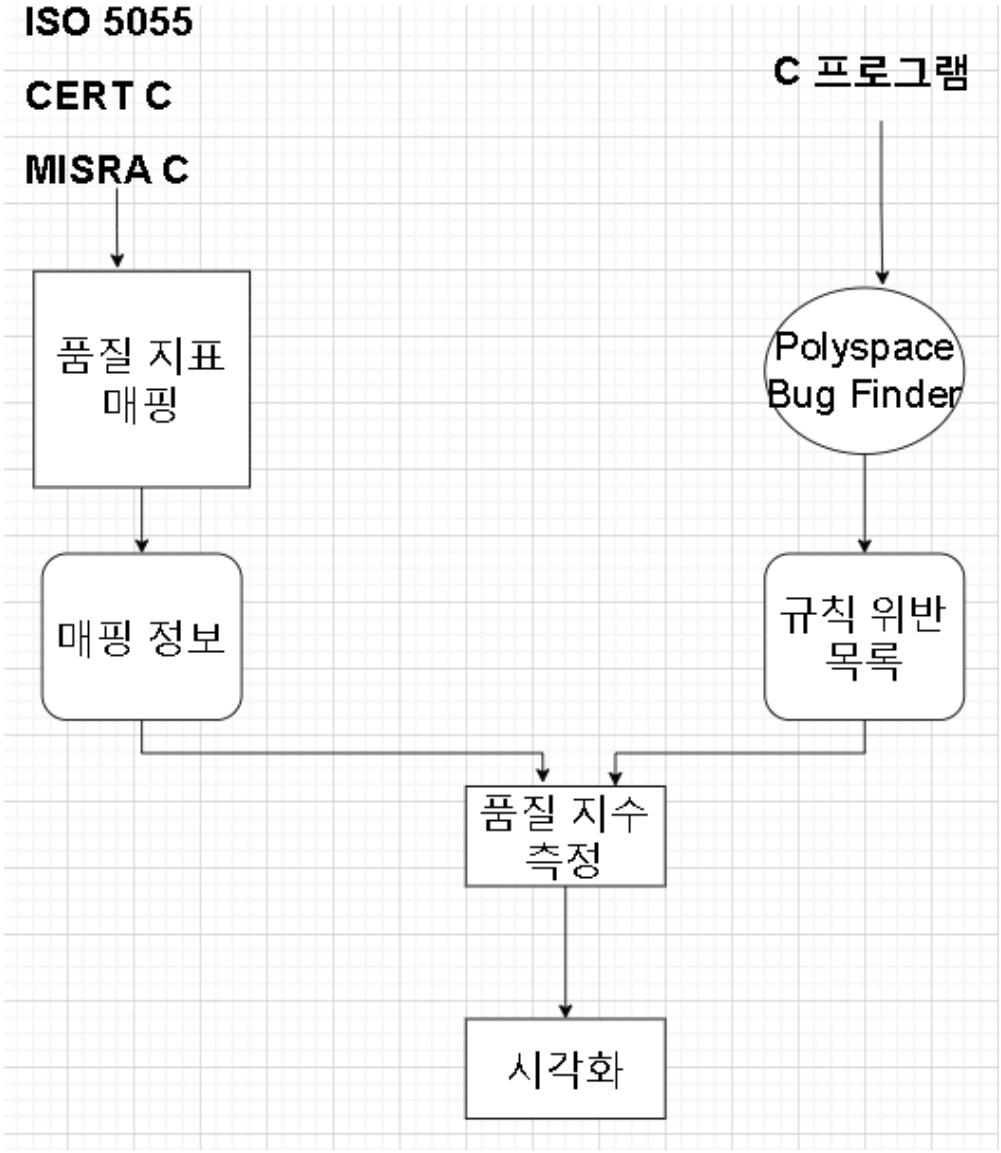
품질 측정의 정확도를 높이고자 MISRA C 2012, SEI CERT C에 ISO/IEC 5055:2021을 추가하여 품질 속성의 매핑에 활용한다.

ISO/IEC 5055:2021은 소프트웨어 품질 평가 및 소스코드 평가를 위한 표준으로서 우리가 품질 평가에 이용할 ISO/IEC 25010:2011 Quality Model의 8가지 품질 속성 중 4가지에 대한 규칙 목록을 제공한다. (Maintainability, Performance Efficiency, Reliability, Security)

해당 규칙들은 이 표준만의 식별자를 가지고 있지 않지만 CWE의 식별자를 가지고 있으며 CWE란 Common Weakness Enumeration으로 보안에 영향을 줄 수 있는 일반적인 소프트웨어 및 하드웨어의 취약점 유형에 대한 목록이다.

Polyspace에서 ISO 5055 표준의 준수 여부를 직접 검사할 수는 없지만 CWE의 준수 여부는 검사할 수 있었기 때문에 CWE의 준수 여부를 통해 간접적으로 확인할 수 있다고 판단하여 Coding Standards와 Quality Model의 품질속성의 매핑에 이용하고자 한다.

2. 설계 상세화



2.1. 품질 속성 매핑

본 프로젝트에서 매핑이란 품질 요구사항과 코딩 규칙들이 어떤 관련이 있는지 파악하여 서로 연결하는 것을 매핑이라고 정의한다. 각각의 코딩 규칙들을 살펴보고, 코딩 규칙들과 매핑했다.

그 매핑 기준은 각각 다음과 같다. 우선적으로 분류한 품질 요구사항들을 간단히 정리해보았다.

Maintainability(유지보수성) : 지정된 수정 사항을 만들기 위해 필요한 노력에 영향을 미치는 속성 집합

Performance Efficiency(성능 효율성) : 지정된 조건 하에서 소프트웨어의 성능 수준과 사용된 자원 양 간의 관계에 영향을 미치는 속성 집합

Reliability(신뢰성) : 지정된 기간 동안 명시된 조건 하에서 소프트웨어의 성능 수준을 유지할 수 있는 능력에 영향을 미치는 속성 집합

Security(안전성) : 제품이나 시스템이 정보와 데이터를 보호하여 사람이나 다른 제품 또는 시스템이 해당하는 권한 유형과 수준에 적합한 데이터 액세스 수준을 갖도록 하는 정도

Portability(이식성) : 소프트웨어가 한 환경에서 다른 환경으로 이전될 수 있는 능력에 영향을 미치는 속성 집합

출처 : ISO 25010 (iso25000.com)

ISO/IEC 9126 - Wikipedia

2.1.1 SEI CERT C

CERT C는 The CERT C Secure Coding Standard가 정의이기 때문에 안전한 코딩을 하는 것을 목표로 한다. 따라서 별도의 표기가 없는 가이드라인은 기본적으로 Security에 매핑을 하였다.

Undefined Behavior라고 표시된 가이드라인은 [ISO/IEC 9899:2011]에 따르면, 프로그램의 동작은 다음 표에 기술된 상황에서 정의되지 않기 때문에 프로그램이 동작하지 않고, 이는 오류로 볼 수 있다. 따라서 Reliability에 관련이 있다고 매핑 하였다.

또한 Unspecified Behavior라고 표시된 가이드라인은 Portability와 관련이 있다고 매핑 하였다. [ISO/IEC 9899:2011]에 따르면, 컴파일러에 따라 동작이 달라질 수 있기 때문이다.

그리고 ISO 5055, MISRA C와 관련 있는 경우를 모두 표시하여 추후 활용할 수 있도록 하였다. 예를 들어, CWE-562에서 DCL30-C와 관련이 있으므로 CWE-562에 Security 관련 품질 요구사항이 없더라도 DCL30-C에 존재하므로 Security 속성과도 관련이 있다고 볼 수 있다.

2.1.2 ISO 5055

CWE는 다른 표준과는 다르게 Portability를 제외한 4개의 품질요구사항에 따라 규칙들을 분류하였다. 그래서 분류된 결과를 토대로 각각의 규칙을 품질요구사항에 매핑 하였다. 그리고 한가지의 규칙이 여러 가지의 품질요구사항을 나타내기도 한다.

이 역시 CERT C와 관계있는 규칙을 확인할 수 있었으므로 표시하여 매핑에 활용할 수 있도록 한다.

2.1.3 MISRA C:2012

MISRA C:2012 표준 문서에서 일부 가이드라인은 Undefined, Unspecified, Implementation의 표시가 있는데 이는 ISO 9899:1990 및 ISO 9899:1999 표준의 내용을 기반으로 하고 있을 때 표시된다. 이에 대한 내용은 MISRA C 표준 문서의 6.10.1 ISO C portability issue references에서 다루고 있으며 각 항목에 대한 설명을 포함하고 있는데 다음과 같다.

-Unspecified

이는 컴파일은 성공적으로 이루어지지만 컴파일러에 따라 동작이 달라질 수 있으며 한 컴파일러에서도 모든 구성에서 일관적으로 동작한다고 할 수 없는 경우이다.

-Undefined

이는 컴파일러에 의해 오류가 표시되지 않을 수 있지만 본질적으로 프로그래밍 오류에 속한다.(함수에 대한 유효하지 않은 매개변수 등)

-Implementation

이는 Unspecified와 유사하게 컴파일러마다 동작이 다를 수 있다. 하지만 한 컴파일러 내에서는 일관된 접근을 취하고 문서화해야 한다는 차이가 있다.

한편 이러한 표시가 없는 가이드라인 역시 존재한다. 이러한 경우 MISRA C 표준 문서의 vision에서 표준에 대해 설명하기를 높은 신뢰성이 요구되는 개발에 사용할 수 있다고 하였다. 즉, 따로 표시되지 않았다면 MISRA C 표준의 가이드라인은 기본적으로 신뢰성(Reliability) 품질 속성에 연관이 있다고 볼 수 있다.

따라서 정리하면 Unspecified와 Implementation은 공통적으로 프로그래밍 오류는 아니지만 컴파일러에 따라 의도하지 않은 동작을 취할 수 있는 문제를 가지고 있다. 즉, 이식성(Portability) 품질 속성과 연관이 있다. Undefined는 검출되지 않았을 수 있을 뿐 프로그래밍 오류에 해당한다. 의도하지 않은 동작을 하거나 아예 동작하지 않을 수 있으므로 신뢰성(Reliability) 품질 속성과 연관이 있다. 아무런 표시가 없는 경우 신뢰성(Reliability) 품질 속성과 연관이 있는 것으로 본다.

이러한 표시는 각 가이드라인에 대한 내용에서도 확인할 수 있지만 Appendix G와 Appendix H에서도 표시하고 있으므로 양 쪽 모두를 확인하여 각 가이드라인에 대해 연관되는 품질 속성을 체

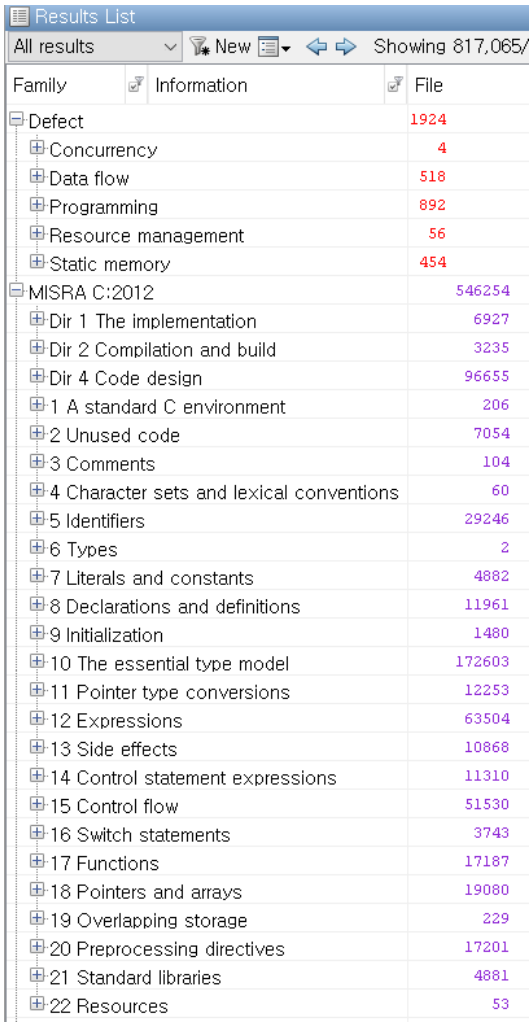
크할 수 있었다.

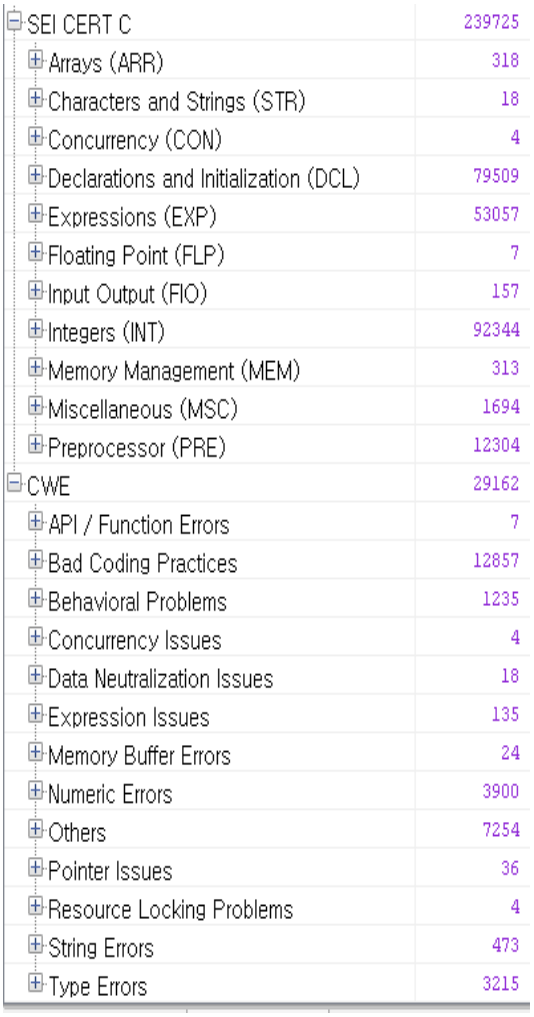
2.1.4 매핑데이터 통합

각각의 표준에 대한 매핑이 완료되면 셋을 통합한 테이블을 만든다. 이 때 ISO 5055와 SEI CERT C에서 표기한 Related guideline에 대한 정보를 활용할 수 있다. 서로 유사한 가이드라인이라면 영향을 주는 품질 속성 역시 공유하고 있다고 판단할 수 있는 것이다. 따라서 연관된 가이드라인에서 체크된 품질 속성을 추가로 체크할 수 있다.

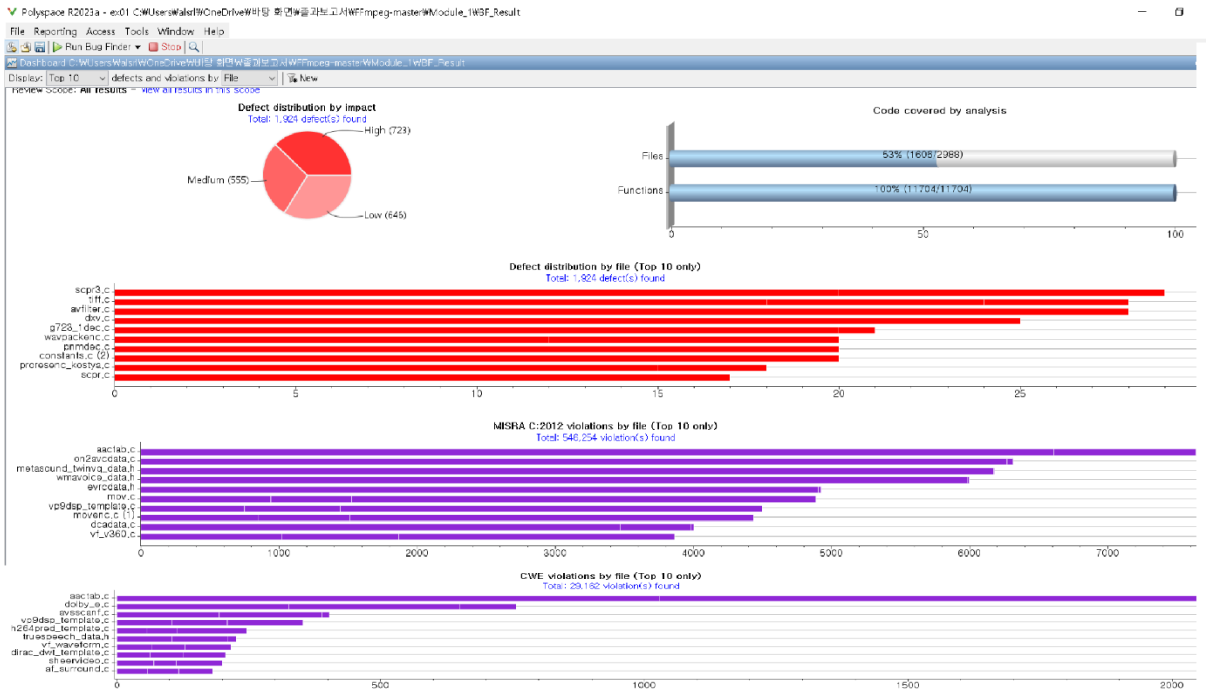
단 이때 ISO 5055와 SEI CERT C 사이에는 Relate가 Exact인 일부 가이드라인이 존재한다. 이는 두 가이드라인이 완전히 일치한다는 뜻으로서 품질 속성의 통합 체크는 진행하되 SEI CERT C 쪽의 가이드라인만 남기도록 한다. 같은 규칙의 위반이 양 쪽에서 검출되면 실제보다 위반 횟수가 높게 계산되어 점수에 왜곡이 생길 수 있기 때문이다.

2.2 Polyspace 분석 결과



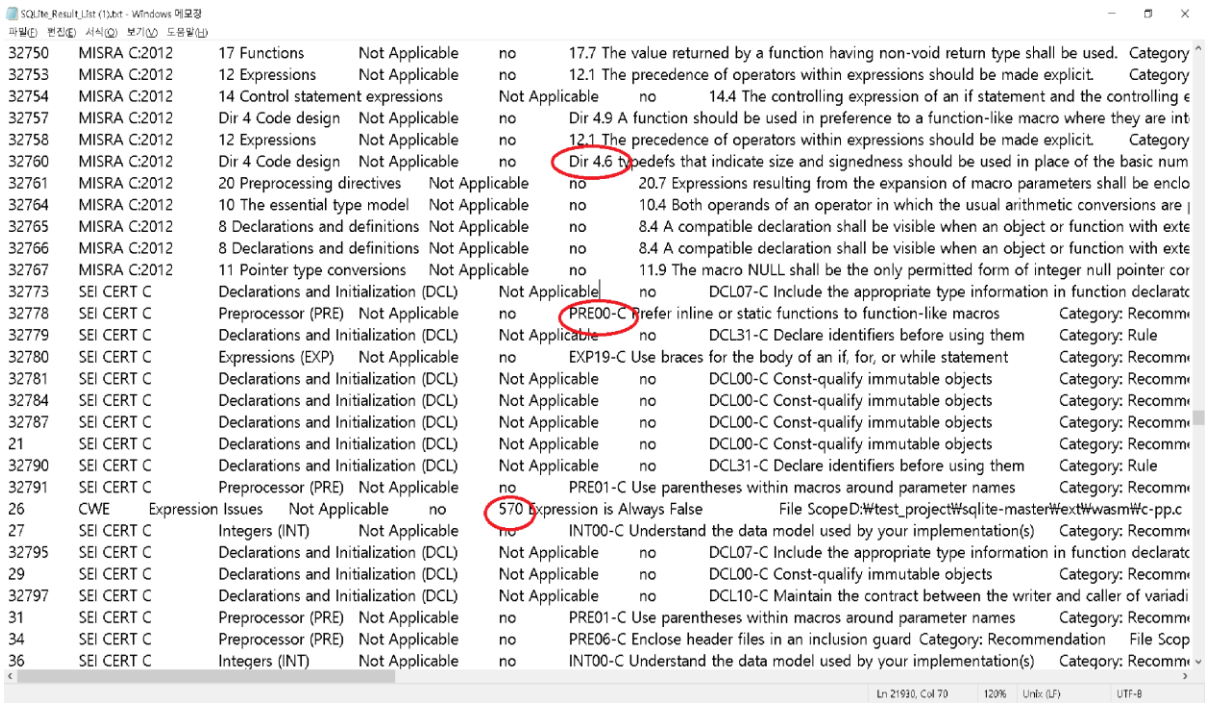


FFmpeg 프로젝트의 C로 작성된 부분만을 대상으로 분석을 진행하였고, MISRA C와 546,254개의 규칙, CERT C와 239,725개의 규칙, CWE와 29,162개의 규칙과 관련이 있음을 알 수 있다. 또한 각각 어떤 규칙들이 위반되었는지 파악할 수 있다.



FFmpeg 프로젝트를 분석하였을 때 위반 사항이 가장 많이 일어나는 파일들도 파악할 수 있다.

또한 결과를 txt로도 분석해 보았다.



32750	MISRA C:2012	17 Functions	Not Applicable	no	17.7 The value returned by a function having non-void return type shall be used. Category
32753	MISRA C:2012	12 Expressions	Not Applicable	no	12.1 The precedence of operators within expressions should be made explicit. Category
32754	MISRA C:2012	14 Control statement expressions	Not Applicable	no	14.4 The controlling expression of an if statement and the controlling e
32757	MISRA C:2012	Dir 4 Code design	Not Applicable	no	Dir 4.9 A function should be used in preference to a function-like macro where they are int
32758	MISRA C:2012	12 Expressions	Not Applicable	no	12.1 The precedence of operators within expressions should be made explicit. Category
32760	MISRA C:2012	Dir 4 Code design	Not Applicable	no	Dir 4.6 typedefs that indicate size and signedness should be used in place of the basic num
32761	MISRA C:2012	20 Preprocessing directives	Not Applicable	no	20.7 Expressions resulting from the expansion of macro parameters shall be enclo
32764	MISRA C:2012	10 The essential type model	Not Applicable	no	10.4 Both operands of an operator in which the usual arithmetic conversions are
32765	MISRA C:2012	8 Declarations and definitions	Not Applicable	no	8.4 A compatible declaration shall be visible when an object or function with exte
32766	MISRA C:2012	8 Declarations and definitions	Not Applicable	no	8.4 A compatible declaration shall be visible when an object or function with exte
32767	MISRA C:2012	11 Pointer type conversions	Not Applicable	no	11.9 The macro NULL shall be the only permitted form of integer null pointer cor
32773	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL07-C Include the appropriate type information in function declarat
32778	SEI CERT C	Preprocessor (PRE)	Not Applicable	no	PRE00-C Prefer inline or static functions to function-like macros Category: Recomm
32779	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL31-C Declare identifiers before using them Category: Rule
32780	SEI CERT C	Expressions (EXP)	Not Applicable	no	EXP19-C Use braces for the body of an if, for, or while statement Category: Recomm
32781	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL00-C Const-qualify immutable objects Category: Recomm
32784	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL00-C Const-qualify immutable objects Category: Recomm
32787	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL00-C Const-qualify immutable objects Category: Recomm
21	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL00-C Const-qualify immutable objects Category: Recomm
32790	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL31-C Declare identifiers before using them Category: Rule
32791	SEI CERT C	Preprocessor (PRE)	Not Applicable	no	PRE01-C Use parentheses within macros around parameter names Category: Recomm
26	CWE	Expression Issues	Not Applicable	no	570 Expression is Always False File ScopeD:₩test_project₩sqlite-master₩ext₩wasm₩c-pp.c
27	SEI CERT C	Integers (INT)	Not Applicable	no	INT00-C Understand the data model used by your implementation(s) Category: Recomm
32795	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL07-C Include the appropriate type information in function declarat
29	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL00-C Const-qualify immutable objects Category: Recomm
32797	SEI CERT C	Declarations and Initialization (DCL)	Not Applicable	no	DCL10-C Maintain the contract between the writer and caller of variadi
31	SEI CERT C	Preprocessor (PRE)	Not Applicable	no	PRE01-C Use parentheses within macros around parameter names Category: Recomm
34	SEI CERT C	Preprocessor (PRE)	Not Applicable	no	PRE06-C Enclose header files in an inclusion guard Category: Recommendation File Scop
36	SEI CERT C	Integers (INT)	Not Applicable	no	INT00-C Understand the data model used by your implementation(s) Category: Recomm

위와 같이 규칙들에 대한 설명과 어떤 규칙들이 위반되었는지를 바로 확인할 수 있다. 이 결과를 이용해 관리자는 개발자들의 작업이 표준에 따라 안전하게 진행되고 있는지 확인할 수 있고, 개발자들은 본인의 코드에서 어떤 부분이 표준에 적합하지 않아 오류를 일으킬 수 있는지 파악할

수 있어 생산성을 향상시킬 수 있다.

2.3 품질 점수 측정

V – 규칙 위반
V_S – 해당 규칙의 심각도(1..3)
V_C – 해당 규칙을 위반한 횟수에 따른 점수(1..3)
S_V – 위반한 규칙 V의 점수
$S_V = \frac{V_S \times V_C}{9}$
$Q_{m,e,r,s,p}$ = 각각의 품질 속성의 점수
$G_{m,e,r,s,p}$ – 각각의 품질속성에 해당하는 규칙의 개수
F – 함수, F_V – 함수가 위반한 규칙의 종류의 수
M – 모듈, M_V – 모듈이 위반한 규칙의 종류의 수
M_F – 모듈에 포함된 함수 개수, M_S – 모듈의 품질 점수
S – 시스템, S_M – 시스템에 포함된 모듈 개수, S_S – 시스템의 품질 점수

$$F_{Q_m} = 1 - \frac{\sum_{k=1}^{F_V} S_{V_k}}{G_m}, F_Q = \frac{F_{Q_m} + F_{Q_e} + F_{Q_r} + F_{Q_s} + F_{Q_p}}{5}$$

$$M_{S_m} = \frac{M_{Q_m} + \frac{\sum_{k=1}^{M_F} F_{Q_m}}{M_F}}{2}, S_{S_m} = \frac{\sum_{k=1}^{S_M} M_{S_m}}{S_M}$$

위 그림은 품질 점수를 측정하기 위한 방법을 정리한 표이다. 먼저 MISRA-C를 예로 들면 Mandatory, Required, Advisory 등 각각의 가이드라인에는 중요도가 있다. 이에 따른 점수를 1~3으로 부여하고, 같은 규칙을 위반한 횟수를 범위로 나누어 1~3의 점수를 부여한다. 그러면 해당 규칙의 위반에 의한 점수는 둘을 곱한 뒤 9로 나눈 값이 된다. (0~1 의 범위를 유지하기 위해 나누었다)

이러한 위반에 의한 점수를 품질 속성 별로 합산한 뒤 해당 품질 속성에 관여하는 규칙의 수로 나누어주면 전체 규칙 대비 위반한 규칙의 점수가 얻어진다. 그리고 이를 1에서 제하면 위반하지 않은, 즉 해당 품질의 점수를 얻을 수 있다.

함수의 경우 이렇게 얻어진 각 품질의 점수를 평균 내어 전체 점수를 계산한다. 모듈의 경우 모듈 자체의 File Scope 규칙 위반에 따른 품질 점수를 모듈에 속한 함수의 품질 점수의 평균과 합한 뒤 2로 나누어 계산한다. 시스템의 경우 별도의 위반을 포함하지 않기 때문에 속한 모듈의 품질 점수의 평균을 각각의 품질 점수로 삼아 계산한다.

2.4 추가 품질 관련 요소

Polyspace에는 Coding Standards의 위반 이외에도 Code Metrics에 대한 측정 내용을 제공한다. Cyclomatic Complexity나 LOC, NPath등 다양한 요소를 확인할 수 있는데 이에 대한 threshold를 설정하여 품질과 관련된 추가적인 정보를 제공할 수 있다.

이와 관련된 정보는 MISRA Report 5 Software Metrics 문서에서 확인할 수 있다. 각각의 Metrics의 설명과 허용 가능한 값의 기준에 대해 다루고 있으므로 각 Metrics의 threshold는 해당 문서의

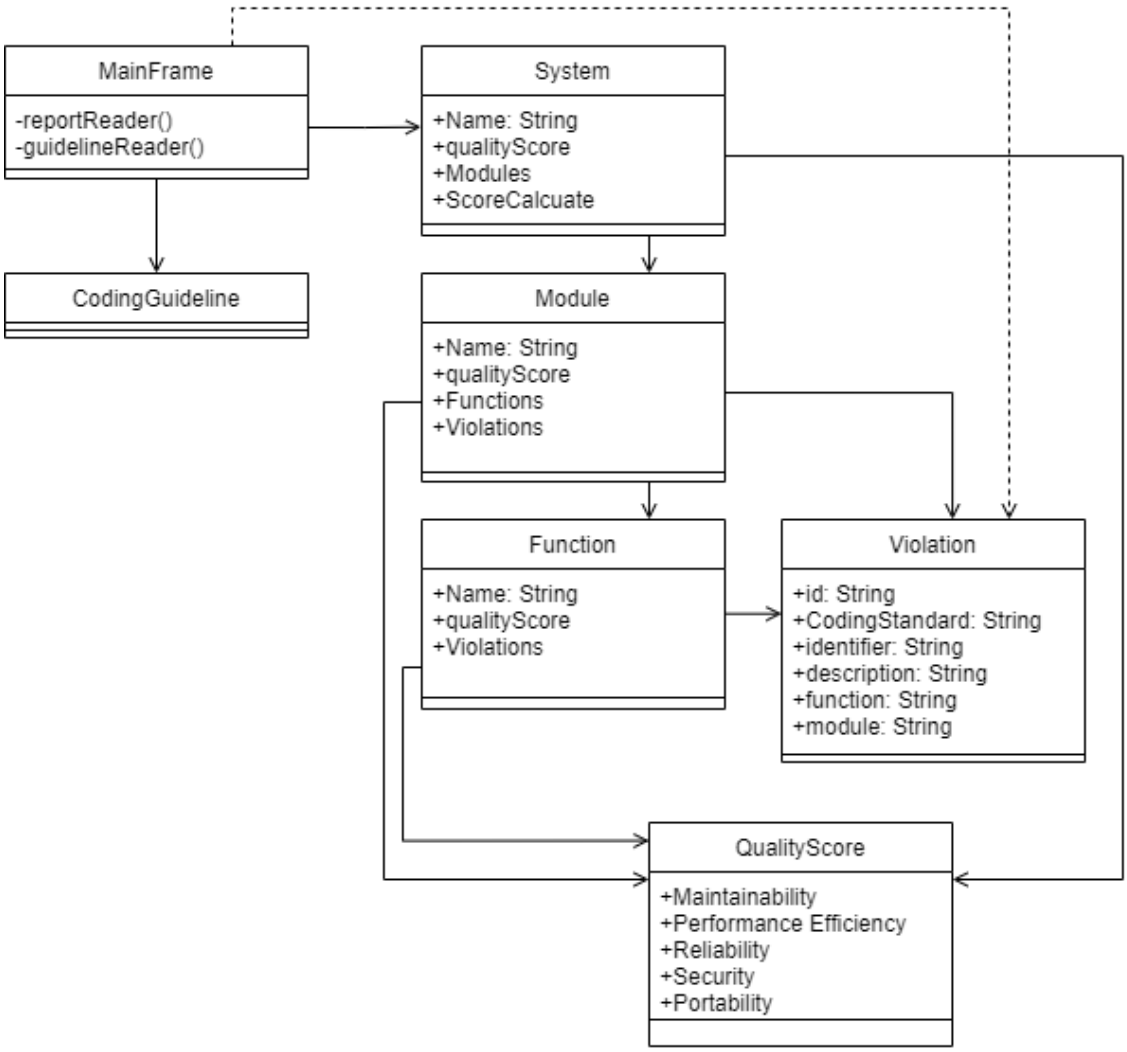
값을 참조하여 설정할 수 있을 것이다.

2.5 측정 결과 시각화

품질 점수의 계산이 완료되면 최종적으로는 이를 사용자가 확인할 수 있도록 하여야 한다. 이에는 시스템, 각 모듈, 함수별로 측정된 전체 점수, 품질별 점수, 위반한 규칙을 확인할 수 있는 구성이 먼저 필수적으로 들어가야 한다.

이외에는 2.4에서 언급한 Code Metrics의 값이 threshold를 넘겼는지에 대한 여부와 구체적인 값에 대해 확인할 수 있도록 만들고자 한다.

3. 구현



3.1 Polyspace 규칙 위반 검사 결과

ID	Family	Group	Check	Function	File
32768	MISRA C:2012	Dir 4 Code design	Dir 4.10 Precautions shall be taken in order to prevent the contents of a header file being included more than once.	File Scope	₩sqlite-master₩test₩tt3_checkpoint.c

32806	MISRA C:2012	11 Pointer type conversions	11.9 The macro NULL shall be the only permitted form of integer null pointer constant.	mergeWorkerFinishHierarchy()	₩sqlite-master₩ext₩lsm1₩lsm_sorted.c
2219	SEI CERT C	Integers (INT)	INT00-C Understand the data model used by your implementation(s)	uintCollFunc()	₩sqlite-master₩ext₩misc₩uint.c
35132	CWE	Numeric Errors	197 Numeric Truncation Error	mergeWorkerLoadHierarchy()	₩sqlite-master₩ext₩lsm1₩lsm_sorted.c

위 표는 Polyspace를 통한 분석 결과의 일부로서 필요한 정보만을 가져왔다. ID는 분석 결과 내부의 식별자이고 Family는 어떤 Coding Standards에 대한 위반사항인지를 나타낸다. Group는 해당 Coding Standards 내부에서의 카테고리이며 Check는 Coding Standards 내부의 식별자로 시작한다. MISRA-C의 경우 Directives는 Dir을 붙이며 Rule에는 별도의 첨자 없이 숫자로만 이루어져있다. CERT C는 전체 식별자를 적어두었으며 CWE는 숫자만을 기재한 것을 확인할 수 있다. Funtion, File은 어떤 파일의 어떤 함수에서 위반이 검출되었는지를 표시한다. 개개의 함수가 아닌 파일에 대한 검출인 경우 Function은 File Scope로 표시한다.

이러한 입력데이터는 각 line 별로 Violation 개체를 생성한다. id는 ID, Coding Standard는 Family, identifier와 description은 Check, function은 Function, module은 File의 내용을 통해 입력된다.

3.2 매핑 데이터 파싱 및 저장

2.1에서 작성한 매핑 데이터에서 실제 프로그램에서 사용되는 부분은 어떤 Coding Standard에 속하는지, 식별자는 무엇인지, 어떤 품질 속성에 영향을 주는 지이다. 전체 엑셀 데이터를 사용하지 않으므로 label과 related guideline에 대한 정보를 제외한 내용만 txt 파일의 형태로 입력에 이용한다.

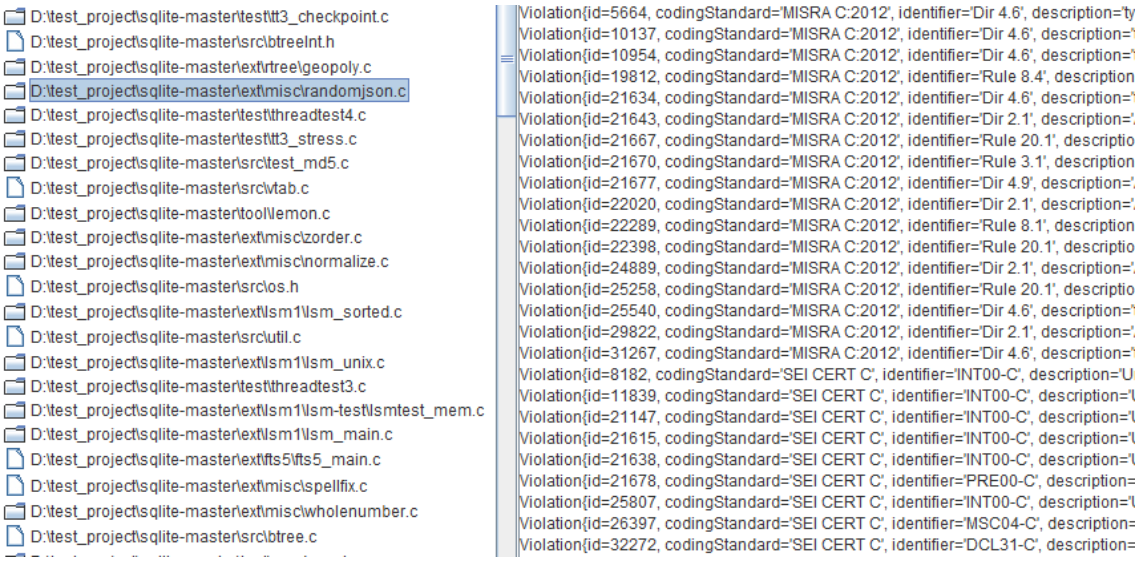
Coding Standard	Identifier	Descripti	Maintainability	Performance Efficien	Reliability	Security	Portability	Severity

		on		cy				
MISRA C:2012	Rule 13.4	…			O		O	Advisory
SEI CERT	MSC00-C	…	O			O		L2
CWE	CWE-404	…		O	O	O		Medium

위 표는 txt에 입력되어야하는 데이터의 label과 내용의 예시이다. 이렇게 입력된 데이터를 line 별로 read하여 각각을 CodingGuideline 개체로 생성하여 list에 저장한다.

CodingGuideline은 속하는 Coding Standard와 Identifier를 String 형태로 가지고 있으며 각 품질 속성에 관여하는지 여부를 Boolean으로 가지고 있다. 생성 시 초기 값은 False이며 매핑 데이터를 read할 때 O로 표시된 항목에 한하여 True로 변경하는 식이다.

3.3 입력 데이터 파싱 및 저장



3.1의 입력 데이터는 3.2의 매핑데이터를 이용하여 파싱을 거친다. 생성된 Violation 개체의 Identifier를 매핑데이터의 CodingGuideline 리스트에서 일치하는 Identifier가 있는지 조회하는 것이다. 만약 존재하지 않는다면 해당 Violation에 대한 품질 속성의 정보가 없으므로 저장하지 않고 넘어간다. 만약 존재한다면 해당 Violation이 가지고 있는 function, module에 대한 정보를 통해 각각을 생성하거나 기존의 개체에 Violation을 추가한다. 이러한 과정을 입력 데이터 전체를 읽을 때까지 반복하면 System과 System에 속한 module의 리스트, 각각의 module과 module의

Violation의 리스트 및 module에 속한 function의 리스트, 그리고 각각의 function과 function의 Violation의 리스트를 얻을 수 있다.

3.4 품질점수 산출

3.2.3의 과정을 거쳐 각 System, Module, Function 개체와 개체의 Violation 리스트가 얻어지면 2.3의 설계에 따라 품질 점수를 산출할 단계이다. 먼저 QualityScore 개체에 Violation 리스트를 입력한다.

```
private Map<String, Integer> guidelineCount =  new HashMap<>();

    public void violationCount(List<Violation> violations) {

        violations.forEach(e-> {  //리스트의 각 Violation에 대하여

            if(guidelineCount.containsKey(e.identifier)) { //이미 포함된 규칙인 경우

                int count = guidelineCount.remove(e.identifier)+1;

                guidelineCount.put(e.identifier,count); //count에 +1하여 저장

            } else {

                guidelineCount.put(e.identifier,1); //없는 경우 1의 count로 저장

            }

        });

    }
```

위 코드는 Violation의 Identifier를 이용하여 각 가이드라인을 몇 번 위반하였는지에 대해 정리하여 저장하는 동작을 한다.

이후 CodingGuideline의 Severity와 위반횟수에 대한 count를 이용하여 해당 Violation의 점수를 계산하고 이를 대응하는 품질속성에 반영한다. 이를 리스트 내의 모든 Violation에 대해 실행하면 해당 Violation 리스트에 의한 품질속성의 감소치를 얻을 수 있다.

이후 2.3의 설계에 따라 계산하되 실제로는 따로 Function을 가지지 않는 Module이나 Violation을 가지지 않는 Module이 존재한다. 이러한 경우 Function을 가지지 않는다면 Function과 같은 방식으로 Violation 리스트의 점수만을 사용하고 Violation을 가지지 않는다면 System과 마찬가지

로 Function 리스트의 점수만을 사용하여 품질점수를 산출한다.

4. 갱신된 과제 추진 계획

5월		6월				7월				8월				9월			
3주	4주	1주	2주	3주	4주	1주	2주	3주	4주	1주	2주	3주	4주	1주	2주	3주	4주
소스코드 확보																	
		MISRA, CERT 조사 Coding Rule 분석															
						Quality Model 품질 요구사항 매핑											
								중간 보고서 작성									
										정적 분석 결과 입력 및 파싱 개발							
										품질지수 및 세부사항 출력 개발							
										세부사항을 출력을 위한 추가 입력 및 파싱 개발							
														품질지수 측정법의 세부 수치 조정			
														테스트 및 디버깅			
																최종 보고서 작성	

5. 보고 시점까지의 과제 수행 내용

5.1 품질 속성 매핑

5.1.1 CERT C

ID	Description	Recommendations/Rules	Reliability	Security	Portability	Related ISO 5055	Related MISRA C	Exact ISO 5055	
EXP20-C	Perform explicit tests to determine success, true and false, and equality	Recommendations		O					
EXP30-C	Do not depend on the order of evaluation for side effects	Rules	O	O	O	CWE-758	Rule 13.2		Undefined Unspecified
EXP32-C	Do not access a volatile object through a nonvolatile reference	Rules	O	O			Rule 11.8		Undefined
EXP33-C	Do not read uninitialized memory	Rules	O	O		CWE-758 CWE-908 CWE-824 CWE-456 CWE-457		CWE-456 CWE-457	Undefined
EXP34-C	Do not dereference null pointers	Rules	O	O		CWE-690 CWE-476		CWE-476	Undefined
EXP35-C	Do not modify objects with temporary lifetime	Rules		O					
EXP36-C	Do not cast pointers into more strictly aligned pointer types	Rules	O	O			Rule 11.1 Rule 11.2 Rule 11.5 Rule 11.7		Undefined
EXP37-C	Call functions with the correct number and type of arguments	Rules	O	O		CWE-628 CWE-685 CWE-686	Rule 8.2 Rule 17.3		Undefined
EXP39-C	Do not access a variable through a pointer of an incompatible type	Rules	O	O	O	CWE-119 CWE-125 CWE-704			Undefined Unspecified
EXP40-C	Do not modify constant objects	Rules	O	O					Undefined
EXP42-C	Do not compare padding data	Rules		O					
EXP43-C	Avoid undefined behavior when using restrict-qualified pointers	Rules	O	O			Rule 8.14		Undefined
EXP44-C	Do not rely on side effects in operands to sizeof, _Alignof, or _Generic	Rules		O	O				Unspecified

SEI CERT C의 코딩 규칙과 품질 요구사항 매핑 테이블의 일부

Maintainablity	0
Performance Efficiency	0
Reliability	53
Security	308
Portability	7

위 표는 SEI CERT C에서 각 품질 지표와 연관 있는 가이드라인의 수를 표시하고 있다. 표준의 특성 상 Security에 연관 되어있는 가이드라인의 수가 가장 많음을 확인할 수 있다.

5.1.2 ISO 5055

	A	B	C	D	E	F	G	H	I	J	K
1	ID	Description	Maintainability	Performance Efficiency	Reliability	Security	Portability	Related CERT C	Related CERT C(CWE More Abstract)	Related CERT C(Exact	Related CERT C(Imprecise)
56	CWE-194	Unexpected Sign Extension			O	O					
57	CWE-195	Signed to Unsigned Conversion Error			O	O					
58	CWE-196	Unsigned to Signed Conversion Error			O	O					
59	CWE-197	Numeric Truncation Error			O	O		INT02-C, INT05-C	FIO34-C, FLP34-C, INT31-C		
60	CWE-248	Uncaught Exception			O						
61	CWE-252	Unchecked Return Value			O	O					ERR33-C, PO
62	CWE-366	Race Condition within a Thread			O	O			CON32-C, CON40-C	CON43-C	
63	CWE-369	Divide by Zero			O	O		FLP03-C		INT33-C	
64	CWE-390	Detection of Error Condition Without Action			O						
65	CWE-391	Unchecked Error Condition			O			ERR00-C	ERR33-C, ERR34-C, POS54-C		FLP32-C
66	CWE-392	Missing Report of Error Condition			O						
67	CWE-394	Unexpected Status Code or Return Value			O						
								MEM00-C, MEM01-			

ISO 5055의 코딩 규칙과 품질 요구사항 매핑 테이블의 일부

Maintainablity	29
Performance Efficiency	18
Reliability	74
Security	73
Portability	0

위 표는 ISO 5055에서 각 품질 지표와 연관 있는 가이드라인의 수를 표시하고 있다. Reliability와 Security의 비중이 높음을 확인할 수 있다.

5.1.3 MISRA C: 2012

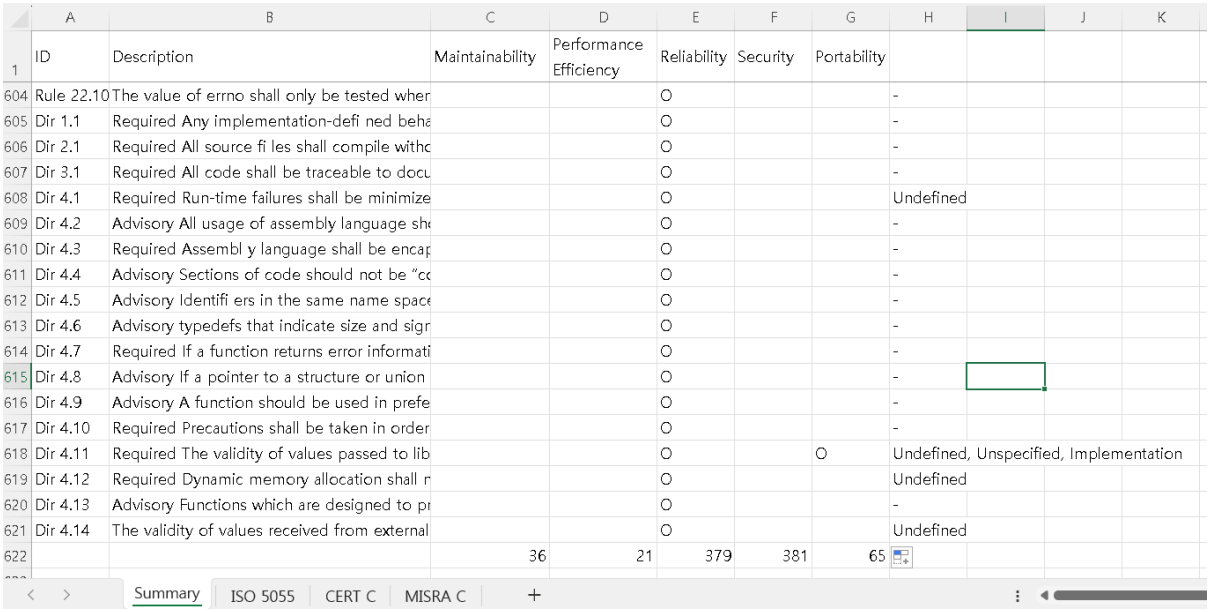
	A	B	E	G	H	I	J	K
1	ID	Description	Reliability	Portability	Portability Issue	AMD 2 Appendix		
167	Dir 4.5	Advisory Identifi ers in the sa	O		-			
168	Dir 4.6	Advisory typedefs that indicat	O		-			
169	Dir 4.7	Required If a function returns	O		-			
170	Dir 4.8	Advisory If a pointer to a stru	O		-			
171	Dir 4.9	Advisory A function should b	O		-			
172	Dir 4.10	Required Precautions shall be	O		-			
173	Dir 4.11	Required The validity of value	O	O	Undefined, Unspecified, Implementation			
174	Dir 4.12	Required Dynamic memory al	O		Undefined			
175	Dir 4.13	Advisory Functions which are	O		-			
176	Dir 4.14	The validity of values receive	O		Undefined			
177			167	38				

MISRA C:2012의 코딩 규칙과 품질 요구사항 매핑 테이블의 일부

Maintainablity	0
Performance Efficiency	0
Reliability	167
Security	0
Portability	38

위 표는 MISRA C:2012에서 각 품질 지표와 연관 있는 가이드라인의 수를 표시하고 있다. 표준의 특성 상 Reliability에 연관 되어 있는 가이드라인의 수가 가장 많음을 확인할 수 있다.

5.1.4 매핑데이터 통합



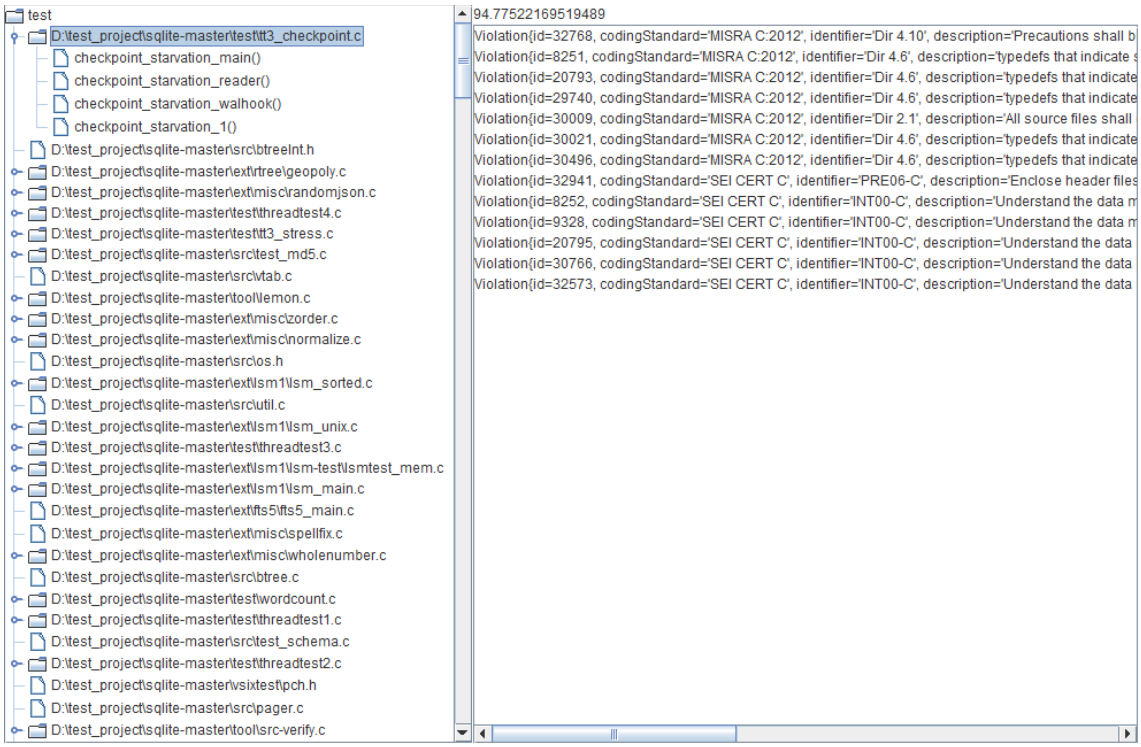
	A	B	C	D	E	F	G	H	I	J	K
1	ID	Description	Maintainability	Performance Efficiency	Reliability	Security	Portability				
604	Rule 22.10	The value of errno shall only be tested wher			O			-			
605	Dir 1.1	Required Any implementation-defi ned beha			O			-			
606	Dir 2.1	Required All source fi les shall compile withc			O			-			
607	Dir 3.1	Required All code shall be traceable to docu			O			-			
608	Dir 4.1	Required Run-time failures shall be minimize			O			Undefined			
609	Dir 4.2	Advisory All usage of assembly language sh			O			-			
610	Dir 4.3	Required Assembl y language shall be encap			O			-			
611	Dir 4.4	Advisory Sections of code should not be "cc			O			-			
612	Dir 4.5	Advisory Identifi ers in the same name space			O			-			
613	Dir 4.6	Advisory typedefs that indicate size and sign			O			-			
614	Dir 4.7	Required If a function returns error informati			O			-			
615	Dir 4.8	Advisory If a pointer to a structure or union			O			-			
616	Dir 4.9	Advisory A function should be used in prefe			O			-			
617	Dir 4.10	Required Precautions shall be taken in order			O			-			
618	Dir 4.11	Required The validity of values passed to lib			O		O	Undefined, Unspecified, Implementation			
619	Dir 4.12	Required Dynamic memory allocation shall r			O			Undefined			
620	Dir 4.13	Advisory Functions which are designed to pr			O			-			
621	Dir 4.14	The validity of values received from external			O			Undefined			
622			36	21	379	381	65				

Summary | ISO 5055 | CERT C | MISRA C

이 모든 것들을 종합하여 하나의 큰 매핑 테이블을 만들었다. 이 매핑 테이블은 ISO 5055, CERT C, MISRA C의 매핑 결과들을 종합하여 각 규칙과 품질 요소들을 매핑 하였으며, 총 621개의 규칙을 정리하였고, 매핑 결과는 다음과 같다.

Maintainablity	36
Performance Efficiency	21
Reliability	379
Security	381
Portability	65

5.2 품질 점수 측정 및 시각화 구현



Polyspace의 규칙 위반 결과를 입력받고 매핑 데이터를 읽어 각 모듈, 함수별로 위반 사항을 저장하는 기능을 구현하였다. 또한 2.5에서 언급한 내용에 따라 각각의 품질 점수를 측정하여 확인할 수 있도록 하였다. 여기에 더해 품질 속성별 점수와 Code Metrics의 내용을 표시할 수 있어야 할 것이며 확인을 용이하게 할 수 있도록 화면 구성에 대한 구상 역시 필요할 것이다.

6. 구성원별 진척도

이름	진척도
전민기	CERT C의 Rules와 Recommendations 정리, 각 Rules와 Recommendations를 품질 요구사항에 매핑
성민우	MISRA C의 Rules를 정리, 각 Rules와 품질 요구사항을 매핑
천동혁	CWE의 규칙들 정리, 각 CWE와 품질 요구사항을 매핑
공통	CERT C, MISRA C, ISO 5055에 대한 스터디 진행, 보고서 작성