

<소스 코드 품질 평가 시스템>

작성자: 지창규

Revision History

Version	Date	Summary
1	2023.02.01	1. Project Overview 작성 2. System Overview 작성 3. Architectural Driver 작성
2	2023.02.14	4. Top Level Design Description 작성
3	2023.02.27	5. Component Level Design Description 작성 6. Architecture Evaluation 작성

내용

1.	Introduction	6
2.	Project Overview	7
2.1.	Project Background.....	7
2.2.	Business Context Diagram	8
2.3.	Stakeholder List	9
2.4.	Business Goal List	10
3.	System Overview.....	11
3.1.	System Context Diagram.....	11
3.2.	External Entity List.....	11
3.3.	External Interface List.....	13
3.4.	System Feature List.....	15
4.	Architectural Driver.....	16
4.1.	Use Case Model	16
4.1.1.	Use Case Diagram	16
4.1.2.	Actor List	17
4.1.3.	Use Case List.....	18
4.1.4.	UC-01 프로젝트 정보 관리	19
4.1.5.	UC-02 프로젝트 결과 조회	20
4.1.6.	UC-03 프로젝트 파일 별 결과 조회.....	21
4.1.7.	UC-04 소스코드 품질 평가	23
4.1.8.	UC-05 저 품질 프로젝트 정보.....	24
4.1.9.	UC-06 시스템 상태 탐지 및 복구	25
4.2.	Quality Attribute Scenario	27
4.2.1.	QA Scenario List.....	27
4.2.2.	QA-01 품질 평가 수행 시간	27
4.2.3.	QA-02 시스템 장애 탐지 및 복구	28
4.2.4.	QA-03 새로운 품질 유형 지원.....	28
4.2.5.	QA-04 새로운 버전관리 시스템 지원	29
4.3.	Constraint	29
4.3.1.	Business Constraint List	29
5.	Top Level Design Description	30
5.1.	System Infrastructure View.....	30
5.1.1.	System Infrastructure Diagram.....	30
5.1.2.	Node Specification.....	30
5.1.3.	Execution Environment Specification.....	33

5.1.4. Communication Path Specification.....	34
5.2. Structure View	36
5.2.1. Overall Structure	36
5.2.2. 소스코드 품질 평가 시스템 관리 서버 – Static Structure Model	36
5.2.3. 품질 평가 준비 서버 – Static Structure Model	40
5.2.4. 품질 분석 서버 – Static Structure Model	44
5.2.5. 품질 분석 결과 동기화/DB 저장 서버 – Static Structure Model	49
5.2.6. 저 품질 프로젝트 알림 서버 – Static Structure Model.....	54
5.2.7. 품질 결과 조회 서버– Static Structure Model	58
5.2.8. 시스템 상태 탐지 서버– Static Structure Model.....	59
5.3. Behavior View.....	61
5.3.1. UC-01 프로젝트 정보 관리 Behavior Model.....	61
5.3.2. UC-02 프로젝트 결과 조회 Behavior Model.....	63
5.3.3. UC-03 프로젝트 파일 별 결과 조회 Behavior Model	64
5.3.4. UC-04 소스코드 품질 평가 Behavior Model.....	66
5.3.5. UC-05 저 품질 프로젝트 정보 Behavior Model	68
5.3.6. UC-06 시스템 상태 탐지 및 복구 Use Case Behavior Model.....	69
5.4. Deployment View	70
5.4.1. Artifact Definition Model.....	70
5.4.2. Artifact Deployment Model	72
5.5. Documenting Design Decisions	72
5.5.1. Design Decision List.....	72
5.5.2. DD-01 품질 평가 성능을 위한 분석 서버 분리	73
5.5.3. DD-02 시스템 장애 탐지를 위한 Tactic 적용.....	78
5.5.4. DD-03 새로운 품질 유형지원을 위한 Component 설계	81
5.5.5. DD-04 새로운 버전관리 시스템 지원을 위한 통합 Gateway 활용	84
6. Component Level Design Description	86
6.1. AnalyzeSourceCode Design Description	86
6.1.1. Overview	86
6.1.2. Static Structure Diagram	87
6.1.3. Element List	88
6.1.4. Design Rationale	89
6.2. VersionControlSystemGateway Design Description	91
6.2.1. Overview	91
6.2.2. Static Structure Diagram	92
6.2.3. Element List	93
6.2.4. Design Rationale	94

7.	Architecture Evaluation.....	97
7.1.	Traceability Summary.....	97

1. Introduction

본 문서는 소스 코드 품질 평가 시스템에 대한 소프트웨어 설계 명세서이다. Project Overview, System Overview, Architectural Drivers, Top Level Design, Component Level Design, Architecture Evaluation을 포함하여 작성되었다.

2. Project Overview

2.1. Project Background

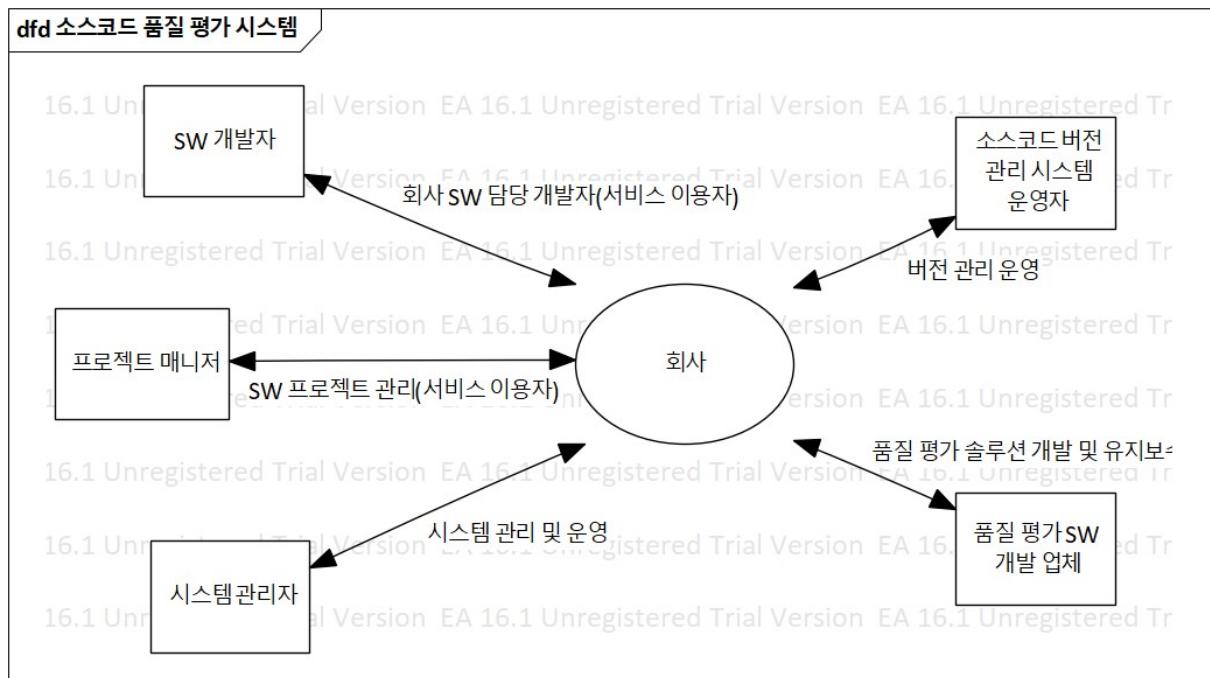
현재 소프트웨어는 변화의 핵심이 되고, 산업에서 차지하는 비중이 커지고 있다. 혁신적인 소프트웨어 기술 확보도 중요하지만, 소프트웨어를 잘 만드는 것이 더욱 중요하다. 소프트웨어 품질이 낮다면 고객에게 신뢰를 높이기 힘들 뿐만 아니라 작은 오작동으로 치명적인 사고가 발생할 수 있다. 한 예로 테슬라 자율주행차를 보게 되면 잘못된 AI 판단으로 사망까지 이르게 된 사건이 있다. 이처럼 중요해지고 있는 소프트웨어 품질을 평가하기 위해선 소스코드 품질 평가 시스템이 필요하다.

여러 회사에서 품질 평가 프로그램을 제공하고 있다. 품질 평가 프로그램에서 사용되는 정적 분석은 Synopsys에서 제공하는 Coverity가 대표적인 솔루션이다. Coverity는 20개 이상의 프로그래밍 언어에서 버그, 코드 Smell, 보안 취약점을 발견하여 사용자에게 제공해준다. 그 뿐만 아니라 심각도, CWE 정보, 결함 발생 위치, 상세한 교정 지침, 데이터 플로우 추적 기능을 가지고 있다.

본 프로젝트의 소스 코드 품질 평가 시스템 대상은 소스코드를 작성하는 [SW개발자], 프로젝트를 관리하는 [프로젝트 매니저]가 주 사용자이다. 소스코드 품질 평가 시스템은 소스 코드를 분석하여 품질을 평가하고 사용자에게 정보를 제공하는 것이 주 기능이다. 검출된 약점을 수정, 보안하여 소프트웨어의 안정성을 강화하고 향후 발생하는 오류 수정 비용을 줄일 수 있다. 소스 코드 품질 평가 시스템의 상세 기능은 아래와 같다.

1. 프로젝트 전체, 소스 파일 별 품질 유형 별 점수 제공
2. 품질 유형 별 규칙 위반 건수 제공
3. 각 규칙 위반 행과 위반 규칙 설명 및 수정 가이드라인 제공
4. 과거 결과 조회 기능
5. 새로운 버전관리 시스템 지원
6. 시스템 장애 복구 기능

2.2. Business Context Diagram



2.3. Stakeholder List

Stakeholder	Description
프로젝트 매니저	<p><input type="checkbox"/> 배경 : 소스코드에 대한 취약점을 발견하고 미리 예방하여 프로젝트의 안정성을 강화하는 목적을 가진 관리 담당 인원이며 소스코드 품질 평가 시스템을 이용하려는 사용자</p> <p><input type="checkbox"/> 관심사 :</p> <ul style="list-style-type: none"> - 프로젝트 품질에 대한 정보를 주기적으로 받고 싶음.
SW개발자	<p><input type="checkbox"/> 배경 : 프로젝트의 SW 개발 담당하는 인원이며 소스코드 품질 평가 시스템을 사용하여 품질을 높이고 싶은 목적을 가진 사용자</p> <p><input type="checkbox"/> 관심사 :</p> <ul style="list-style-type: none"> - 정확한 품질 결과 데이터를 받고 싶음. - 품질 유형 별 규칙 위반의 수정 가이드라인을 반기 원함.
시스템 관리자	<p><input type="checkbox"/> 배경 : 소스코드 품질 평가 시스템 관리 담당자이며 시스템 이상 발생시 즉각 대응하여 안정적으로 가동될 수 있도록 하는 관리자</p> <p><input type="checkbox"/> 관심사 :</p> <ul style="list-style-type: none"> - 시스템의 이상 여부를 실시간으로 파악하고 싶음. - 시스템이 24시간동안 안정적으로 가동되길 원함. - 시스템 사용 모니터링을 위해 사용 현황을 알기 원함.
품질 평가 SW 개발 업체	<p><input type="checkbox"/> 배경 : 전문적인 소스코드 품질 평가 솔루션을 제공하는 업체</p> <p><input type="checkbox"/> 관심사 :</p> <ul style="list-style-type: none"> - 타 경쟁업체보다 질 높은 서비스를 제공하여 시장의 점유율을 높이고 싶어함. - 기업 목적인 이윤 창출을 위해 높은 솔루션 라이센스 구입비를 받기 원함
소스코드 버전 관리 시스템 운영자	<p><input type="checkbox"/> 배경 : 소스코드 버전 관리 서비스를 제공하는 업체 운영자</p> <p><input type="checkbox"/> 관심사 :</p> <ul style="list-style-type: none"> - 버전 관리 서버 안정성을 위해 큰 리소스를 잡지 않길 원함.

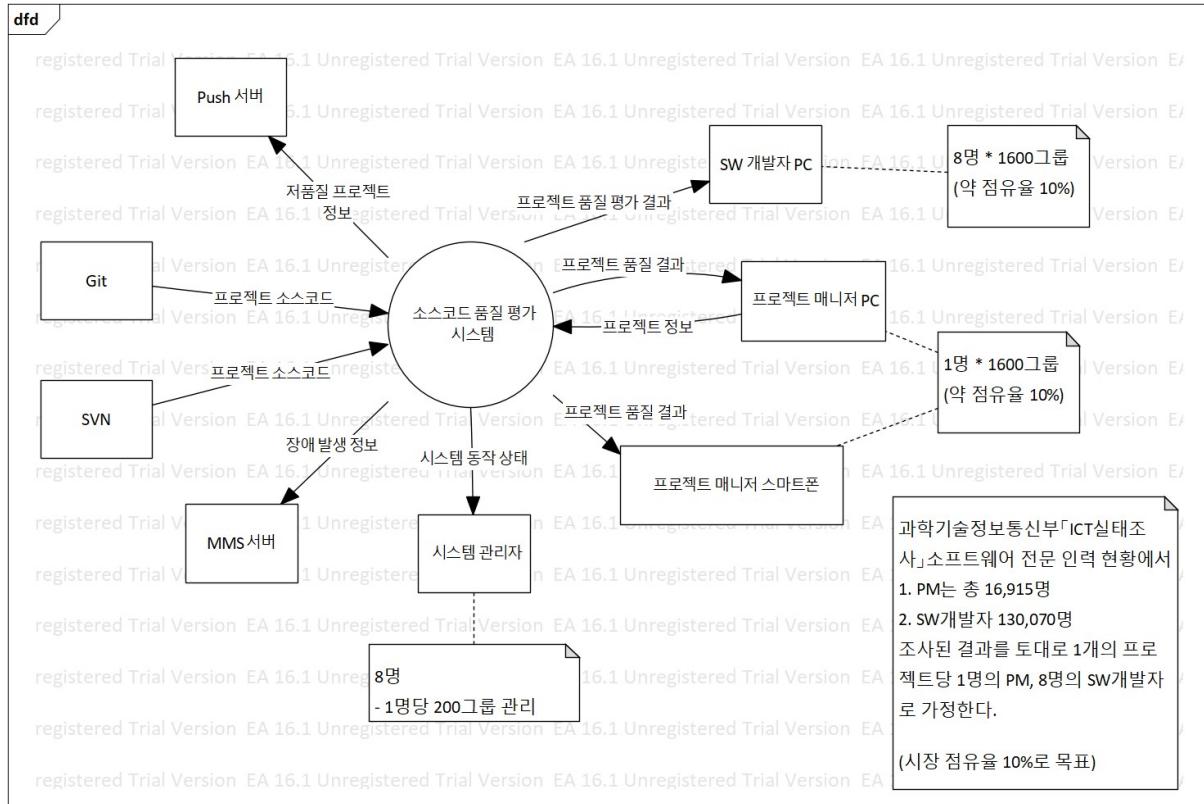
2.4. Business Goal List

Stakeholder	Business Goal		
	ID	Statement	I
프로젝트 매니저	BG-01	<input type="checkbox"/> SW 품질 관련 이슈가 발생하면 고객에게 신뢰가 낮아지기 때문에 예방 차원으로 이슈가 1건도 발생하지 않게 SW의 품질을 높이고 싶어함. <input type="checkbox"/> 정해진 개발 기간안에 프로젝트가 완료되고 싶어함.	상
SW개발자	BG-02	<input type="checkbox"/> SW 품질 관련 이슈가 발생하면 고객에게 신뢰가 낮아지기 때문에 예방 차원으로 이슈가 1건도 발생하지 않게 SW의 품질을 높이고 싶어함.	상
시스템 관리자	BG-03	<input type="checkbox"/> 시스템이 안정적으로 24시간동안 가동되길 원함 ■ 시스템 가동률 99.9% 유지 <input type="checkbox"/> 시스템 장애 발생시 복구가 되길 원함. ■ 1분내로 자가 복구	중
품질 평가 SW 개발 업체	BG-04	<input type="checkbox"/> 이윤 창출을 위해 높은 개발비를 받길 원함. ■ 연 매출 20% 향상 <input type="checkbox"/> 유지보수를 지원하여 안정적인 매출 유지하길 원함. ■ 10년 이상 유지보수 계약 체결	상
소스코드 버전 관리 시스템 운영자	BG-05	<input type="checkbox"/> 소스코드 버전 관리 시스템 서비스를 제공하여 이윤 창출을 원함. ■ 연 매출 5% 향상 ■ 국내 시장 점유율 5% 향상	하

* I : Importance

3. System Overview

3.1. System Context Diagram



* 참고. 과학기술정보통신부 [ICT실태조사]

https://kosis.kr/statHtml/statHtml.do?orgId=127&tblId=DT_127005_B029&conn_path=12

3.2. External Entity List

Name	Description
프로젝트 매니저 PC	<ul style="list-style-type: none"> <input type="checkbox"/> 유형: 사용자 <input type="checkbox"/> 역할: 프로젝트 품질 결과를 보여준다. <input type="checkbox"/> 핵심기대사항 <ul style="list-style-type: none"> ■ 프로젝트 전체적인 품질 결과데이터를 도표, 그래프를 활용하여 보여주길 원한다. ■ 품질 결과 정확도가 90% 이상이길 기대한다.
프로젝트 매니저 스마트폰	<ul style="list-style-type: none"> <input type="checkbox"/> 유형: 사용자 <input type="checkbox"/> 역할: 프로젝트 품질 결과를 보여준다. 저 품질 프로젝트의 정보를 Push 알림을 통해 받는다. <input type="checkbox"/> 핵심기대사항 <ul style="list-style-type: none"> ■ 모바일 환경에서 확인하기 때문에 명료하게 보여주길 원한다. ■ 품질 결과 정확도가 90% 이상이길 기대한다. ■ 저 품질 프로젝트의 정보를 즉시 Push 알림을 받길 원한다.

SW개발자 PC	<input type="checkbox"/> 유형: 사용자 <input type="checkbox"/> 역할: 프로젝트 품질 평가 세부 결과를 보여준다. <input type="checkbox"/> 핵심기대사항 <ul style="list-style-type: none"> ■ 소스코드 파일마다 품질 세부 결과를 보여주길 원한다. ■ 품질 위반 유형에 대한 수정 가이드라인을 제공해 주길 원한다. ■ 품질 결과 정확도가 90% 이상이길 기대한다.
시스템 관리자	<input type="checkbox"/> 유형: 사용자 <input type="checkbox"/> 역할: 소스코드 품질 평가 시스템의 상태를 관제하며 장애 발생시 대응해주는 업무를 수행한다. <input type="checkbox"/> 숙련도: 시스템 관리에 대한 지식이 깊으며, 본 시스템의 전반적인 기능을 숙지하고 있다. 장애 발생시 긴급 대처 능력이 뛰어난다. <input type="checkbox"/> 핵심기대사항 <ul style="list-style-type: none"> ■ 시스템 상태를 볼 수 있는 페이지가 있길 원한다. ■ 시스템 장애 발생시 5분안에 자가 복구가 되길 원한다. ■ 시스템 장애 발생시 즉시 알림을 받길 원한다.
Push 서버	<input type="checkbox"/> 유형: 시스템 <input type="checkbox"/> 역할: 소스코드 품질 평가 결과가 저 품질 프로젝트인 경우 프로젝트 매니저 스마트폰에 Push 알림을 전달해주는 서버이다. <input type="checkbox"/> 품질수준: <ul style="list-style-type: none"> ■ 가용성 99.9% ■ 1초당 100개 Push 알림 가능
Git	<input type="checkbox"/> 유형: 시스템 <input type="checkbox"/> 역할: SW개발자가 작성한 코드를 형상 관리해주는 시스템이다. 소스코드를 업로드, 다운로드에 대한 기능을 제공한다. <input type="checkbox"/> 품질수준: <ul style="list-style-type: none"> ■ 가용성 99.9% 이상 ■ 신뢰성 99.9% 이상 ■ N명 사용자에 대한 다운로드 속도: $(10\text{Mb}/\text{s})/\text{N명}$ ■ 동시 사용자 관계없이 업로드 속도: 1Mb/s
SVN	<input type="checkbox"/> 유형: 시스템 <input type="checkbox"/> 설명: SW개발자가 작성한 코드를 형상 관리해주는 시스템이다. 소스코드를 업로드, 다운로드에 대한 기능을 제공한다. <input type="checkbox"/> 품질수준: <ul style="list-style-type: none"> ■ 가용성 99.9% 이상 ■ 신뢰성 99.9% 이상 ■ N명 사용자에 대한 다운로드 속도: $(10\text{Mb}/\text{s})/\text{N명}$ ■ 동시 사용자 관계없이 업로드 속도: 1Mb/s
MMS 서버	<input type="checkbox"/> 유형: 시스템 <input type="checkbox"/> 역할: 시스템 장애 발생시 시스템 관리자에게 즉시 MMS로 장애 발생을 전송한다. <input type="checkbox"/> 품질수준: <ul style="list-style-type: none"> ■ 가용성 99.9% ■ 1초당 20명 MMS 발송 가능 (시스템 관리자에게 발송)

3.3. External Interface List

Name	Description
프로젝트 정보	<ul style="list-style-type: none"> <input type="checkbox"/> 유형 <ul style="list-style-type: none"> - System Interface: HTTPS - User Interface: Web <input type="checkbox"/> 역할: 품질 평가할 프로젝트에 대해서 입력한 설정 정보를 전달한다. <ol style="list-style-type: none"> 1. 소스코드 버전관리 주소 2. 품질 결과 보고서 받는 시작 설정 3. 품질 유형 선택 (Maintenance, Performance, Security) 4. 저 품질 프로젝트 기준 점수 입력 <input type="checkbox"/> 특성 <ul style="list-style-type: none"> - 데이터 포맷: JSON - 입출력 규모: 1Kbyte 이하 - 입출력 빈도: 프로젝트 초기단계 1건 (설정 정보 수정이 필요한 경우에 전송)
프로젝트 품질 평가 결과	<ul style="list-style-type: none"> <input type="checkbox"/> 유형 <ul style="list-style-type: none"> - System Interface: HTTPS - User Interface: Web, Mobile Application <input type="checkbox"/> 역할: 소스코드 품질 평가 결과가 사용자에게 맞게 보고서를 전송한다. <ul style="list-style-type: none"> - SW개발자 PC: 파일 별 세부사항 품질 평가 보고서 - 프로젝트 매니저 PC: 품질 평가 종합 보고서 - 프로젝트 매니저 스마트폰: 품질 평가 요약 보고서 <input type="checkbox"/> 특성 <ul style="list-style-type: none"> - 데이터 포맷: PDF (Web에선 PDF뷰어로 출력) - 입출력 규모: 25Mbyte 이하 - 입출력 주기/빈도: 프로젝트당 1건/1일 (1600그룹 목표로 하루 1600건)
저 품질 프로젝트 정보	<ul style="list-style-type: none"> <input type="checkbox"/> 유형 <ul style="list-style-type: none"> - System Interface: HTTPS <input type="checkbox"/> 역할: 프로젝트 매니저가 설정한 [4. 저 품질 프로젝트 기준 점수]보다 낮은 경우 프로젝트 매니저 스마트폰에 Push 알림을 전송할 수 있도록 Push서버에 정보를 보낸다. <input type="checkbox"/> 특성 <ul style="list-style-type: none"> - 데이터 포맷: JSON - 입출력 규모: 1kbyte 이하 - 입출력 주기/빈도: 프로젝트당 최대 1건/1일 (최대 1600건)
프로젝트 소스코드	<ul style="list-style-type: none"> <input type="checkbox"/> 유형 <ul style="list-style-type: none"> - System Interface: HTTPS <input type="checkbox"/> 역할: 품질 평가를 위해 프로젝트 매니저가 설정한 [1. 소스코드 버전관리 주소]를 참고하여 버전관리 시스템(Git, SVN)으로부터 소스코드를 전송받는다. <input type="checkbox"/> 특성 <ul style="list-style-type: none"> - 데이터 포맷: Project 폴더 (평가될 모든 소스코드 파일) - 입출력 규모: 1GByte 이하 - 입출력 주기/빈도: 프로젝트당 최대 1건/1일 (최대 1600건)
시스템 동작 상태	<ul style="list-style-type: none"> <input type="checkbox"/> 유형 <ul style="list-style-type: none"> - System Interface: HTTPS - User Interface: Web

	<ul style="list-style-type: none"> <input type="checkbox"/> 역할: 시스템관리자에게 시스템의 상태들을 종합하여 시스템 상태 페이지에 보여준다. <input type="checkbox"/> 특성 <ul style="list-style-type: none"> - 데이터 포맷: HTML - 입출력 규모: 1KByte 이하 - 입출력 주기/빈도: 8건 / 10초 (시스템 관리자 8명에게 10마다 정보 업데이트)
장애 발생 정보	<ul style="list-style-type: none"> <input type="checkbox"/> 유형 <ul style="list-style-type: none"> - System Interface: HTTPS <input type="checkbox"/> 역할: 시스템 장애가 발생한 경우 시스템 관리자에게 문자로 장애 발생을 즉시 알려주기 위해 MMS서버로 장애 발생 정보를 보낸다. <input type="checkbox"/> 특성 <ul style="list-style-type: none"> - 데이터 포맷: 문자열 - 입출력 규모: 250Byte 이하

3.4. System Feature List

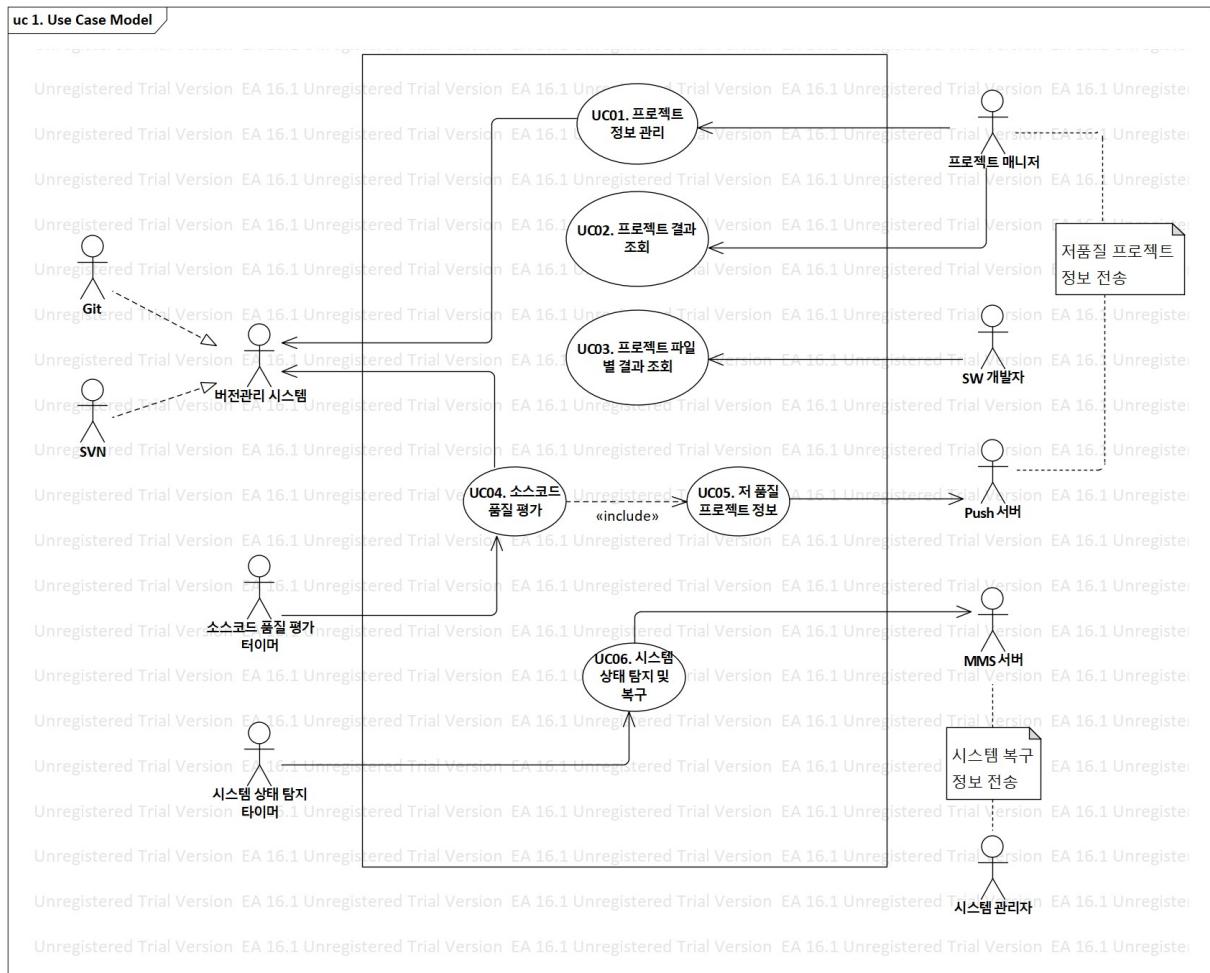
ID	Title	Description	I	Related Business Goal ID
SF-01	국제 표준 품질 유형을 기준으로 한 높은 정확도의 소스코드 품질 평가 제공	국제 표준 품질 유형에 부합하는 90%이상의 정확도를 가진 품질 평가를 해주어 소프트웨어의 잠재적인 결함을 찾아 품질 향상에 도움이 된다.	상	BG-01 BG-02
SF-02	자동화 품질 평가 수행	일정 시간마다 자동으로 품질 분석을 수행하여 사용자에게 품질 결과 보고서를 제공해주어 프로젝트 관리를 용이하게 해준다.	상	BG-01 BG-05
SF-03	품질 위반 유형 수정 가이드라인 제공	사용자에게 품질 위반 유형에 대한 수정 가이드라인을 제시하여 코드 결함을 해결하는데 도움을 준다.	상	BG-02
SF-04	자가 장애 탐지 및 복구	시스템의 장애를 탐지하여 1분안에 자동 복구 기능이 제공된다.	상	BG-03
SF-05	뛰어난 유지보수와 높은 완성도 시스템 제공	10년간 유지보수를 제공하며 버그 없는 완성도 높은 시스템을 제공한다.	중	BG-04

* I: Importance

4. Architectural Driver

4.1. Use Case Model

4.1.1. Use Case Diagram



4.1.2. Actor List

Name	Description
프로젝트 매니저	<p>프로젝트에 대해서 전반적으로 관리하는 인원이다. 프로젝트의 SW 품질 결과 데이터를 조회하여 이상 없이 프로젝트를 관리하는데 목적을 가지고 있다. 소스 코드 품질 평가 시스템의 주 사용자이다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 프로젝트 정보들을 시스템에 기록한다. 프로젝트 정보 리스트 <ol style="list-style-type: none"> 1. 소스코드 버전관리 주소 2. 품질 결과 보고서 받는 시각 설정 3. 품질 유형 선택 (Maintenance, Performance, Security) 4. 저 품질 프로젝트 기준 점수 입력 ■ 품질 결과 보고서를 조회한다.
SW개발자	<p>프로젝트 SW 개발담당 인원이다. 자신이 작성한 코드에 대한 품질 결과 데이터를 조회하여 시스템에서 제공하는 수정가이드라인을 통해 품질을 높인다. 소스 코드 품질 평가 시스템의 주 사용자이다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 작성한 파일(코드)에 대한 품질 결과 보고서를 조회한다. ■ 버전관리 시스템에 소스코드를 업로드/다운로드하여 관리한다.
시스템 관리자	<p>소스코드 품질 평가 시스템의 관리 담당이다. 시스템의 상태를 Web을 통해 상시 점검한다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 시스템 상태를 Web을 통해 상시 점검한다.
Push 서버	<p>품질 평가 결과가 저 품질인 경우 프로젝트 매니저 스마트폰에 Push알림을 보내는 서버이다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 시스템으로부터 저 품질 프로젝트 정보를 담당 프로젝트 매니저 스마트폰에 Push알림을 보낸다.
MMS 서버	<p>시스템 장애 발생 후 자가 복구가 완료되면 장애 발생 시간, 장애 유형, 복구 완료 시간을 시스템 관리자에게 MMS서버를 통해 문자메시지로 전달한다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 시스템 관리자에게 시스템 장애 정보를 문자메시지로 전달한다.
버전관리 시스템	<p>SW개발자가 작성한 코드를 형상 관리해주는 시스템이다. 소스코드를 업로드, 다운로드에 대한 기능을 제공한다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 품질 평가할 소스코드를 시스템에 제공해준다.
소스코드 품질 평가 타이머	<p>품질 평가 수행을 위해 소스코드 품질 평가 시스템에 trigger를 보낸다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 소스코드 품질 평가 시스템에 일정시간마다 품질 평가 타임아웃을 알린다.
시스템 상태 탐지 타이머	<p>소스코드 품질 평가 시스템의 상태를 탐지하기 위해 trigger를 보낸다.</p> <ul style="list-style-type: none"> <input type="checkbox"/> Actor의 역할 <ul style="list-style-type: none"> ■ 소스코드 품질 평가 시스템에 일정시간마다 시스템 상태 탐지 타임아웃을 알린다.

4.1.3. Use Case List

ID	Title	Description	Priority		System Feature ID
			I	D	
UC-01	프로젝트 정보 관리	<p>프로젝트 매니저가 품질 평가할 프로젝트의 정보를 관리한다. 데이터 처리 기능인 CRUD으로 관리한다.</p> <ul style="list-style-type: none"> * 프로젝트 정보 리스트 1. [소스코드 버전관리 주소] 2. [품질 결과 보고서 받는 시각 설정] 3. [품질 유형 선택 (M, P, S)] 4. [저 품질 프로젝트 기준 점수 입력] 	중	하	SF-05
UC-02	프로젝트 결과 조회	프로젝트 매니저가 시스템에게 프로젝트 종합 품질 결과 보고서를 조회한다.	상	하	SF-01
UC-03	프로젝트 파일 별 결과 조회	SW개발자가 시스템에게 파일 별 품질 결과 보고서를 조회한다. 품질 결과 보고서는 품질 규칙 위반 항목, 수정 가이드라인을 제공한다.	상	하	SF-01 SF-03
UC-04	소스코드 품질 평가	소스코드 품질 평가 타이머로부터 Trigger를 받는 경우 시스템은 품질 평가를 수행한다. 결과 점수가 [저 품질 프로젝트 기준 점수 입력]보다 낮은 경우 [UC-05 저 품질 프로젝트 정보]의 Use Case를 수행하도록 프로젝트정보들을 전달한다.	상	상	SF-01 SF-02
UC-05	저 품질 프로젝트 정보	[UC-04 소스코드 품질 평가]으로부터 Trigger를 받는 경우 전달받은 프로젝트 정보들을 Push서버에 전달한다.	상	하	SF-01
UC-06	시스템 상태 탐지 및 복구	시스템 상태 탐지 타이머로부터 Trigger를 받는 경우 시스템 상태를 탐지하여 시스템 상태 페이지에 화면을 갱신한다. 만약 장애 발생한 경우에는 자가 복구를 한다. 자가 복구가 완료된 시점에 장애 발생 시간, 장애 발생 유형, 복구 시간 정보들을 MMS서버에 전달한다.	상	중	SF-04

* I: Importance, D: Difficulty

4.1.4. UC-01 프로젝트 정보 관리

4.1.4.1. Scenario List

Scenario Title	Kind	Description
프로젝트 정보 등록	기본	프로젝트 매니저가 품질 평가할 프로젝트 정보를 등록한다.
프로젝트 정보 조회	선택	프로젝트 매니저가 등록된 프로젝트 정보를 조회한다.
프로젝트 정보 수정	선택	프로젝트 매니저가 등록된 프로젝트 정보를 수정한다.
프로젝트 정보 삭제	선택	프로젝트 매니저가 등록된 프로젝트 정보를 삭제한다.
버전관리 주소 접근 실패	예외	프로젝트 매니저가 입력한 [소스코드 버전관리 주소]가 유효하지 않는 주소인 경우 프로젝트 매니저에게 즉시 웹페이지를 통해 “버전관리 주소 접근 실패” 오류 메시지를 띄워준다.
DB 접속 실패	예외	프로젝트 정보가 저장되어 있는 DB서버 접속 오류가 발생하여 프로젝트 CRUD 기능을 수행하지 못하고 프로젝트 매니저에게 웹페이지를 통해 “DB 접속 실패” 오류 메시지를 띄워준다.

4.1.4.2. 프로젝트 정보 등록(기본)

4.1.4.2.1. Pre condition

Title	Description
프로젝트 정보 관리 페이지 접속	프로젝트 매니저는 프로젝트 정보 관리 페이지에 접속된 상태

4.1.4.2.2. Post condition

Title	Description
프로젝트 정보 DB 서버 저장	시스템은 프로젝트 매니저가 입력한 데이터를 DB서버에 저장된다.

4.1.4.2.3. Flow of events

Step No.	Description
1	프로젝트 매니저는 [소스코드 버전관리 주소], [품질 결과 보고서 받는 시각 설정], [품질 유형 선택 (M, P, S)], [저 품질 프로젝트 기준 점수 입력]의 정보를 시스템에 입력한다.
2	프로젝트 매니저는 프로젝트 정보 등록 버튼을 클릭한다.
3	시스템은 [소스코드 버전관리 주소]에 입력되어 있는 데이터를 참조하여 버전관리 시스템에 유효 주소인지 확인한다.
4	IF (버전관리 주소 INVALID)

5	프로젝트 매니저에게 페이지 화면에 버전관리 접근 실패 오류 메시지를 띄워준다.
6	ELSE
7	IF (DB 접속 실패)
8	시스템은 프로젝트 매니저에게 웹페이지 화면에 “DB 접속 실패” 오류 메시지를 띄워준다.
9	ELSE
10	시스템은 프로젝트 매니저로부터 입력된 데이터를 DB에 저장한다.
11	시스템은 소스코드 품질 평가시스템의 [품질 분석 수행 스케줄링 리스트]에 추가 한다.
12	ENDIF
13	ENDIF

4.1.5. UC-02 프로젝트 결과 조회

4.1.5.1. Scenario List

Scenario Title	Kind	Description
프로젝트 결과 조회	기본	프로젝트 매니저는 Web으로 시스템에게 프로젝트 결과보고서를 조회한다.
Mobile환경 프로젝트 결과 조회	선택	프로젝트 매니저는 스마트폰으로 시스템에게 프로젝트 결과보고서를 조회한다.
DB 접속 실패	예외	프로젝트 결과 보고서가 저장되어 있는 DB 접속 오류가 발생하여 프로젝트 조회 기능을 수행하지 못하고 프로젝트 매니저에게 웹페이지를 통해 “DB 접속 실패” 오류 메시지를 띄워준다.

4.1.5.2. 프로젝트 결과 조회 (기본)

4.1.5.2.1. Pre-condition

Title	Description
프로젝트 조회 페이지 접속	프로젝트 매니저는 프로젝트 조회 페이지에 접속된 상태

4.1.5.2.2. Post condition

Title	Description
프로젝트 결과 보고서 전달	프로젝트 매니저는 시스템으로부터 전달받은 프로젝트 결과 보고서를 확인 한다.

4.1.5.2.3. Flow of events

Step No.	Description
1	프로젝트 매니저는 Web, App을 통해 결과보고서 조회 버튼을 클릭한다.
2	시스템은 프로젝트 매니저에 맞는 [조회 가능 결과 보고서 리스트]를 화면에 출력한다.
3	프로젝트 매니저는 [조회 가능 결과 보고서 리스트]에서 조회할 결과 보고서를 선택한다.
4	프로젝트 매니저는 결과보고서 조회 버튼을 클릭한다.
5	IF (DB 접속 오류)
6	시스템은 웹페이지를 통해 프로젝트 매니저에게 “DB 접속 실패” 오류 메시지를 띄워준다.
7	ELSE
8	시스템은 DB서버로부터 저장되어 있는 데이터를 조회한다.
9	IF (Mobile 환경인 경우)
10	시스템은 프로젝트 매니저 Mobile 환경에 맞게 레이아웃을 조정하여 App에 결과 보고서를 출력한다.
11	ELSE (Web 환경인 경우)
12	시스템은 프로젝트 매니저에게 Web에 결과 보고서를 출력한다.
13	ENDIF
14	ENDIF

4.1.6. UC-03 프로젝트 파일 별 결과 조회

4.1.6.1. Scenario List

Scenario Title	Kind	Description
프로젝트 파일 별 결과 조회	기본	SW개발자는 시스템에게 프로젝트 파일 별 결과 보고서를 조회한다.
품질 유형 별 결과 조회	선택	SW개발자는 Maintenance, Performance, Security 중 선택하여 결과 보고서를 조회한다.
수정 가이드라인 제공	선택	SW개발자는 품질 규칙 위반에 대한 수정 가이드라인 제공 옵션을 선택하여 결과 보고서를 조회한다.
DB 접속 실패	예외	프로젝트 결과 보고서가 저장되어 있는 DB 접속 오류가 발생하여 프로젝트 조회 기능을 수행하지 못하고 SW개발자에게 웹페이지를 통해 “DB 접속 실패” 오류 메시지를 띄워준다.

4.1.6.2. 프로젝트 파일 별 결과 조회 (기본)

4.1.6.2.1. Pre condition

Title	Description
프로젝트 파일 별 조회 페이지 접속	SW개발자는 프로젝트 파일 별 조회 페이지에 접속된 상태

4.1.6.2.2. Post condition

Title	Description
프로젝트 파일 별 결과 보고서 전달	SW개발자는 시스템으로부터 전달받은 프로젝트 파일 별 결과 보고서를 확인한다.

4.1.6.2.3. Flow of events

Step No.	Description
1	SW개발자는 Web을 통해 결과보고서 조회 리스트 버튼을 클릭한다.
2	시스템은 SW개발자에 맞는 [조회 가능 파일 별 결과 보고서 리스트]를 화면에 출력한다.
3	SW개발자는 [조회 가능 파일 별 결과 보고서 리스트]에서 조회할 결과 보고서를 선택한다.
4	SW개발자는 결과보고서 조회 버튼을 클릭한다.
5	IF (DB 접속 오류)
6	시스템은 웹페이지를 통해 SW개발자에게 “DB 접속 실패” 에러 메시지를 띄워준다.
7	ELSE
8	시스템은 DB서버로부터 저장되어 있는 데이터를 조회한다.
9	IF (품질 유형 옵션 선택)
10	시스템은 SW개발자가 선택한 품질 유형만 결과 보고서를 출력한다.
11	IF(수정 가이드라인 옵션 선택)
12	시스템은 SW개발자가 선택한 품질 유형만 수정 가이드라인을 추가하여 출력한다.
13	ENDIF
19	ENDIF

4.1.7. UC-04 소스코드 품질 평가

4.1.7.1. Scenario List

Scenario Title	Kind	Description
소스코드 품질 평가	기본	소스코드 품질 평가 타이머가 3분마다 시스템에 Trigger를 보낸다. 시스템은 [품질 분석 수행 스케줄링 리스트] 데이터를 참조하여 현재 시간에 수행해야 할 프로젝트의 소스코드를 품질 평가한다.
저 품질 프로젝트 정보 판단	선택	품질 결과 점수가 프로젝트 매니저가 입력한 [저 품질 프로젝트 기준 점수 입력]보다 낮은 경우 [UC-05 저 품질 프로젝트 정보]가 수행된다.
버전관리 주소 접근 실패	예외	프로젝트 매니저가 입력한 [소스코드 버전관리 주소]가 유효하지 않는 주소인 경우 “버전관리 주소 접근 실패” 오류 메시지를 로그에 저장한다.
DB 접속 실패	예외	DB 접속 오류가 발생하여 품질 결과 데이터 저장 기능을 수행하지 못하고 “DB 접속 실패” 오류 메시지를 로그에 저장한다.

4.1.7.2. 소스코드 품질 평가(기본)

4.1.7.2.1. Pre condition

Title	Description
타이머 정상 동작	소스코드 품질 평가 타이머는 정상 동작하고 있다.

4.1.7.2.2. Post condition

Title	Description
품질 결과 데이터 DB 저장	프로젝트 품질 결과 데이터가 DB서버에 저장된다.

4.1.7.2.3. Flow of events

Step No.	Description
1	소스코드 품질 평가 타이머는 시스템에 Trigger한다.
2	시스템은 [품질 분석 수행 스케줄링 리스트]을 조회한다.
3	DO
4	시스템은 [소스코드 버전관리 주소] 참조하여 버전관리 시스템(Git, SVN)으로부터 소스코드를 다운로드 받는다.
5	IF (버전관리 주소 접근 실패)
6	시스템은 “버전관리 접근 실패” 오류 메시지를 로그에 저장한다.

7	ELSE
8	시스템은 프로젝트 품질 평가를 수행한다.
9	IF (DB 접속 실패)
10	시스템은 “DB 접속 실패” 에러 메시지를 로그에 저장한다.
11	ELSE
12	시스템은 프로젝트 품질 평가 결과 데이터를 DB에 저장한다.
13	ENDIF
14	IF (점수가 [저 품질 프로젝트 기준 점수 입력]보다 낮은 경우)
15	시스템은 [UC-05 저 품질 프로젝트 정보(프로젝트 식별자, 프로젝트 결과)] Use Case를 수행한다.
16	ENDIF
17	ENDIF
18	WHILE(현재 시간에 수행해야 할 [품질 분석 수행 스케줄링 리스트]의 모든 프로젝트)

4.1.8. UC-05 저 품질 프로젝트 정보

4.1.8.1. Scenario List

Scenario Title	Kind	Description
Push 서버에 저 품질 프로젝트 정보 전송	기본	시스템은 (프로젝트 식별자, 프로젝트 결과)를 바탕으로 프로젝트 매니저에게 저 품질 프로젝트 정보와 함께 Push서버에 전송한다.

4.1.8.2. Push 서버에 저 품질 프로젝트 정보 전송(기본)

4.1.8.2.1. Pre condition

Title	Description
소스코드 품질 평가 가능 수행 완료 상태	시스템은 소스코드 품질 평가 가능 수행 완료된 상태이다.

4.1.8.2.2. Post condition

Title	Description
Push서버에 저 품질 프로젝트 정보 전송	시스템은 Push서버에 저 품질 프로젝트 정보를 전송한다.

4.1.8.2.3. Flow of events

Step No.	Description
1	시스템은 (프로젝트 식별자, 프로젝트 결과)를 바탕으로 프로젝트 정보를 전달받는다.
2	시스템은 Push 서버에 저 품질 프로젝트 매니저 정보, 프로젝트 이름, 프로젝트 품질 점수 데이터를 전송한다.

4.1.9. UC-06 시스템 상태 탐지 및 복구

4.1.9.1. Scenario List

Scenario Title	Kind	Description
시스템 상태 탐지 및 복구	기본	시스템 상태 탐지 타이머는 1분 주기로 서버의 동작 상태를 확인 한다. 시스템은 시스템 상태 웹페이지에 업데이트한다. 서버 장애 발생 확인 시 서버를 재가동하여 복구한다. MMS서버에 장애 발생 시간, 유형, 복구 시간을 전송한다.
시스템 복구 실패	선택	시스템은 서버 장애 발생 확인 후 서버를 재가동하여 복구 시도를 했지만 복구가 안된 경우 MMS서버에 장애 발생 시간, 유형, 복구 실패 정보를 전송한다.
시스템 상태 탐지 오작동	예외	시스템 상태 탐지 기능 자체가 장애가 발생하여 시스템 탐지가 불 가능하다. 이 경우 MMS서버에 시스템 상태 탐지 장애 발생 정보를 전송한다.

4.1.9.2. 시스템 상태 탐지 및 복구(기본)

4.1.9.2.1. Pre condition

Title	Description
타이머 정상 동작	시스템 상태 탐지 타이머는 정상 동작하고 있다.

4.1.9.2.2. Post condition

Title	Description
시스템 상태 페이지 업데이트 및 복구	시스템 상태 페이지에 서버 동작 상태가 업데이트 된다. 장애 발생시 서버는 재가동 되어 복구된다.

4.1.9.2.3. Flow of events

Step No.	Description
1	시스템 상태 탐지 타이머는 시스템에 Trigger한다.
2	DO

3	시스템은 서버에 상태 정보를 조회한다.
4	IF(장애 발생 시)
5	시스템은 장애 서버를 재 가동하여 복구한다.
6	WHILE(복구 시간)
7	시스템은 장애 서버 상태 정보를 조회한다.
8	IF (복구 실패)
9	시스템은 MMS서버에 장애 발생 시간, 유형, 복구 실패 정보를 전송한다.
10	ENDIF
11	ELSE
12	시스템은 시스템 상태 페이지에 서버 상태를 업데이트 한다.
13	ENDIF
14	WHILE (시스템 모든 서버)

4.2. Quality Attribute Scenario

4.2.1. QA Scenario List

ID	Title	Type	Priority		Related Use Case	System Feature ID
			I	D		
QA-01	품질 평가 수행 시간	Performance	상	상	UC-04 UC-05	SF-02
QA-02	시스템 장애 탐지 및 복구	Availability	상	중	UC-06	SF-04
QA-03	새로운 품질 유형 지원	Maintainability	상	중	UC-04	SF-05
QA-04	새로운 버전관리 시스템 지원	Maintainability	중	하	UC-01 UC-04	SF-05

4.2.2. QA-01 품질 평가 수행 시간

QA Type	Performance
Description	<p>3분마다 소스코드 품질 평가 타이머가 Trigger시 소스코드 품질 평가 시스템은 4가지 동작을 수행한다.</p> <ol style="list-style-type: none"> 1. 소스코드 다운로드(30초) 2. 소스코드 품질 평가 수행(19분) 3. 품질 결과 데이터 DB서버 저장(20초) 4. 프로젝트 매니저 스마트폰에 저 품질 프로젝트 정보 Push알림(10초) <p>소스코드 품질 평가 시스템은 4가지 과정을 총 20분안에 완료되어야 한다.</p>
Source of Stimulus	3분마다 Trigger하는 소스코드 품질 평가 타이머
Stimulus	<p>품질 평가 수행 요청</p> <ul style="list-style-type: none"> - 최대 동시 1600개 프로젝트 품질 평가 수행 요청 가능 * 시장 점유율 10% 목표 (국내 기준 16000개의 프로젝트)
Artifact	소스코드 품질 평가 시스템
Environment	<ul style="list-style-type: none"> * 소스코드 품질 평가 시스템이 정상 동작 중인 상태 <ul style="list-style-type: none"> - 서버의 CPU / Memory 사용량 50% 이하 수준 - 네트워크 속도 10Gbps * 버전 관리 시스템이 정상적으로 운용되는 상태 <ul style="list-style-type: none"> - 네트워크 속도 300Mbps * Push서버가 정상적으로 운용되는 상태 <ul style="list-style-type: none"> - 서버의 CPU / Memory 사용량 50% 이하 수준 - 네트워크 속도 1Gbps
Response	<p>버전 관리 시스템으로부터 소스코드를 다운받는다.</p> <p>소스코드 품질 평가를 수행한다.</p> <p>품질 평가 결과 데이터가 DB서버에 저장한다.</p>

	저 품질 프로젝트인 경우 프로젝트 매니저 스마트폰에 Push 알림 전송한다. 최신 결과 보고서를 조회한다.
Response Measure	<p>버전관리 시스템으로부터 소스코드 다운로드: 30초</p> <ul style="list-style-type: none"> - 버전관리 시스템에 최대 저장 용량: 1GB - 네트워크 속도: 300Mbps(37.5MB/S) - 다운로드 시간: $1\text{GB} (1024\text{MB}) / 37.5\text{MB/S} = 27\text{초} (\text{약 } 30\text{초})$ <p>소스코드 품질 평가 수행 시간: 19분</p> <ul style="list-style-type: none"> - 5000LOC당 품질 분석 1초 소요 - 최대 용량 1GB/1LOC당 200Byte 가정 = 500만 LOC - $500\text{만 LOC} / 5000\text{LOC} = 1000\text{초} = \text{약 } 17\text{분} \text{ 소요}$ <p>품질 평가 결과 데이터 DB서버 저장: 20초</p> <ul style="list-style-type: none"> - 품질 평가 결과 데이터 용량: 1MB <p>저 품질 프로젝트 정보 프로젝트 매니저 스마트폰에 Push 알림: 10초</p> <p>위 4가지 과정은 총 20분안에 완료한다.</p>

4.2.3. QA-02 시스템 장애 탐지 및 복구

QA Type	Availability
Description	서버에 장애가 발생 시 1분안에 탐지하고 장애 서버를 재가동하여 3분안에 복구 완료한다. 또한 시스템 관리자에게 1분안에 문자메시지로 장애발생시간, 장애유형, 복구시간을 통보한다. (총 5분)
Source of Stimulus	시스템내 장애를 탐지하는 서버 외 전체 서버
Stimulus	서버 장애 발생
Artifact	소스코드 품질 평가 시스템
Environment	소스코드 품질 평가 시스템 정상 동작 상태 <ul style="list-style-type: none"> - CPU/Memory 사용량이 50% 이하 수준 - 네트워크속도 10Gbps
Response	장애 서버를 탐지하고 재가동하여 복구한다. 시스템 관리자에게 문자메시지로 장애발생시간, 장애유형, 복구시간을 통보한다.
Response Measure	장애 서버 탐지(1분) + 복구시간(3분) + 문자메시지 전송(1분) 총 5분 이내로 탐지, 복구 및 통보를 완료한다. <ul style="list-style-type: none"> - 가용성 99.9% (연간 장애 시간: 8시간 45분) : 월 8회 장애 발생시, 연간 장애 8시간 ($5\text{분} * 8\text{회} * 12\text{개월} / 60\text{분}$)

4.2.4. QA-03 새로운 품질 유형 지원

QA Type	Maintainability
Description	새로운 품질 유형 추가가 필요한 경우 추가할 새로운 품질 유형에 대한 결과 조회가 빠른 시간(1MM)안에 완료되어야 한다.

Source of Stimulus	소스코드 품질 평가 시스템을 구매한 고객사
Stimulus	새로운 품질 유형 추가 요청
Artifact	소스코드 품질 평가 시스템
Environment	시스템이 시장에서 운영되고 있는 상태 기존 기능에 대한 모든 Test Case를 보유한 상태 새로운 품질 유형이 추가가 용이하도록 설계된 상태
Response	새로운 품질 유형이 반영되고 추가된 새로운 품질 유형에 대한 개발 및 검증을 수행한다.
Response Measure	프로젝트 결과 조회 시 새로운 품질 유형에 대한 정보가 추가되었으면 반영됨을 인식한다. 또한 개발, 검증 및 기능 반영은 1MM 이내 완료되어야 한다.

4.2.5. QA-04 새로운 버전관리 시스템 지원

QA Type	Maintainability
Description	새로운 버전관리 시스템 추가가 필요한 경우 추가할 새로운 버전관리 시스템과 본 시스템의 연동이 빠른 시간(1MM)안에 완료되어야 한다.
Source of Stimulus	소스코드 품질 평가 시스템을 구매한 고객사
Stimulus	새로운 버전관리 시스템 추가 요청
Artifact	소스코드 품질 평가 시스템
Environment	시스템이 시장에서 운영되고 있는 상태 기존 버전관리 시스템에 대한 모든 Test Case를 보유한 상태 새로운 버전관리 시스템 추가가 용이하도록 설계된 상태
Response	새로운 버전관리 시스템이 연동되고 추가된 새로운 버전관리 시스템에 대한 개발 및 검증을 수행한다.
Response Measure	프로젝트 매니저가 프로젝트 정보 등록할 때 새로운 버전관리 시스템에 저장된 [소스코드 버전관리 주소]를 입력하여 오류 없이 수행되면 정상적으로 기능이 연동됨을 인식한다. 또한 개발, 검증 및 기능 반영은 1MM 이내 완료되어야 한다.

4.3. Constraint

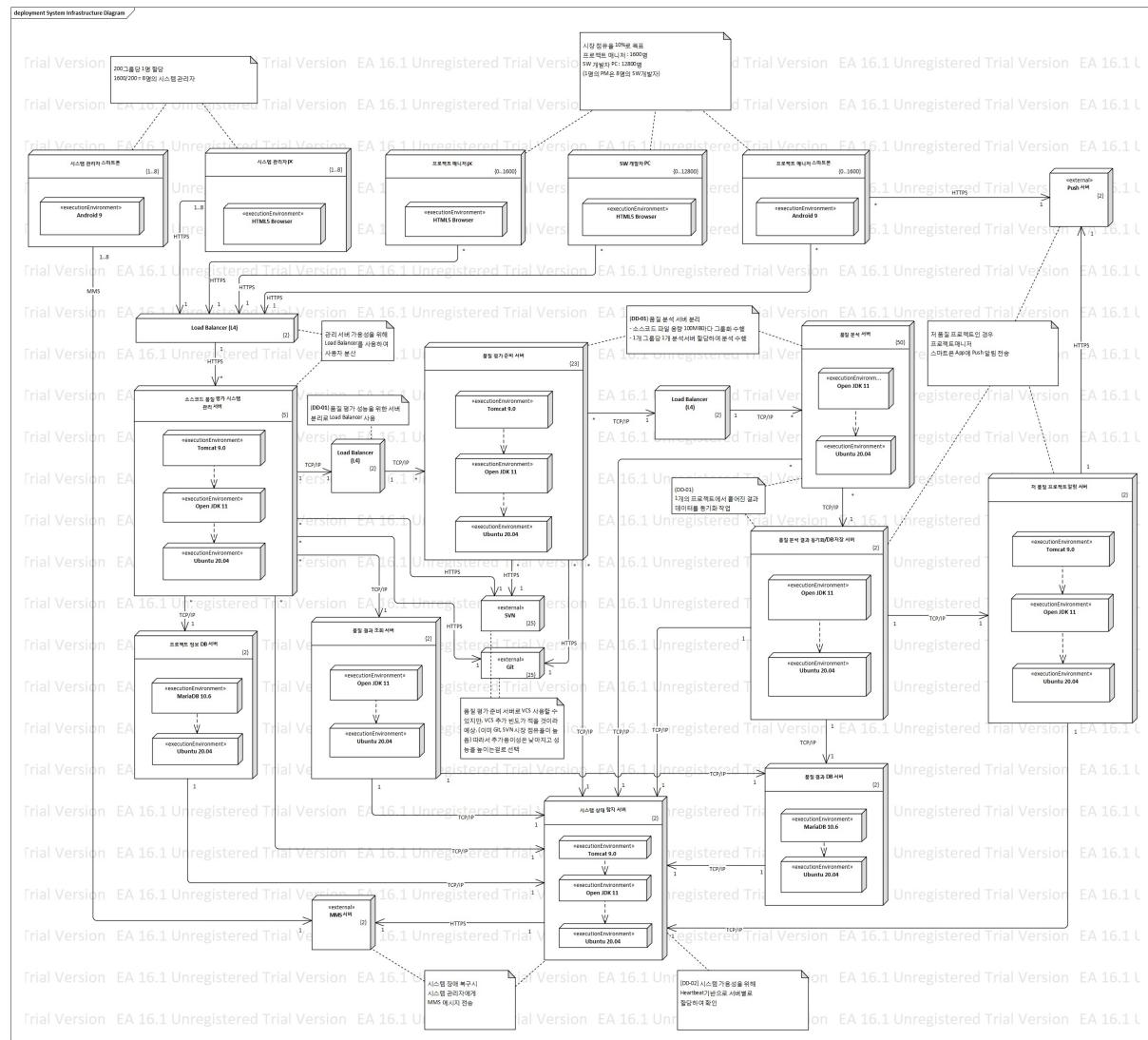
4.3.1. Business Constraint List

본 과제의 범위에 포함되지 않음.

5. Top Level Design Description

5.1. System Infrastructure View

5.1.1. System Infrastructure Diagram



5.1.2. Node Specification

Name	Description
시스템 관리자 스마트폰	<ul style="list-style-type: none"> * 역할: 시스템 관리자는 서버 장애 복구 완료가 되면 시스템 관리자 스마트폰에 MMS메시지를 받는다. * 특성: <ul style="list-style-type: none"> - 스마트폰의 수명 평균 수명 5년을 고려하여 안드로이드 9.0 이상 지원 - MMS 메시지 수신 가능 * Multiplicity: 1600그룹당 1명의 시스템 관리자로 최대 8명

시스템 관리자 PC	<ul style="list-style-type: none"> * 역할: 시스템 관리자는 PC를 통해 시스템 상태를 표시해주는 웹페이지에 접속하여 전반적인 서버 현황을 확인한다. * 특성: 서버 현황 조회용도로 사용되므로 고사양 불필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 8개 - RAM: 8GB - ROM: 256GB - Ethernet: 10Gbps * Multiplicity: 1그룹당 1명의 시스템 관리자로 최대 8명
프로젝트 매니저 PC	<ul style="list-style-type: none"> * 역할: 프로젝트 매니저는 PC를 통해 품질 평가 진행이 필요한 프로젝트를 관리(CRUD)하고 진행하거나/진행했던 프로젝트 품질 결과 보고서를 조회한다. * 특성: 프로젝트 관리, 보고서 조회용도로 사용되므로 고사양 불필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 8개 - RAM: 8GB - ROM: 256GB - Ethernet: 10Gbps * Multiplicity: 1그룹당 1명의 프로젝트 매니저로 최대 1600명
프로젝트 매니저 스마트폰	<ul style="list-style-type: none"> * 역할: 품질 결과가 [저 품질 프로젝트 기준 점수 입력] 보다 낮은 경우 프로젝트 매니저 스마트폰 App으로부터 알림을 받는다. * 특성: <ul style="list-style-type: none"> - 스마트폰의 수명 평균 수명 5년을 고려하여 안드로이드 9.0 이상 지원 * Multiplicity: 1그룹당 1명의 프로젝트 매니저로 최대 1600명
SW개발자 PC	<ul style="list-style-type: none"> * 역할: SW개발자는 자신이 작성한 코드에 대해 유지보수성, 성능, 보안의 품질 결과와 품질 규칙 위반 수정 가이드라인을 제공받는다. * 특성: SW 개발자에겐 고사양 필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 64개 - RAM: 64GB - ROM: 2TB - Ethernet: 10Gbps * Multiplicity: 1그룹당 8명의 SW개발자로 최대 12800명
소스코드 품질 평가 시스템 관리 서버	<ul style="list-style-type: none"> * 역할: 소스코드 품질 평가 시스템의 전반적인 서비스를 관리하는 서버이다. 프로젝트 정보 관리(CRUD), 결과 보고서 조회, 품질 평가 수행의 기능을 수행할 수 있도록 운영하는 서버이다. 시스템 관리자에게 시스템 서버 현황에 대해 실시간으로 제공한다. SW개발자에게 결과 보고서 조회할 수 있도록 지원한다. 프로젝트 매니저에게 프로젝트 정보 관리, 결과 보고서 조회할 수 있도록 지원한다. * 특성: 시스템의 메인 서버로 빠른 처리가 필요하여 고사양 필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 64개 - RAM: 64GB - ROM: 256GB - Ethernet: 10Gbps * Multiplicity: 가용성을 위하여 5대로 운영
품질 평가 준비 서버	<ul style="list-style-type: none"> * 역할: 소스코드 품질 평가 시스템 관리 서버로부터 품질 평가 대상인 정보들을 받아 품질 평가 준비하는 서버이다. 품질 평가 수행 서버는 버전관리 시스템으로부터 소스코드를 다운받게 된다. 다운받은 소스파일들은 100MB 기준으로 Load Balancer에 전달하여 분석 요청을 한다. * 특성: 프로젝트 소스를 다운로드가 필요하여 ROM 대 용량 필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 64개 - RAM: 64GB

	<ul style="list-style-type: none"> - ROM: 256GB - Ethernet: 10Gbps <p>* Multiplicity: 프로젝트 개수: 시장 점유율 10% 목표인 1600개 프로젝트 프로젝트 품질 평가 분석 최대 소요 시간: 20분 1개의 서버가 하루에 분석할 수 있는 프로젝트 최대 개수: 72개 (최악의 경우) 품질 평가 서버 최소 개수 = $1600 \div 72 = 22.2 \approx 23$개의 서버 필요</p>
Load Balancer	<ul style="list-style-type: none"> * 역할: 품질 평가 준비 서버에서 오는 품질 분석 서버의 부하를 분산 (<u>DD-01</u>) <ul style="list-style-type: none"> - 소스코드 품질 평가 시스템 관리 서버 > 품질 평가 준비 서버 - 품질 평가 준비 서버 > 품질 분석 서버 (분석이 필요한 소스파일) - 사용자 > 소스코드 품질 평가 시스템 관리 서버 * 특성: 부하 분산을 위한 Auto-Scaling 지원 * Multiplicity: 가용성을 위하여 Active/Standby 2대로 운영
품질 분석 서버	<ul style="list-style-type: none"> * 역할: 전달받은 소스파일을 품질 유형에 대해 분석한다. 1개의 프로젝트는 여러 개의 소스파일이 존재하여 결과 데이터에 대한 동기화가 필요하다. 따라서 DB에 저장하지 않고 동기화 서버에 결과 데이터를 전달한다. (<u>DD-01</u>) * 특성: 시스템 성능을 결정하는 서버로 성능 보장을 위해 고 사양 필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 64개 - RAM: 64GB - ROM: 256GB - Ethernet: 10Gbps * Multiplicity: 가용성 보장을 위하여 Active/Standby 50개 서버 운영 DD-01 품질 평가 성능(평가 시간 20분 이내)을 위한 분석 서버 분리를 보장하기 위하여, 복수 개 서버 운영/Multi-Thread 활용
품질 분석 결과 동기화/DB 저장 서버	<ul style="list-style-type: none"> * 역할: 품질 분석 서버로부터 받은 결과 데이터를 프로젝트 동기화 작업을 하는 서버이다. 동기화가 완료되면 DB서버에 데이터를 저장한다. 또한 프로젝트 매니저가 설정한 [저 품질 프로젝트 기준 점수 입력]보다 낮은 경우는 저 품질 프로젝트 알림에 프로젝트 정보를 전달한다. (<u>DD-01</u>) * 특성: 동기화작업을 하기 때문에 처리속도가 중요하여 고 사양 필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 64개 - RAM: 64GB - ROM: 1TB - Ethernet: 10Gbps * Multiplicity: 가용성을 위하여 Active/Standby 2대로 운영 다수의 프로젝트를 처리하기 위해 Multi-Thread 활용
저 품질 프로젝트 알림 서버	<ul style="list-style-type: none"> * 역할: 품질 분석 결과 동기화/DB 저장 서버로부터 받은 저 품질 프로젝트 정보를 담당 프로젝트 매니저 스마트폰에게 Push알림을 보내기 위해 Push서버에 프로젝트 정보를 전달한다. * 특성: 저 품질 프로젝트 알림을 위한 서버로 고 사양 불필요 <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 16개 - RAM: 16GB - ROM: 256 GB - Ethernet: 10Gbps * Multiplicity: 가용성을 위하여 Active/Standby 2대로 운영
품질 결과 조회 서버	<ul style="list-style-type: none"> * 역할: 프로젝트 매니저, SW개발자가 품질 결과 보고서를 조회하는 정보들을 서버로부터 Read하여 보고서형식으로 변환 후 파일을 관리서버에 제공한다. * 특성: DB 데이터를 읽어 보고서 형식을 변환하는 서버로 고 사양 필요

	<ul style="list-style-type: none"> - CPU: 3.6GHz/코어 64개 - RAM: 64GB - ROM: 1TB - Ethernet: 10Gbps <p>* Multiplicity: 가용성을 위하여 Active/Standby 2대로 운영</p>
시스템 상태 탐지 서버	<p>* 역할: 시스템내 운영중인 전체 서버의 장애를 <u>(DD-02) heartbeat</u> 신호로 탐지하고 장애 발생 시 해당 서버를 복구한다.</p> <p>* 특성: 시스템 상태 탐지 및 복구로 고 사양 불필요</p> <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 16개 - RAM: 16GB - ROM: 256TB - Ethernet: 10Gbps <p>* Multiplicity: 가용성을 위하여 Active/Standby 2대로 운영</p>
프로젝트 정보 DB 서버	<p>역할: 소스코드 품질 평가 대상 프로젝트의 정보들을 저장하는 DB서버이다.</p> <p>* 특성:</p> <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 16개 - RAM: 16GB - ROM: 256TB - Ethernet: 10Gbps <p>* Multiplicity: 가용성을 위하여 Active/Standby 2대로 운영</p>
품질 결과 DB 서버	<p>역할: 소스코드 품질 분석 데이터를 저장하는 DB서버이다.</p> <p>* 특성:</p> <ul style="list-style-type: none"> - CPU: 3.6GHz/코어 16개 - RAM: 16GB - ROM: 2TB (저장되는 데이터가 많을 것이라 예상) - Ethernet: 10Gbps <p>* Multiplicity: 가용성을 위하여 Active/Standby 2대로 운영</p>

5.1.3. Execution Environment Specification

Node	Name	Description
품질 결과 조회 서버 품질 평가 준비 서버 품질 분석 서버 저 품질 프로젝트 알림 서버 시스템 상태 탐지 서버 품질 분석 결과 동기화/DB 저장 서버 소스코드 품질 평가 시스템 관리 서버 프로젝트 정보 DB 서버 품질 결과 DB 서버	Ubuntu 20.04	많은 서버 노드에서 사용되므로 개발 비용과 성능을 고려하여 오픈소스인 Ubuntu를 활용. 안정성, 유지 보수성을 고려하여 2020년 4월 23일날 출시한 20.04 LTS버전을 사용
품질 결과 조회 서버 품질 평가 준비 서버 품질 분석 서버	Open JDK 11	많은 서버 노드에서 사용되므로 개발 비용과 성능을 고려하여 오픈소스인 OpenJDK를 활용. Tomcat을 안정적으로 지원하여 OpenJDK 11버전 사용

저 품질 프로젝트 알림 서버 시스템 상태 탐지 서버 품질 분석 결과 동기화/DB 저장 서버 소스코드 품질 평가 시스템 관리 서버		(OpenJDK 8버전은 향후 몇 년 안에 지원 X)
품질 평가 준비 서버 저 품질 프로젝트 알림 서버 시스템 상태 탐지 서버 소스코드 품질 평가 시스템 관리 서버	Tomcat 9.0	많은 서버 노드에서 사용되므로 개발 비용과 성능을 고려하여 오픈소스인 Tomcat를 활용. (OpenJDK 11과 호환 가능)
프로젝트 정보 DB 서버 품질 결과 DB 서버	MariaDB 10.6	오픈 소스인 관계형 데이터베이스 시스템이며 개발 비용과 성능을 고려하여 MariaDB를 활용. (2026년 7월까지 MariaDB10.6 지원)
시스템 관리자 PC 프로젝트 매니저 PC SW개발자 PC	HTML5 Browser	OS 상관없이 Web socket, HTTPS 모두 지원하는 HTML5 Browser 사용
시스템 관리자 스마트폰 프로젝트 매니저 스마트폰	Android 9.0	스마트폰 수명(5년) 고려하여 2018년에 출시한 Android 9.0 이상 지원

5.1.4. Communication Path Specification

Path	Description
시스템 관리자 PC – Load Balancer	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
프로젝트 매니저 PC – Load Balancer	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
SW개발자 PC – Load Balancer	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
프로젝트 매니저 스마트폰 – Load Balancer	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
Load Balancer – 소스코드 품질 평가 시스템 관리 서버	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
Load Balancer – 소스코드 품질 평가 시스템 관리 서버	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
Load Balancer – 소스코드 품질 평가 시스템 관리 서버	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
소스코드 품질 평가 시스템 관리 서버 – 프로젝트 정보 DB 서버	TCP/IP – 내부 DB 서버와 연결하기 때문에 보안성 고민 없이 응답속도가 빠른 TCP/IP 통신
소스코드 품질 평가 시스템 관리 서버 –	TCP/IP – 내부 시스템 연결로 TCP/IP 통신

품질 결과 조회 서버	
소스코드 품질 평가 시스템 관리 서버 – Load Balancer	TCP/IP – 내부 시스템 연결로 TCP/IP 통신
소스코드 품질 평가 시스템 관리 서버 – SVN	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
소스코드 품질 평가 시스템 관리 서버 – Git	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
Load Balancer – 품질 평가 준비 서버	TCP/IP – 내부 시스템 연결로 TCP/IP 통신
소스코드 품질 평가 시스템 관리 서버 – 시스템 상태 탐지 서버	TCP/IP – heartbeat방식을 위해 응답속도가 빠른 TCP/IP 통신
프로젝트 정보 DB 서버 – 시스템 상태 탐지 서버	TCP/IP – heartbeat방식을 위해 응답속도가 빠른 TCP/IP 통신
품질 결과 조회 서버 – 시스템 상태 탐지 서버	TCP/IP – heartbeat방식을 위해 응답속도가 빠른 TCP/IP 통신
품질 결과 조회 서버 – 품질 결과 DB 서버	TCP/IP – 내부 DB 서버와 연결하기 때문에 보안은 고려 대상이 아니므로 응답속도가 빠른 TCP/IP 통신
품질 평가 준비 서버 – SVN	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
품질 평가 준비 서버 – Git	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
품질 평가 준비 서버 – Load Balancer	TCP/IP – 내부 시스템 연결로 TCP/IP 통신
Load Balancer – 품질 분석 서버	TCP/IP – 내부 시스템 연결로 TCP/IP 통신
품질 분석 서버 – 시스템 상태 탐지 서버	TCP/IP – heartbeat방식을 위해 응답속도가 빠른 TCP/IP 통신
품질 분석 서버 – 품질 분석 결과 동기화/DB 저장 서버	TCP/IP – 내부 시스템 연결로 TCP/IP 통신
품질 분석 결과 동기화/DB 저장 서버 – 시스템 상태 탐지 서버	TCP/IP – heartbeat방식을 위해 응답속도가 빠른 TCP/IP 통신
품질 분석 결과 동기화/DB 저장 서버 – 품질 결과 DB 서버	TCP/IP – 내부 DB 서버와 연결하기 때문에 보안은 고려 대상이 아니므로 응답속도가 빠른 TCP/IP 통신
품질 분석 결과 동기화/DB 저장 서버 – 저 품질 프로젝트 알림 서버	TCP/IP – 내부 시스템 연결로 TCP/IP 통신
품질 결과 DB 서버 – 시스템 상태 탐지 서버	TCP/IP – heartbeat방식을 위해 응답속도가 빠른 TCP/IP 통신
저 품질 프로젝트 알림 서버 – 시스템 상태 탐지 서버	TCP/IP – heartbeat방식을 위해 응답속도가 빠른 TCP/IP 통신
저 품질 프로젝트 알림 서버 – Push 서버	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용
시스템 상태 탐지 서버 – MMS 서버	HTTPS – 외부 시스템과 통신하기 때문에 보안을 위해 HTTPS 사용

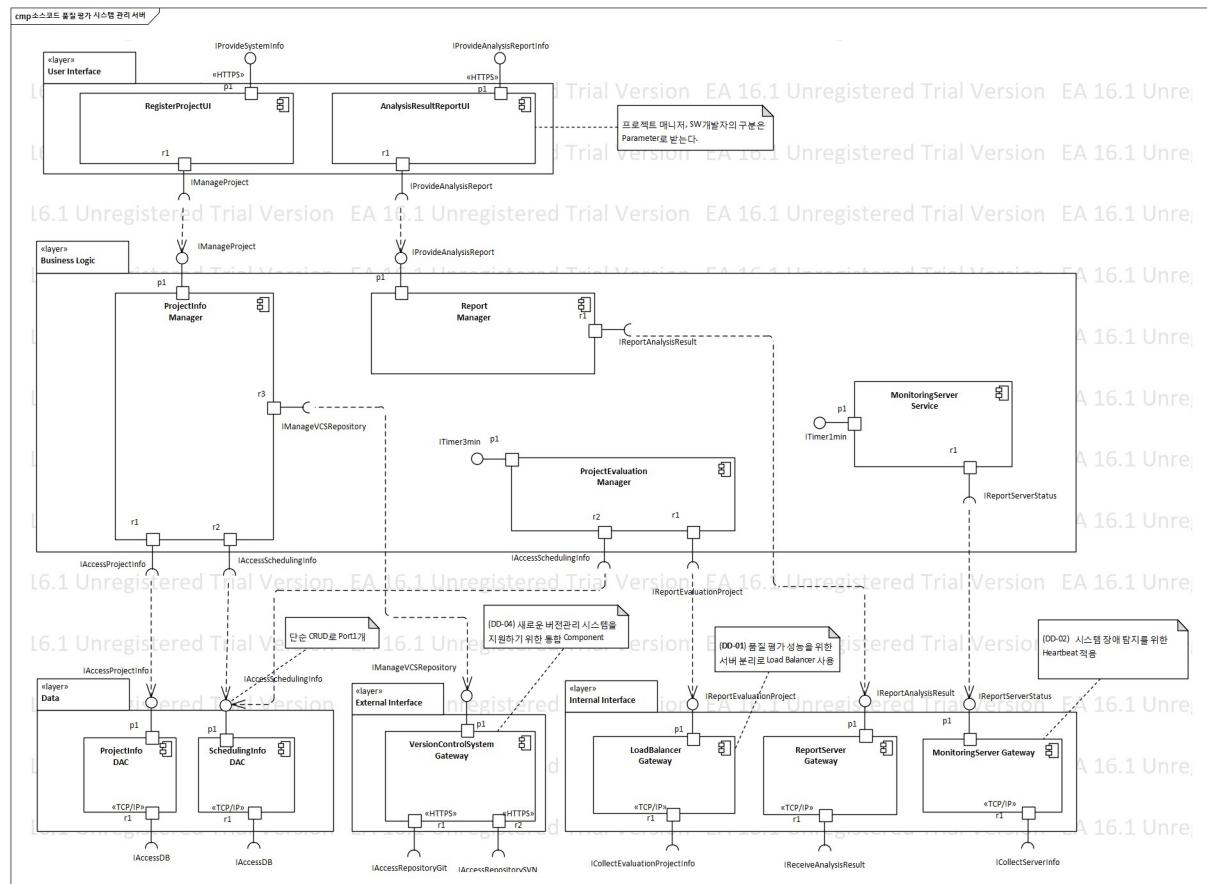
5.2. Structure View

5.2.1. Overall Structure

생략 (Static Structure Model 대체)

5.2.2. 소스코드 품질 평가 시스템 관리 서버 - Static Structure Model

5.2.2.1. Static Structure Diagram



5.2.2.2. Element List

Name	Responsibility	Relevant ADs
User Interface Layer		
RegisterProjectUI	프로젝트 매니저가 프로젝트 정보 등록(CRUD포함)할 때 입력된 Data를 전달받는다.	UC-01, QA-01
AnalysisResult ReportUI	프로젝트 매니저가 조회할 프로젝트에 대한 정보, SW개발자가 조회할 프로젝트 정보와 조회할 품질 유형에 대한 정보에 대한 Data를 전달받는다.	UC-02, UC-03
Business Logic Layer		

ProjectInfo Manager	프로젝트 정보가 저장되어 있는 프로젝트 정보 DB에 접근 한다. 프로젝트 정보들을 Read/Write한다. 품질 평가 수행 리스트 DB에 접근하여 정보들을 Read/Write한다.	UC-01, UC-02, UC-03, QA-01
ReportManager	프로젝트매니저, SW개발자에게 프로젝트 보고서를 제공하기 위해 품질 결과 조회 서버에 조회할 프로젝트 정보를 전달한다.	UC-02, UC-03, QA-01
ProjectEvaluation Manager	매 3분마다 수행 리스트를 DB에서 조회하여 수행 평가할 프로젝트 정보를 품질 분석 준비 서버에 전달한다.	UC-04, UC-05, QA-01
MonitoringServer Service	매 1분마다 시스템 상태 탐지 서버에서 모니터링 할 수 있도록 <u>(DD-02)</u> Heartbeat 신호를 전달한다.	UC-06, QA-02
Data Layer		
ProjectInfo DAC	프로젝트 정보들과 사용자의 정보가 저장된 Database에 접근하여 DB 값을 읽고/쓴다.	UC-01, UC-02, UC-03, UC-04, QA-01
SchedulingInfo DAC	프로젝트 분석 스케줄링의 정보가 저장된 Database에 접근하여 DB 값을 읽고/쓴다.	UC-01, UC-04, QA-01
Internal Interface		
ReportServer Gateway	조회할 프로젝트 정보들과 함께 품질 평가 조회 서버에 전달한다.	UC-06, QA-02
Load Balancer	품질 평가할 프로젝트 정보들과 함께 Load Balancer에 전달한다. Load Balancer는 품질 평가 준비 서버가 부하되지 않게 분산시켜준다. <u>(DD-01)</u>	UC-04, UC-05, QA-01
MonitoringServer Gateway	Heartbeat 신호를 시스템 상태 탐지 서버에 전달한다.	UC-06, QA-02
External Interface		
VersionControl SystemGateway	VCS 별 Repository 주소 유효성과 Repository 다운로드에 대한 요청을 전달하는 Gateway <u>(DD-04)</u>	UC-01, QA-04

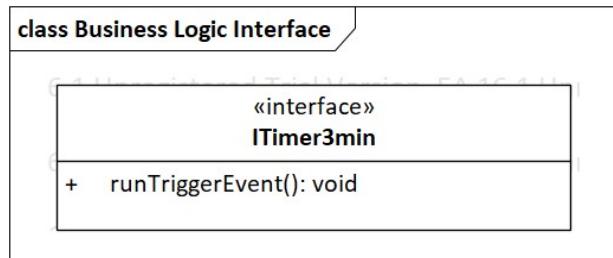
5.2.2.3. Business logic Layer

5.2.2.3.1. ProjectEvaluationManager Specification

5.2.2.3.1.1. Interface List

Name	Kind	Description
ITimer3min	Provided	품질 분석 타이머로부터 매 3분마다 Trigger를 받는 인터페이스
IAccessSchedulingInfo	Required	[품질 분석 수행 스케줄링 리스트]를 조회하는 인터페이스
IReportEvaluationProject	Required	[품질 분석 수행 스케줄링 리스트]의 시간에 맞춰서 프로젝트 품질 분석 수행 요청하는 인터페이스

5.2.2.3.1.2. ITimer3min Interface Specification



Operation	Responsibility
runTriggerEvent	Trigger신호를 받으면 품질 평가 수행 요청 flow를 실행한다.

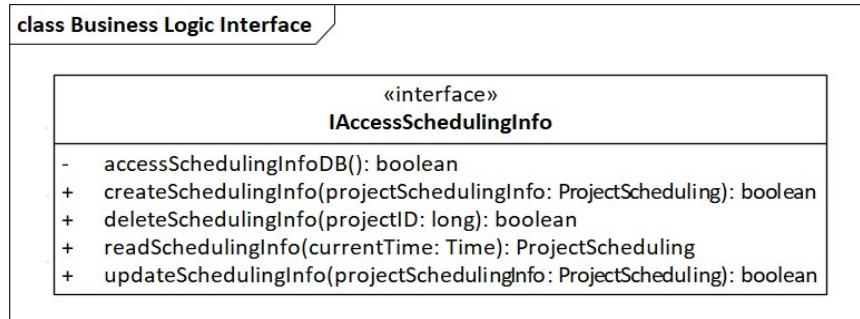
5.2.2.4. Data Layer

5.2.2.4.1. ProjectInfo Specification

5.2.2.4.2. Interface List

Name	Kind	Description
IAccessProjectInfo	Provided	프로젝트 정보에 CRUD 관리 인터페이스
IAccessSchedulingInfo	Provided	프로젝트 스케줄링 데이터에 CRUD 관리 인터페이스
IAccessDB	Required	프로젝트 정보가 저장되어 있는 DB에 접근하는 인터페이스

5.2.2.4.3. IAccessSchedulingInfo Interface Specification



Operation	Responsibility
accessSchedulingInfoDB	스케줄링 DB에 접근한다
createSchedulingInfo	스케줄링 DB에 데이터를 생성한다.
deleteSchedulingInfo	DB에 저장되어 있는 스케줄링 데이터를 삭제한다.
readSchedulingInfo	DB에 저장되어 있는 스케줄링 데이터를 읽는다.
updateSchedulingInfo	DB에 저장되어 있는 스케줄링 데이터를 수정한다.

«dataType» Project	«dataType» ProjectManager	«dataType» QualityType	«dataType» ProjectScheduling
<ul style="list-style-type: none"> - language: String - lowQualityScore: double - projectDeveloper: int - projectId: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean 	<ul style="list-style-type: none"> - projectInfo: Project - projectSize: byte - scheduleTime: Time

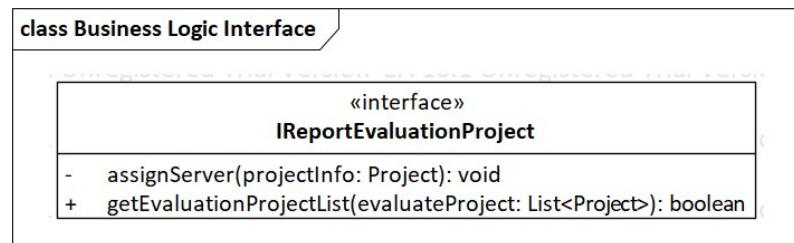
5.2.2.5. Internal Interface Layer

5.2.2.5.1. ProjectInfo Specification

5.2.2.5.2. Interface List

Name	Kind	Description
IReportEvaluationProject	Provided	[품질 분석 수행 스케줄링 리스트]의 시간에 맞춰서 프로젝트 품질 분석 수행 요청 인터페이스
ICollectEvaluationProjectInfo	Required	품질 평가 수행서버에 요청하는 인터페이스

5.2.2.5.3. IReportEvaluationProject Interface Specification

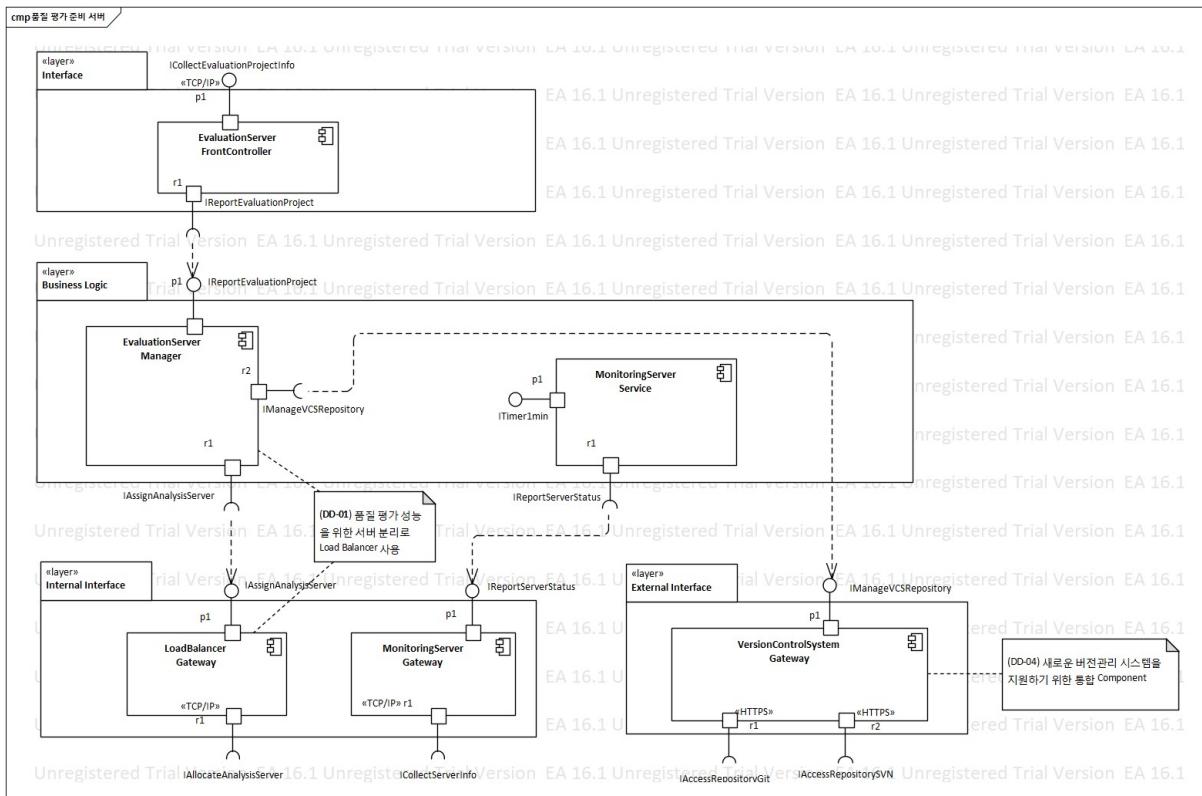


Operation	Responsibility
assignServer	전달받은 프로젝트를 품질 분석을 위해 Load Balancer에 전달 한다.
getEvaluationProjectList	품질 평가 수행할 프로젝트 리스트를 받는다.

«dataType» Project	«dataType» ProjectManager	«dataType» QualityType
<ul style="list-style-type: none"> - language: String - lowQualityScore: double - projectDeveloper: int - projectId: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

5.2.3. 품질 평가 준비 서버 – Static Structure Model

5.2.3.1. Static Structure Diagram



5.2.3.2. Element List

Name	Responsibility	Relevant ADs
Interface Layer		
EvaluationServer FrontController	품질 분석 프로젝트의 정보를 소스코드 품질 평가 시스템 관리서버로부터 Data를 받아 처리하다. 프로젝트 품질 분석 기능 수행을 위해 Business Logic내 Component에 전달한다.	UC-04, QA-01
Business Logic Layer		
EvaluationServer Manager	전달받은 프로젝트의 정보를 참고하여 VersionControl SystemManager Component에 버전관리 주소를 전달한다. 버전관리 시스템으로부터 소스코드가 다운로드가 되었다면 (DD-01) 품질 평가 수행 시간을 위해서 100MB기준으로 파일을 그룹화시켜 분석 서버에 전달한다.	UC-04, QA-01
MonitoringServer Service	매 1분마다 시스템 상태 탐지 서버에서 모니터링 할 수 있도록 Heartbeat 신호를 전달한다.	UC-06, QA-02
Internal Interface		
AnalysisServer LoadBalancer	100MB기준으로 그룹화된 파일을 분석할 수 있도록 분석 서버에게 파일과 프로젝트 정보를 전달한다. (DD-01) 품질 평가 수행 시간을 위해서 분석 서버를 분산시키는 Load Balancer에게 전달한다.	UC-04, QA-01

MonitoringServer Gateway	Heartbeat 신호를 시스템 상태 탐지 서버에 전달한다.	UC-06, QA-02
External Interface		
VersionControl SystemGateway	VCS 별 Repository 주소 유효성과 Repository 다운로드에 대한 요청을 전달하는 Gateway (<u>DD-04</u>)	UC-04, QA-01, QA-04

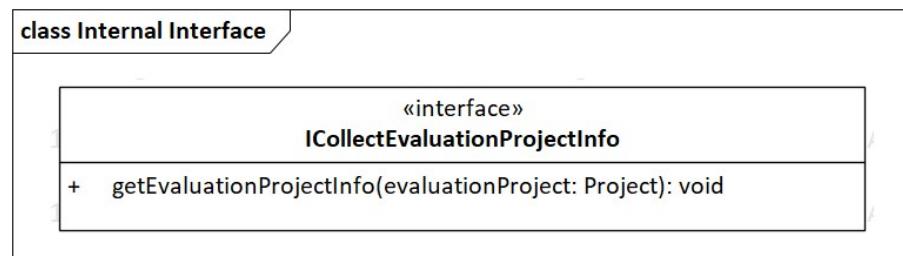
5.2.3.3. Interface Layer

5.2.3.3.1. EvaluationServerGateway Specification

5.2.3.3.2. Interface List

Name	Kind	Description
ICollectEvaluationProjectInfo	Provided	프로젝트 평가 정보를 받는 인터페이스
IEvaluationServer	Required	품질 분석하기 위해 프로젝트 정보를 전송하는 인터페이스

5.2.3.3.2.3. ICollectEvaluationProjectInfo Interface Specification



Operation	Responsibility
getEvaluationProjectInfo	품질 평가할 프로젝트 정보를 전달받는다.
«dataType» Project	
- Language: String - lowQualityScore: double - projectDeveloper: int - projectId: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String	«dataType» ProjectManager
	- id: long - nickname: String - password: long - phoneNumber: String
	«dataType» QualityType
	- maintenance: boolean - performance: boolean - security: boolean

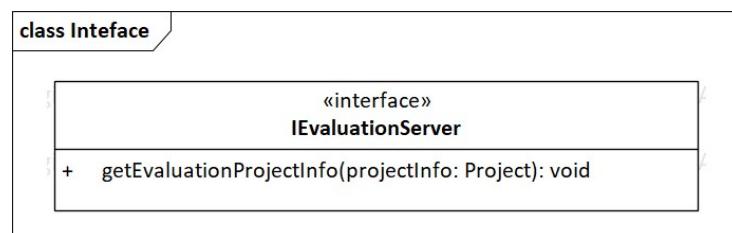
5.2.3.4. Business Logic Layer

5.2.3.4.1. EvaluationServerManager Specification

5.2.3.4.2. Interface List

Name	Kind	Description
IEvaluationServer	Provided	품질 분석할 프로젝트 정보를 받는 인터페이스
IManageSourceCode	Required	품질 분석할 프로젝트 정보의 버전관리 주소로 소스코드를 다운로드하는 인터페이스
IAssignAnalysisServer	Required	100MB마다 그룹화된 파일들을 품질 분석하기 위해 분석 서버 할당하는 인터페이스

5.2.3.4.3. IEvaluationServer Interface Specification



Operation	Responsibility
getEvaluationProjectInfo	품질 평가할 프로젝트 정보를 전달받는다.

«dataType» Project	«dataType» ProjectManager	«dataType» QualityType
<ul style="list-style-type: none"> - Language: String - lowQualityScore: double - projectDeveloper: int - projectID: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

5.2.3.5. External Interface Layer

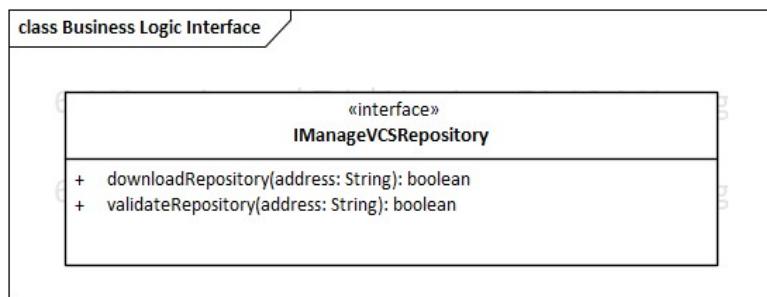
5.2.3.5.1. VersionControlSystemGateway Specification

5.2.3.5.2. Interface List

Name	Kind	Description
IManageVCSRepository	Provided	버전관리 시스템의 기능을 제공하는 인터페이스
IAccessRepositoryGit	Required	Git 버전관리 시스템의 기능을 사용하는 인터페이스

IAccessRepositorySVN	Required	SVN 버전관리 시스템의 기능을 사용하는 인터페이스
----------------------	----------	------------------------------

5.2.3.5.3. IManageVCSRepository Interface Specification



Operation	Responsibility
validateRepository	전달받은 Repository Address가 유효한지 여부를 알려준다.
downloadRepository	전달받은 Repository Address의 프로젝트 파일들을 다운받는다.

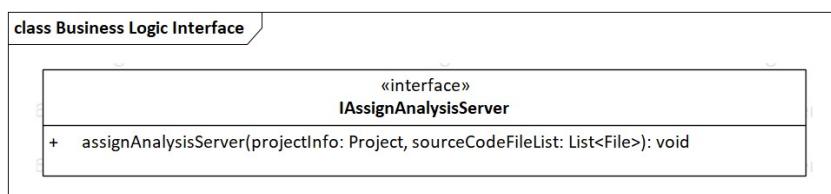
5.2.3.6. Internal Interface Layer

5.2.3.6.1. AnalysisServer Load Balancer Specification

5.2.3.6.2. Interface List

Name	Kind	Description
IAssignAnalysisServer	Provided	품질 분석할 프로젝트 정보를 받는 인터페이스
IAllocateAnalysisServer	Required	품질 분석 서버에 프로젝트에 대한 정보, 파일들을 전송하는 인터페이스

5.2.3.6.2.3. IAssignAnalysisServer Interface Specification

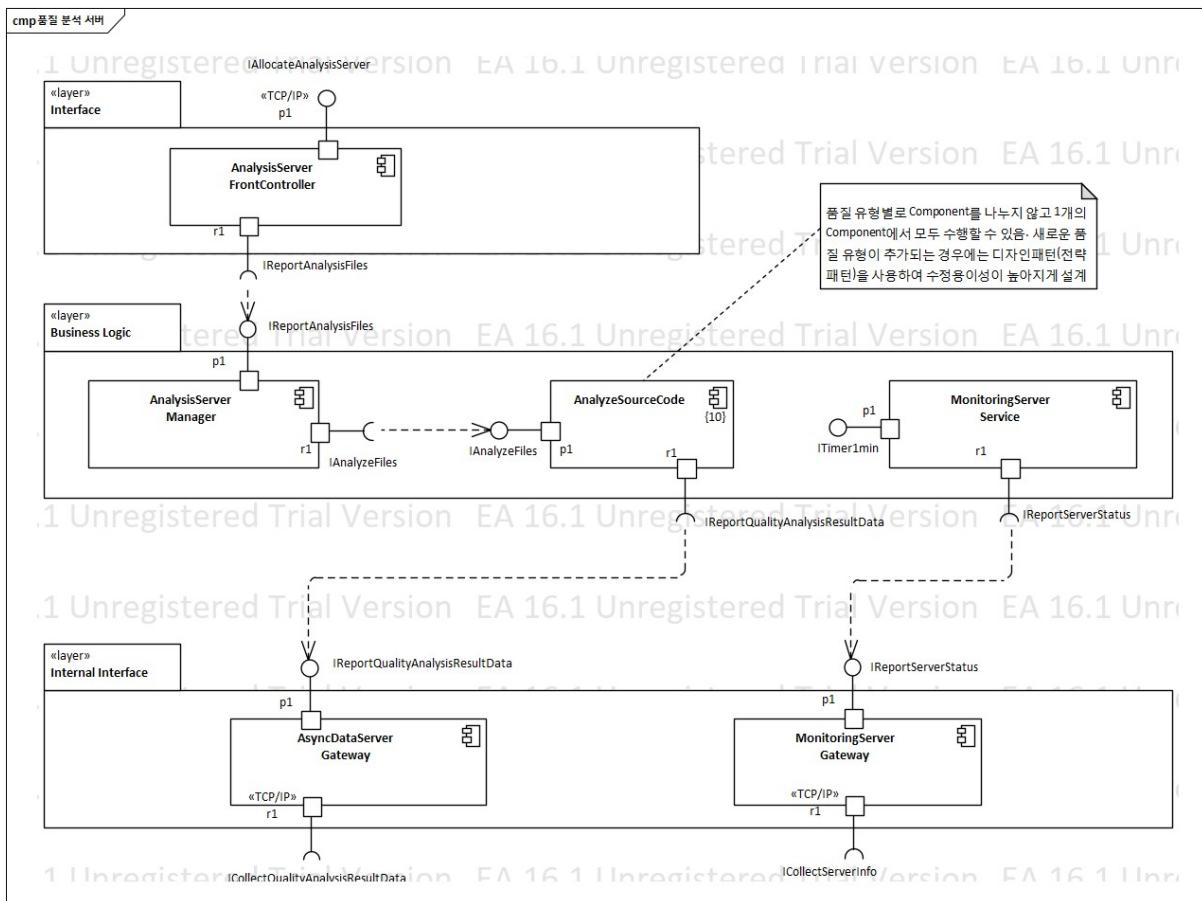


Operation	Responsibility
assignAnalysisServer	품질 분석을 위해 프로젝트의 정보와 소스코드 파일을 전달받는다. 소스코드 파일은 총 100MB로 제한되어 있다.

«dataType» Project	«dataType» ProjectManager	«dataType» QualityType
<ul style="list-style-type: none"> - language: String - lowQualityScore: double - projectDeveloper: int - projectId: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

5.2.4. 품질 분석 서버 – Static Structure Model

5.2.4.1. Static Structure Diagram



5.2.4.2. Element List

Name	Responsibility	Relevant ADs
Interface Layer		
AnalysisServer FrontController	100MB 파일과 프로젝트 정보를 전달받아 분석할 수 있도록 처리한다. Business Logic의 AnalysisServerManager Component에 전달한다.	UC-04, QA-01

Business Logic Layer		
AnalysisServer Manager	<p>전달받은 <u>10MB</u> 파일을 품질 유형별로 분석한다. 분석을 위해 AnalyzeSourceCode Component에 품질 분석하도록 Data를 전달한다.</p> <p>* 10MB: 100MB으로 나뉘어서 각 서버에 전달되고 컴포넌트는 10개로 나뉘어 1개의 컴포넌트당 10MB</p>	QA-01
Analyze SourceCode	전달받은 파일들을 품질 유형별로 분석하여 동기화/저장서버에 결과 데이터를 JSON으로 변환하여 전달한다.	UC-04, QA-01
MonitoringServer Service	매 1분마다 시스템 상태 탐지 서버에서 모니터링 할 수 있도록 Heartbeat 신호를 전달한다.	UC-06, QA-02
Internal Interface		
AsyncDataServer Gateway	프로젝트의 분할된 파일을 동기화를 위해 분석된 결과 데이터를 전달한다.	UC-04, QA-01
MonitoringServer Gateway	Heartbeat 신호를 시스템 상태 탐지 서버에 전달한다.	UC-06, QA-02

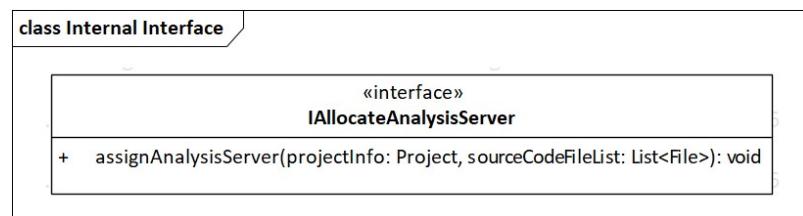
5.2.4.3. Interface Layer

5.2.4.3.1. AnalysisServerFrontController Specification

5.2.4.3.2. Interface List

Name	Kind	Description
IAllocateAnalysisServer	Provided	품질 분석 서버에 프로젝트에 대한 정보, 파일들을 전달받는 인터페이스
IReportAnalysisFiles	Required	품질 분석 서버에 프로젝트에 대한 정보를 전달받기 위한 인터페이스

5.2.4.3.3. IAllocateAnalysisServer Interface Specification



Operation	Responsibility
assignAnalysisServer	품질 분석을 위해 프로젝트의 정보와 소스코드 파일을 전달받는다. 소스코드 파일은 총 100MB로 제한되어 있다.

«dataType» Project	«dataType» ProjectManager	«dataType» QualityType
<ul style="list-style-type: none"> - language: String - lowQualityScore: double - projectDeveloper: int - projectID: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

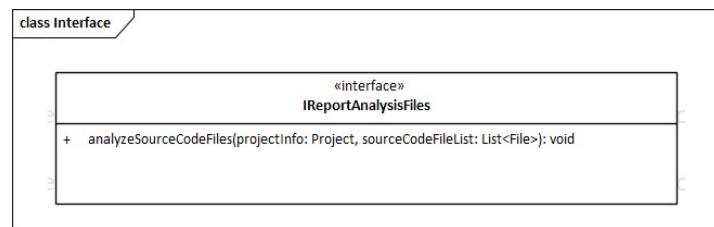
5.2.4.4. Business logic Layer

5.2.4.4.1. AnalysisServerManager Specification

5.2.4.4.2. Interface List

Name	Kind	Description
IReportAnalysisFiles	Provided	품질 분석 할 소스코드를 전달받는 인터페이스.
IAnalyzeFiles	Required	소스코드를 품질 분석하기 위해 전달받는 인터페이스.

5.2.4.4.3. IReportAnalysisFiles Interface Specification



Operation	Responsibility
analyzeSourceCodeFiles	소스코드 파일과 함께 프로젝트 정보들을 전달받아 품질 분석을 진행한다.

«dataType» Project	«dataType» ProjectManager	«dataType» QualityType
<ul style="list-style-type: none"> - language: String - lowQualityScore: double - projectDeveloper: int - projectID: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

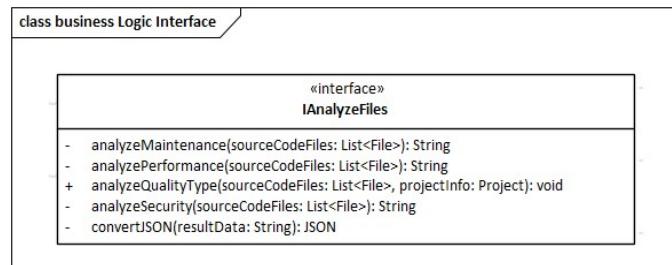
5.2.4.4.4. AnalyzeSourceCode Specification

5.2.4.4.5. Interface List

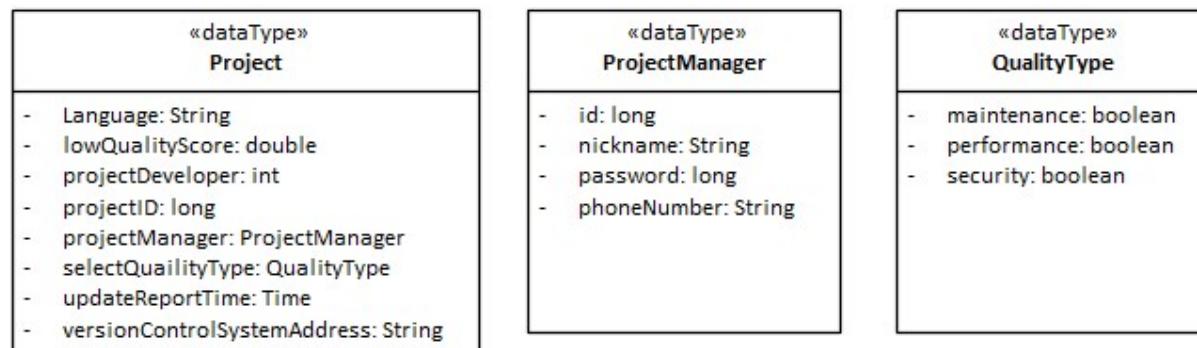
Name	Kind	Description

IAnalyzeFiles	Provided	소스코드를 품질 분석하는 인터페이스.
IReportQualityAnalysisResultData	Required	품질 분석 결과 데이터가 동기화 작업이 필요하여 수집하기 위한 인터페이스

5.2.4.4.6. IAnalyzeFiles Interface Specification



Operation	Responsibility
analyzeQualityType	전달받은 소스코드를 품질 유형별 분석할 수 있도록 관여한다.
analyzeMaintenance	소스코드를 유지보수성에 대한 품질 유형을 분석한다.
analyzePerformance	성능에 대한 품질 유형을 분석한다.
analyzeSecurity	보안에 대한 품질 유형을 분석한다.
convertJSON	동기화 서버에 데이터를 전달하기 위해 String형태인 결과값을 JSON형태로 변환한다.



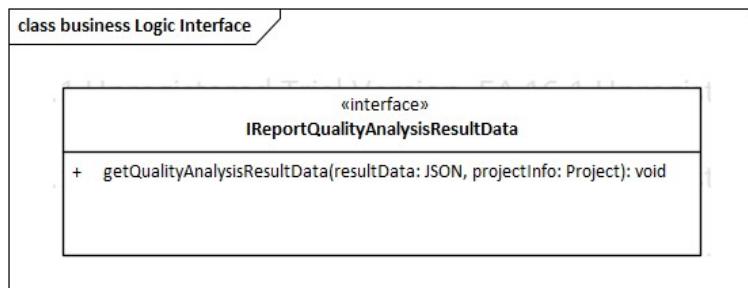
5.2.4.5. Internal Interface Layer

5.2.4.5.1. AsyncDataServerGateway Specification

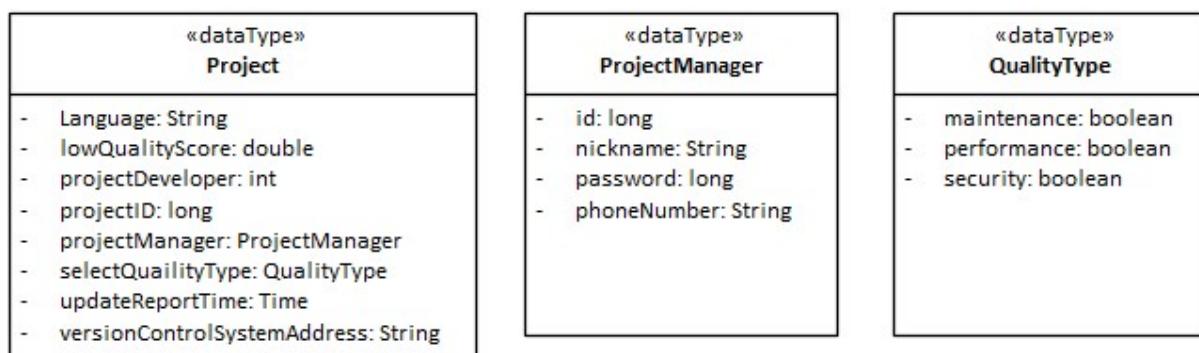
5.2.4.5.2. Interface List

Name	Kind	Description
IReportQualityAnalysisResultData	Provided	품질 분석 결과 데이터를 받는 인터페이스
ICollectQualityAnalysisResultData	Required	품질 분석 결과 데이터를 동기화서버에 전달하기 위한 인터페이스

5.2.4.5.3. IReportQualityAnalysisResultData Interface Specification

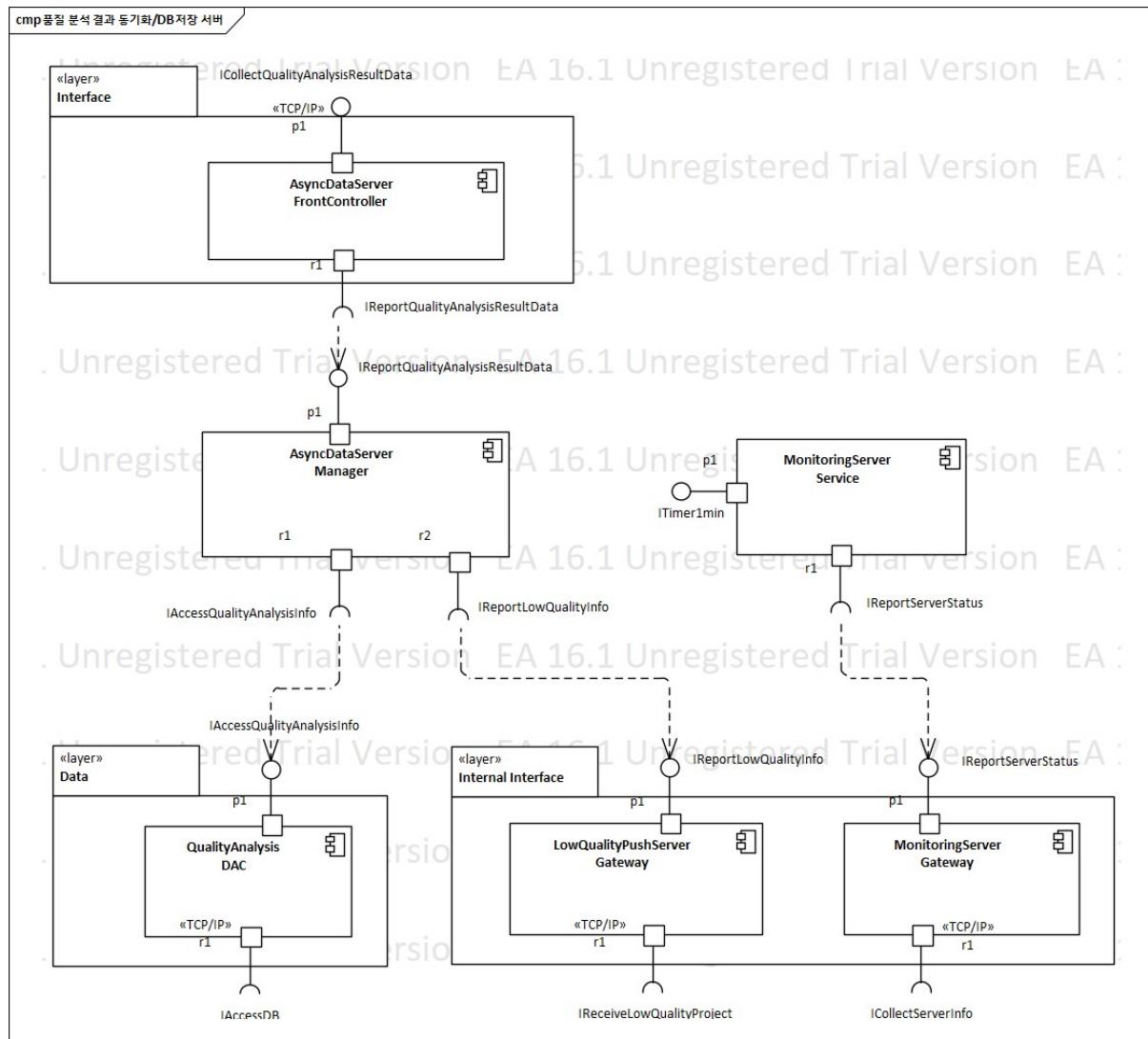


Operation	Responsibility
getQualityAnalysisResultData	JSON형식으로 품질 분석 결과 데이터를 전달받는다.



5.2.5. 품질 분석 결과 동기화/DB 저장 서버 – Static Structure Model

5.2.5.1. Static Structure Diagram



5.2.5.2. Element List

Name	Responsibility	Relevant ADs
User Interface Layer		
AsyncDataServer FrontController	프로젝트의 분할 파일에 대한 분석 결과 데이터를 받아 동기화를 수행한다. 동기화 작업을 위해 Business Logic의 AsyncDataServerManager Component에 데이터를 전달 한다.	UC-04, QA-01
Business Logic Layer		
AsyncDataServer Manager	여러 분석 서버로부터 모든 분석 결과 데이터를 받을 때까지 대기한다. 모두 전달받으면 결과 데이터를 변환하여 DB에 저장한다. 결과 점수가 저 품질 기준 점수보다 낮으면 저 품질 프로젝트 정보를 담당 프로젝트 매니저에게 알림을 보내기 위해 저 품질 알림 서버에 프로젝트 정보를 전달한다.	UC-04, UC-05, QA-01

MonitoringServer Service	매 1분마다 시스템 상태 탐지 서버에서 모니터링 할 수 있도록 Heartbeat 신호를 전달한다.	UC-06, QA-02
Data Layer		
QualityAnalysis DAC	프로젝트 결과 데이터가 저장되는 Database에 접근하여 DB 값을 읽고/쓴다.	UC-04, QA-01
Internal Interface		
LowQualityPush ServerGateway	저 품질 프로젝트 정보를 저 품질 프로젝트 알림 서버에 전달한다.	UC-05, QA-01
MonitoringServer Gateway	Heartbeat 신호를 시스템 상태 탐지 서버에 전달한다.	UC-06, QA-02

5.2.5.3. Interface Layer

5.2.5.3.1. AsynDataServerGateway Specification

5.2.5.3.2. Interface List

Name	Kind	Description
ICollectQualityAnalysisResultData	Provided	품질 분석 결과데이터를 JSON 형식으로 받는 인터페이스
IReportQualityAnalysisResultData	Required	동기화를 위해 품질 분석 결과 데이터를 전달하는 인터페이스

5.2.5.3.3. ICollectQualityAnalysisResultData Interface Specification

class Internal Interface	
<pre>«interface» ICollectQualityAnalysisResultData + getQualityAnalysisResultData(resultData: JSON, projectInfo: Project): void</pre>	
Operation	Responsibility
getQualityAnalysisResultData	JSON형식으로 품질 분석 결과 데이터를 받는다.

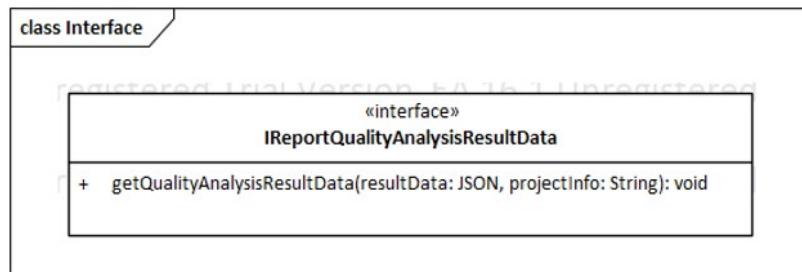
5.2.5.4. Business logic Layer

5.2.5.4.1. ProjectEvaluationManager Specification

5.2.5.4.2. Interface List

Name	Kind	Description
IReportQualityAnalysisResultData	Provided	DB에 저장하기 위해선 프로젝트의 품질 분석 결과 데이터의 동기화가 필요하여 결과 데이터를 JSON형식으로 받는 인터페이스
IAccessQualityAnalysisInfo	Required	품질 결과 데이터를 저장하기 위해 데이터를 전달하는 인터페이스
IReportLowQualityInfo	Required	저 품질 프로젝트인 경우 Push알림을 프로젝트 매니저 스마트폰에 전송하기 위해 저 품질 프로젝트에 대한 정보를 전달하는 인터페이스

5.2.5.4.3. IReportQualityAnalysisResultData Interface Specification



Operation	Responsibility
getQualityAnalysisResultData	JSON형식으로 품질 분석 결과 데이터를 받는다.

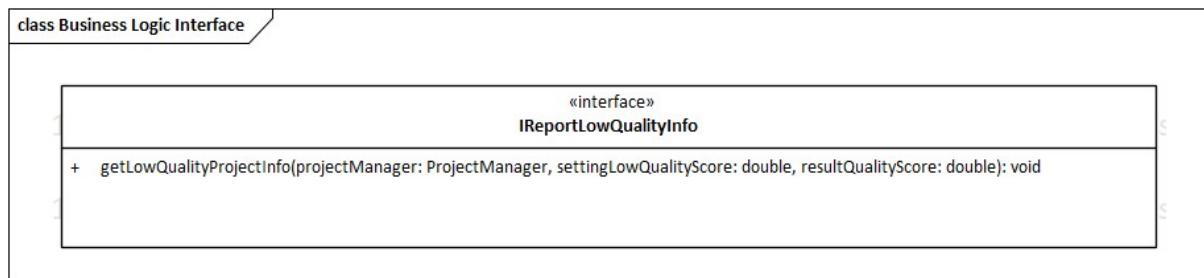
5.2.5.5. Internal Interface Layer

5.2.5.5.1. LowQualityPushServerGateway Specification

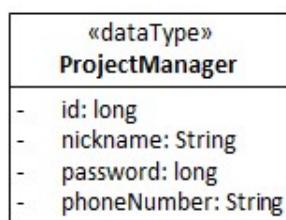
5.2.5.5.2. Interface List

Name	Kind	Description
IReportLowQualityInfo	Provided	프로젝트 매니저 스마트폰에 저 품질 프로젝트 Push 알림을 보내기 위해 저 품질 프로젝트 정보를 받는 인터페이스
IReceiveLowQualityProject	Required	저 품질 프로젝트 Push알림을 보내기 위해 저 품질 프로젝트 정보를 전달하는 인터페이스

5.2.5.3. IReportLowQualityInfo Interface Specification



Operation	Responsibility
getLowQualityProjectInfo	저 품질 기준 점수보다 낮은 프로젝트의 정보를 전달받는다.



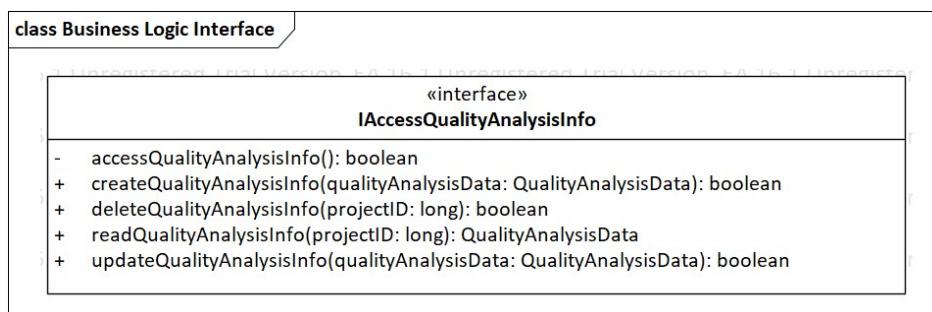
5.2.5.4. Data Layer

5.2.5.4.1. QualityAnalysisDAC Specification

5.2.5.4.2. Interface List

Name	Kind	Description
IAccessQualityAnalysisInfo	Provided	품질 결과 DB의 CRUD 관리 인터페이스
IAccessDB	Required	품질 결과 데이터가 저장되어 있는 DB에 접근하는 인터페이스

5.2.5.4.3. IAccessQualityAnalysisInfo Interface Specification



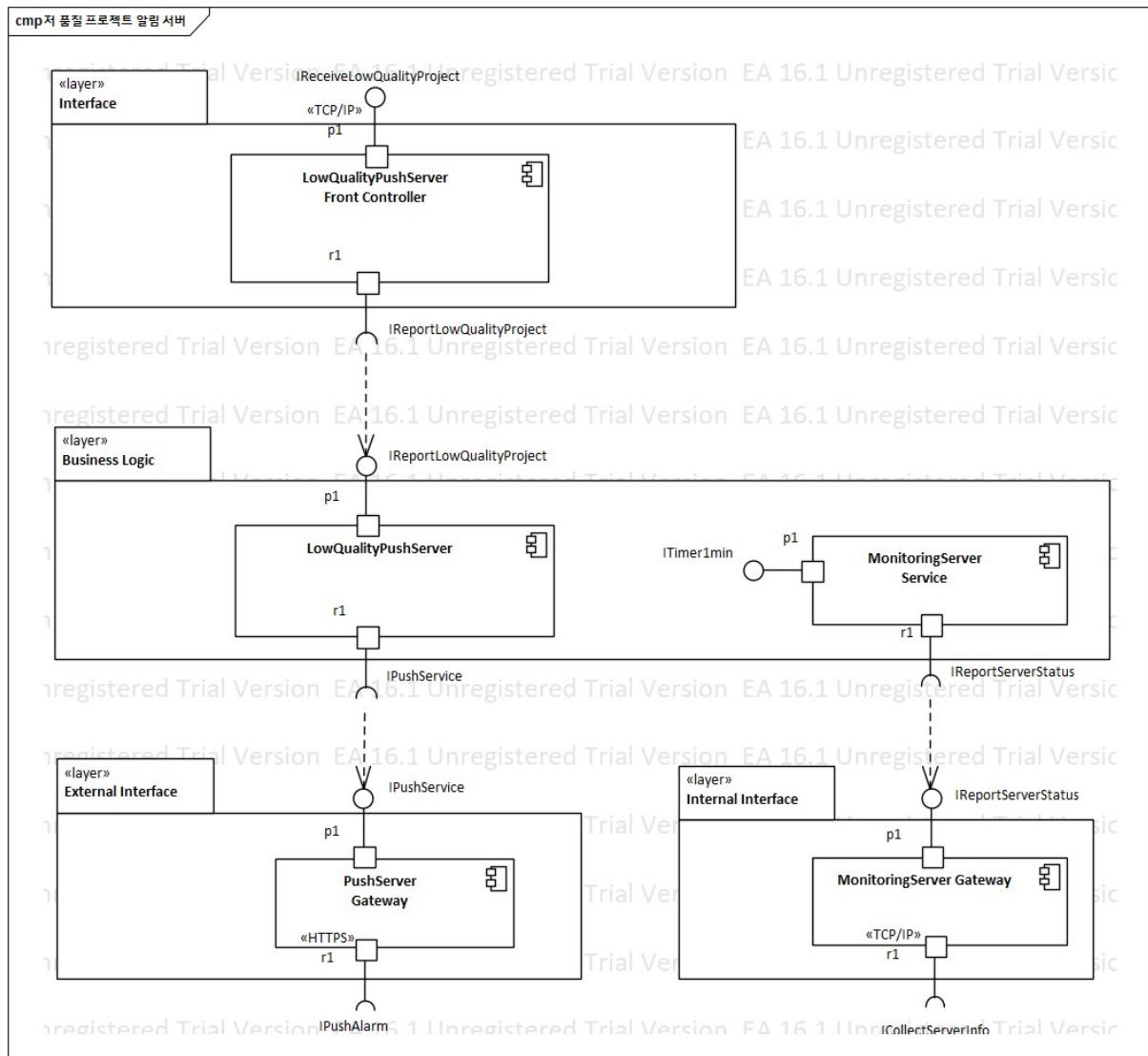
Operation	Responsibility
accessQualityAnalysisInfo	품질 분석 DB에 접근한다

createQualityAnalysisInfo	품질 분석 DB에 데이터를 생성한다.
deleteQualityAnalysisInfo	DB에 저장되어 있는 품질 분석 데이터를 삭제한다.
readQualityAnalysisInfo	DB에 저장되어 있는 품질 분석 데이터를 읽는다.
updateQualityAnalysisInfo	DB에 저장되어 있는 품질 분석 데이터를 수정한다.

«dataType» QualityAnalysisData
<ul style="list-style-type: none"> - maintenanceAnalysisData: String - maintenanceGuideline: String - maintenanceScore: int - performanceAnalysisData: String - performanceGuideline: String - performanceScore: int - securityAnalysisData: String - securityGuideline: String - securityScore: int - totalScore: int

5.2.6. 저 품질 프로젝트 알림 서버 – Static Structure Model

5.2.6.1. Static Structure Diagram



5.2.6.2. Element List

Name	Responsibility	Relevant ADs
Interface Layer		
LowQualityPushServerFrontController	전달받은 저 품질 프로젝트 정보를 Business Logic의 LowQualityPushServer Component에 데이터를 전달한다.	UC-04, UC-05, QA-01
Business Logic Layer		
LowQualityPushServer	저 품질 프로젝트 정보를 담당 프로젝트 매니저 스마트폰에 Push 알림을 받을 수 있도록 Push서버에 요청한다.	UC-04, UC-05, QA-01
MonitoringServerService	매 1분마다 시스템 상태 탐지 서버에서 모니터링 할 수 있도록 Heartbeat 신호를 전달한다.	UC-06, QA-02

Internal Layer		
MonitoringServer Gateway	Heartbeat 신호를 시스템 상태 탐지 서버에 전달한다.	UC-06, QA-02
External Interface		
PushServer Gateway	프로젝트매니저 스마트폰에 Push알림을 전달할 수 있도록 Push서버에 전달한다.	UC-04, UC-05, QA-01

5.2.6.3. Interface Layer

5.2.6.3.1. LowQualityPushServerFrontController Specification

5.2.6.3.2. Interface List

Name	Kind	Description
IReceiveLowQualityProject	Provided	저 품질 프로젝트의 정보를 받는 인터페이스
ILowQualityProject	Required	저 품질 프로젝트의 정보를 Push서버에 보내기 위해서 전달하는 인터페이스

5.2.6.3.3. IReceiveLowQualityProject Interface Specification



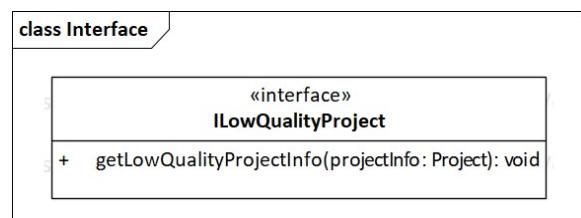
5.2.6.4. Business logic Layer

5.2.6.4.1. LowQualityPushServer Specification

5.2.6.4.2. Interface List

Name	Kind	Description
ILowQualityProject	Provided	저 품질 프로젝트의 정보를 받는 인터페이스
IPushService	Required	Push알람을 보내기 위해 대상, 정보를 전달하는 인터페이스

5.2.6.4.3. ILowQualityProject Interface Specification



Operation	Responsibility
getLowQualityProjectInfo	저 품질 프로젝트 정보를 전달받는다.

«dataType» Project	«dataType» ProjectManager	«dataType» QualityType
<ul style="list-style-type: none"> - Language: String - lowQualityScore: double - projectDeveloper: int - projectID: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

5.2.6.5. External Interface Layer

5.2.6.5.1. PushServerGateway Specification

5.2.6.5.2. Interface List

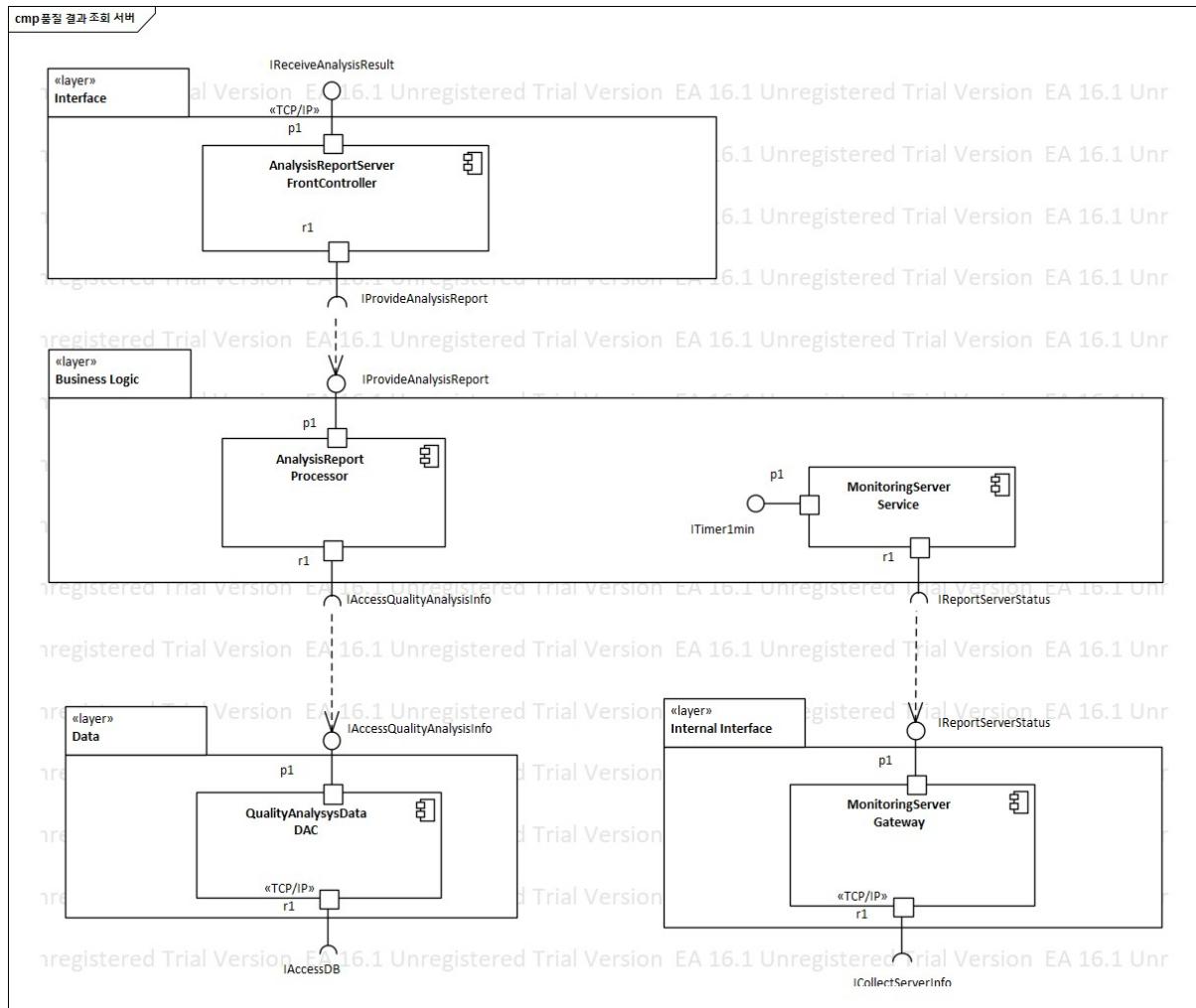
Name	Kind	Description
IPushService	Provided	Push알람을 보내기 위해 대상, 정보를 전달받는 인터페이스
IPushAlarm	Required	스마트폰에 정보를 알림보내는 인터페이스

5.2.6.5.3. IPushService Interface Specification

class Business Logic Interface	
«interface» IPushService	
+ getPushInfo(message: String, projectManager: ProjectManager): void	
Operation	
getPushInfo	프로젝트 매니저 스마트폰에 보낼 메시지의 정보와 프로젝트 매니저의 정보를 받는 인터페이스
«dataType» ProjectManager	
<ul style="list-style-type: none">- id: long- nickname: String- password: long- phoneNumber: String	

5.2.7. 품질 결과 조회 서버 – Static Structure Model

5.2.7.1. Static Structure Diagram



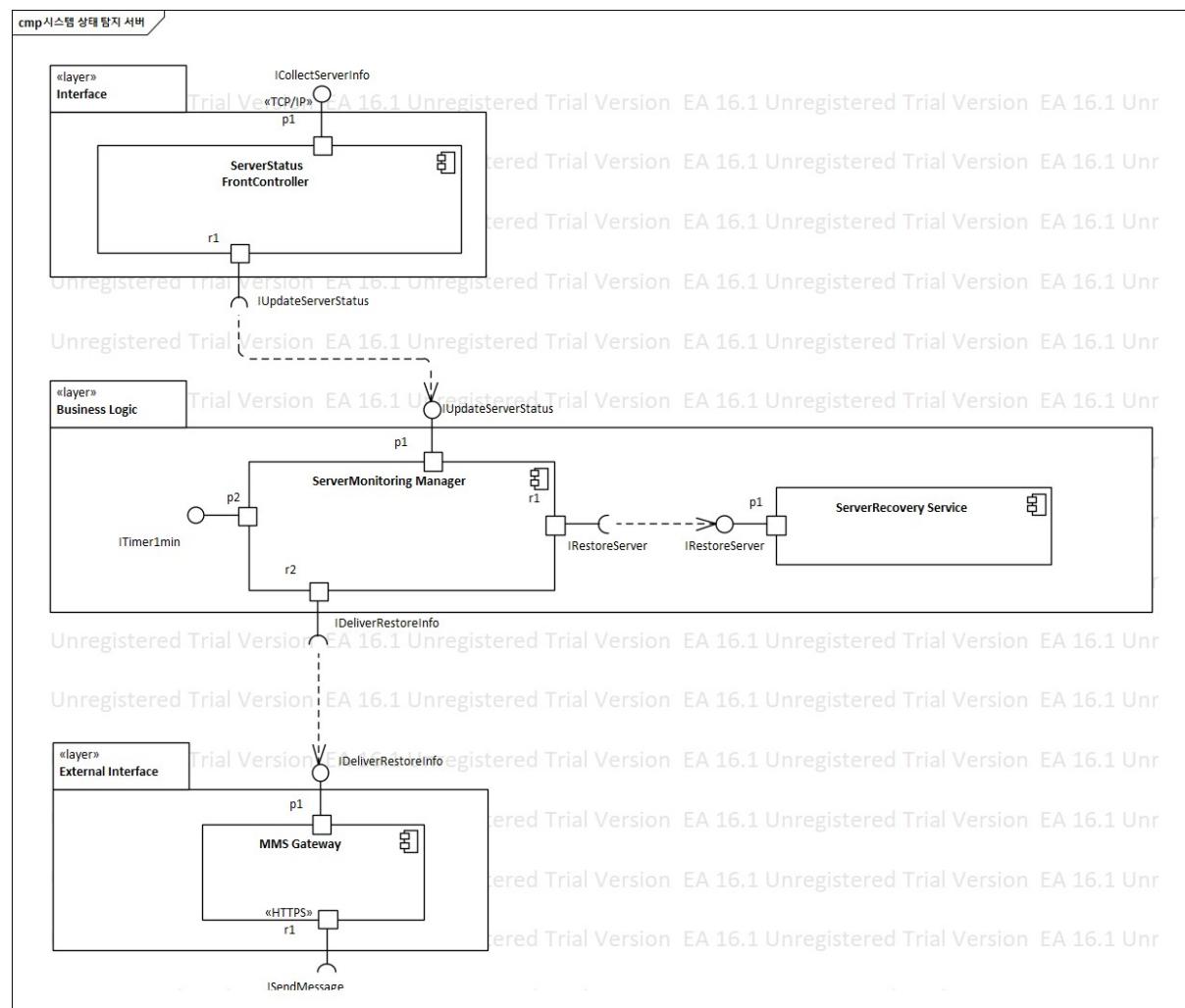
5.2.7.2. Element List

Name	Responsibility	Relevant ADs
Interface Layer		
AnalysisReportFrontController	소스코드 품질 평가 시스템 관리 서버로부터 전달받은 요청 결과 보고서에 대한 요청 데이터를 Business Logic의 AnalysisReportProcessor에 전달한다.	UC-02, UC-03
Business Logic Layer		
AnalysisReportProcessor	품질 분석 결과 데이터가 저장되어 있는 DB로부터 데이터 Read후 결과 보고서 형식으로 파일을 변환한다. 변환된 파일을 관리 서버에 전달한다.	UC-02, UC-03
MonitoringServerService	매 1분마다 시스템 상태 탐지 서버에서 모니터링 할 수 있도록 Heartbeat 신호를 전달한다.	UC-06, QA-02
Data Layer		

QualityAnalysis DataDAC	프로젝트 결과 데이터가 저장되는 Database에 접근하여 DB 값을 읽고/쓴다.	UC-02, UC-03
Internal Interface		
MonitroingServer Gateway	Heartbeat 신호를 시스템 상태 탐지 서버에 전달한다.	UC-06, QA-02

5.2.8. 시스템 상태 탐지 서버 – Static Structure Model

5.2.8.1. Static Structure Diagram



5.2.8.2. Element List

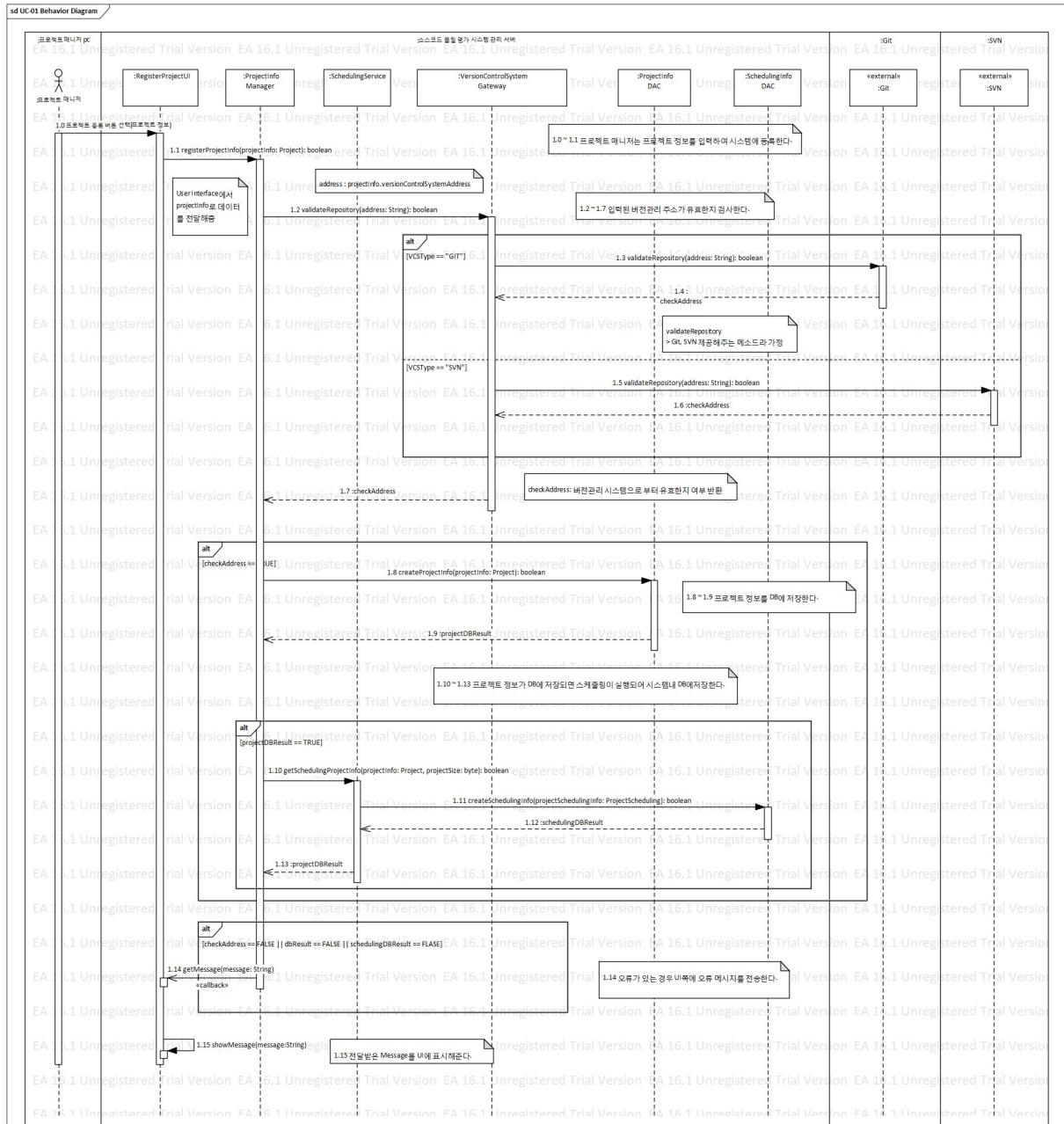
Name	Responsibility	Relevant ADs
Interface Layer		
ServerStatus FrontController	시스템 상태 탐지 서버를 제외한 서버에서 Heartbeat 신호를 받으면 Business Logic의 ServerMonitoringManager에게 전달한다.	UC-06, QA-02

Business Logic Layer		
ServerMonitoring Manager	매 1분마다 전달받은 Heartbeat 신호를 처리한다. 1분마다 Heartbeat 신호를 전달받지 못한 서버는 장애로 판단하여 ServerRecoveryService Component에 복구 기능을 요청한다. 시스템 관리자에게 장애 서버, 장애 발생 시간, 장애 유형, 복구 완료 시간을 MMS서버에 전달한다.	UC-06, QA-02
ServerRecovery Service	복구 요청 전달받은 서버를 재가동하여 복구한다.	UC-06, QA-02
External Interface		
MMS Gateway	전달받은 장애 서버, 장애 발생 시간, 장애 유형, 복구 완료 시간을 담당 시스템 관리자에게 MMS메시지 통보하기 위해 MMS서버에 전달한다.	UC-06, QA-02

5.3. Behavior View

5.3.1. UC-01 프로젝트 정보 관리 Behavior Model

5.3.1.1. Behavior Diagram



«dataType»	Project
- Language: String - lowQualityScore: double - projectDeveloper: int - projectId: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String	

«dataType»	ProjectManager
- id: long - nickname: String - password: long - phoneNumber: String	

«dataType»	QualityType
- maintenance: boolean - performance: boolean - security: boolean	

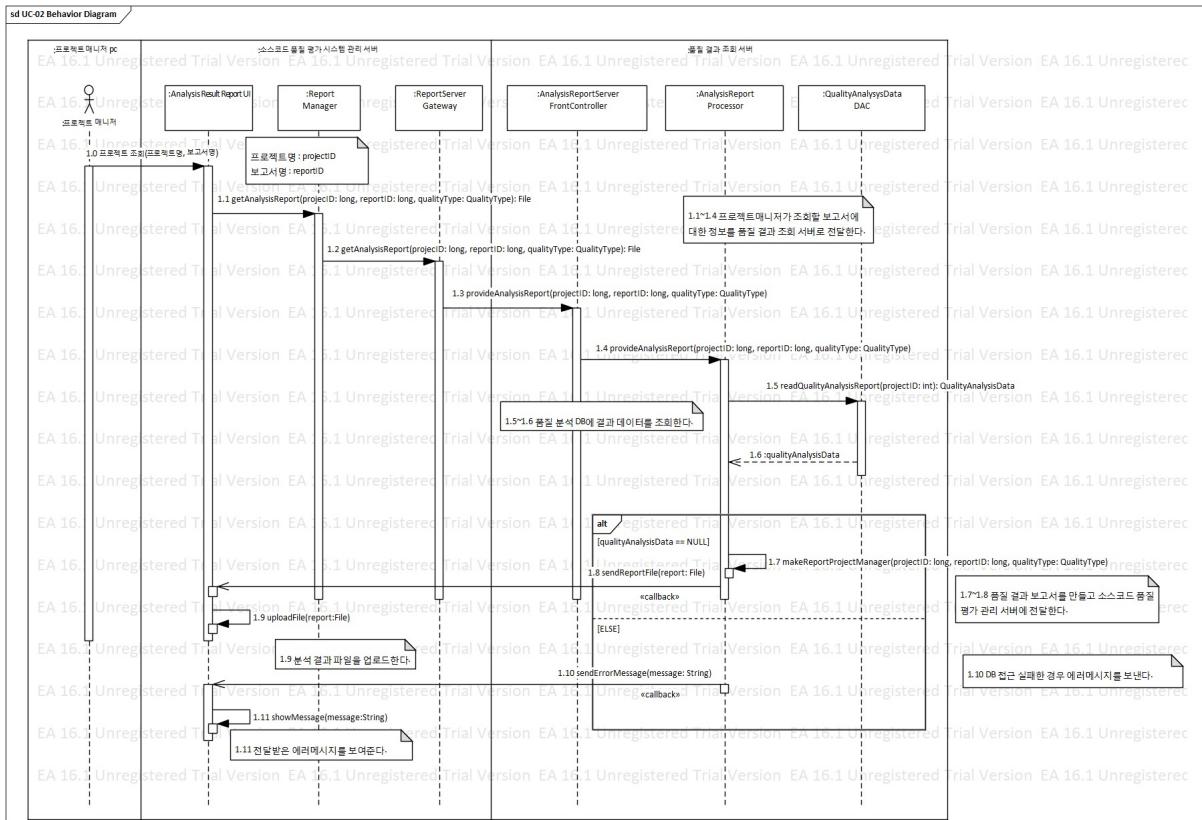
«dataType»	ProjectScheduling
- projectInfo: Project - projectSize: byte - scheduleTime: Time	

5.3.1.2. Behavior Description

No.	UC Step	Behavior Step
1	프로젝트 매니저는 [소스코드 버전관리 주소], [품질 결과 보고서 받는 시각 설정], [품질 유형 선택 (M, P, S)], [저 품질 프로젝트 기준 점수 입력]의 정보를 시스템에 입력한다.	[1.0] 프로젝트 매니저는 프로젝트 정보들을 입력한다.
2	프로젝트 매니저는 프로젝트 정보 등록 버튼을 클릭한다.	[1.0] RegisterProjectUI에 프로젝트 정보들과 이벤트를 보낸다.
3	시스템은 [소스코드 버전관리 주소]에 입력되어 있는 데이터를 참조하여 버전관리 시스템에 유효 주소인지 확인한다.	[1.2~1.7] 시스템은 버전관리 시스템에게 프로젝트 소스코드 주소가 유효한지 검사한다. Git, SVN 2가지 시스템을 지원하기 때문에 버전관리 시스템에 맞게 유효 검사를 진행한다.
4	IF (버전관리 주소 INVALID)	
5	프로젝트 매니저에게 페이지 화면에 버전관리 접근 실패 오류 메시지를 띄워준다.	[1.15] 주소가 유효하지 않는 경우 UI에 오류 메시지를 전송한다.
6	ELSE	
7	IF (DB 접속 실패)	
8	시스템은 프로젝트 매니저에게 웹페이지 화면에 “DB 접속 실패” 오류 메시지를 띄워준다.	[1.15] DB 접속 실패인 경우 UI에 오류 메시지를 전송한다.
9	ELSE	
10	시스템은 프로젝트 매니저로부터 입력된 데이터를 DB에 저장한다.	[1.8~1.9] 전달받은 프로젝트 정보들을 DB에 저장한다. 저장이 완료되면 Flag를 반환해준다.
11	시스템은 소스코드 품질 평가시스템의 [품질 분석 수행 스케줄링 리스트]에 추가한다.	[1.10~1.13] 프로젝트 등록 성공되면 프로젝트 품질 분석하기 위해 스케줄링 DB에 저장한다.
12	ENDIF	
13	ENDIF	

5.3.2. UC-02 프로젝트 결과 조회 Behavior Model

5.3.2.1. Behavior Diagram



[DataType]

«dataType»	«dataType»
QualityAnalysisData	QualityType
<ul style="list-style-type: none"> - maintenanceAnalysisData: String - maintenanceGuideline: String - maintenanceScore: int - performanceAnalysisData: String - performanceGuideline: String - performanceScore: int - securityAnalysisData: String - securityGuideline: String - securityScore: int - totalScore: int 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

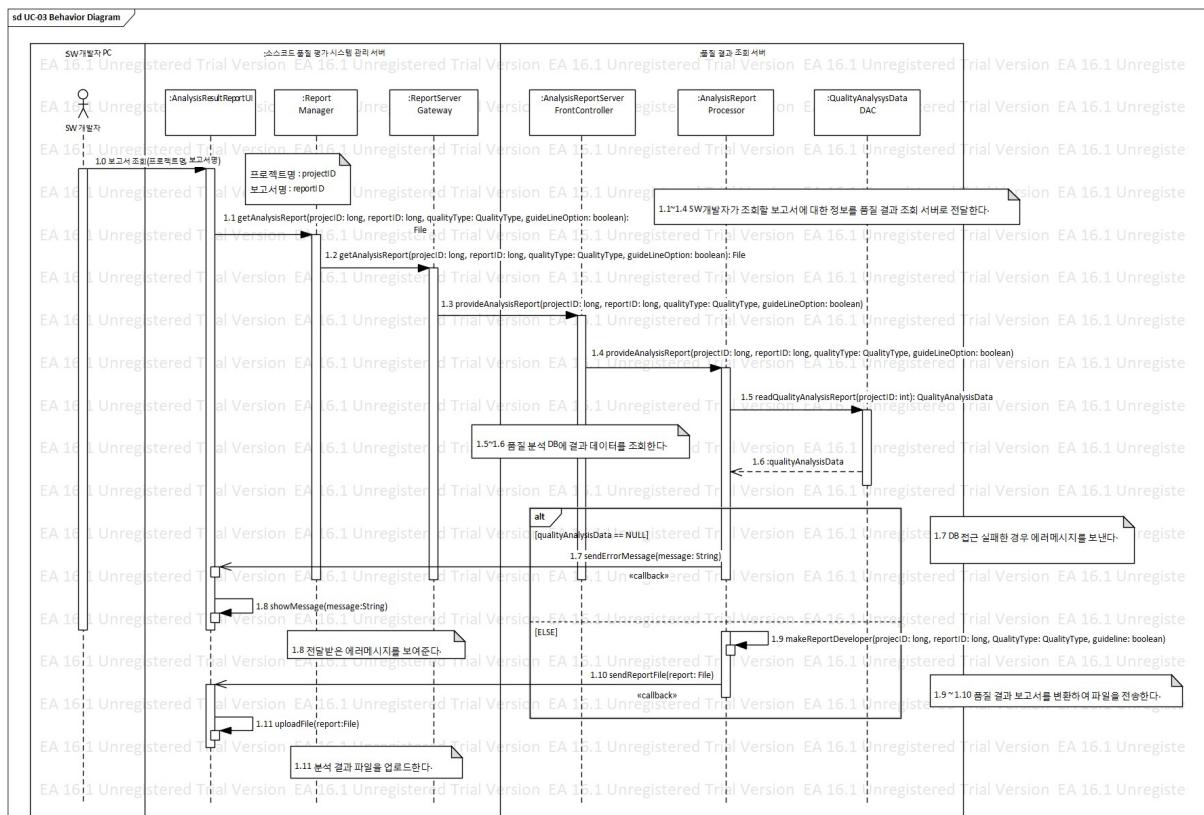
5.3.2.2. Behavior Description

No.	UC Step	Behavior Step
1	프로젝트 매니저는 Web, App을 통해 결과보고서 조회 리스트 버튼을 클릭한다.	[1.0] 프로젝트 매니저는 UI를 통해 조회할 보고서의 정보를 전달해준다. * UI단 생략
2	시스템은 프로젝트 매니저에 맞는 [조회 가능 결과 보고서 리스트]를 화면에 출력한다.	

3	프로젝트 매니저는 [조회 가능 결과 보고서 리스트]에서 조회할 결과 보고서를 선택한다.	
4	프로젝트 매니저는 결과보고서 조회 버튼을 클릭한다.	
5	IF (DB 접속 오류)	
6	시스템은 웹페이지를 통해 프로젝트 매니저에게 “DB 접속 실패” 오류 메시지를 띄워준다.	[1.9] DB 접근 실패한 경우 Analysis ResultReportUI에 오류메시지를 보낸다.
7	ELSE	
8	시스템은 DB서버로부터 저장되어 있는 데이터를 조회한다.	[1.5~1.6] 품질 결과 DB에서 조회할 결과 데이터를 Read한다.
9	IF (Mobile 환경인 경우)	
10	시스템은 프로젝트 매니저 Mobile 환경에 맞게 레이아웃을 조정하여 App에 결과 보고서를 출력한다.	[1.7~1.8] Read한 데이터를 품질 결과 보고서 형식으로 변환하여 전달한다. UI에서 Mobile 환경에 맞춰 출력해준다.
11	ELSE (Web 환경인 경우)	
12	시스템은 프로젝트 매니저에게 Web에 결과 보고서를 출력한다.	[1.7~1.8] Read한 데이터를 품질 결과 보고서 형식으로 변환하여 전달한다. UI에서 Web 환경에 맞춰 출력해준다.
13	ENDIF	
14	ENDIF	

5.3.3. UC-03 프로젝트 파일 별 결과 조회 Behavior Model

5.3.3.1. Behavior Diagram



[Data Type]

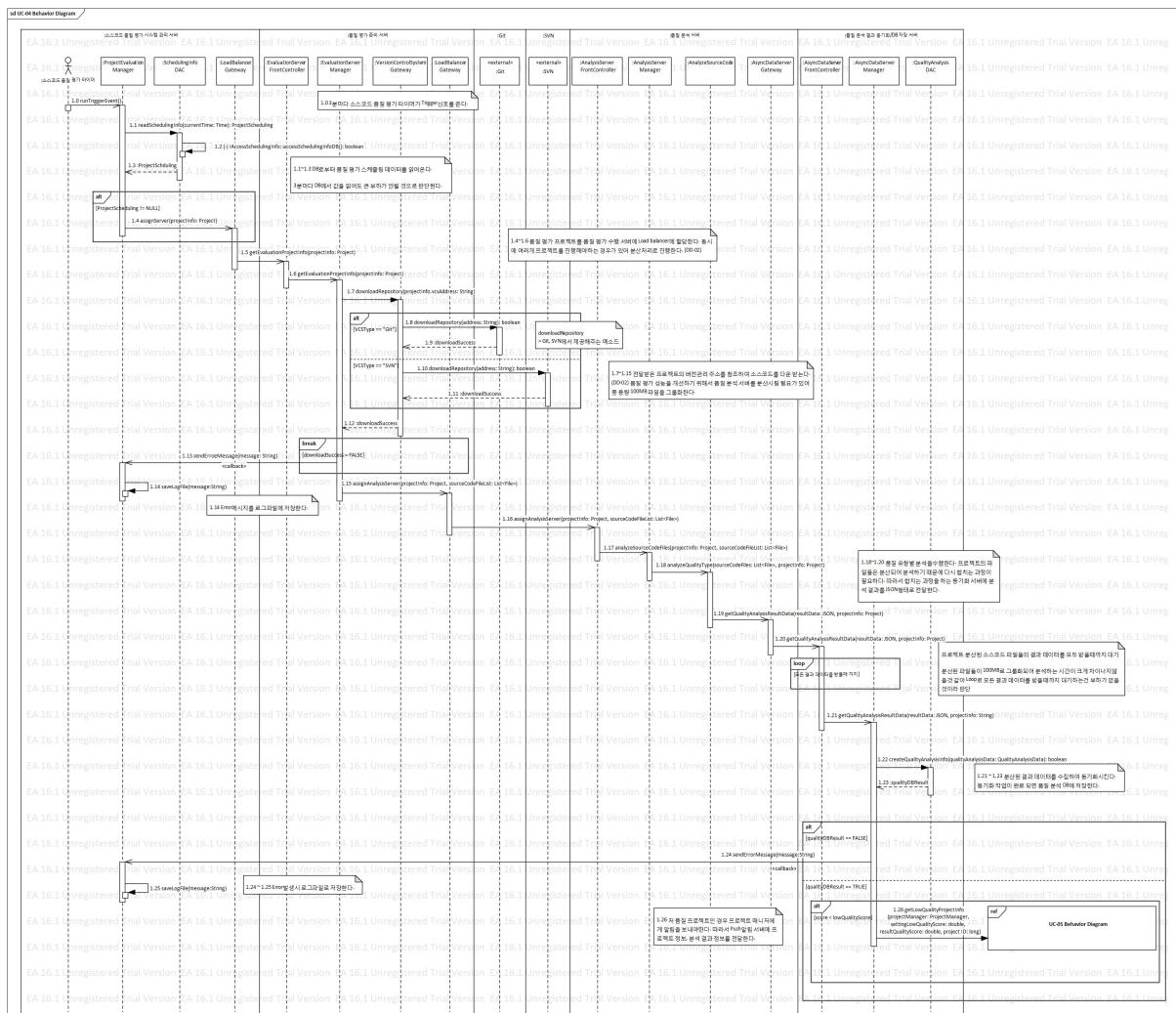
«dataType» QualityAnalysisData	«dataType» QualityType
<ul style="list-style-type: none"> - maintenanceAnalysisData: String - maintenanceGuideline: String - maintenanceScore: int - performanceAnalysisData: String - performanceGuideline: String - performanceScore: int - securityAnalysisData: String - securityGuideline: String - securityScore: int - totalScore: int 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

5.3.3.2. Behavior Description

No.	UC Step	Behavior Step
1	SW개발자는 Web을 통해 결과보고서 조회 리스트 버튼을 클릭한다.	[1.0] 프로젝트 매니저는 UI를 통해 조회할 보고서의 정보를 전달해준다.
2	시스템은 SW개발자에 맞는 [조회 가능 파일 별 결과 보고서 리스트]를 화면에 출력한다.	
3	SW개발자는 [조회 가능 파일 별 결과 보고서 리스트]에서 조회할 결과 보고서를 선택한다.	
4	SW개발자는 결과보고서 조회 버튼을 클릭한다.	
5	IF (DB 접속 오류)	
6	시스템은 웹페이지를 통해 SW개발자에게 “DB 접속 실패” 에러 메시지를 띄워준다.	[1.7 ~1.8] DB 접근 실패한 경우 Analysis ResultReportUI에 오류메시지를 보낸다.
7	ELSE	
8	시스템은 DB서버로부터 저장되어 있는 데이터를 조회한다.	[1.5~1.6] 품질 결과 DB에서 조회할 결과 데이터를 Read한다.
9	IF (품질 유형 옵션 선택)	
10	시스템은 SW개발자가 선택한 품질 유형만 결과 보고서를 출력한다.	[1.9~1.11] SW개발자가 선택한 품질 유형에 대해서 Read한 데이터를 품질 결과 보고서 형식으로 변환하여 전달한다.
11	IF(수정 가이드라인 옵션 선택)	
12	시스템은 SW개발자가 선택한 품질 유형만 수정 가이드라인을 추가하여 출력한다.	[1.9~1.11] 수정 가이드 라인 옵션을 클릭한 경우 Read한 데이터를 품질 결과 보고서에 포함하여 변환 후 전달한다.
13	ENDIF	
14	ENDIF	

5.3.4. UC-04 소스코드 품질 평가 Behavior Model

5.3.4.1. Behavior Diagram



[Data Type]

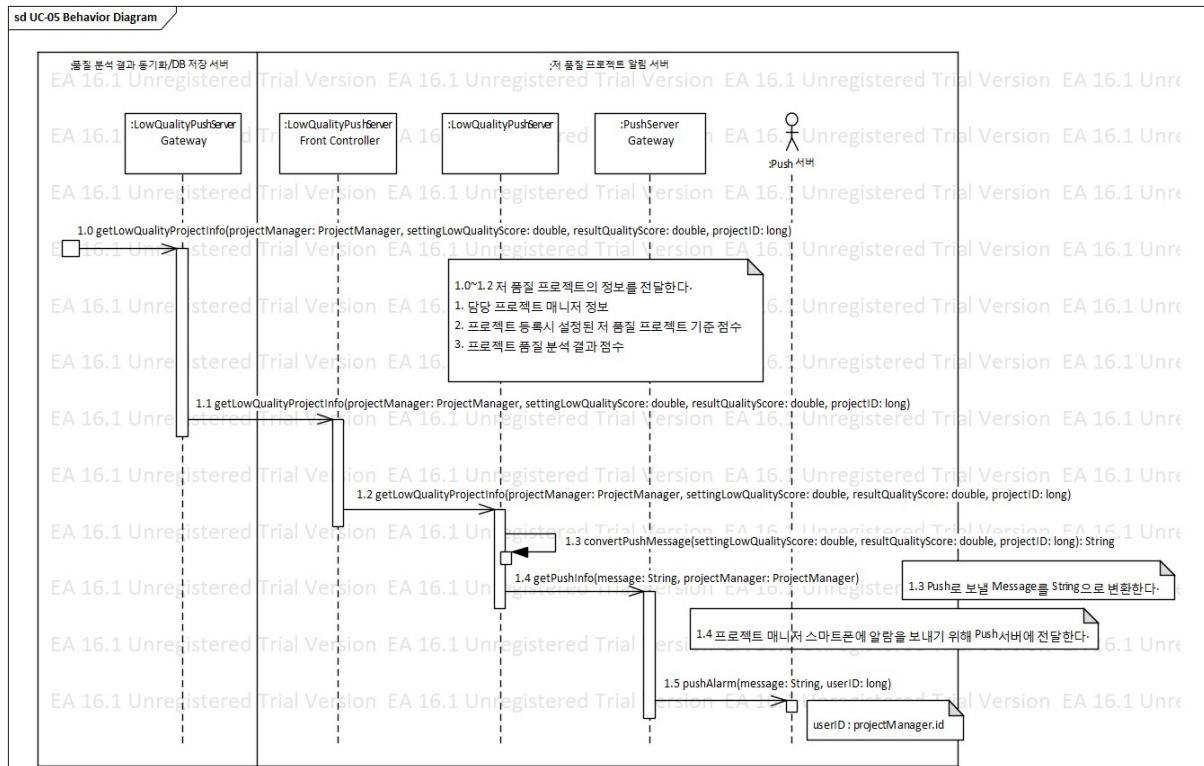
«dataType» Project	«dataType» ProjectManager	«dataType» QualityType
<ul style="list-style-type: none"> - language: String - lowQualityScore: double - projectDeveloper: int - projectID: long - projectManager: ProjectManager - selectQualityType: QualityType - updateReportTime: Time - versionControlSystemAddress: String 	<ul style="list-style-type: none"> - id: long - nickname: String - password: long - phoneNumber: String 	<ul style="list-style-type: none"> - maintenance: boolean - performance: boolean - security: boolean

5.3.4.2. Behavior Description

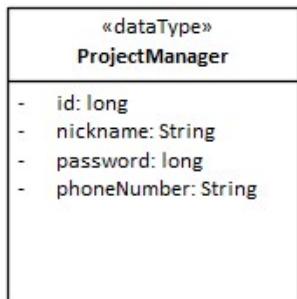
No.	UC Step	Behavior Step
1	소스코드 품질 평가 타이머는 시스템에 Trigger 한다.	[1.0] 3분마다 타이머의 Trigger를 받는다.
2	시스템은 [품질 분석 수행 스케줄링 리스트]을 조회한다.	[1.1 ~ 1.3] DB에서 품질 분석 수행 리스트를 조회한다.
3	DO	
4	시스템은 [소스코드 버전관리 주소] 참조하여 버전관리 시스템(Git, SVN)으로 부터 소스코드를 다운로드 받는다.	[1.7 ~ 1.15] 프로젝트 버전관리 타입에 맞춰서 소스코드를 다운로드 받는다. (DD-01) 품질 평가 수행 성능을 위해 분산하여 1개의 서버는 1개의 프로젝트만 작업할 수 있도록 병렬화를 시킨다. 총 23개의 서버가 존재하며 동시에 23개의 품질 분석이 가능하다.
5	IF (버전관리 주소 접근 실패)	
6	시스템은 “버전관리 접근 실패” 오류 메시지를 로그에 저장한다.	[1.13 ~ 1.14] 버전관리 시스템에서 오류가 발생하는 경우 Error Message를 전달한다.
7	ELSE	
8	시스템은 프로젝트 품질 평가를 수행한다.	[1.15 ~ 1.17] (DD-01) 분산된 품질 평가 준비 서버에서 품질 분석 서버를 한번 더 분산시키게 된다. 100MB 파일마다 품질 분석 서버를 분산시켜 품질 평가가 진행된다. [1.18 ~ 1.19] 프로젝트 정보에서 선택된 품질 유형별로 분석을 수행한다. 수행한 결과 데이터는 JSON형식으로 변환한다. [1.20 ~ 1.21] 분석 결과 데이터를 저장하기 위해서는 분산된 결과 데이터를 동기화 작업이 필요하다. 모든 결과 데이터를 전달 받으면 동기화 작업을 한다.
9	IF (DB 접속 실패)	
10	시스템은 “DB 접속 실패” 에러 메시지를 로그에 저장한다.	[1.24~1.25] DB 오류가 발생하면 Error Message를 전송 후 Log 파일에 저장한다.
11	ELSE	
12	시스템은 프로젝트 품질 평가 결과 데이터를 DB에 저장한다.	[1.21 ~ 1.23] 동기화 작업이 완료된 DB데이터를 저장한다.
13	ENDIF	
14	IF (점수가 [저 품질 프로젝트 기준 점수 입력] 보다 낮은 경우)	
15	시스템은 [UC-05 저 품질 프로젝트 정보(프로젝트 식별자, 프로젝트 결과)] Use Case를 수행 한다.	[1.26] 저 품질 기준 점수보다 낮은 경우 UC-05를 실행시키기 위해 저 품질 프로젝트 정보를 전달한다.
16	ENDIF	
17	ENDIF	
18	WHILE(현재 시간에 수행해야 할 [품질 분석 수행 스케줄링 리스트]의 모든 프로젝트)	(DD-01) 해당 UC Step은 분산 시스템을 사용하기 때문에 대체된다.

5.3.5. UC-05 저 품질 프로젝트 정보 Behavior Model

5.3.5.1. Behavior Diagram



[Data Type]

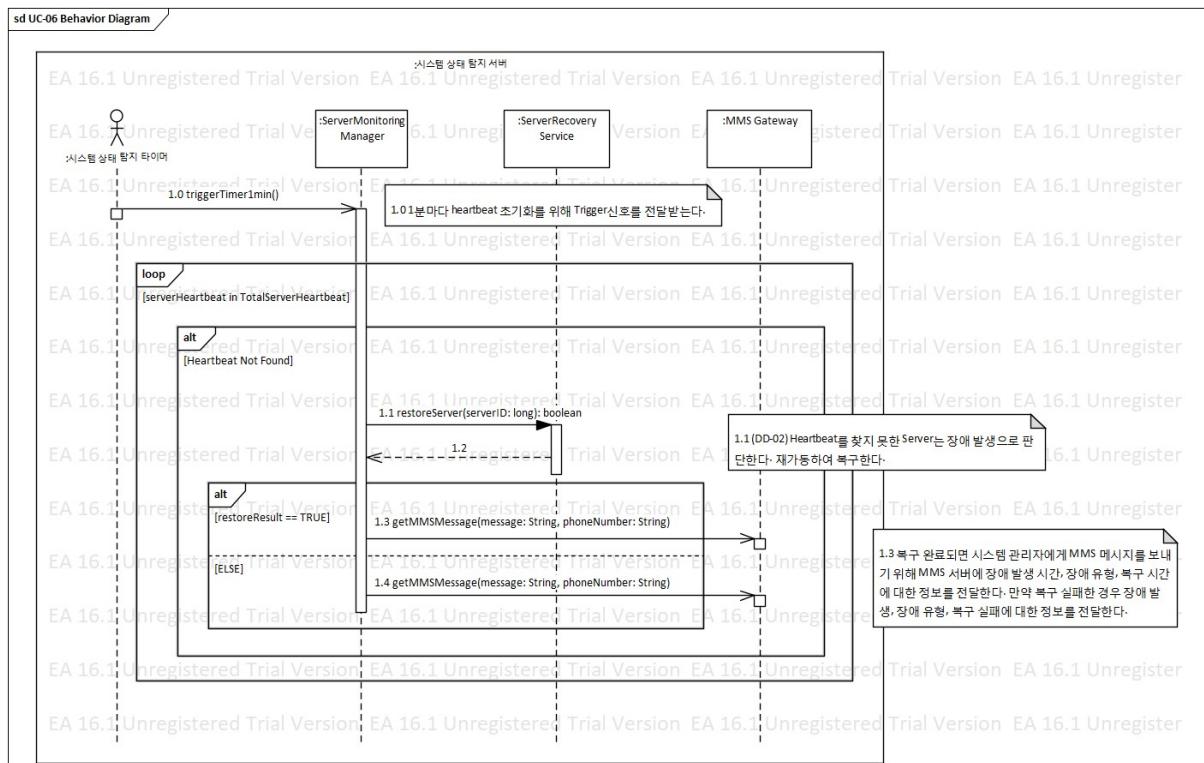


5.3.5.2. Behavior Description

No.	UC Step	Behavior Step
1	시스템은 (프로젝트 식별자, 프로젝트 결과)를 바탕으로 프로젝트 정보를 전달받는다.	[1.0 ~ 1.2] Async/DB서버로부터 저 품질 프로젝트의 정보를 전달받는다
2	시스템은 Push 서버에 저 품질 프로젝트 매니저 정보, 프로젝트 이름, 프로젝트 품질 점수 데이터를 전송한다.	[1.3 ~ 1.4] 프로젝트 매니저 스마트폰에 알람을 보내기 위해 Push서버에 정보를 전송한다.

5.3.6. UC-06 시스템 상태 탐지 및 복구 Use Case Behavior Model

5.3.6.1. Behavior Diagram



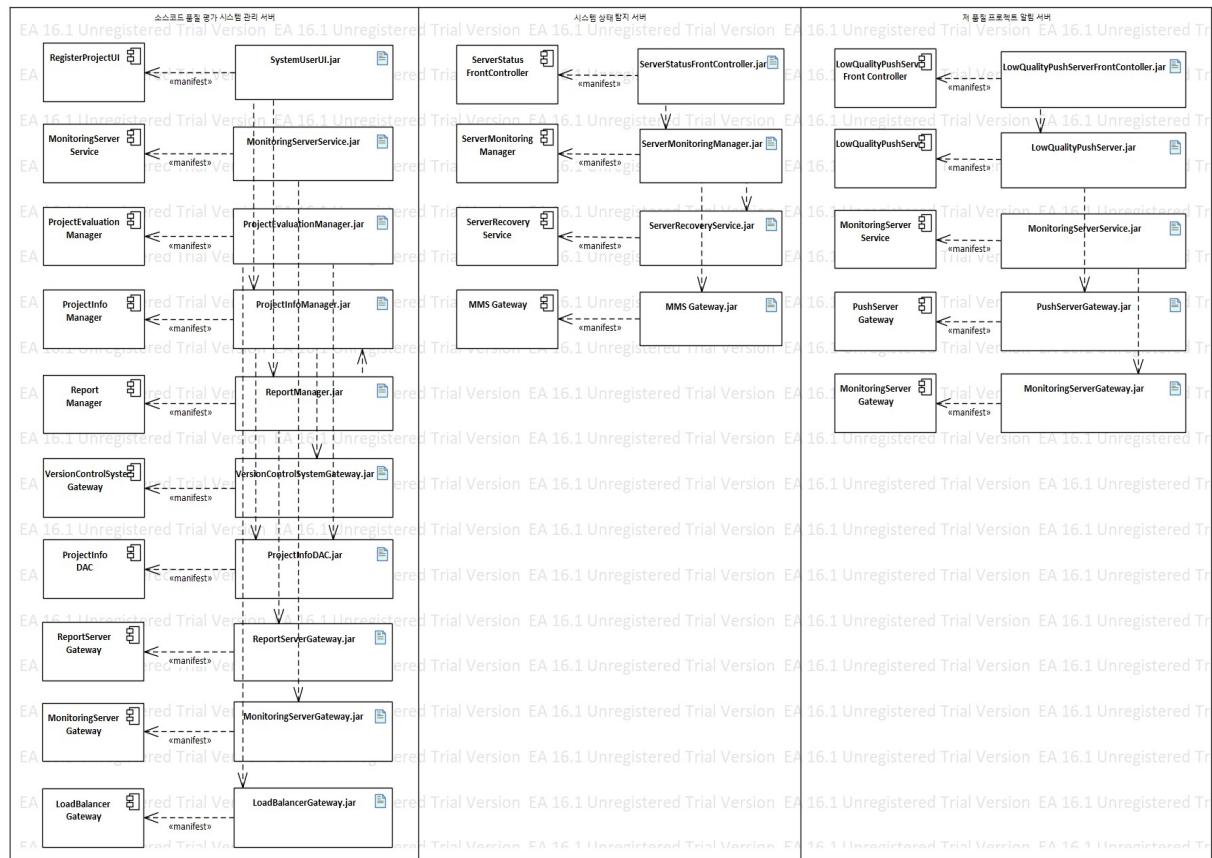
5.3.6.2. Behavior Description

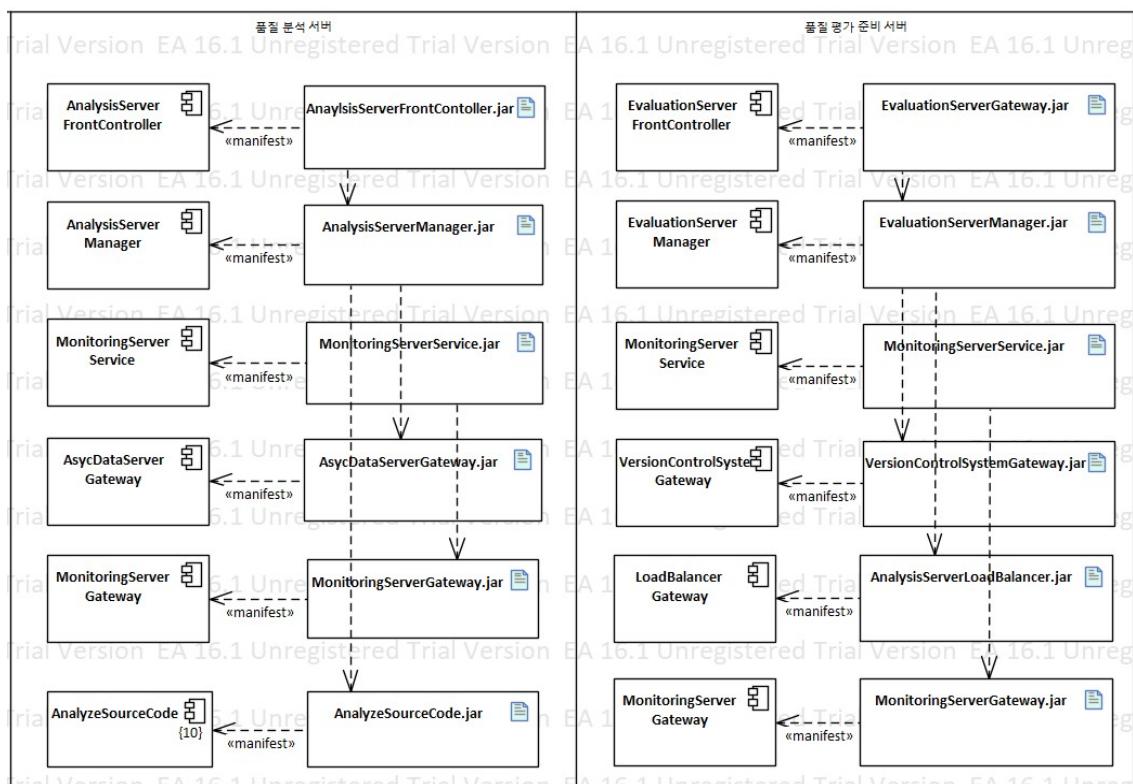
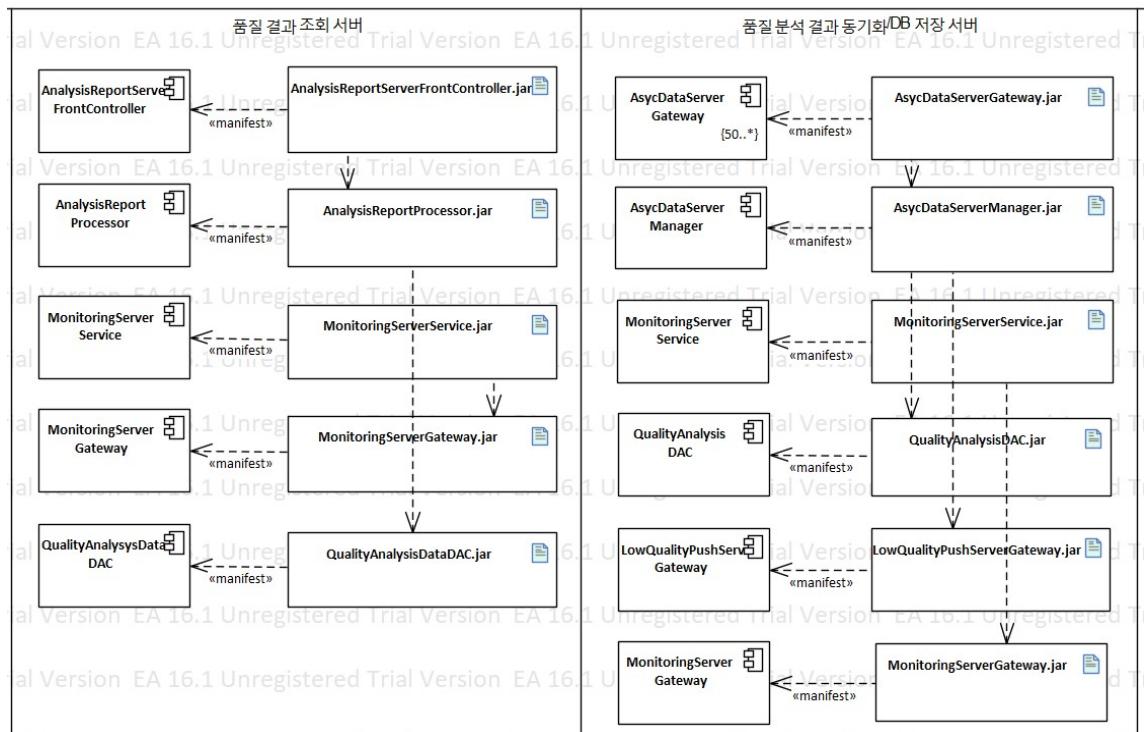
No.	UC Step	Behavior Step
1	시스템 상태 탐지 타이머는 시스템에 Trigger 한다.	[1.0] 시스템 상태 탐지 타이머는 1분마다 Trigger를 전달한다.
2	DO	
3	시스템은 서버에 상태 정보를 조회한다.	(DD-03) Heartbeat 신호를 검사한다.
4	IF(시스템 장애 발생 시)	
5	시스템은 장애 서버를 재 가동하여 복구한다.	[1.1~1.2] heartbeat신호가 안온 서버는 재 가동하여 복구한다.
6	WHILE(복구 시간)	
7	시스템은 장애 서버 상태 정보를 조회한다.	
8	IF (복구 실패)	
9	시스템은 MMS서버에 장애 발생 시간, 유형, 복구 실패 정보를 전송한다.	[1.4] 복구 실패한 경우에는 시스템 관리자에게 장애 발생 시간, 장애 유형, 복구 실패 메시지를 전달하기 위해 MMS 서버에 전달한다.
10	ENDIF	
11	ELSE	
12	시스템은 시스템 상태 페이지에 서버 상태를 업데이트 한다.	[1.3] 복구 성공한 경우에는 시스템 관리자에게 장애 발생 시간, 장애 유형, 복구 시간 메시지를 전달하기 위해 MMS 서버에 전달한다.
13	ENDIF	
14	WHILE (시스템 모든 서버)	1분마다 시스템 상태 탐지 서버를 제외하고 Heartbeat 신호를 검사한다.

5.4. Deployment View

5.4.1. Artifact Definition Model

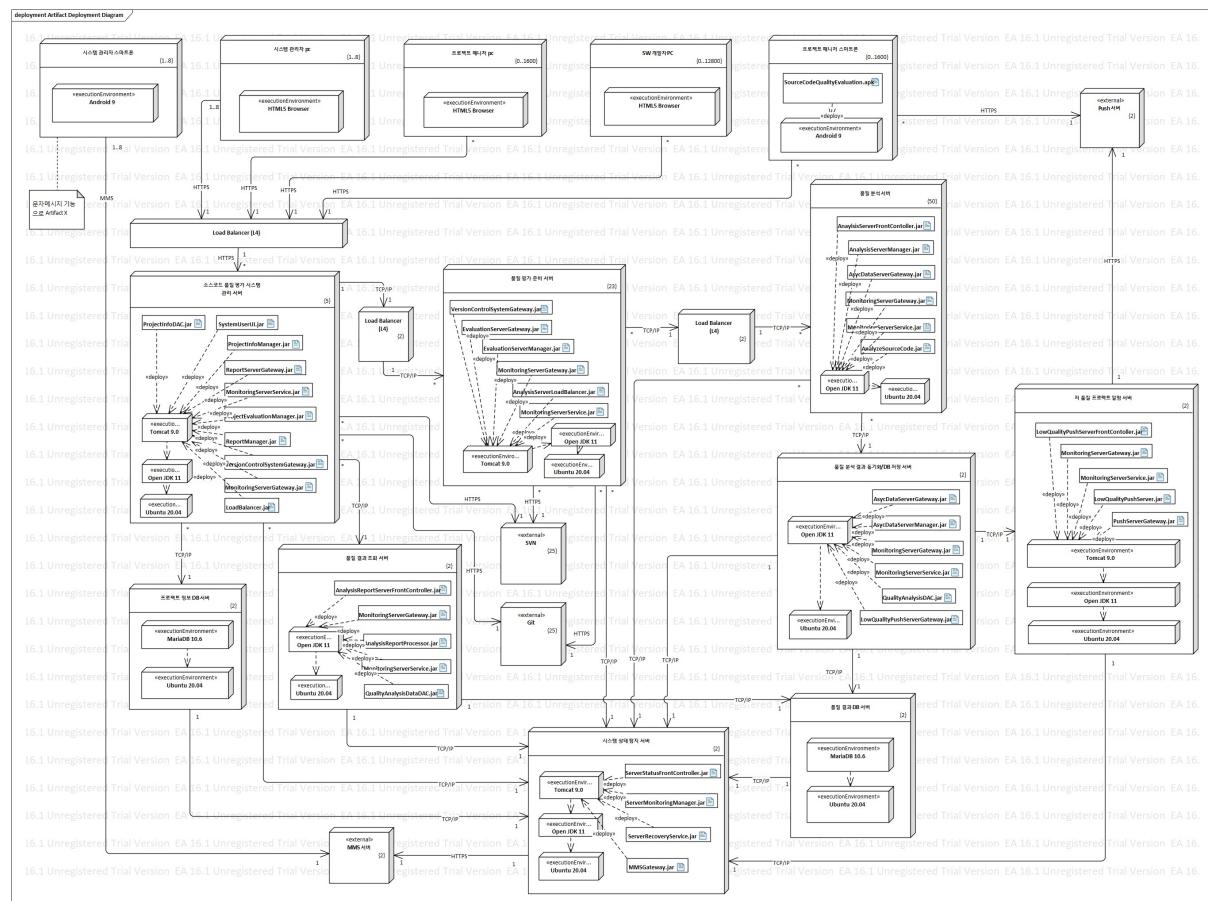
5.4.1.1. Artifact Definition Diagram





5.4.2. Artifact Deployment Model

5.4.2.1. Artifact Deployment Diagram



5.5. Documenting Design Decisions

5.5.1. Design Decision List

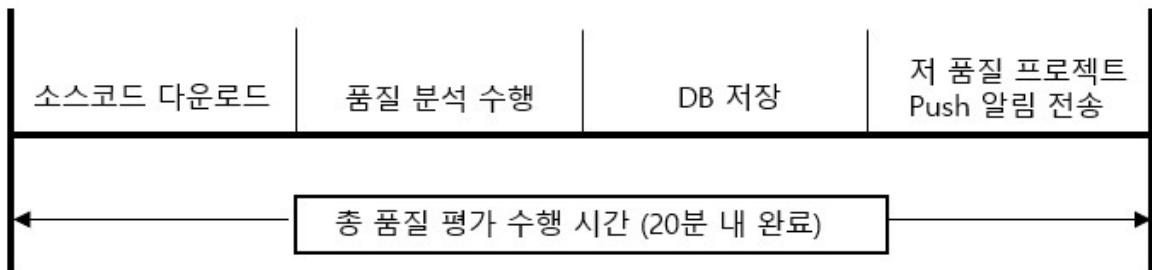
ID	Title	목표 QAs
DD-01	성능과 가용성을 위한 분석 서버 분리	QA-01(품질 평가 수행 시간) QA-02(시스템 장애 탐지 및 복구)
DD-02	시스템 장애 탐지를 위한 Heartbeat 적용	QA-02(시스템 장애 탐지 및 복구)
DD-03	새로운 품질 유형 지원을 위한 단일 Component 설계	QA-03(새로운 품질 유형 지원)
DD-04	새로운 버전관리 시스템 지원을 위한 통합 Gateway 활용	QA-04(새로운 버전관리 시스템 지원)

5.5.2. DD-01 품질 평가 성능을 위한 분석 서버 분리

5.5.2.1. Design Goal

[관련 QA] QA-01

- ▶ 3분마다 소스코드 품질 평가 타이머가 Trigger시 소스코드 품질 평가 시스템은 품질 평가를 총 20분안에 완료되어야 한다.



<성능>

본 시스템은 매일 1600개 프로젝트의 품질 평가하고, 프로젝트 매니저와 SW 개발자에게 결과 보고서를 제공해야 한다. 그러나 모든 프로젝트를 분석하는 것은 많은 시간이 소요된다. 시간 소요를 증가시키는 원인은 복잡한 로직과 계산식이 많기 때문이다. 서버의 성능을 최신 사양으로 업그레이드하여 분석 속도를 높일 수 있지만, 설계적인 측면에서도 문제를 해결해야 한다.

[관련 QA] QA-02

- ▶ 서버에 장애가 발생 시 1분안에 탐지하고 장애 서버를 재가동하여 3분안에 복구 완료한다. 또한 시스템 관리자에게 1분안에 문자메시지로 장애발생시간, 장애유형, 복구시간을 통보한다. (총 5분)

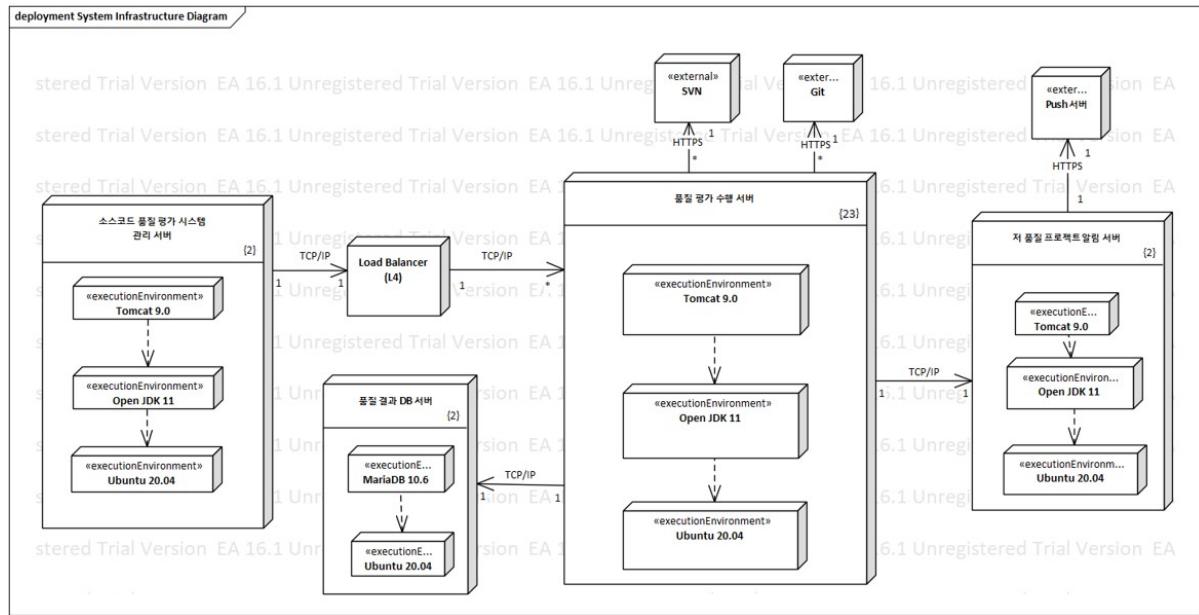
<가용성>

서버의 가용성은 매우 중요하다. 만약 품질 분석 서버가 1개라도 장애가 발생한다면 하루에 모든 프로젝트를 평가하는 것이 불가능할 수 있다. 따라서 품질 분석 서버가 99.9%의 가용성을 유지할 수 있도록 설계해야 한다.

5.5.2.2. Design Approach List

5.5.2.2.1. Design Approach #1 Description: 통합 품질 분석 서버

각 프로젝트는 1개의 품질 분석서버에 할당 받아 분석을 시작한다. 1개의 품질 분석서버로 품질 분석을 진행하면 불필요한 데이터 전송이 발생하지 않는다. 그만큼 Overhead가 발생하지 않고 품질 분석 서버의 성능을 100% 사용할 수 있는 장점을 가지고 있다. 통합 품질 분석 서버의 구조는 아래와 같다.



해당 Approach를 설명하기 위해선 품질 평가의 Flow에 대해 언급이 필요하다.

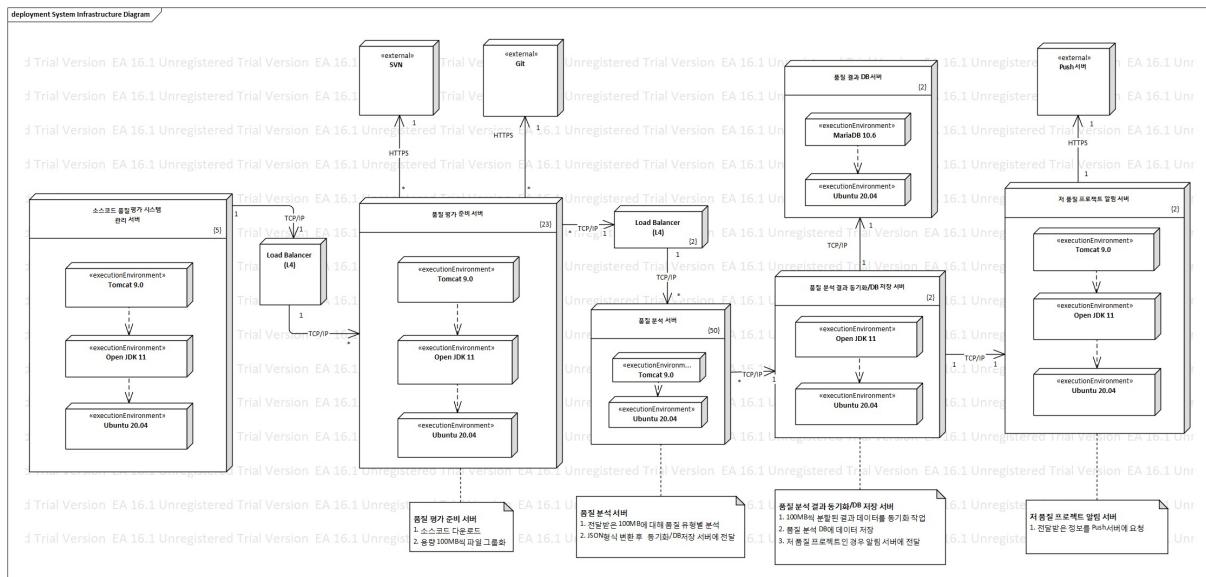
1. 버전관리 소스코드 다운로드
2. 소스코드 품질 분석 수행
3. 품질 결과 데이터 DB서버 저장
4. 저 품질 프로젝트인 경우 담당 프로젝트 매니저 스마트폰에 Push 알림

통합 품질 분석 서버로 설계가 되어진다면 품질 평가 수행 서버에서 위 4가지 과정을 수행한다. 이 설계는 두 가지 이점을 가진다. 첫째, 시스템 상태를 감지할 때 품질 평가 준비 서버(총 23대)만 감지하면 된다. 둘째, 시스템 구현 및 유지 보수가 용이해지며 시스템의 복잡도도 감소한다. 그러나 이 설계는 단점도 가지고 있다. 품질 평가 준비 서버에 장애가 발생하면 1대의 서버를 사용할 수 없으므로 가용성 측면에서 단점이 존재한다. 또한, 소스 코드가 매우 복잡하거나 (QA-03) 새로운 품질 유형이 추가되는 경우 품질 분석 시간이 증가하여 20분 이내에 분석이 완료되지 않을 수 있다. 분석이 20분 안에 완료되지 않는 경우가 빈번하게 발생하면 프로젝트 매니저와 SW 개발자는 최신 결과 보고서를 조회할 수 없게 된다.

정리하면, 시스템의 복잡도가 낮아져 시스템 구현 및 유지보수가 편리해지지만 추가용이성과 가용성 측면에서 위험이 존재한다.

5.5.2.2. Design Approach #2 Description: 품질 분석 서버 분리

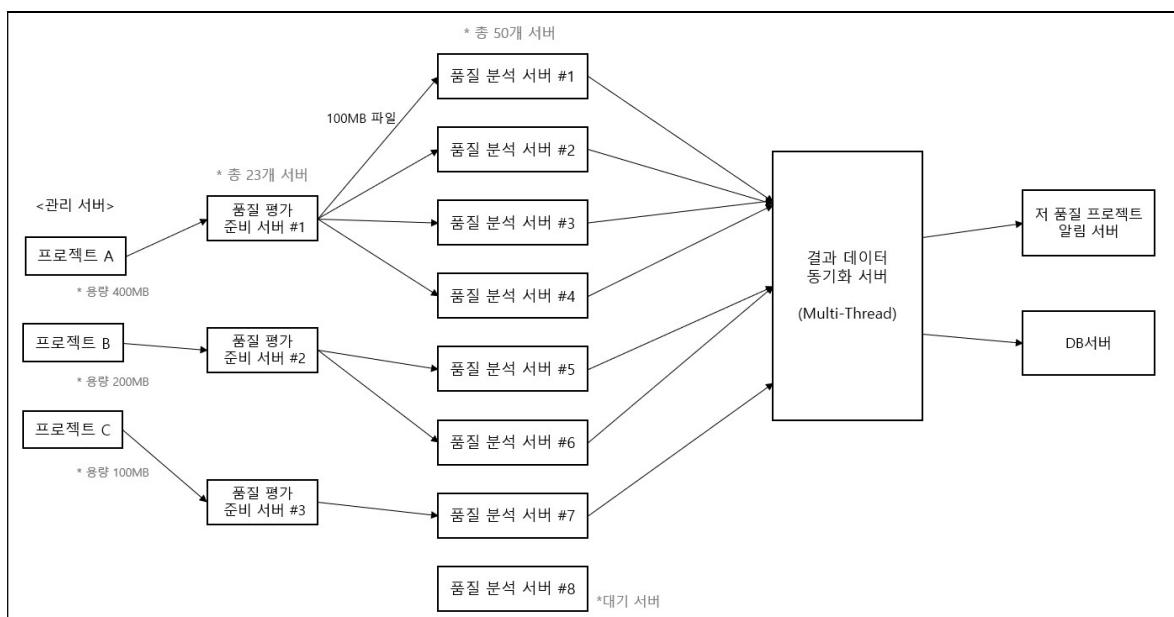
1개의 프로젝트는 여러 개의 소스코드 파일을 분리하여 분산 분석을 수행한다. 소스코드 파일을 분석 서버에 할당하는 과정 속에서 Overhead가 증가할 것으로 보인다. 하지만 프로젝트의 용량이 큰 경우에는 여러 분석 서버에 요청이 가능하여 성능은 대폭 개선될 것이라 예상된다. 전체 프로젝트의 분석 평균 소요 시간은 증가될 것으로 보인다. 그렇지만 최대 분석 소요 시간은 대폭 줄어든다. Maintain multiple copies of computations의 Tactic을 활용하여 성능을 개선하게 된다.



소스코드 품질 평가 시스템 관리 서버 (타이머 발생시) → 품질 평가 준비 서버 → 품질 분석 서버 → 결과 데이터 동기화 → DB서버 저장 → 저 품질 프로젝트 알림 순으로 진행하게 된다.

품질 평가의 4가지 과정을 분리하여 품질 분석을 진행하게 된다.

- (1) 품질 평가 준비 서버는 버전관리 시스템으로부터 소스코드를 다운받고 100MB씩 파일을 그룹화 시킨다.
- (2) 품질 분석 서버는 전달받은 그룹화된 파일들을 품질 유형에 대해 분석을 한다. 분석이 끝나면 결과 데이터를 JSON형식으로 변환하여 동기화/DB서버에 전달한다.
- (3) 품질 분석 결과 동기화/DB 저장 서버는 분산된 결과 데이터를 동기화 시킨다. 동기화가 된 결과 데이터는 DB에 저장한다. 또한 분석 점수가 저 품질 프로젝트 기준 점수보다 낮은 경우 프로젝트 매니저 스마트폰에게 알림을 보내기 위해 저 품질 프로젝트 알림 서버에 정보를 전달한다.
- (4) 전달받은 저 품질 프로젝트 정보들을 Push서버에 요청한다.



위 과정처럼 품질 평가가 진행하게 된다. 서버를 분리하게 되면 트레이드 오프로 단점과 장점이 존재한다. 단점은 서버의 개수가 상당히 늘어나게 되어 시스템 상태 탐지 서버는 탐지해야 할 서버 수가 늘어난다. 또한 시스템의 복잡도가 높아져 구현하기 어려울 가능성이 존재한다. 통신과 처리 과정이 많아져 Overhead가 발생하게 된다. 하지만 Overhead는 (QA-01)을 달성하기 위해 문제는 없다고 판단된다. 그 이유는 아래와 같다.

최악의 경우인 프로젝트 용량 1GB/품질 분석 시간 20분 = 1200초

품질 분석 서버 분리 적용

1. 위 프로젝트 용량 1GB를 100MB씩 그룹화
2. 100MB씩 그룹화된 파일을 10개의 품질 분석 서버에 전달
3. 1개의 품질 분석 서버는 10개의 Process를 가지고 있어 1개의 Process는 10MB씩 품질 분석 수행
4. 1개의 Process당 10MB 품질 분석 소요시간은 $1200초/100 = 12초 + \alpha$ 예상

품질 분석 서버 분리의 Design Approach를 적용하면 Overhead가 초래해도 품질 분석 소요 시간은 대폭 감소할 것이라 판단된다.

장점은 품질 평가가 20분 이내 모두 수행될 가능성이 매우 높아지며 분석 서버의 분산으로 더 빠른 분석이 가능하다. 그 이유는 계산량이 많고 용량이 큰 프로젝트를 분할하여 분석하기 때문에 성능이 대폭 개선될 것이라 예상된다. 평균 분석 소요시간은 증가하지만 최대 분석 시간은 대폭 감소하게 된다. 따라서 (QA-01) 품질 평가 수행 시간을 보장할 수 있다. 또한 (QA-03) 새로운 품질 유형이 추가되는 경우 분석 시간이 증가하지만 분산하여 분석하기 때문에 20분 이내에 모두 수행이 가능할 것이라 예상된다. 장애가 발생하는 경우에도 분산되어있기 때문에 가용성 측면에도 도움이 된다. 즉 성능, 가용성, 추가용이성이 높아지는 이점을 얻을 수 있다.

5.5.2.3. Decision and Rationale

품질 평가 분석 소요시간이 많은 경우 시간을 대폭 개선할 수 있는 DA#2를 선택한다. 모든 프로젝트에 대한 품질 평가 분석 평균 소요 시간과 Overhead가 증가되는 단점이 있다. 그렇지만 품질 분석 시간이 20분 내로 진행되어야 하기 때문에 Overhead의 영향은 성능에 미약하다고 판단된다. 만약 시스템 규모가 작다면 DA#1를 선택하겠지만, 성장 가능성은 고려하였을 때 시장 점유율이 늘어나거나 (QA-03) 품질 유형이 추가되는 경우 성능을 보장하기에 DA#2가 적합한 방법이라 판단된다.

Quality Attribute		Analysis	DA #1: 통합 품질 분석 서버	DA #2: 품질 분석 서버 분리 (Selected)
ID	Title			
QA-01	품질 평가 수행 시간	Pros	<ul style="list-style-type: none"> (+) Overhead가 없어 서버의 성능을 100% 사용 가능하다. (+) 품질 분석에 대한 모든 과정이 한 개의 서버에서 이루어지므로 시스템 복잡도가 낮아진다. (InfraStructure 관점) 	<ul style="list-style-type: none"> (++) 1개의 프로젝트가 여러 개의 품질 분석 서버를 사용되어 소요 시간이 대폭 개선된다.

		Cons	<p>(-) 서버 장애가 발생하면 시스템 전체 성능이 낮아진다. (품질 분석 서버 23개 모두 활성화된 상태에서 1개가 장애가 발생하는 경우 장애가 발생한 서버의 프로젝트는 대기상태로 전환되면 전체 시스템 성능 저하를 초래한다.)</p>	<p>(-) 분석 서버를 분리시켜 데이터 전달, 서버 할당하는 Overhead가 발생한다. (분석 시간이 20분 내로 되어야하기 때문에 QA에 영향이 미약하다고 판단)</p> <p>(-) 품질 분석 서버를 분산하기 위해 시스템 구현 난이도와 시스템 복잡도가 높아진다.</p>
QA-02	<u>시스템 장애 탐지 및 복구</u>	Pros	<p>(+) 시스템 복잡도가 낮아 구현/유지보수가 쉽다.</p>	<p>(+) 특정 서버에 장애 발생 시 다른 서버의 동작에 영향이 없다.</p> <p>(+) 분산 시스템으로 장애 서버가 발생 시 대기 서버로 전환되기 때문에 가용성이 높아진다.</p>
		Cons	<p>(-) 장애 발생시 전체 기능 멈추어 가용성이 낮아진다.</p>	<p>(-) 시스템 탐지할 서버 개수가 증가한다</p>
QA-03	<u>새로운 품질 유형 지원</u>	Pros		<p>(+) 품질 유형이 추가되는 경우 그룹화되는 파일 용량을 낮춰 (ex 100MB > 50MB) 성능을 보장할 수 있다. 즉 품질 유형에 대한 추가용이성이 용이하다.</p>
		Cons	<p>(-) 품질 유형이 추가되는 경우 처리양이 많아져 성능을 보장하기 위해 Resource를 늘리거나 최적화로 성능을 높여야 한다. 따라서 추가용이성이 좋지 않다.</p>	

5.5.3. DD-02 시스템 장애 탐지를 위한 Heartbeat 적용

5.5.3.1. Design Goal

[관련 QA] QA-02

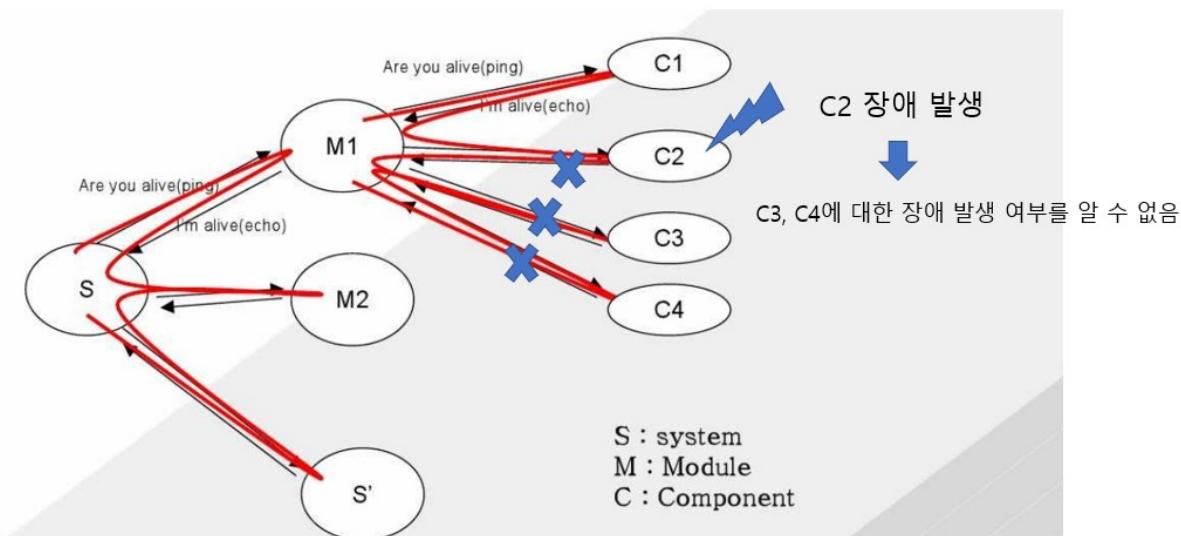
- 서버에 장애가 발생 시 1분안에 탐지하고 장애 서버를 재가동하여 3분안에 복구 완료한다.
또한 시스템 관리자에게 1분안에 문자메시지로 장애발생시간, 장애유형, 복구시간을 통보한다. (총 5분)

본 시스템은 99.9%의 가용성을 요구하고 있다. 1분 이내 탐지가 안되는 경우 99.9%의 가용성을 달성하기 어렵다. 따라서 장애 발생 시 1분 이내 탐지, 3분 이내 복구를 목표로 한다. 이를 달성하기 위해선 1분마다 효율적으로 모든 서버의 동작 상태를 감지해야 한다. 모든 서버를 탐지하고 복구를 지원할 수 있는 Tactic이 필요하다. 또한 장애가 발생한 서버만 재가동하여 복구될 수 있도록 설계가 되어야 한다.

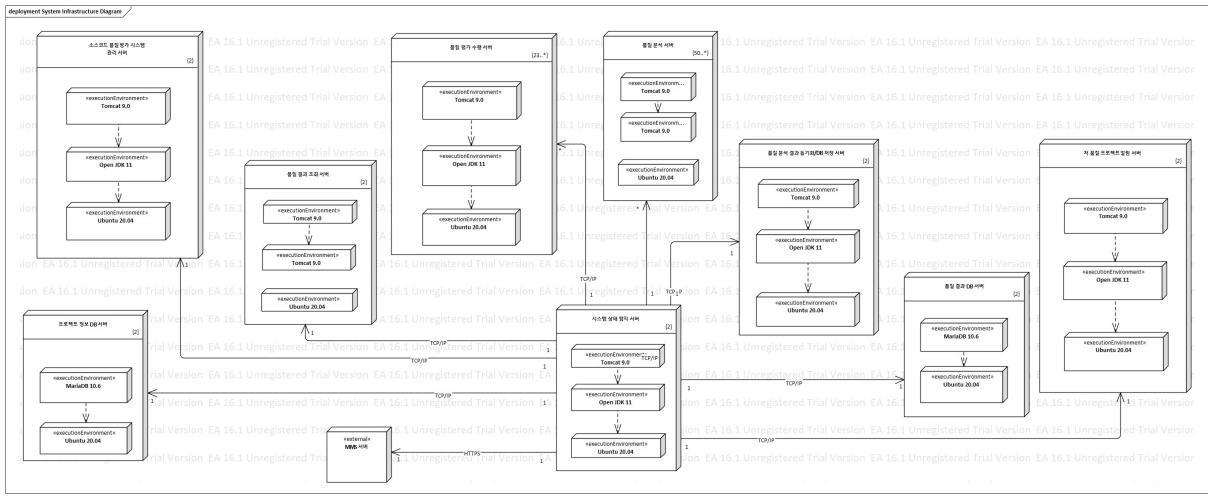
5.5.3.2. Design Approach List

5.5.3.2.1. Design Approach #1 Description: Ping/Echo

Ping/Echo은 모니터 컴포넌트가 모니터할 컴포넌트에 주기적으로 신호를 전송하여 확인하는 방법이다. 응답이 없으면 장애로 탐지하게 된다. 본 시스템은 1분마다 시스템을 탐지하기 때문에 메인 모니터 시스템은 1분의 주기로 Ping을 전송한다.



System은 Module에게 Ping을 전송하여 장애판단을 하고 Module은 Component에게 Ping을 전송하여 장애판단을 하게 된다. 장애 발생하게 되면 장애가 발생한 컴포넌트 이후에 Ping을 받아야하는 컴포넌트는 장애 여부를 알 수가 없다. 즉 1개의 서버만 장애가 발생한 건지, 여러 개의 서버가 장애 발생 여부에 대한 정확성이 떨어진다. 장애가 발생하면 3분안에 복구가 되어야 하는데 모든 서버를 동시에 탐지하지 못하게 된다면 동시에 장애 복구가 수행되지 않고 순차적으로 장애 복구가 수행될 것이다. 구현에 있어서 단순하겠지만 가용성이 떨어지게 된다.

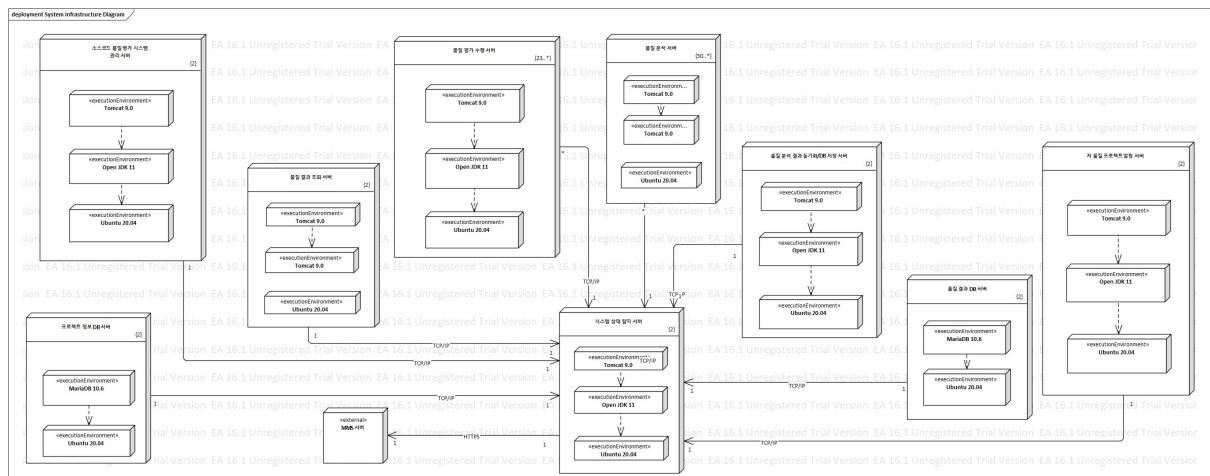


위와 같이 본 시스템은 시스템 상태 탐지 서버로부터 각 서버에 Ping을 보내어 장애 여부를 탐지한다.

5.5.3.2.2. Design Approach #2 Description: Heartbeat

Heartbeat는 모니터할 컴포넌트가 모니터 컴포넌트에게 신호를 보내는 방법이다. 모니터할 컴포넌트는 일정 기간 동안 heartbeat 시그널을 받지 못하면 장애 발생으로 간주한다. 따라서 모든 모니터 할 컴포넌트는 heartbeat 신호를 전송할 수 있도록 구현되어야 한다. 또한, 각 컴포넌트는 일정한 주기로 heartbeat 신호를 전송해야 한다. 시스템은 1분 이내에 장애를 탐지해야 하므로, 각 컴포넌트는 1분마다 heartbeat 시그널을 전송해야 한다. 그러나 통신으로 인한 Overhead가 발생할 수 있으므로, 10초의 buffer 시간을 추가해야 한다. 따라서, 모니터 컴포넌트는 50초 이내에 모든 heartbeat 시그널을 받아야 한다.

본 시스템은 최소 96개의 서버를 가지고 있다. 모니터 컴포넌트는 1분 동안 96개의 신호를 받을 수 있는 성능이 필요하다. 각 신호를 보낼 컴포넌트는 1분마다 heartbeat를 실행하여 heartbeat 전송 기능이 부하를 유발하지 않는 것으로 판단된다. 1초마다 heartbeat 신호를 보내는 것이 아니라 1분마다 보내기 때문에 모니터할 컴포넌트에서 부하가 예상되지 않는다.



각 서버는 시스템 상태 탐지 서버에게 Heartbeat 신호를 보내는 구조이다.

5.5.3.3. Decision and Rationale

DA #1은 매우 큰 단점이 존재한다. 장애가 여러 개 동시에 발생한 경우를 가정하면 Ping/Echo는 최초 장애 발생 서버만 탐지하고 나서 서버를 복구한다. 최초 장애 발생 서버가 복구 완료되면 탐지하지 못한 서버는 다시 장애 탐지가 될 것이고 또한 복구가 시작이 될 것이다. 서버는 독립적으로 구성이 되어있지만 복구 작업은 Chain형식으로 진행된다. DA #2는 시스템 탐지 및 복구 관련 복잡도가 높을 것이라 예상되지만 1분마다 모든 장애 서버를 복구할 수 있어 가용성 측면에서 DA#2가 더 적합하다.

Quality Attribute		Analysis	DA #1: Ping/Echo	DA #2: Heartbeat (Selected)
ID	Title			
<u>QA-02</u>	<u>시스템 장애 탐지 및 복구</u>	Pros	(+) 시스템 구현 복잡도가 낮음	(++) 서버 단위로 장애를 탐지하고 동시에 복구 진행 가능하여 가용성이 높아짐.
		Cons	(--) 장애가 여러 개가 동시에 발생하는 경우 최초 장애 탐지한 서버만 복구 진행. 남은 장애 서버는 순차적으로 복구되어 가용성 감소.	(-) 시스템 탐지 및 복구 관련 한 시스템 복잡도 증가

5.5.4. DD-03 새로운 품질 유형지원을 위한 단일 Component 설계

5.5.4.1. Design Goal

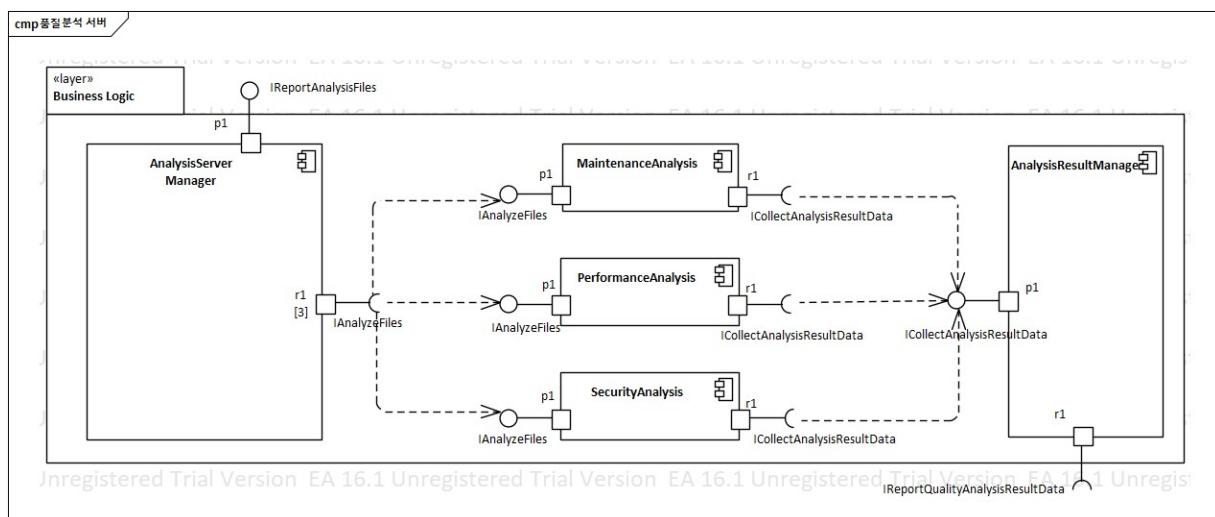
[관련 QA] QA-03

- 새로운 품질 유형 추가가 필요한 경우 추가할 새로운 품질 유형에 대한 결과 조회가 빠른 시간(1MM)안에 완료되어야 한다.

본 시스템은 유지보수성, 성능, 보안 3가지 품질 평가 유형으로 평가할 수 있다. 새로운 품질 유형에 대한 추가 지원 요청이 있을 경우 추가 지원을 해야 한다. 만약 유지보수성, 성능, 보안 3가지만 고려하여 설계한 경우, 새로운 품질 유형이 추가될 때 기존의 품질 유형의 코드도 변경이 될 수 있다. 이로 인해 Shotgun Surgery의 코드 스멜이 발생한다. 따라서 새로운 품질 유형을 추가할 때 기존의 품질 유형에 대한 기능은 변경되지 않도록 설계되어야 한다.

5.5.4.2. Design Approach List

5.5.4.2.1. Design Approach #1 Description: 품질 유형별 분리 Component

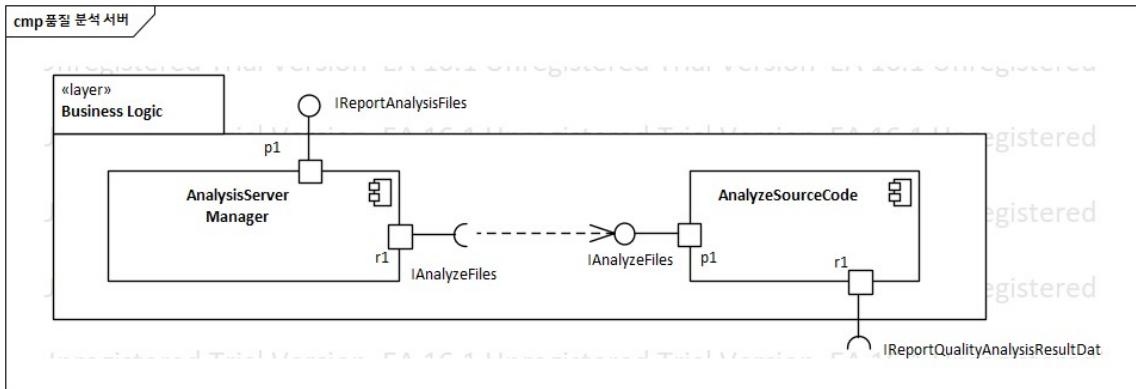


유지보수성, 성능, 보안 각각 Component를 만들어 품질 분석하는 설계이다. 새로운 품질 유형이 추가되는 경우 Component를 만들어 새로운 품질 유형에 대한 분석을 지원한다. AnalysisServerManager Component에서 품질 유형별 Component로 소스코드 파일들을 전달하는 Overhead가 발생한다. 소스코드 파일의 용량이 그룹화된 100MB이기 때문에 Overhead의 영향이 클 것으로 예상된다.

해당 Overhead를 줄이기 위해서 Tactic인 Reduce Overhead를 적용하면 소스코드 파일들을 로컬 저장소에 저장한다. 그리고 품질 유형 Component는 로컬 저장소에 저장되어 있는 파일들을 접근하는 방식으로하면 Overhead는 감소한다. 하지만 분석 결과 데이터인 String은 AnalysisResultManager Component에 전달할 때 Overhead가 발생한다. 따라서 품질 유형별 분리 Component로 설계하면 Overhead는 발생하게 된다. 각 품질 유형별 분석 시간이 다르기 때문에 AnalysisResultManager Component는 모든 분석이 될 때까지 대기해야 하는 단점이 존재한다.

새로운 품질 유형이 추가되는 경우 AnalysisServerManager와 AnalysisResultManager는 1개의 인터페이스를 사용하기 때문에 영향이 적다. 시스템 복잡도는 높아지지만 추가 용이성은 용이해진다.

5.5.4.2.2. Design Approach #3 Description: 품질 유형 통합 Component



유지보수성, 성능, 보안에 대한 Component를 통합하여 분석하는 설계이다. 3가지 품질 유형에 대한 분석을 모두 수행하고 동기화 서버에 결과 데이터를 전달한다. 새로운 품질 유형이 추가되는 경우에는 통합 Component안에 추가된다. 통합 Component에 추가하면 Component Level Design에서 디자인 패턴을 적용하여 OCP를 만족할 수 있도록 설계되어야 한다. AnalyzeSourceCode Component에서 모든 품질 분석이 진행되기 때문에 Overhead는 발생하지 않는다.

5.5.4.3. Decision and Rationale

품질 유형 통합 Component인 DA #2를 선택한다. 그 이유는 분리 Component (DA #1)로 설계하면 Component간 통신의 Overhead가 발생하게 된다. 새로운 품질 유형이 추가될 때마다 Overhead는 증가된다. 반대로 통합 Component는 Overhead가 발생하지 않는다. 즉 새로운 품질 유형이 추가되어도 분석에 대한 시간만 증가되며 Overhead는 발생하지 않는다. 새로운 품질 유형은 Component Level Design 단계에서 디자인 패턴을 활용하여 OCP를 만족할 수 있는 설계를 구성하면 된다.

Quality Attribute		Analysis	DA #1: 품질 유형별 분리 Component	DA #2: 품질 유형 통합 Component (Selected)
ID	Title			
QA-03	<u>새로운 품질 유형 지원</u>	Pros	(+) 분리 Component로 되어있기 때문에 추가할 때 Component만 추가하면 되어 추가용이성이 높다.	(+) 한 개의 Component에서 모든 품질 유형 분석이 수행되기 때문에 파일 전송과 같은 Overhead가 없다.
		Cons	(-) Component에 소스코드 파일을 전달하는 Overhead가 발생한다. (로컬 저장소에 저장된 파일을 접근하면 Overhead가 해결됨) (-) 분석 결과 데이터를 취합하는 Component에 전달하는 Overhead가 발생한	- <small>* 단일 Component로 구성하여 노드간 통신 Overhead X</small>

		<p>다.</p> <p>(-) 품질 유형마다 분석 시 간이 상이하여 모든 품질 유형 분석이 완료될 때까지 취합하는 Component는 대 기상태에 빠진다.</p>	
--	--	--	--

5.5.5. DD-04 새로운 버전관리 시스템 지원을 위한 통합 Gateway 활용

5.5.5.1. Design Goal

[관련 QA] QA-04

- 새로운 버전관리 시스템 추가가 필요한 경우 추가할 새로운 버전관리 시스템과 본 시스템의 연동이 빠른 시간(1MM)안에 완료되어야 한다.

본 시스템이 지원하고 있는 버전 관리 시스템은 Git과 SVN이다. 현재 버전 관리 시스템은 표준화가 안되어 있어 새로운 버전 관리 시스템이 추가되는 경우 빠른 대응할 수 있도록 설계 구조가 필요하다. 본 시스템에서 버전관리 시스템은 총 2가지 기능만 필요하다.

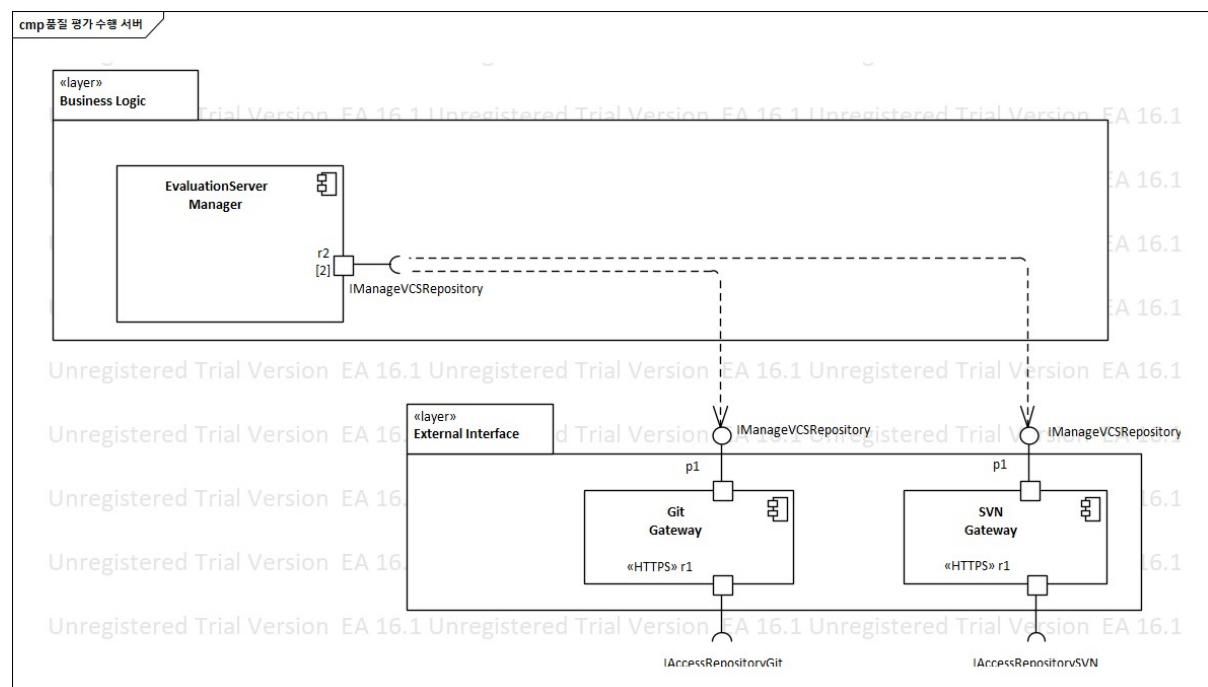
1. 소스코드 다운로드
2. 버전관리 주소 유효성 확인

문제 발생 원인: 버전 관리 시스템의 표준화가 되어 있지 않아 새로운 버전관리 시스템이 추가되면 시스템에서 작동할 수 있도록 수정이 필요함.

5.5.5.2. Design Approach List

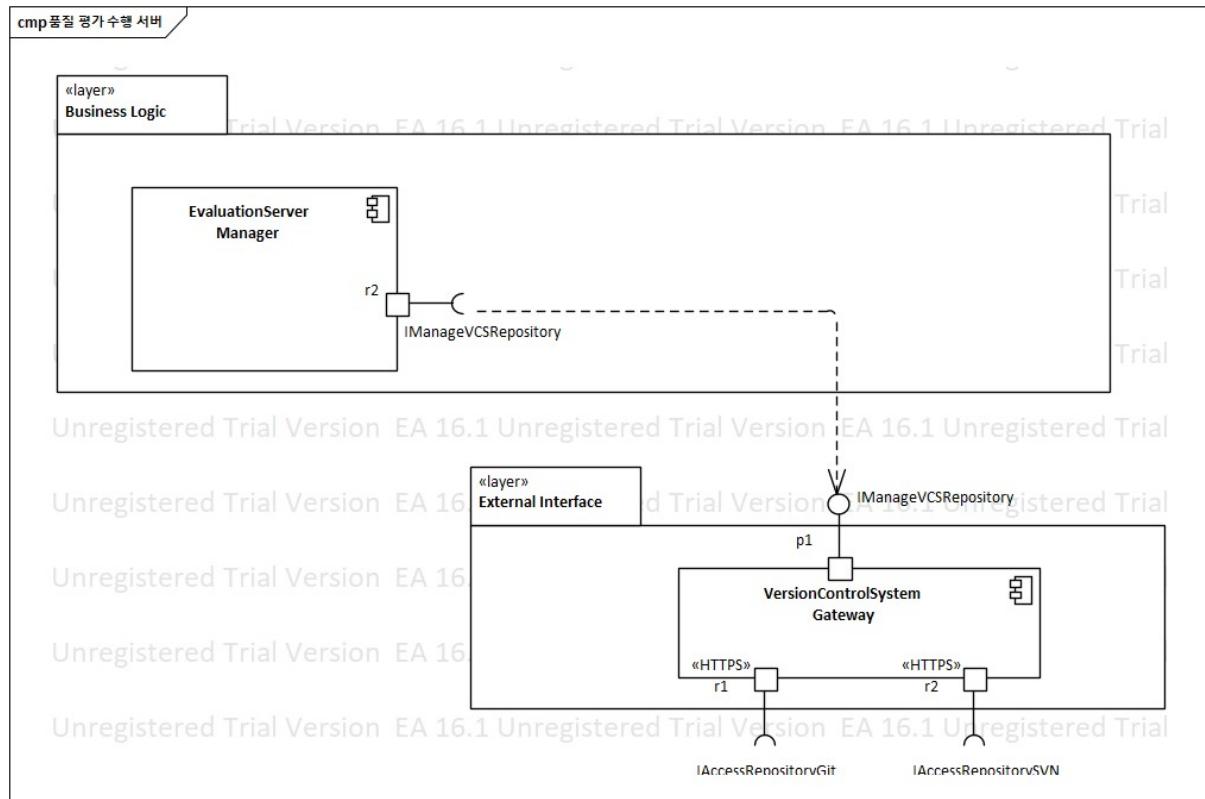
5.5.5.2.1. Design Approach #1 Description: 시스템 별 Gateway 생성

새로운 버전관리 시스템별로 Gateway를 생성하는 방식이다. 새로운 버전 관리 시스템이 추가될 때마다 Component를 추가해야 한다. 시스템별로 Gateway가 나눠져 있어 유지보수, 테스트에선 용이하다. 소스코드 다운로드/버전관리 주소 유효성 확인 2가지 기능만 사용하므로 새로운 버전 관리 시스템 추가 시 2가지 기능만 구현하면 된다.



5.5.5.2.2. Design Approach #2 Description: 통합 Gateway 생성

모든 버전관리 시스템을 1개의 Gateway로 설계하는 방식이다. 새로운 버전관리 시스템이 추가되는 경우에는 VersionControlSystemGateway Component에서 구현하여 추가 지원해주면 된다. 단순한 기본 2가지 기능이기 때문에 구현 복잡도는 높지 않다고 판단된다. 추가용이성을 높이기 위해 Component Level Design에서 OCP와 SRP를 만족하는 Design Pattern으로 설계되어야 한다.



5.5.5.3. Decision and Rationale

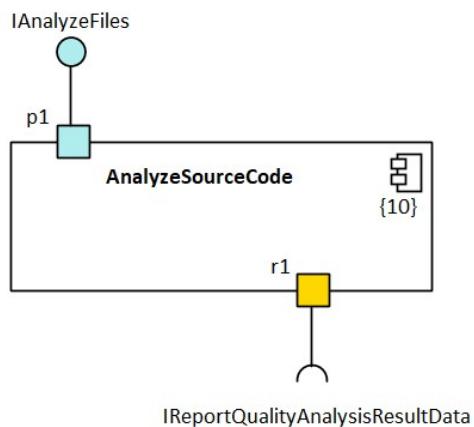
본 시스템은 버전 관리 시스템의 소스 코드를 다운로드하고 유효성을 검사하는 두 가지 기능을 사용한다. 이러한 기능들은 매우 복잡하지 않기 때문에 Top Level Design에서 컴포넌트를 분리할 필요가 없으며, Component Level Design 단계에서 높은 추가용이성을 가진 설계가 충분히 가능하다. 따라서 통합 Component인 DA#2를 선택한다.

Quality Attribute		Analysis	DA #1: 시스템별로 Component 생성	DA #2: 통합 Component 생성 (Selected)
ID	Title		DA #1: 시스템별로 Component 생성	DA #2: 통합 Component 생성 (Selected)
QA-04	<u>새로운 버전관리 시스템 지원</u>	Pros	(+) 새로운 버전관리 시스템이 추가되는 경우 유지보수 및 테스트에 용이하다	(+) 1개의 Component를 사용하기 때문에 중복 코드가 적어진다.
		Cons	(-) 버전관리 시스템 기능의 중복된 코드 발생한다. (Code smell: Duplicate code)	(-) OCP를 만족하는 디자인 패턴을 활용하여 구현되어야 하기 때문에 시스템 구현 난이도가 높아진다.

6. Component Level Design Description

6.1. AnalyzeSourceCode Design Description

6.1.1. Overview



[Interface]

«interface»
IAnalyzeFiles
+ analyzeSourceCodeFiles(sourceCodeFileList: List<File>, projectInfo: Project): void

«interface»
IReportQualityAnalysisResultData
+ getQualityAnalysisResultData(resultData: JSON): boolean

[Data Types]

«dataType»
Project
- Language: String
- lowQualityScore: double
- projectDeveloper: int
- projectId: long
- projectManager: ProjectManager
- selectQualityType: QualityType
- updateReportTime: Time
- versionControlSystemAddress: String

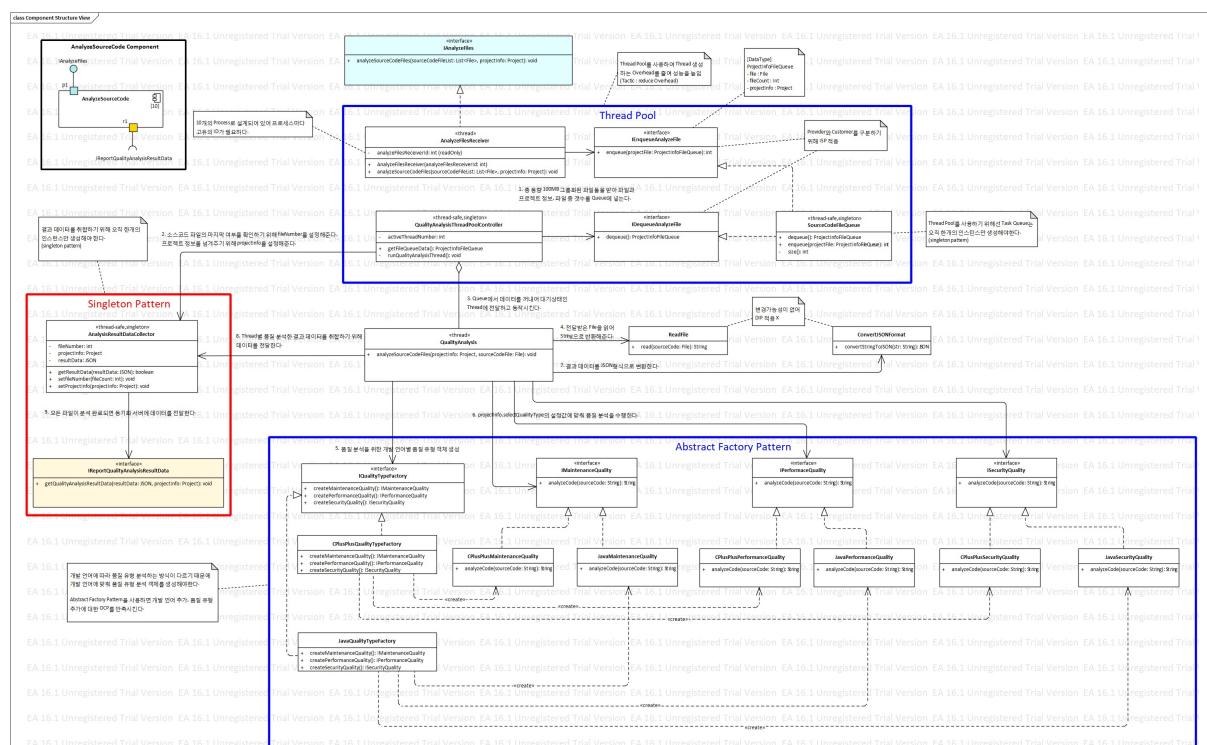
«dataType»
ProjectManager
- id: long
- nickname: String
- password: long
- phoneNumber: String

«dataType»
QualityType
- maintenance: boolean
- performance: boolean
- security: boolean

항목	설명
개요	10MB씩 그룹화된 파일들을 품질 유형에 맞게 분석한 후, 각 파일에서 추출한 분석 데이터를 전달한다.

컴포넌트 기능 요구사항	<p><input type="checkbox"/> IAnalysisFiles</p> <ul style="list-style-type: none"> - AnalysisServerManager(Business Logic Layer)로부터 전달받은 10MB씩 그룹화된 파일들을 품질 유형별로 분석한다. 분석에 대한 설정들은 [ProjectInfo] 데이터에 저장되어 있다. projectInfo.selectQualityType에서 설정되어 있는 품질 유형별로 분석한다. - 분산된 데이터를 동기화하기 위해 AsyncDataServerGateway에 전달한다. 동기화작업을 할 수 있게 품질 분석 결과 데이터를 JSON형식으로 변경해야 한다. <p>* 10MB: 100MB으로 나뉘어서 각 서버에 전달되고 컴포넌트는 10개로 나뉘어 1개의 컴포넌트당 10MB</p>
컴포넌트 품질 요구사항	<p><input type="checkbox"/> QA-01 품질 평가 수행 시간</p> <ul style="list-style-type: none"> - “소스코드 품질 평가 시스템은 품질 평가를 총 20분안에 완료되어야 한다.” <p>> 품질 평가는 총 20분안에 완료되어야 하기 때문에 분석 서버를 분산시켜 진행하게 된다. AnalyzeSourceCode 컴포넌트는 분산된 파일들을 분석하는 컴포넌트이다. 서버가 분산되어 있어 10MB 그룹화된 소스코드 파일들을 20분 이내에 분석 완료되어야 한다.</p> <p>* 10MB: 100MB으로 나뉘어서 각 서버에 전달되고 컴포넌트는 10개로 나뉘어 1개의 컴포넌트당 10MB</p> <p><u>* QA-03 새로운 품질 유형 지원</u></p> <ul style="list-style-type: none"> - “새로운 품질 유형 추가가 필요한 경우 추가할 새로운 품질 유형에 대한 결과 조회가 빠른 시간(1MM)안에 완료되어야 한다.” <p>> 새로운 품질 유형이 추가되면 기존 품질 유형의 기능에는 영향이 없어야 한다. 새로운 품질 유형에 대한 기능을 추가하면서 기존 품질 유형의 성능을 유지해야 한다. 또한 추가 용이성이 높게 설계되어야 한다.</p>

6.1.2. Static Structure Diagram



- 1) AnalyzeFilesReceiver는 IAnalyzeFiles Interface를 통해서 품질 분석해야 하는 소스코드 파일과 프로젝트 정보를 전달 받는다.
- 2) Thread Pool를 사용하기 위해 SourceCodeFileQueue에 소스코드 파일과 프로젝트 정보, 파일 개수를 Queue에 저장한다.
- 3) QualityAnalysisThreadController는 Queue에서 받은 파일의 수를 AnalysisResultData의 fileNumber에 설정한다. 이 설정은 파일의 마지막 여부를 판단하는 데 필요하게 된다. 또한 프로젝트의 정보가 담긴 ProjectInfo를 설정해준다.
- 4) QualityAnalysisThreadController는 Queue에서 받은 소스코드 파일과 프로젝트 정보, 파일 개수를 Thread에 전달하여 실행시킨다.
- 5) Abstract Factory Pattern을 사용하여 개발 언어(C++, Java)의 맞춰 품질 유형 분석 객체를 생성한다.
 C++ : CPlusPlusMaintenanceQuality, CPlusPlusPerformanceQuality, CPlusPlusSecurityQuality 생성
 Java: JavaMaintenanceQuality, JavaPerformanceQuality, JavaSecurityQuality 생성
- 6) projectInfo.selectQualityType을 참고하여 프로젝트 매니저가 선택한 품질 유형에 대해서만 품질 분석을 수행한다. 품질 분석 결과 데이터는 JSON형식을 맞춘 String으로 반환된다.
- 7) 품질 분석 결과 데이터인 String을 ConvertJSONFormat을 통해 JSON Object로 변환한다.
- 8) JSON Object으로 변환된 데이터를 AnalysisResultDataCollector 클래스에 전달한다.
- 9) AnalysisResultDataCollector은 전달받은 데이터를 멤버 변수인 resultData에 추가한다.
- 10) AnalysisResultDataCollector는 3)에서 설정된 fileNumber 멤버변수를 참고하여 마지막 여부를 판단한다.
- 11) 분석 결과 데이터가 마지막인 경우 AsynDataServerGateway의 IReportQualityAnalysisResult Data Interface를 통해 결과 데이터와 프로젝트 정보들을 전달한다.

6.1.3. Element List

Name	Responsibility
IAnalyzeFiles	그룹화된 소스코드 파일들을 받아서 프로젝트 매니저가 설정한 품질 유형에 대해 분석을 제공하는 Interface
AnalyzeFilesReceiver	그룹화된 소스코드 파일들을 받는다. 성능을 높이기 위해 Thread Pool을 사용하게 되어 Queue에 소스코드 파일을 넣는다.
IEnqueueAnalysisFile	[Provider] Queue에 추가하는 Interface (ISP 적용)
IDequeueAnalysisFile	[Customer] Queue에서 데이터를 가져오는 Interface (ISP 적용)
SourceCodeFileQueue	Queue에 프로젝트 정보, 파일 개수, 소스코드 파일을 추가 및 제거하는 기능을 제공한다. 프로세스당 오직 1개의 Queue만 존재해야 한다. 따라서 singleton pattern을 사용한다.
QualityAnalysisThread PoolController	Queue에서 데이터를 받아 Thread에 전달하여 실행시킨다. 성능을 높이기 위해 Thread pool을 사용한다. 프로세스당 오직 1개의

	ThreadPoolController만 존재해야 하므로 singleton pattern을 사용한다.
QualityAnalysis	소스코드 파일을 품질 분석 전체 Flow를 담당한다.
ReadFile	소스코드 파일(.cpp, .java 등)을 String으로 읽어 반환한다.
ConvertJSONFormat	JSON형식으로 된 String을 JSON Object으로 반환한다.
IQualityTypeFactory (AbstractFactory)	개발 언어의 품질 유형 분석 Set을 생성하는 AbstractFactory Interface
CPlusPlusQualityTypeFactory (ConcreteFactory)	C++ 개발 언어의 품질 유형 분석 Set을 생성하는 ConcreteFactory
JavaQualityTypeFactory (ConcreteFactory)	Java 개발 언어의 품질 유형 분석 Set을 생성하는 ConcreteFactory
IMaintenanceQuality (AbstractProduct)	유지보수성 품질 분석 수행 Interface
CPlusPlusMaintenanceQuality (ConcreteProduct)	C++ 의 유지보수성 품질 분석 수행
JavaMaintenanceQuality (ConcreteProduct)	Java의 유지보수성 품질 분석 수행
IPerformanceQuality (AbstractProduct)	성능 품질 분석 수행 Interface
CPlusCPlusPerformanceQuality (ConcreteProduct)	C++ 의 성능 품질 분석 수행
JavaPerformanceQuality (ConcreteProduct)	Java의 성능 품질 분석 수행
ISecurityQuality (AbstractProduct)	보안 품질 분석 수행 Interface
CPlusPlusSecurityQuality (ConcreteProduct)	C++ 의 보안 품질 분석 수행
JavaSercuirtQuality (ConcreteProduct)	Java의 보안 품질 분석 수행
AnalysisResultDataCollector	100MB 그룹화된 소스코드 파일들의 결과 데이터를 취합한다. 여러 Thread에서 결과 데이터를 전달하기 때문에 오직 1개의 인스턴스만 생성되어야 한다. 따라서 singleton pattern을 사용한다.
IReportQualityAnalysis ResultData	AsynDataServerGateway에서 제공해주는 Interface이며 결과 데이터를 전달받게 된다.

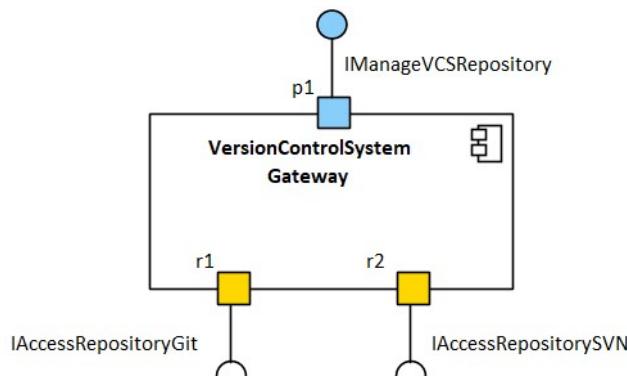
6.1.4. Design Rationale

QA	Relevant Elements	Description
QA-01 품질 평가 수행 시간	IAnalyzeFiles AnalyzeFilesReceiver IFileQueue SourceCodeFileQueue QualityAnalysisThread PoolController QualityAnalysis AnalysisResultData Collector IReportQualityAnalysis ResultData	<p>□ 10개의 멀티 프로세스 적용 - 10개의 멀티 프로세스를 적용하여 100MB 그룹화된 파일을 각 프로세스마다 10MB씩 할당하여 품질 분석을 수행한다.</p> <p>□ 10MB 그룹화된 소스코드 파일을 병렬 처리하기 위하여 Thread Pool(multi-Thread, Queue) 적용 - 전달받은 파일은 10MB 단위로 그룹화되어 있으며, 이는 여러 소스코드 파일로 이루어져 있다. 이러한 파일들을 순차적으로 하나씩 분석하는 것보다 병렬 처리를 통해 분석하는 것이 더 효율적이라 판단된다. 따라서 Thread Pool을 사용하여 Thread 생성 과정에서 발생하는 오버헤드를 줄이는 Reduce Overhead Tactic을 적용하였다.</p> <p>- Singleton pattern 적용</p> <ol style="list-style-type: none"> 1. SourceCodeFileQueue > 프로세스에서 오직 1개의 Queue만 생성 2. QualityAnalysisThreadPoolController > Thread를 제어하기 때문에 오직 1개의 객체만 생성 3. AnalysisResultDataCollector > 결과 데이터 분실 예방을 위해 오직 1개의 객체만 생성
QA-03 새로운 품질 유형 지원	IQualityTypeFactory CPlusPlusQuality TypeFactory JavaQualityTypeFactory IMaintenanceQuality CPlusPlusMaintenance Quality JavaMaintenanceQuality IPerformanceQuality CPlusCPlusPerformance Quality JavaPerformanceQuality ISecurityQuality CPlusPlusSecurityQuality JavaSercuirtQuality	<p>□ 새로운 품질 유형 추가되는 경우 빠른 지원을 위한 Abstract Factory Pattern 적용</p> <ul style="list-style-type: none"> - 새로운 품질 유형이 추가되면 기존 품질 유형과 전혀 다른 성격의 품질 유형 분석이 필요하다. 또한 같은 품질 유형이라도 개발 언어에 따라서 다른 분석이 필요하다. - 품질 유형의 변경점에 대해서 독립적이어야 하고, 개발 언어에 따라 품질 유형 Set이 적용되어야 한다. - Abstract Factory Pattern을 이용하여 품질 유형 분석 Set을 생성하는 역할을 Encapsulation하고, 인터페이스를 사용하여 구현을 분리함으로써 DIP, SRP, OCP를 모두 만족시키도록 설계하였다. - 그 외 다른 후보 설계: Strategy Pattern <pre> classDiagram class QualityAnalysis { <<thread>> + QualityAnalysis(projectId: Project, sourceCodeFile: File) + setAnalysisStrategy(IAalysisQuality): void } class IAnalysisQuality { <<interface>> + analyzeCode(sourceCode: string): JSON } class CPlusPlusQuality { + analyzeCode(sourceCode: string): JSON } class CPlusPlusPerformanceQuality { + analyzeCode(sourceCode: string): JSON } class CPlusPlusSecurityQuality { + analyzeCode(sourceCode: string): JSON } class JavaMaintenanceQuality { + analyzeCode(sourceCode: string): JSON } class JavaPerformanceQuality { + analyzeCode(sourceCode: string): JSON } QualityAnalysis --> IAnalysisQuality CPlusPlusQuality --> IAnalysisQuality CPlusPlusPerformanceQuality --> IAnalysisQuality CPlusPlusSecurityQuality --> IAnalysisQuality JavaMaintenanceQuality --> IAnalysisQuality JavaPerformanceQuality --> IAnalysisQuality </pre> <p>품질 유형 모두 IAnalysisQuality Interface를 implement함</p>

		<p>여 품질 유형 분석 기능을 제공해준다. 품질 유형에 맞게 전략을 수정하여 행위를 변경하는 패턴이다. 품질 분석은 런타임 중에 품질 유형 분석 전략을 변경해야 하므로 Strategy Pattern도 알맞은 디자인 패턴이라 판단된다. 그러나 Abstract Factory Pattern을 사용한 이유는 개발 언어와 품질 유형을 분리시키는 것이 중요하다고 판단했기 때문이다. 예를 들어 Java 개발 언어인 프로젝트를 품질 분석할 때 Java 개발 언어의 성격에 맞는 품질 유형 Set만 사용하고, C++ 개발 언어 관련 품질 유형 Set은 사용하지 않는다. 따라서 필요한 개발 언어에 대해서만 객체를 생성하는 것이 올바른 선택이라 생각한다. 따라서 Abstract Factory Pattern을 선택하여 설계하였다.</p>
--	--	---

6.2. VersionControlSystemGateway Design Description

6.2.1. Overview



«interface»
IMANAGEVCSREPOSITORY

- + downloadRepository(address: String): boolean
- + validateRepository(address: String): boolean

«interface»
IAccessssRepositoryGIT

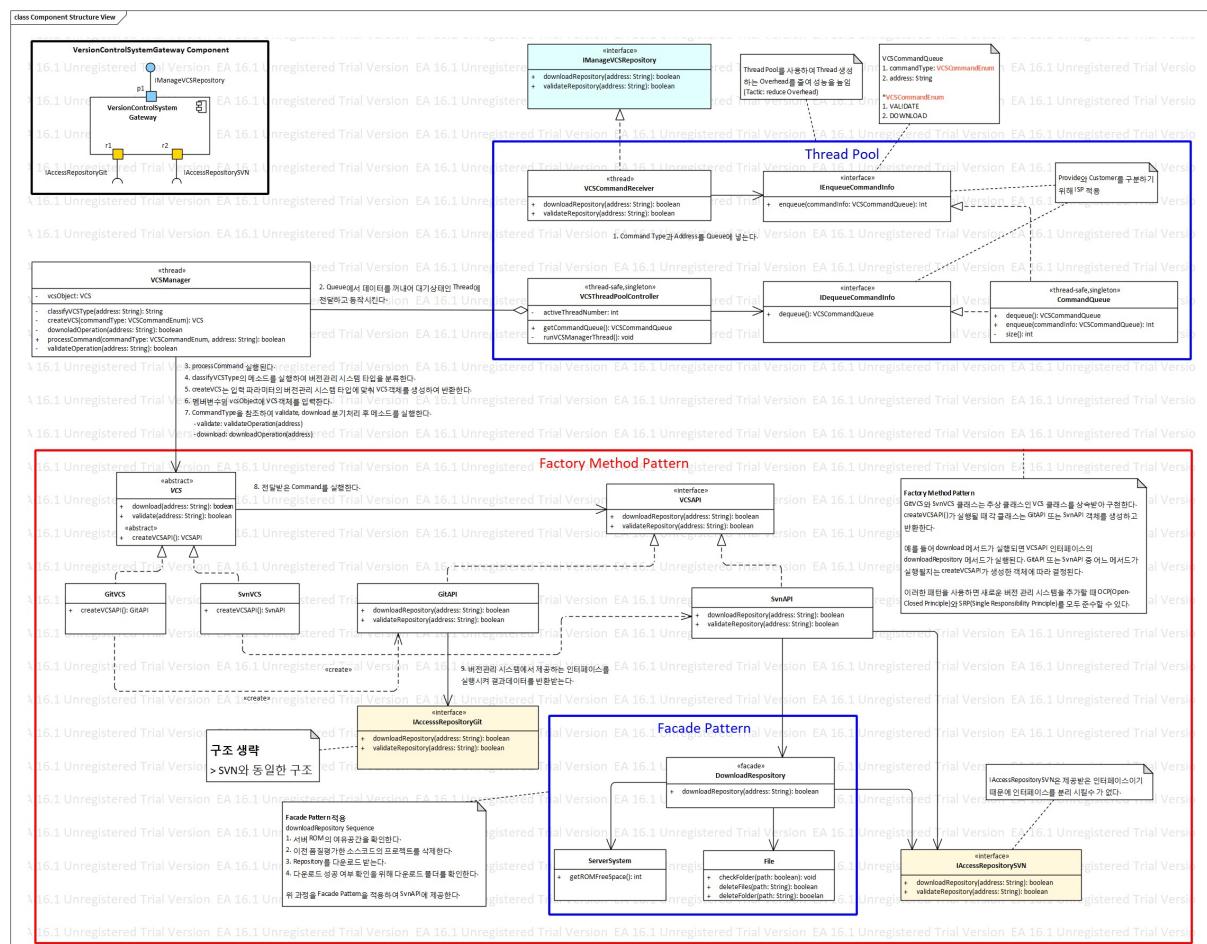
- + downloadRepository(address: String): boolean
- + validateRepository(address: String): boolean

«interface»
IAccessRepositorySVN

- + downloadRepository(address: String): boolean
- + validateRepository(address: String): boolean

항목	설명
개요	버전관리 시스템별로 Repository 파일 다운로드, Repository 유효성 검사 기능을 제공해주는 컴포넌트이다.
컴포넌트 기능 요구사항	<p><input type="checkbox"/> IManageVCSRepository 전달받은 Repository Address를 참고하여 버전관리 시스템별(Git, SVN)로 분류한다. 다운로드/유효성 검사에 대한 기능을 제공한다.</p> <ul style="list-style-type: none"> - downloadRepository 전달받은 Repository Address를 접근하여 프로젝트 파일을 다운로드한다. 다운로드 성공 여부를 Return해준다. - validateRepository 전달받은 Repository Address를 접근하여 유효성 검사를 한다. 유효성 검사에 대한 결과값을 Return해준다.
컴포넌트 품질 요구사항	<p><input type="checkbox"/> QA-04 새로운 버전관리 시스템 지원</p> <ul style="list-style-type: none"> - “새로운 버전관리 시스템 추가가 필요한 경우 추가할 새로운 버전 관리 시스템과 본 시스템의 연동이 빠른 시간(1MM)안에 완료되어야 한다.” > 새로운 버전관리 시스템이 추가되더라도 기존 버전 관리 시스템의 기능에는 영향이 없어야 하며, 성능도 유지되어야 한다. 또한 유연하고 빠른 대응을 위해 높은 추가용이성을 가진 설계가 되어야 한다.

6.2.2. Static Structure Diagram



- 1) ProjectInfoManager Component, EvaluationServerManager Component로부터 IManageVCSRepository Interface를 통해서 전달받은 Command Type과 Address 데이터를 Thread

Pool 사용하기 위해 Queue에 넣는다.

- 2) Queue에서 데이터를 꺼내어 대기상태인 Thread에 전달하고 실행시킨다.
 - 3) 작업 요청을 전달받은 Thread는 processCommand가 실행된다.
 - 4) 버전 관리 시스템의 종류를 알기 위해서 classifyVCSType을 실행시켜 종류를 Return받는다.
 - 5) FactoryMethodPattern의 Creator부분인 VCS객체를 생성한다. 버전 관리 시스템의 종류에 맞춰 객체가 생성되어 생성한 객체를 Return해준다.
 - 6) VCSManager클래스의 멤버변수인 vcsObject에 반환된 객체를 입력한다.
 - 7) CommandType을 참조하여 전달된 Command가 실행된다.
- * CommandType이 VALIDATE인 경우
 > VCSManager클래스의 validateOperation실행
- * CommandType이 DOWNLOAD인 경우
 > VCSManager클래스의 downloadOperation실행
- 8) DOWNLOAD인 경우 (한가지 예를 들어 설명) VCS클래스의 download메서드가 실행된다.
 - 9) VCSAPI의 인터페이스인 downloadRepository가 실행되는데 GitAPI, SvnAPI 둘 중 어떤 메서드가 실행되는지 판단여부는 5)에서 생성된 객체에서 결정된다.
 - 10) 요청한 명령에 맞게 실행한다. (Download인 경우 Façade Pattern 적용)

6.2.3. Element List

Name	Responsibility
IManageVCSRepository	버전 관리 시스템별 Repository 유효성검사, 다운로드기능을 제공해주는 Interface
VCSCmdReceiver	외부 Component에서 요청을 전달받는다. 성능을 높이기 위해 Thread Pool을 사용한다. 작업을 저장하는 Queue에 전달받은 요청 데이터를 넣는다.
IEnqueueCommandInfo	[Provider] VCS 동작 관련 데이터를 추가하는 Interface (ISP 적용)
IDequeueCommandInfo	[Customer] VCS 동작 관련 데이터를 가져오는 Interface (ISP 적용)
CommandQueue	Queue에 CommandType, Repository Address를 추가 및 제거하는 기능을 제공한다. 오직 1개의 Queue만 존재해야 한다. 따라서 singleton pattern을 사용한다.
VCSThreadPoolController	Queue에서 데이터를 받아 Thread에 전달하여 실행시킨다. 성능을 높이기 위해 Thread pool을 사용한다. 오직 1개의 객체만 생성되어야 하기 때문에 singleton pattern을 사용한다.
VCSManager	버전 관리 시스템의 전반적인 Flow를 담당한다. 버전 관리 시스템의 종류를 분류하고 Factory Method Pattern의 Client로 VCS 객체를 생성하여 요구된 Command동작을 실행한다.
VCS (Creator)	Factory Method Pattern의 Creator부분으로 종류가 다른 버전 관리 시스템 객체를 반환하는 팩토리 메서드가 Abstract로 설계된다. 새로운 버전 관리 시스템이 추가된다면 VCS를 상속받아 구현하면 된다. 따

	라서 OCP에 만족하게 하는 역할을 한다.
GitVCS (ConcreteCreator)	Abstract 클래스인 VCS를 상속받아 구현한 클래스이다. GitAPI 객체를 생성하여 반환한다.
SvnVCS (ConcreteCreator)	Abstract 클래스인 VCS를 상속받아 구현한 클래스이다. SvnAPI 객체를 생성하여 반환한다.
VCSAPI (Product)	버전관리 시스템의 동작을 담당하는 Interface이다. 새로운 버전관리 시스템이 추가되면 해당 Interface를 구현해서 추가하면 된다. 따라서 OCP를 만족하며 또한 SRP를 만족시켜 더 쉽게 유지관리가 가능하다.
GitAPI (Concrete Product)	VCSAPI 인터페이스를 Git에 맞춰 구현한 클래스
SvnAPI (Concrete Product)	VCSAPI 인터페이스를 SVN에 맞춰 구현한 클래스
DownloadRepository (Façade)	Façade가 적용된 클래스로 Download에 대한 기능을 제공해준다. * DownloadRepository Sequence <ul style="list-style-type: none"> 1. 서버 ROM의 여유공간을 확인한다. 2. 이전 품질평가한 Repository를 삭제한다. 3. Repository를 다운로드 받는다. 4. 다운로드 성공 여부 확인을 위해 다운로드 폴더를 확인한다.
ServerSystem	Server시스템을 접근하는 클래스
File	파일을 저장/삭제/확인기능을 제공하는 클래스
IAccessRepositoryGit	Git서버에서 제공하는 Interface (validate, download기능)
IAccessRepositorySVN	SVN서버에서 제공하는 Interface (validate, download기능)

6.2.4. Design Rationale

QA	Relevant Elements	Description
QA-04 새로운 버전 관리 시스템 지원	VCSManager VCS GitVCS SvnVCS VCSAPI GitAPI SvnAPI	<p><input type="checkbox"/> 새로운 버전 관리 시스템이 추가되는 경우 빠른 지원을 위한 Factory Method Pattern 적용</p> <ul style="list-style-type: none"> - 새로운 버전 관리 시스템이 추가될 때, 기존 버전 관리 시스템 (Git, SVN)에 영향을 미치지 않고 추가될 수 있어야 하므로 OCP를 만족하는 설계가 필요하다. - Factory Method Pattern을 적용하여 기존 버전 관리 시스템의 코드를 수정하지 않고 새로운 버전 관리 시스템을 도입할 수 있다. <p>Factory Method Pattern</p> <ol style="list-style-type: none"> 1. GitVCS와 SvnVCS 클래스는 추상 클래스인 VCS 클래스를 상속받아 구현한다. 2. createVCSAPI()가 실행될 때 각 클래스는 GitAPI 또는 SvnAPI 객체를 생성하고 반환한다. 3. 예를 들어 download 메서드가 실행되면 VCSAPI 인터페이스의 downloadRepository 메서드가 실행된다. 4. GitAPI 또는 SvnAPI 중 어느 메서드가 실행될지는

createVCSAPI가 생성한 객체에 따라 결정된다.

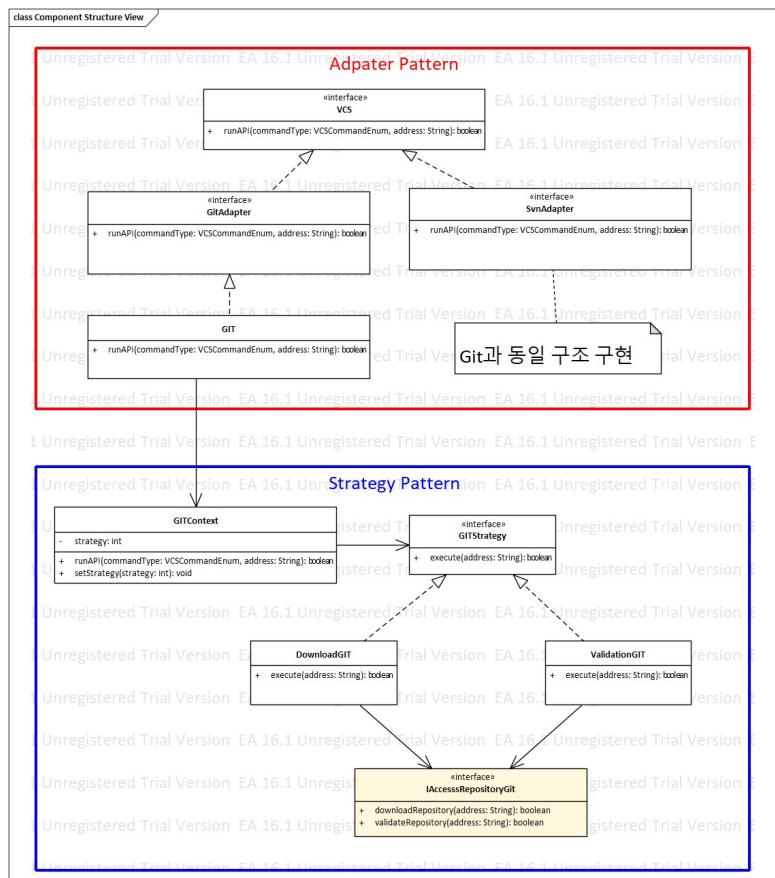
위 과정은 Factory Method Pattern을 본 시스템에 적용하였을 때 시스템의 Sequence를 간략하게 설명한 내용이다.

만약 ‘XYZ’의 버전 관리 시스템이 추가되는 경우를 가정하면

- VCS 클래스를 상속받아 XyzVCS를 구현한다.
(기존 코드를 훼손하지 않음 > OCP만족)
- VCSAPI 인터페이스를 상속받아 XyzAPI를 구현한다.
(기존 코드를 훼손하지 않음 > OCP만족)
- (XYZ 버전 관리 시스템의 기능만 책임 부여 > SRP만족)

따라서 새로운 버전 관리 시스템을 추가할 때 OCP와 SRP를 모두 준수할 수 있는 Factory Method Pattern이 적합한 패턴으로 판단하여 설계하였다.

- 그 외 다른 후보 설계: Adapter Pattern + Strategy Pattern



Git과 SVN의 객체를 생성하기 위해 Adapter 패턴을 적용하고, 생성된 객체의 동작 구현 부분에는 Strategy 패턴을 적용하여 새로운 버전 관리 시스템을 추가할 때 기존 시스템에 영향을 주지 않도록 설계할 수 있다. 하지만 Adapter Pattern과 Strategy Pattern을 사용하는 대신 Factory Method Pattern을 사용한 이유는 다음과 같다.

	<p>첫째, Adapter Pattern과 Strategy Pattern을 함께 사용하는 설계는 복잡도가 높아진다. 만약 새로운 버전 관리 시스템을 추가할 때 Adapter 인터페이스를 생성하고, 이를 구현하기 위해 Context와 Strategy 함수를 모두 구현해야 한다. 반면, Factory Method Pattern은 인터페이스를 생성하지 않고 상속을 통해 간단히 구현할 수 있다.</p> <p>둘째, Client인 VCSManager는 적절한 전략을 선택하기 위해 모든 전략에 대해 알고 있어야 한다. Factory Method Pattern은 메소드로 제공되기 때문에 Client 입장에서 간단히 이해할 수 있다. 이는 코드 유지보수를 위해 Factory Method Pattern이 더 적합하다는 것을 의미한다.</p> <p>셋째, download와 validate 외에 다른 동작이 추가되는 경우, Adapter Pattern과 Strategy Pattern을 사용하면 코드 수정이 많아질 수 있다. 입력 파라미터와 반환 값이 다른 경우에는 Strategy Pattern을 사용할 수 없다. 반면, Factory Method Pattern은 문제없이 추가할 수 있다.</p> <p>따라서, Adapter Pattern과 Strategy Pattern을 사용하는 대신 <u>Factory Method Pattern</u>을 선택하는 것이 더 적합하며, 이 세 가지 패턴 OCP를 만족시키기 때문에 유연한 코드 설계가 가능하다.</p>
--	--

7. Architecture Evaluation

7.1. Traceability Summary

Architectural Driver		Design Elements
ID	Title	
UC-01	프로젝트 정보 관리	<p><input type="checkbox"/> Structure View [소스코드 품질 평가 시스템 관리 서버]</p> <ul style="list-style-type: none"> - RegisterProjectUI: 프로젝트 정보 등록에 대해서 입력된 프로젝트 정보 데이터를 수신한다. - ProjectInfoManager: 프로젝트 정보 관리의 전체 Workflow (버전 관리 시스템 주소 유효성 검사 > ProjectInfo DB저장 > 프로젝트 품질 분석 리스트 스케줄링 > 프로젝트 품질 분석 리스트 DB 저장)을 관리한다. - ProjectInfoDAC: 프로젝트에 대한 정보들을 Database에 접근하여 읽고/쓴다. - SchedulingInfoDAC: 프로젝트 품질 분석 리스트에 대한 정보들을 Database에 접근하여 읽고/쓴다. - VersionControlSystemGateway: 프로젝트 등록시 버전 관리 시스템 주소 유효성 검사를 제공한다. <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.01 프로젝트 정보 관리 (전체)
UC-02	프로젝트 결과 관리	<p><input type="checkbox"/> Structure View [소스코드 품질 평가 시스템 관리 서버]</p> <ul style="list-style-type: none"> - AnalysisResultReportUI: 프로젝트 매니저로부터 프로젝트 결과 보고서 조회 요청 데이터를 수신한다. - ReportManager: 프로젝트 결과 보고서 조회 요청 데이터를 ReportServerGateway에 전달한다. - ReportServerGateway: 프로젝트 결과 보고서 조회 요청 데이터를 품질 결과 조회 서버에 전달한다. <p>[품질 결과 조회 서버]</p> <ul style="list-style-type: none"> - AnalysisReportServerFrontController: 소스코드 품질 평가 시스템 관리 서버로부터 전달받은 결과 보고서 조회 요청 데이터를 수신한다. - AnalysisReportProcessor: 품질 분석 DB에서 결과 데이터를 읽어 보고서 형식으로 변환하여 품질 분석 결과 보고서를 제공한다. - QualityAnalysisDataDAC: 프로젝트 결과 데이터에 대한 정보들을 Database에 접근하여 읽고/쓴다. <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.02 프로젝트 결과 관리 (전체)
UC-03	프로젝트 파일 별 결과 조회	<p><input type="checkbox"/> Structure View</p> <ul style="list-style-type: none"> - AnalysisResultReportUI: SW개발자로부터 품질 분석 파일 별 결과 보고서 조회 요청 데이터를 수신한다. - ReportManager: 품질 분석 파일 별 결과 보고서 조회 요청 데이터를 수신한다.

		<p>이터를 ReportServerGateway에 전달한다.</p> <ul style="list-style-type: none"> - ReportServerGateway: 품질 분석 파일 별 결과 보고서 조회 요청 데이터를 품질 결과 조회 서버에 전달한다. <p>[품질 결과 조회 서버]</p> <ul style="list-style-type: none"> - AnalysisReportServerFrontController: 소스코드 품질 평가 시스템 관리 서버로부터 전달받은 품질 분석 파일 별 결과 보고서 조회 요청 데이터를 수신한다. - AnalysisReportProcessor: 품질 분석 DB에서 결과 데이터를 읽어 보고서 형식으로 변환하여 품질 분석 파일 별 결과 보고서를 제공한다. 그리고 품질 규칙 위반 유형, 수정 가이드라인도 제공한다. - QualityAnalysisDataDAC: 프로젝트 결과 데이터에 대한 정보들을 Database에 접근하여 읽고/쓴다. <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.03 프로젝트 파일 별 결과 조회 (전체)
UC-04	소스코드 품질 평가	<p><input type="checkbox"/> Structure View</p> <p>[소스코드 품질 평가 시스템 관리 서버]</p> <ul style="list-style-type: none"> - ProjectEvaluationManager: Trigger 신호를 받은 경우 품질 분석 수행 리스트가 저장되어 있는 DB를 읽어 품질 분석할 프로젝트 정보들을 품질 평가 준비 서버에 전달한다. - SchedulingInfoDAC: 프로젝트 품질 분석 리스트에 대한 정보들을 Database에 접근하여 읽고/쓴다. - LoadBalancerGateway: 부하되지 않게 Load Balancer를 사용한다. 프로젝트 품질 분석에 대한 정보를 전달한다. <p>[품질 평가 준비 서버]</p> <ul style="list-style-type: none"> - EvaluationServerFrontController: 소스코드 품질 평가 시스템 관리 서버로부터 전달받은 프로젝트 품질 분석에 대한 정보를 수신한다. - EvaluationServerManager: 프로젝트 소스코드를 다운받기 위해 VersionControlSystemGateway에 Repository Address 정보를 전달한다. 소스코드 다운로드가 완료되면 LoadBalancer Gateway에 100MB 그룹화하여 소스코드 파일과 프로젝트 정보를 전달한다. - VersionControlSystemGateway: 전달받은 Repository 주소에 접근하여 소스코드를 다운로드 받는다. - LoadBalancerGateway: LoadBalancer에 100MB 그룹화된 파일과 프로젝트 정보들을 전달한다. <p>[품질 분석 서버]</p> <ul style="list-style-type: none"> - AnalysisServerFrontController: 100MB 그룹화된 파일과 프로젝트 정보들을 수신한다. - AnalysisServerManager: 100MB 그룹화된 파일들을 10MB씩 나누어 분석하기 위해 AnalyzeSourceCode에 데이터를 전달한다. - AnalyzeSourceCode: 10MB 파일과 프로젝트 정보들을 전달받아 유지보수성, 성능, 보안에 대한 분석을 수행한다. 완료된 분석 데이터는 JSON형식으로 변환한다. 품질 결과 동기화/DB 저장 서버에 전달하기 위해 AsynDataServerGateway에 전달한다. - AsynDataServerGateway: 분산된 파일들을 동기화 작업을 위해 JSON형식으로 전달받은 품질 분석 결과 데이터를 전달한다. <p>[품질 분석 결과 동기화/DB 저장 서버]</p>

		<ul style="list-style-type: none"> - AsynDataServerFrontController: 품질 분석 결과 데이터를 수신한다. - AsynDataServerManager: 분산된 파일들이 모두 품질 분석이 수행될 때까지 대기한다. 모든 분석 결과 데이터를 전달받으면 품질 분석 결과 데이터에 대한 정보들을 Database에 저장한다. 저 품질 기준 점수보다 낮은 경우 프로젝트 매니저에게 Push알림을 보낸다. - QualityAnalysisDAC: 품질 분석 결과 데이터에 대한 정보들을 접근하여 읽고/쓴다. <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.04 소스코드 품질 평가 (전체)
UC-05	저 품질 프로젝트 정보	<p><input type="checkbox"/> Structure View</p> <p>[품질 결과 분석 동기화/DB 저장 서버]</p> <ul style="list-style-type: none"> - LowQualityPushServerGateway: 담당 프로젝트 매니저 정보와 저 품질 기준 점수, 품질 분석 점수를 저 품질 프로젝트 알림 서버에 전달한다. <p>[저 품질 프로젝트 알림 서버]</p> <ul style="list-style-type: none"> - LowQualityPushServerFrontController: 담당 프로젝트 매니저 정보와 저 품질 기준 점수, 품질 분석 점수를 수신한다. - LowQualityPushServer: 전달받은 데이터를 통해서 Push알림 Message를 작성한 후 Push서버에 전달한다. - PushServerGateway: 외부 Push서버에 전달받은 Message와 담당 프로젝트 매니저 ID 정보를 전달한다. <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.05 저 품질 프로젝트 정보 (전체)
UC-06	시스템 상태 탐지 및 복구	<p><input type="checkbox"/> Structure View</p> <p>[시스템 상태 탐지 서버]</p> <ul style="list-style-type: none"> - ServerStatusFrontController: 시스템 상태 탐지 서버를 제외한 모든 서버가 전송하는 Heartbeat 신호를 수신한다. - ServerMonitoringManager: 1분마다 Trigger 신호를 받은 경우 Heartbeat 신호를 전달받지 못한 서버는 장애로 판단하여 ServerRecoveryService에 복구 기능을 요청한다. 장애가 발생한 경우 시스템 관리자에게 장애 서버, 장애 발생 시간, 장애 유형, 복구 완료 시간을 MMS메시지로 전달한다. - ServerRecoveryService: 복구 요청 전달받은 서버를 재가동하여 복구한다. - MMSGateway: 전달받은 장애 서버, 장애 발생 시간, 장애 유형, 복구 완료 시간을 MMS서버에 전달한다. <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.06 시스템 상태 탐지 및 복구 (전체)
QA-01	품질 평가 수행 시간	<p><input type="checkbox"/> System Infrastructure View</p> <ol style="list-style-type: none"> 1. 소스코드 품질 평가 시스템 관리 서버: 3분마다 Trigger 발생 2. Load Balancer(L4): 23대 품질 평가 준비 서버 분산 처리를 위해 L4로드 밸런서 적용 3. 품질 평가 준비 서버: 소스코드 다운로드 → 100MB 그룹화하여 품질 분석 서버에 전달 4. Load Balancer(L4): 50대 품질 분석 서버 분산 처리를 위해 L4로드 밸런서 적용 5. 품질 분석 서버: 품질 유형에 맞게 분석 6. 품질 분석 결과 동기화/DB 서버: 분산된 결과 파일을 동기화 작

		<p>업 및 DB 저장 / 저 품질 프로젝트인 경우 알림 서버에 전달</p> <p>7. 저 품질 프로젝트 알림 서버: 담당 프로젝트 매니저에게 저 품질 프로젝트에 대한 내용을 Push 알림 * DD-01 품질 평가 성능을 위한 분석 서버 분리</p> <p>□ Structure View</p> <ol style="list-style-type: none"> 소스코드 품질 평가 시스템 관리 서버 > ProjectEvaluationManager: 3분마다 Trigger 신호 수신 품질 평가 준비 서버 > EvaluationServerManager: 분산하여 품질 분석하기 위해 100MB 그룹화 진행 VersionControlSystemGateway: 소스코드 다운로드 품질 분석 서버 > AnalysisServerManager: analyzeSourceCode 컴포넌트에 10MB씩 전달한다. AnalyzeSourceCode: 품질 유형별 분석 수행 품질 분석 결과 동기화/DB 서버 > AsyncDataServerManager: 분산된 결과 데이터 동기화 작업 저 품질 프로젝트 알림 서버 > LowQualityPushServer: 담당 프로젝트 담당자에게 저 품질 프로젝트 정보 Push 알림 전송 <p>□ Behavior View</p> <ul style="list-style-type: none"> - UC.04 소스코드 품질 평가 (전체) - UC.05 저 품질 프로젝트 정보 (전체) <p>□ Component Level Design</p> <p>[AnalyzeSourceCode]</p> <p>10MB 파일들을 각 파일마다 품질 분석을 수행하기 위해 Thread Pool(Multi-Thread, Queue)를 활용하여 성능을 높임 <u>(적용 Tactic: Reduce Overhead)</u></p> <ul style="list-style-type: none"> - AnalysisFilesReceiver, IFileQueue, SourceCodeFileQueue, QualityAnalysisThreadPoolController
QA-02	시스템 장애 탐지 및 복구	<p>□ System Infrastructure View</p> <ol style="list-style-type: none"> 시스템 상태 탐지 서버: 시스템 상태 탐지 서버를 제외한 모든 서버로부터 Heartbeat신호 수신 * DD-02 시스템 장애 탐지를 위한 Tactic 적용 <p>□ Structure View</p> <ol style="list-style-type: none"> ServerMonitoringManager: 시스템 상태 탐지 서버를 제외한 모든 서버에서 Heartbeat신호를 전달받지 못한 경우 장애 판단. 장애 복구가 완료되면 시스템 관리자에게 장애 관련 정보 MMS 메시지 전송 ServerRecoveryService: 장애 판단된 서버를 제 가동하여 복구 수행 <p>□ Behavior View</p> <ul style="list-style-type: none"> - UC.06 시스템 상태 탐지 및 복구 (전체)
QA-03	새로운 품질 유형 지원	<p>□ System Infrastructure View</p> <ol style="list-style-type: none"> 품질 분석 서버: 품질 유형별 분석 수행 후 결과 데이터 전달 <p>□ Structure View</p> <ol style="list-style-type: none"> AnalysisServerManager: 품질 분석 준비 서버로부터 전달받은 100MB 그룹화된 파일을 품질 유형 분석을 위해 10MB씩 AnalyzeSourceCode Component{10}에 전달

		<p>2. AnalyzeSourceCode: 품질 유형 분석을 수행하고 분산된 결과 데이터의 동기화작업이 필요하기 때문에 동기화 서버에 전달</p> <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.04 소스코드 품질 평가 [1.17 ~ 1.19] <p><input type="checkbox"/> Component Level Design</p> <p>[AnalyzeSourceCode]</p> <p>새로운 품질 유형 추가에 대해 유연하게 대응하기 위해 <u>Abstract Factory Pattern</u> 적용. 개발 언어에 대해 Factory를 만들고 각 개발 언어의 품질 유형 Set를 생성하게 함. 새로운 품질 유형 추가 시 품질 유형 Set을 추가하는 형태로 기존 코드에 영향없이 추가 지원. 더불어 개발 언어가 추가되었을 때도 유연하게 대응 가능</p> <ul style="list-style-type: none"> - IQualityTypeFactory, CPlusPlusQualityTypeFactory, JavaQualityTypeFactory, IMaintenanceQuality, CPlusPlusMaintenanceQuality, JavaMaintenanceQuality, IPerformanceQuality, CPlusCPlusPerformanceQuality, JavaPerformanceQuality, ISecurityQuality, CPlusPlusSecurityQuality, JavaSercuirtQuality,
QA-04	새로운 버전관리 시스템 지원	<p><input type="checkbox"/> Structure View</p> <p>VersionControlSystemGateway: 유지 보수성을 위하여 버전관리 시스템별 Gateway를 두지 않고 통합 Gateway 설계</p> <p>* DD-04 새로운 버전관리 시스템 지원을 위한 통합 Component 설계</p> <p><input type="checkbox"/> Behavior View</p> <ul style="list-style-type: none"> - UC.01 프로젝트 정보 관리 [1.2 ~ 1.7] - UC.04 소스코드 품질 평가 [1.7 ~ 1.12] <p><input type="checkbox"/> Component Level Design</p> <p>[VersionControlSystemGateway]</p> <p>새로운 버전 관리 시스템 추가에 대해 유연하게 대응하기 위해 <u>Factory Method Pattern</u> 적용. 새로운 버전 관리 시스템에 대해 Creator 클래스를 상속받아 구현하고 Product 인터페이스를 상속 받아 새로운 버전 관리 시스템의 동작을 구현함. 기존 버전 관리 시스템 코드에 영향없이 추가 지원 가능</p> <ul style="list-style-type: none"> - VCS, GitVCS, SvnVCS, VCSAPI, GitAPI, SvnAPI