ARMS (Atlassian RMS) SDS - Software Design Specification

- 1. 프로젝트 개요
- 1. 개요
  - 1.1 목적
  - 1.2 범위
  - 1.3 관련 문서
  - 1.4 개발자 스펙
- 2. 설계 고려사항
  - 2.1 소프트웨어 설계 목표
  - 2.2 제약사항
    - 2.2.1 H/W 제약사항
    - 2.2.2 S/W 제약사항
    - 2.2.3 N/W 제약사항
  - 2.3 기타
- 3. 설계
  - 3.1 구성
    - 3.1.1 소프트웨어 배치
    - 3.1.2 소프트웨어 아키텍쳐
    - 3.1.3 Component Design
      - 3.1.3.1 통합 계정 관리 Manager (Component)
      - 3.1.3.2 통합 권한 관리 Manager (Component)
      - 3.1.3.3 로컬 계정 관리 Manager (Component)
      - 3.1.3.4 연동 계정 관리 Manager (Component)
      - 3.1.3.5 사용자별 맞춤 Manager (Component)
      - 3.1.3.6 수집 호스트 분류 Manager (Component)
      - 3.1.3.7 자동 설정 등록 Manager (Component)
      - 3.1.3.8 요구사항 등록 Manager (Component)
      - 3.1.3.9 요구사항 리뷰 Manager (Component)
      - 3.1.3.10 요구사항 권한 Manager (Component)
      - 3.1.3.11 요구사항 버전 Manager (Component)
      - 3.1.3.12 요구사항 승인, 연동 Manager (Component)
      - 3.1.3.13 요구사항 이력 Manager (Component)
      - 3.1.3.14 요구사항 진척 Manager (Component)
      - 3.1.3.15 요구사항 알람 Manager (Component)
      - 3.1.3.16 제품(서비스) 등록 Manager (Component)
      - 3.1.3.17 제품(서비스) ALM Mapping Manager (Component)
      - 3.1.3.18 제품(서비스) - 요구사항 - Issue Tracker Mapping Manager (Component)
      - 3.1.3.19 제품(서비스) - 요구사항 - WIKI Mapping Manager (Component)
      - 3.1.3.20 제품(서비스) - 요구사항 - VCS Mapping Manager (Component)
      - 3.1.3.21 제품(서비스) - 요구사항 - Code Quality Mapping Manager (Component)
      - 3.1.3.22 제품(서비스) - 요구사항 - CICD Mapping Manager (Component)
    - 3.1.4 Data Design
    - 3.1.5 Interface Design
      - 3.1.5.1 Module Interface
      - 3.1.5.2 Component Interface Design
      - 3.1.5.2.1 통합 계정 관리 Manager (Component)
      - 3.1.5.2.2 통합 권한 관리 Manager (Component)
      - 3.1.5.2.3 로컬 계정 관리 Manager (Component)
      - 3.1.5.2.4 연동 계정 관리 Manager (Component)
      - 3.1.5.2.5 사용자별 맞춤 Manager (Component)
      - 3.1.5.2.6 수집 호스트 분류 Manager (Component)
      - 3.1.5.2.7 자동 설정 등록 Manager (Component)
      - 3.1.5.2.8 요구사항 등록 Manager (Component)
      - 3.1.5.2.9 요구사항 리뷰 Manager (Component)
      - 3.1.5.2.10 요구사항 권한 Manager (Component)
      - 3.1.5.2.11 요구사항 버전 Manager (Component)
      - 3.1.5.2.12 요구사항 승인, 연동 Manager (Component)
      - 3.1.5.2.13 요구사항 이력 Manager (Component)
      - 3.1.5.2.14 요구사항 진척 Manager (Component)
      - 3.1.5.2.15 요구사항 알람 Manager (Component)
      - 3.1.5.2.16 제품(서비스) 등록 Manager (Component)
      - 3.1.5.2.17 제품(서비스) ALM Mapping Manager (Component)
      - 3.1.5.2.18 제품(서비스) - 요구사항 - Issue Tracker Mapping Manager (Component)
      - 3.1.5.2.19 제품(서비스) - 요구사항 - WIKI Mapping Manager (Component)
      - 3.1.5.2.20 제품(서비스) - 요구사항 - VCS Mapping Manager (Component)
      - 3.1.5.2.21 제품(서비스) - 요구사항 - Code Qaulity Mapping Manager (Component)
      - 3.1.5.2.22 제품(서비스) - 요구사항 - CICD Mapping Manager (Component)
    - 3.1.6 Database Description
      - 3.1.6.1 ERD (Entity Relationship Diagram)
      - 3.1.6.2 Class Diagram
- 4. 구현
  - 4.1 소스트리 구성
  - 4.2 빌드 설계
  - 4.3 배포 설계
    - 4.3.1 배포 방법
    - 4.3.2 업데이트 방법

---

1. 프로젝트 개요

1. 개요

1.1 목적

이슈 기반 요구사항 관리 시스템을 개발하여,

제품별 요구사항의 생명주기(추가-채택-변경-삭제) 및 이력을 관리하고,

이와 함께 Atlassian ALM Tool Chain과 통합하여

Business Intelligence와 Atlassian 제품군에 없는 Requirement Management System 을 제공하는

RMS (for atlassian)시스템을 개발하는 것을 목표로 aRMS를 설계합니다.

1.2 범위

본 설계의 범위는 아래 시스템의 기능에 대한 설계입니다.

구분	기능
Auth System	인증 시스템으로써, 권한 관리와 계정 관리 및 사용자 맞춤형 서비스를 제공합니다.
Product(Service) System	제품(서비스)를 관리하며, 연관하는 ALM Toolchain 과 Mapping 하여 데이터 싱크를 제공한다.
Requirement System	요구사항을 관리하며, 연관하는 제품(서비스)와 Mapping 하여, 제품 (서비스) - 요구사항 - ALM 을 연결한다.
Monitoring System	다양한 어플리케이션의 조합으로 인하여, 각 시스템간의 성능 및 안정성을 보장하기 위하여 JAVA 기반 모니터링을 지원한다.

1.3 관련 문서

구분	참조 URL
Project Charter	ARMS (Atlassian RMS) Project Charter
SRS	ARMS (Atlassian RMS) SRS - Software Requirement Specification
SDS	ARMS (Atlassian RMS) SDS - Software Design Specification

1.4 개발자 스펙

구분	필요 Skill
View Part	Html+Css, Bootstrap, jQuery, Flex, Json, Xml NodeJS, Bower, Grunt, Ajax, DWR
Server Part	Apache(modjk), Tomcat
Framework Part	Struts, Spring, iBatis, Hibernate, Spring-integration, Spring-security, Spring-Boot, Spring-DW, Spring-WebFlow, Spring-Data(JPA), Spring-Batch, Spring-WebServices, Spring-Mobile, Spring-MVC
Tool Part	Quartz, Ehcache, MemCache, Redis, Apache-Commons, EgovFramework(Component)
CI Part	Junit, Maven, Hudson, Jenkins, Bamboo, Nexus, Jira, Fisheye, Crucible, Confluence, Sonar
Database Part	MySql, Oracle, MS-sql, postgres, Hadoop, Spark, Hive
Mobile Part	Android, PhoneGap
Search Engine Part	Elastic Search, Kibana, Logstash, Beats, InfluxDB, Grafana
Management Part	PMBOK, MicroService, CBD, PLE, Prototype, PMS, ALM
Virtual Image Part	Docker, Kubernetes, Docker Swarm

Microservice Part	Netty - Zookeeper - Finagle - Kafka

2. 설계 고려사항

2.1 소프트웨어 설계 목표

jsTree Service Framework를 활용하여 Tree 알고리즘 기준의 Requirement Management System 을 설계합니다.

aRMS의 설계 목표는 Frontend 개발과 Backend 개발을 분리하는게 첫번째 입니다.

두번째 목표는 비지니스 로직은 Backend에 있되, Java 개발의 engine 및 framework 설정은 Core-Module로 분리하여

실제 비지니스 로직이 아닌 설정은 분리하는게 두번째 입니다.

세번째 목표는 패키지 및 라이브러리 관리 툴을 maven 을 활용하고, 관련한 라이브러리 역시 Lib-Module로 분리하여

손쉬운 패키지 업데이트를 지원하도록 하는게 세번째 입니다.

aRMS와 관련한 설계목표는 개발 후 → 자동으로 빌드하며 → Core-Module은 Nexus에 Upload되고, Docker는 Docker Hub에 Upload되도록

Source Write after Deploy 까지 One Shot Flow CI/CD를 제공하고

Source 설계 목표는 비지니스 코드인 제품 코드는 Web-Module 에서만 사용하도록 구성했습니다.

2.2 제약사항

2.2.1 H/W 제약사항

하드웨어는 Monitoring Server 시스템으로 인하여 많은 자원이 필요합니다.

따라서, 다음의 하드웨어 최소 스펙과 권장 스펙을 지정하도록 하겠습니다.

spec	설명	cpu	mem	disk
최소 스펙	1 Server All in one	12 croe	32 gb	500 gb
권장 스펙	4 + 1 Server Cluster	per 12 core	per 16 gb	NFS Server 2TB~

2.2.2 S/W 제약사항

aRMS는 S/W 제약사항이 존재하지 않고, 100% OpenSource를 활용하여 개발하도록 합니다.

2.2.3 N/W 제약사항

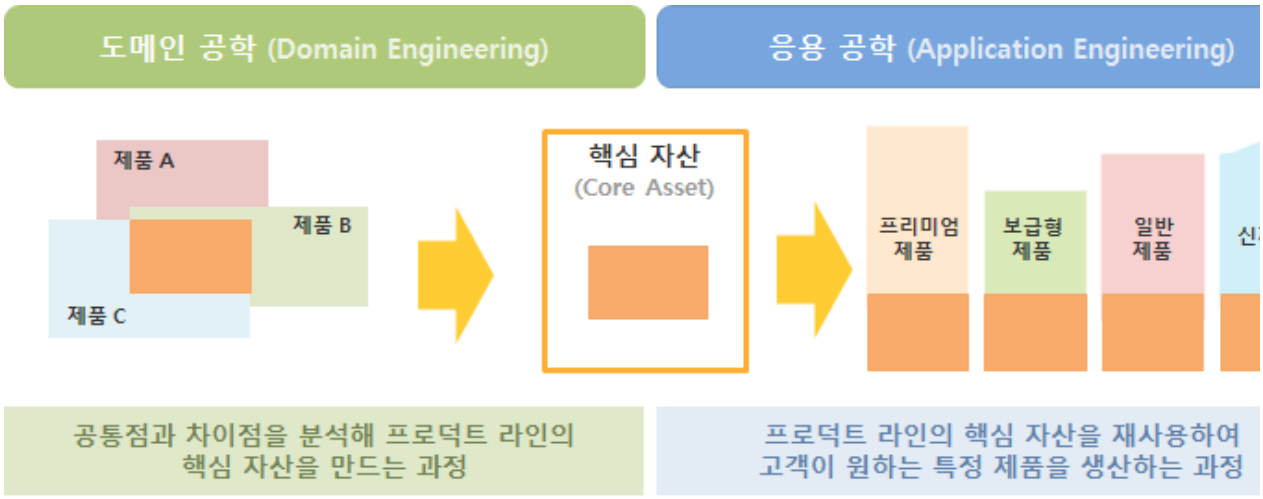
aRMS는 Atlassian ALM 제품군과 통신이 필요합니다.

2.3 기타

PLE (Product Line Engineering) 기법을 활용하여, 재사용을 극대화한 프로젝트 구조를 적용합니다.

- aRMS는 자체적인 Static Code Analysis (SonarQube)를 적용하여 코드 품질을 유지하도록 합니다.

- aRMS는 BitBucket (혹은 Github)을 활용하여 Git으로 형상관리를 적용하도록 합니다.



3. 설계

3.1 구성

3.1.1 소프트웨어 배치

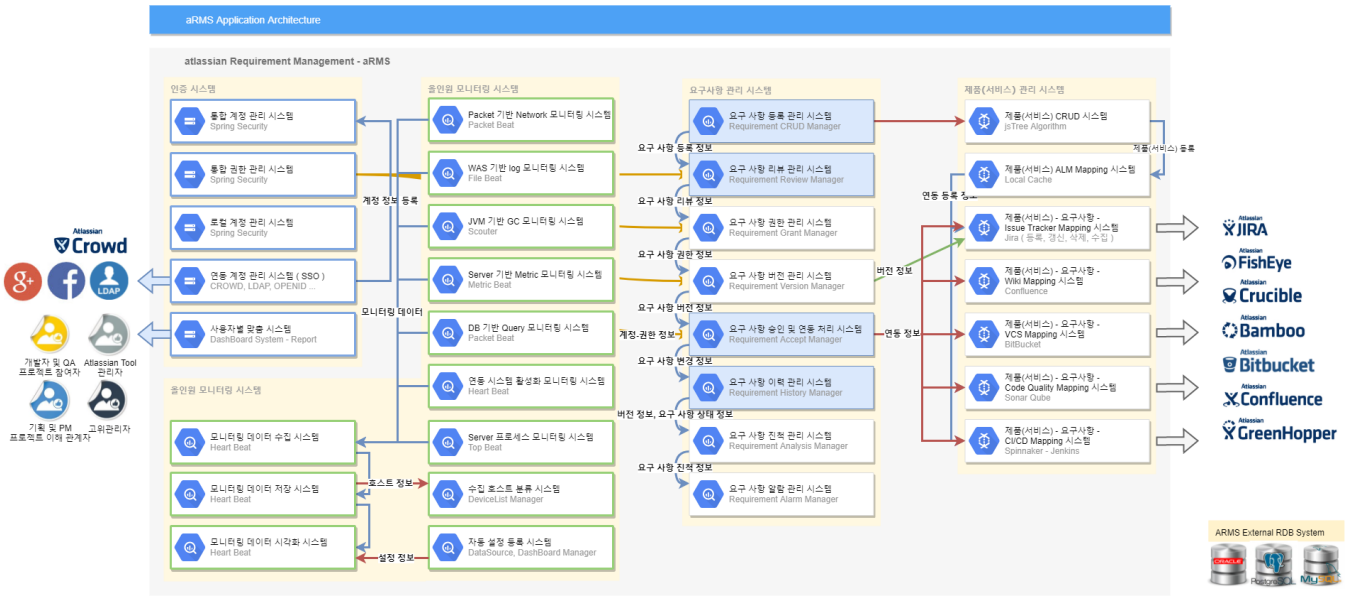
소프트웨어 배치는 SRS (Software Requirement Specification) 에서 산출된 비지니스 아키텍쳐 → 시스템 구성도 → **시스템 아키텍쳐**로 점차 구체화 되기때문에, 시스템 아키텍쳐로 소프트웨어 배치도를 갈음하도록 합니다.

3.1.2 소프트웨어 아키텍쳐

소프트웨어 아키텍쳐는 다음과 같이 구성합니다.

시스템 아키텍쳐를 기반으로 구체화된 어플리케이션 아키텍쳐로 본 장을 갈음하도록 합니다.

어플리케이션 아키텍쳐로 각 시스템의 세부 컴포넌트와 모듈을 표시하고 상호 작용하는 목적이 기술되어 있기 때문입니다.



3.1.3 Component Design

소프트웨어 구성요소인 각 Component와 그 하위 module을 정의하고, 각각의 기능 및 의존관계 등을 기술합니다.

(**input 값인 소프트웨어 아키텍쳐를 기반으로 컴포넌트의 역할과 주요 호출 등을 서술합니다.**)

3.1.3.1 통합 계정 관리 Manager (Component)

로컬 + 연동 계정의 풀에서 사용자 계정을 관리 할 수 있도록 지원하는 메니저

3.1.3.2 통합 권한 관리 Manager (Component)

로컬 + 연동 계정의 풀에서 사용자 계정에 대한 권한을 관리 할 수 있도록 지원하는 메니저

또한 전체 계정을 대상으로 권한을 정의하고 할당 할 수 있도록 지원하는 메니저

3.1.3.3 로컬 계정 관리 Manager (Component)

로컬 내부에서 계정을 관리 할 수 있도록 하는 메니저

3.1.3.4 연동 계정 관리 Manager (Component)

연동하는 계정 툴. 예를 들어 LDAP, CROWD 등의 계정을 로드하도록 지원하는 메니저

연동하는 계정을 연결 할 수 있도록 설정을 지원하는 메니저

3.1.3.5 사용자별 맞춤 Manager (Component)

전체 사용자를 일정한 권한을 기준으로 맞춤형 서비스를 제공하고, 제품(서비스) 별 대시보드를 제공하는 메니저

3.1.3.6 수집 호스트 분류 Manager (Component)

모니터링 시스템에 적재되는, 데이터를 기준으로 호스트를 분류하여 리스트를 제공하도록 하는 메니저

3.1.3.7 자동 설정 등록 Manager (Component)

모니터링 시스템의 데이터 통합 컨트롤러를 InfluxDB로 수집하기 때문에,

InfluxDB에 자동으로 등록할 수 있는 DataSource와 DashBoard를, 미리 등록된 template 를 import 할 수 있도록 지원하는 메니저

3.1.3.8 요구사항 등록 Manager (Component)

요구사항을 등록 할 수 있도록 제품(서비스) 를 표시하고 사용자로 하여금 요구사항을 등록 할 수 있도록 지원하는 메니저

3.1.3.9 요구사항 리뷰 Manager (Component)

등록된 요구사항은 제품(서비스)에 의해 구조화 되고, 해당 제품(서비스)의 요구사항은 관리되어야 할 대상으로

각 요구사항은, 미리 정의된 프로세스인 (등록->리뷰→승인) 일련의 과정을 거치게 한다.

또한, 제픔(서비스)의 일련의 과정을 관리 및 권한을 권한자로 하여금 처리할 수 있도록 한정한다.

3.1.3.10 요구사항 권한 Manager (Component)

요구사항 권한 메니저는 요구사항 리뷰 메니저에 제공할, 제품(서비스)의 일련의 과정 프로세스를 관리하고,

각 프로세스를 처리할 수 있는 권한자를 관리하는 메니저 이다.

3.1.3.11 요구사항 버전 Manager (Component)

요구사항 버전 관리 메니저는 제품(서비스)의 출시와 관련되며, 이슈 트래커에 등록할 버전을 지정하여

관리되고, 연동할 수 있는 컴포넌트이다.

3.1.3.12 요구사항 승인, 연동 Manager (Component)

요구사항은 최종 과정인 승인과, 더불어서 제품(서비스)에 연동된 Issue tracker 및 wiki 등에 후속 조치를 담당하는 메니저 입니다.

3.1.3.13 요구사항 이력 Manager (Component)

요구사항의 모든 기록을 담당하는 이력 메니저이며, 변동사항 발생시 등록된 일련의 프로세스를 재시작하는 메니저입니다.

또한 변경 또는 프로세스가 진행된 내역을 조회할 수 있으며, 연동된 시스템 역시 확인 할 수 있도록 기능을 제공합니다.

3.1.3.14 요구사항 진척 Manager (Component)

요구사항 진척 메니저는 제품(서비스)에 등록된 요구사항의 연결된 이슈와 버전정보를 바탕으로 진척도를 관리하는 메니저 입니다.

3.1.3.15 요구사항 알람 Manager (Component)

요구사항 일련의 프로세스에 따라서, 제품(서비스) 의 이해관계자 * 연동된 시스템의 사용자까지 * 에게 알람을 발송합니다.

알람 발송 시, mail 을 default로 하고 추가 알람 방식은 추후 제공하도록 합니다.

3.1.3.16 제품(서비스) 등록 Manager (Component)

aRMS의 기준 데이터 중에 하나인 제품(서비스)를 등록하고 관리하도록 합니다.

3.1.3.17 제품(서비스) ALM Mapping Manager (Component)

제품(서비스)에서 연동할 툴의 게이트를 제공하는 메니저입니다.

실제 연동 설정 데이터를 관리하지 않습니다. (이유는 1:N 의 관계를 가질 수도 있기 때문입니다)

3.1.3.18 제품(서비스) - 요구사항 - Issue Tracker Mapping Manager (Component)

제품(서비스)의 Issue Tracker 연동 설정을 제공하는 맵핑 메니저 입니다.

3.1.3.19 제품(서비스) - 요구사항 - WIKI Mapping Manager (Component)

제품(서비스)의 Wiki 연동 설정을 제공하는 맵핑 메니저입니다.

위키를 등록하는 이유는 향 후 요구사항을 자동으로 위키에 문서로 저장 관리 해 주기 때문입니다. (not write , only read)

3.1.3.20 제품(서비스) - 요구사항 - VCS Mapping Manager (Component)

제품(서비스)의 VCS 연동 설정을 제공하는 맵핑 메니저입니다.

VCS를 통해 관련 요구사항의 하위 이슈와 연관된 브랜치와 소스의 상태를 추적하여 수치화 하는데 집중되 있습니다.

3.1.3.21 제품(서비스) - 요구사항 - Code Quality Mapping Manager (Component)

제품(서비스)의 CodeQuality 툴 연동 설정을 제공하는 맵핑 메니저 입니다.

이 경우, Issue tracker 나 VCS에서 제공되는 api 가 있다면 자동 수집하도록 하며,

연관 이슈에서 파생된 코드의 변경은 추적하여 수집하고, 이외에도 추가적으로 마스터 브랜치에 대한 코드 퀄리티를 추적합니다.

3.1.3.22 제품(서비스) - 요구사항 - CICD Mapping Manager (Component)

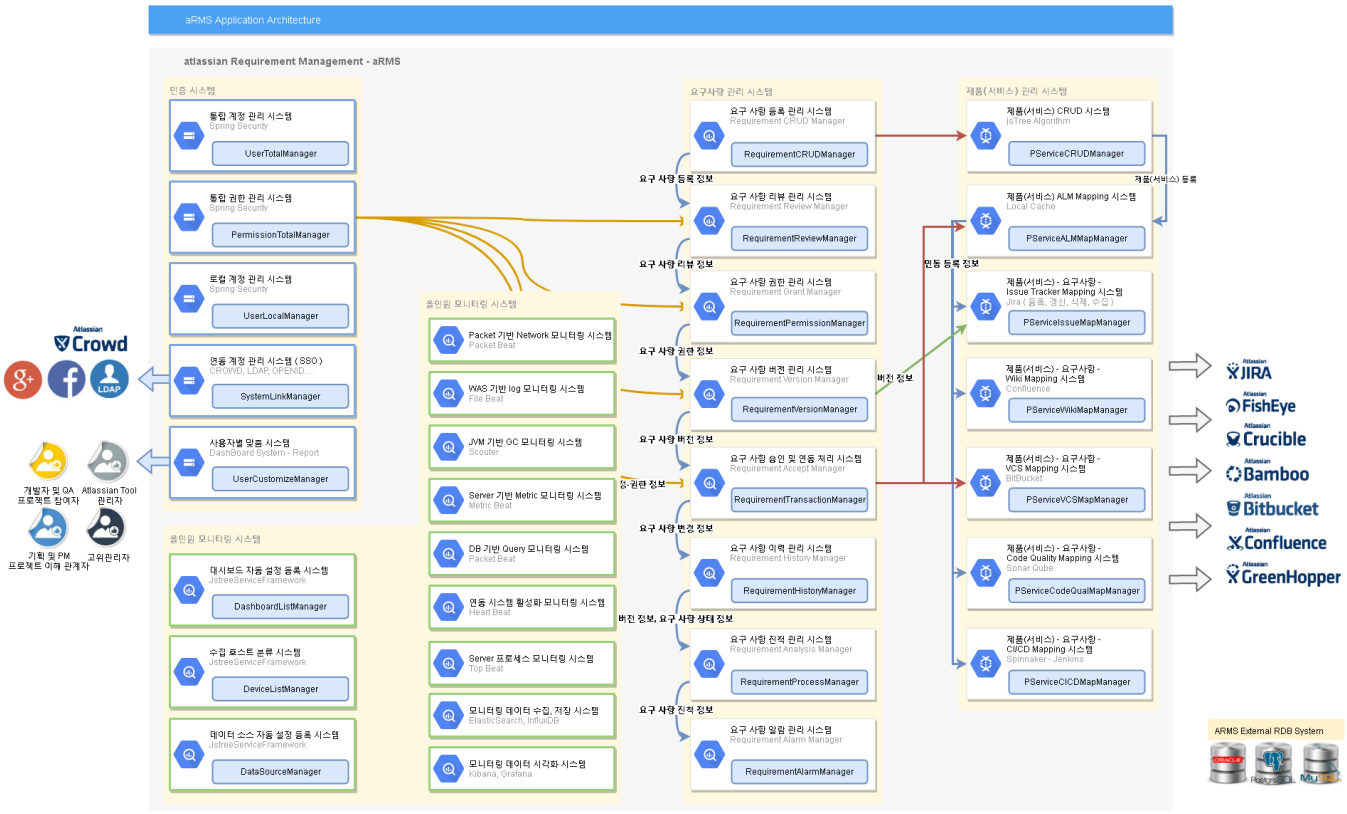
제품(서비스)의 연관된 이슈에서 커밋, 푸쉬된 VCS정보로 부터 확인 할 수 있는 CICD 빌드 결과나, 배포 현황을 볼 수 있도록 비주얼한 서비스를 제공합니다.

3.1.4 Data Design

데이터 디자인은 다음과 같이 구성합니다.

어플리케이션 아키텍처를 기반으로 구체화된 컴포넌트와 모듈이 사용하는 데이터를 설계하는 데이터 아키텍처로 갈음합니다.

(소프트웨어 Component 간에 전달되는 자료구조를 설명합니다.)



3.1.5 Interface Design

인터페이스 디자인의 전제는 시스템 정의에서 상세화된 컴포넌트의 **데이터 아키텍처가 Input 값이며,**
(어떤 데이터를 input하고 어떤 데이터를 return 할 것인지 정의되어야 한다)

컴포넌트 기준으로 사용되는 인터페이스가 상세화되는것이 인터페이스 디자인 Output이다.

3.1.5.1 Module Interface

모듈은 가장 상위에 위치하는 구현의 단위, 컴포넌트는 런타임 엔티티를 참조하는 단위라고 생각하면 됩니다. **따라서, 모듈과 컴포넌트는 상위와 하위관계를 명확히 구분짓기 어렵고 서로 다른 개념으로 바라보아야 합니다.** **쉽게 설명해서, 모듈이란 실질적으로 구현이 된 단위를 의미한다.** **반면, 컴포넌트는 실제적으로 동작하고있는 엔티티로써 활동중인 독립적인 단위를 중점적으로 보고 있다.**
객체지향 프로그래밍 언어를 사용하는 프로젝트에서는, 컴포넌트란, 잘 동작하는 기능 단위를 말하는 것이며, 컴포넌트에서 사용하는 데이터의 단위는 VO (Value Object) 또는 DTO (Data Transfer Object) 를 의미하게 됩니다. 이 데이터 단위 표시는 데이터베이스 ERD 와 연관관계를 가지게 되며, ERD 와 Mapping 관계를 가지지 않는 VO 혹은 DTO 는 Class 명세에 표시됩니다.

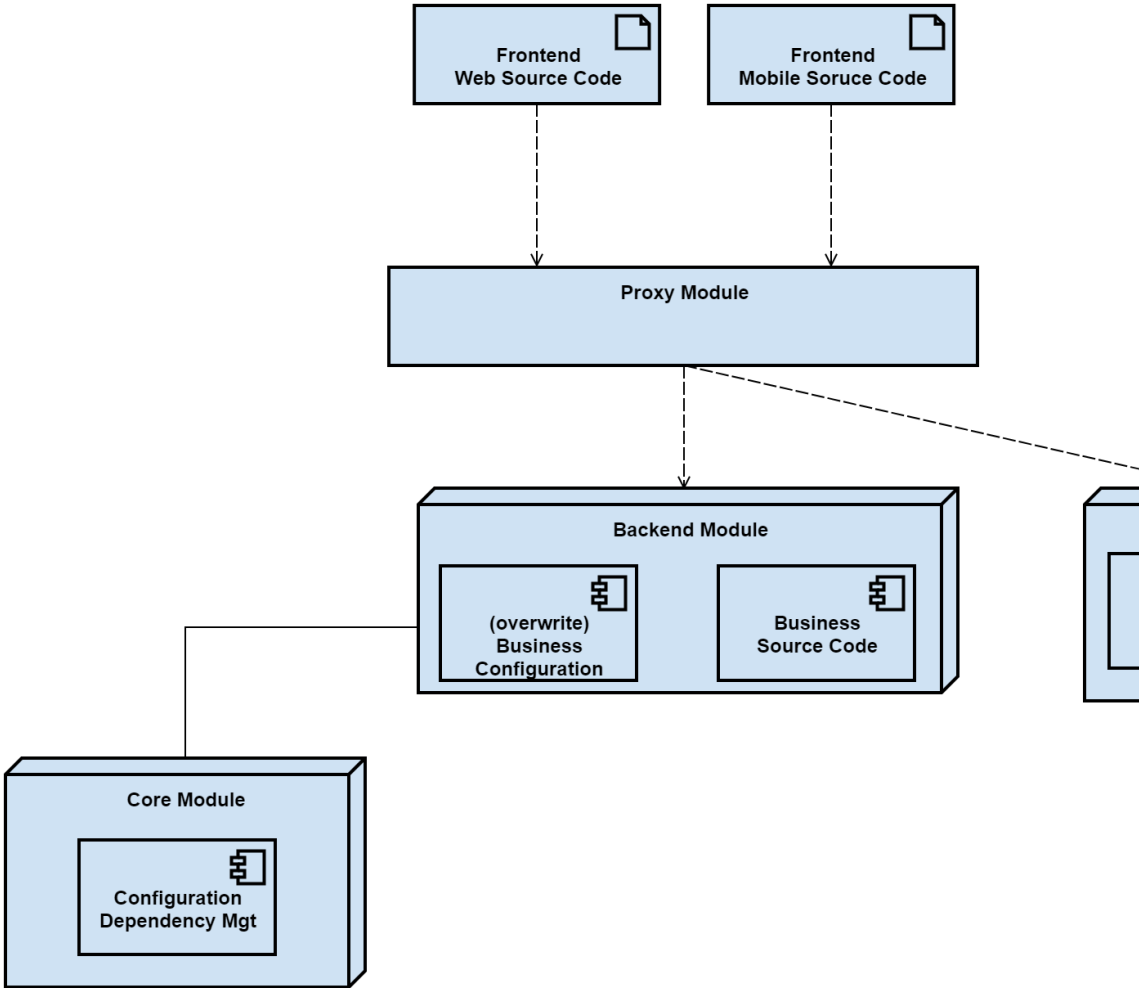
core - module source code : https://github.com/jstree/JavaServiceTree-CoreModule/blob/master/pom.xml

backend - module source code : https://github.com/jstree/JavaServiceTree-Backend-ServerProject/blob/master/pom.xml

frontend - module sorce code : https://github.com/jstree/JavaServiceTree-Frontend-WebProject/blob/master/bower.json

aRMS는 총 3개의 모듈로 구성되어 있다. (core, backend, frontend)
매우 전통적인 방식으로 구성되어 있으며, core 모듈은 dependency lib 를 관리하며, struts ~ spring, 그리고 ibatis ~ hibernate 까지 망라하는 설정을 제공하고 있으며, backend 는 business logic 에서 사용할 최소한의 properties 및 설정과 코드를 가지고 있는 구조로 되어 있으며, frontend 는 단순히 webserver에서 static file을 transfer 하는 역할에 집중되어 있다.
향 후, backend 및 각종 opensource tool을 proxy 연결하기 위하여, netflix의 zuul을 도입한 module이 추가될 예정이다.
mobile과 같은 native app은 현재 선택되어있지 않다.

따라서, module interface는 그 목적대로 모듈간의 관계를 설명하는데 있어, 그 복잡성을 가지지 않고, 전통적인 구조를 유지하는 바본 설명을 하기 module interface architecture 구성도로 대치하도록 하겠다.



3.1.5.2 Component Interface Design

3.1.5.2.1 통합 계정 관리 Manager (Component)

3.1.5.2.1.1 getUserFromAllDirectory

3.1.5.2.1.2 getUsersFromAllDirectory

3.1.5.2.2 통합 권한 관리 Manager (Component)

3.1.5.2.2.1 getPermissionFromUser

3.1.5.2.2.2 getPermission

3.1.5.2.2.3 putPermission

3.1.5.2.2.4 updatePermission

3.1.5.2.2.5 deletePermission

3.1.5.2.2.6 getAllPermissionList

3.1.5.2.3 로컬 계정 관리 Manager (Component)

3.1.5.2.3.1 getLocalUser

3.1.5.2.3.1 putLocalUser

3.1.5.2.3.1 updateLocalUser

3.1.5.2.3.1 deleteLocalUser

3.1.5.2.3.1 getAllLocalUser

3.1.5.2.4 연동 계정 관리 Manager (Component)

3.1.5.2.4.1 getRelatedAccoutTool

3.1.5.2.4.2 putRelatedAccoutTool

3.1.5.2.4.3 updateRelatedAccoutTool

3.1.5.2.4.4 deleteRelatedAccoutTool

3.1.5.2.4.5 getAllRelatedAccoutTool

3.1.5.2.4.6 connectRelatedAccoutTool

3.1.5.2.5 사용자별 맞춤 Manager (Component)

3.1.5.2.5.1 getDeveloperViewTemplate

3.1.5.2.5.2 setPerDeveloperView

3.1.5.2.5.3 getPerDeveloperView

3.1.5.2.5.4 updatePerDeveloperView

3.1.5.2.5.5 deletePerDeveloperView

3.1.5.2.6 수집 호스트 분류 Manager (Component)

3.1.5.2.6.1 Anonymous - getDeviceList

3.1.5.2.6.2 Anonymous - searchNode

3.1.5.2.6.3 Anonymous - getPaginatedChildNode

3.1.5.2.6.4 Anonymous - getNode

3.1.5.2.6.5 Anonymous - getChildNode

3.1.5.2.6.6 Anonymous - getMonitor

3.1.5.2.6.7 Anonymous - updateList

3.1.5.2.6.8 Admin - addNode

3.1.5.2.6.9 Admin - removeNode

3.1.5.2.6.10 Admin - alterNode

3.1.5.2.6.11 Admin - alterNodeType

3.1.5.2.6.12 Admin - moveNode

3.1.5.2.6.13 Admin - getChildNode

3.1.5.2.7 자동 설정 등록 Manager (Component)

3.1.5.2.7.1 updateDashBoard

3.1.5.2.7.2 updateDataSource

3.1.5.2.7.3 updateDeviceHost

3.1.5.2.8 요구사항 등록 Manager (Component)

3.1.5.2.8.1 Admin - Requirement addNode

3.1.5.2.8.2 Admin - Requirement alterNode

3.1.5.2.8.3 Admin - Requirement alterTypeNode

3.1.5.2.8.4 Admin - Requirement removeNode

3.1.5.2.8.5 Admin - Requirement moveNode

3.1.5.2.8.6 Admin - Requirement getNode

3.1.5.2.9 요구사항 리뷰 Manager (Component)

3.1.5.2.9.1 Admin - Requirement Review addNode

3.1.5.2.9.2 Admin - Requirement Review alterNode

3.1.5.2.9.3 Admin - Requirement Review alterTypeNode

3.1.5.2.9.4 Admin - Requirement Review removeNode

3.1.5.2.9.5 Admin - Requirement Review moveNode

3.1.5.2.9.6 Admin - Requirement Review getNode

3.1.5.2.10 요구사항 권한 Manager (Component)

3.1.5.2.10.1 Admin - Requirement Permission addNode

3.1.5.2.10.2 Admin - Requirement Permission alterNode

3.1.5.2.10.3 Admin - Requirement Permission alter type Node

3.1.5.2.10.4 Admin - Requirement Permission removeNode

3.1.5.2.10.5 Admin - Requirement Permission moveNode

3.1.5.2.10.6 Admin - Requirement Permission getNode

3.1.5.2.11 요구사항 버전 Manager (Component)

3.1.5.2.11.1 Admin - Requirement Version addNode

3.1.5.2.11.2 Admin - Requirement Version alterNode

3.1.5.2.11.3 Admin - Requirement Version alterTypeNode

3.1.5.2.11.4 Admin - Requirement Version removeNode

3.1.5.2.11.5 Admin - Requirement Version moveNode

3.1.5.2.11.6 Admin - Requirement Version getNode

3.1.5.2.12 요구사항 승인, 연동 Manager (Component)

3.1.5.2.12.1 Admin - Requirement Transaction addNode

3.1.5.2.12.2 Admin - Requirement Transaction alterNode

3.1.5.2.12.3 Admin - Requirement Transaction alterTypeNode

3.1.5.2.12.4 Admin - Requirement Transaction removeNode

3.1.5.2.12.5 Admin - Requirement Transaction moveNode

3.1.5.2.12.6 Admin - Requirement Transaction getNode

3.1.5.2.13 요구사항 이력 Manager (Component)

3.1.5.2.13.1 Admin - Requirement History addNode

3.1.5.2.13.2 Admin - Requirement History alterNode

3.1.5.2.13.3 Admin - Requirement History alterTypeNode

3.1.5.2.13.4 Admin - Requirement History removeNode

3.1.5.2.13.5 Admin - Requirement History moveNode

3.1.5.2.13.6 Admin - Requirement History getNode

3.1.5.2.14 요구사항 진척 Manager (Component)

3.1.5.2.14.1 Admin - Requirement Info 프로덕트(서비스) crawler - addNode

3.1.5.2.14.2 Admin - Requirement Info 프로덕트(서비스) crawler - alterNode

3.1.5.2.14.3 Admin - Requirement Info 프로덕트(서비스) crawler - alterTypeNode

3.1.5.2.14.4 Admin - Requirement Info 프로덕트(서비스) crawler - removeNode

3.1.5.2.14.5 Admin - Requirement Info 프로덕트(서비스) crawler - moveNode

3.1.5.2.14.6 Admin - Requirement Info 프로덕트(서비스) crawler - getNode

3.1.5.2.15 요구사항 알람 Manager (Component)

3.1.5.2.15.1 Admin - Requirement Alarm - configuration

3.1.5.2.16 제품(서비스) 등록 Manager (Component)

3.1.5.2.16.1 Product(Service) Manager - addNode

3.1.5.2.16.2 Product(Service) Manager - alterNode

3.1.5.2.16.3 Product(Service) Manager - alterTypeNode

3.1.5.2.16.4 Product(Service) Manager - removeNode

3.1.5.2.16.5 Product(Service) Manager - moveNode

3.1.5.2.16.6 Product(Service) Manager - getNode

3.1.5.2.17 제품(서비스) ALM Mapping Manager (Component)

3.1.5.2.17.1 Product(Service) - ALM Mapping Manager - addNode

3.1.5.2.17.2 Product(Service) - ALM Mapping Manager - alterNode

3.1.5.2.17.3 Product(Service) - ALM Mapping Manager - alterTypeNode

3.1.5.2.17.4 Product(Service) - ALM Mapping Manager - removeNode

3.1.5.2.17.5 Product(Service) - ALM Mapping Manager - moveNode

3.1.5.2.17.6 Product(Service) - ALM Mapping Manager - getNode

3.1.5.2.18 제품(서비스) - 요구사항 - Issue Tracker Mapping Manager (Component)

3.1.5.2.18.1 Product(Service) - Issue Tracker Mapping Manager - addNode

3.1.5.2.18.2 Product(Service) - Issue Tracker Mapping Manager - alterNode

3.1.5.2.18.3 Product(Service) - Issue Tracker Mapping Manager - alterTypeNode

3.1.5.2.18.4 Product(Service) - Issue Tracker Mapping Manager - removeNode

3.1.5.2.18.5 Product(Service) - Issue Tracker Mapping Manager - moveNode

3.1.5.2.18.6 Product(Service) - Issue Tracker Mapping Manager - getNode

3.1.5.2.19 제품(서비스) - 요구사항 - WIKI Mapping Manager (Component)

3.1.5.2.19.1 Product(Service) - Issue Wiki Mapping Manager - addNode

3.1.5.2.19.2 Product(Service) - Issue Wiki Mapping Manager - alterNode

3.1.5.2.19.3 Product(Service) - Issue Wiki Mapping Manager - alterTypeNode

3.1.5.2.19.4 Product(Service) - Issue Wiki Mapping Manager - removeNode

3.1.5.2.19.5 Product(Service) - Issue Wiki Mapping Manager - moveNode

3.1.5.2.19.6 Product(Service) - Issue Wiki Mapping Manager - getNode

3.1.5.2.20 제품(서비스) - 요구사항 - VCS Mapping Manager (Component)

3.1.5.2.20.1 Product(Service) - Issue VCS Mapping Manager - addNode

3.1.5.2.20.2 Product(Service) - Issue VCS Mapping Manager - alterNode

3.1.5.2.20.3 Product(Service) - Issue VCS Mapping Manager - alterTypeNode

3.1.5.2.20.4 Product(Service) - Issue VCS Mapping Manager - removeNode

3.1.5.2.20.5 Product(Service) - Issue VCS Mapping Manager - moveNode

3.1.5.2.20.6 Product(Service) - Issue VCS Mapping Manager - getNode

3.1.5.2.21 제품(서비스) - 요구사항 - Code Qaulity Mapping Manager (Component)

3.1.5.2.21.1 Product(Service) - Issue CodeQulity Mapping Manager - addNode

3.1.5.2.21.2 Product(Service) - Issue CodeQulity Mapping Manager - alterNode

3.1.5.2.21.3 Product(Service) - Issue CodeQulity Mapping Manager - alterTypeNode

3.1.5.2.21.4 Product(Service) - Issue CodeQulity Mapping Manager - removeNode

3.1.5.2.21.5 Product(Service) - Issue CodeQulity Mapping Manager - moveNode

3.1.5.2.21.6 Product(Service) - Issue CodeQulity Mapping Manager - getNode

3.1.5.2.22 제품(서비스) - 요구사항 - CICD Mapping Manager (Component)

3.1.5.2.22.1 Product(Service) - Issue CICD Mapping Manager - addNode

3.1.5.2.22.2 Product(Service) - Issue CICD Mapping Manager - alterNode

3.1.5.2.22.3 Product(Service) - Issue CICD Mapping Manager - alterTypeNode

3.1.5.2.22.4 Product(Service) - Issue CICD Mapping Manager - removeNode

3.1.5.2.22.5 Product(Service) - Issue CICD Mapping Manager - moveNode

3.1.5.2.22.6 Product(Service) - Issue CICD Mapping Manager - getNode

3.1.6 Database Description

영구적으로 사용되는 데이터는 데이터를 관리하는 시스템인 Database System에 해당합니다.

또한 데이터베이스 스키마의 데이터를 Mapping 하는 VO, DTO로 보통 표기되기에 ERD와 VO관계를 파악하는 것으로 갈음할 수 있다.

따라서, ERD - 데이터딕셔너리와 Class 다이어그램으로 갈음한다.

3.1.6.1 ERD (Entity Relationship Diagram)

> **ERD는 클래스 상세화를 통하여 어떤 데이터를 데이터베이스에 적제해야 할지 결정 한 후 작성될 수 있다.**
>
> **데이터 설계단계는 각 컴포넌트의 메소드 인터페이스에서 활용할 데이터객체 (VO, DTO)가 정의된 후 상세화 및 명세가 가능하다.**
>
> **따라서, 클래스 다이어그램이 작성된 후 프로젝트 종료시점에 적시하도록 한다. (DDL, DML : script)**

3.1.6.2 Class Diagram

프로젝트 진행 중 빌드 시 자동으로 클래스 다이어그램을 export 하여 제공할 수 있도록 합니다.

관련 링크는 다음과 같습니다 → http://www.313.co.kr/DeveloperPortal/analysis/

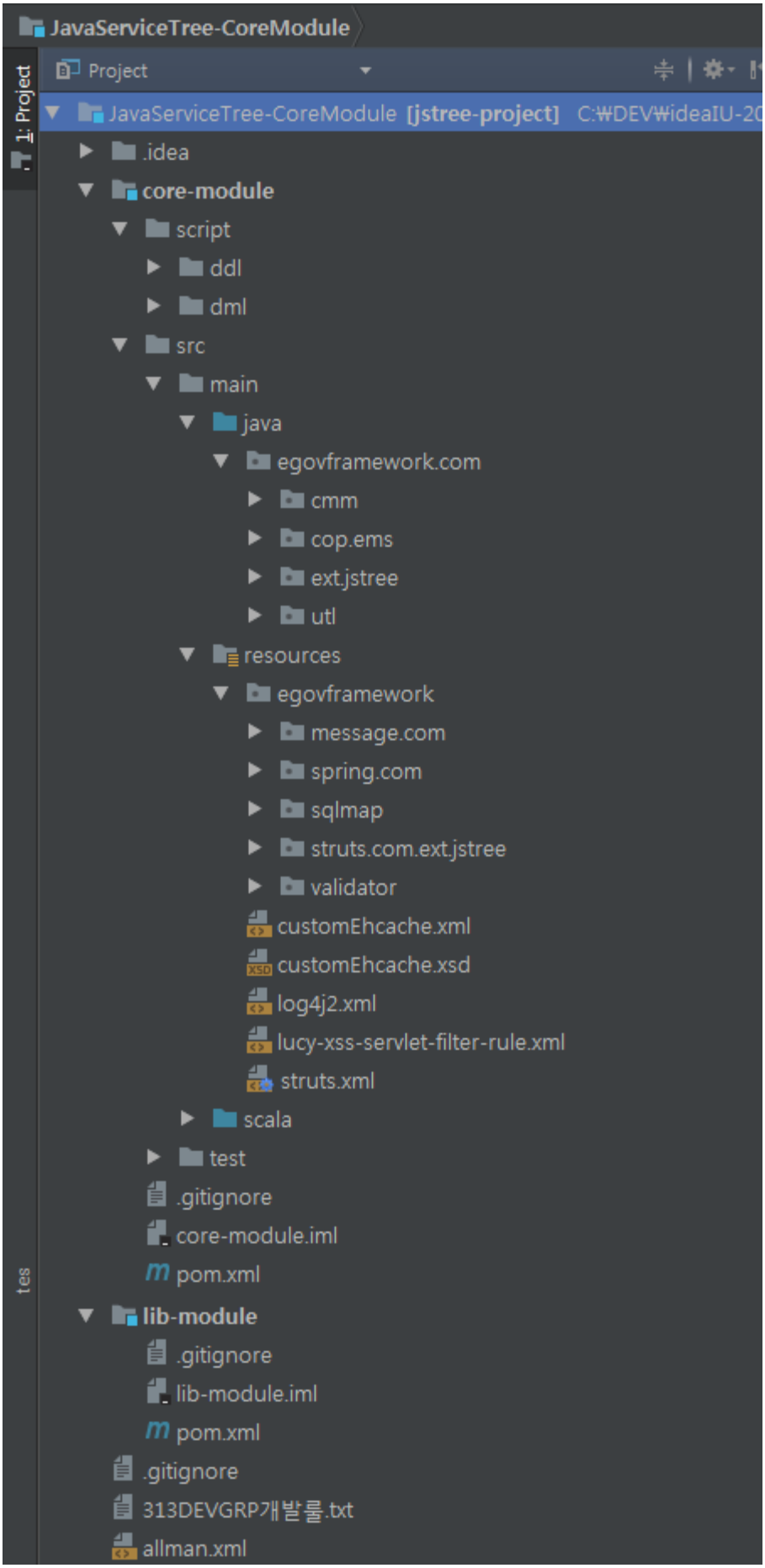
4. 구현

4.1 소스트리 구성

aRMS (Atlassian Requirement System)은 시스템의 엔트로피를 줄이고 재사용성을 극대화하기 위하여 PLE (Product Line Engineering) 기법을 사용한다.

기본 구조는 Maven 프로젝트 관례를 따르며 다음과 같은 기본 구조를 가지고 있다.

구분	설명
	Core-module script → database ddl, dml (기본 구성 셋트 등록) src/main/java → maven 전통 구조



egovframework.com → 향후 관공서 프로젝트 대응 용도 패키지

cmm → 전자 정보 표준 프레임워크 공통 모듈 3.6 기반

cop.ems → 이메일 메니저

ext.jstree → tree base 알고리즘 구현체

→ 1. spring - DWR Version

→ 2. spring - hibernate Version

→ 3. spring - ibatis Version

→ 4. struts - ibatis Version

→ 5. support util

resource → 전통적인 프로퍼티 및 설정 파일

→ message.com : 메세지 파일 (i18n 국제화 대응)

→ sping.com : 스프링 설정 파일

→ sqlmap : ibatis, mybatis 설정 파일

→ struts.com : struts 설정 파일

→ validator : 각종 dto, vo 에 대한 유효성 검사

scala → 스칼라 대응 (이후 groovy, lamda, kotlin 대응 추가)

test → tdd 대응

lib-module : dependency 분리 및 관리 모듈

pom.xml

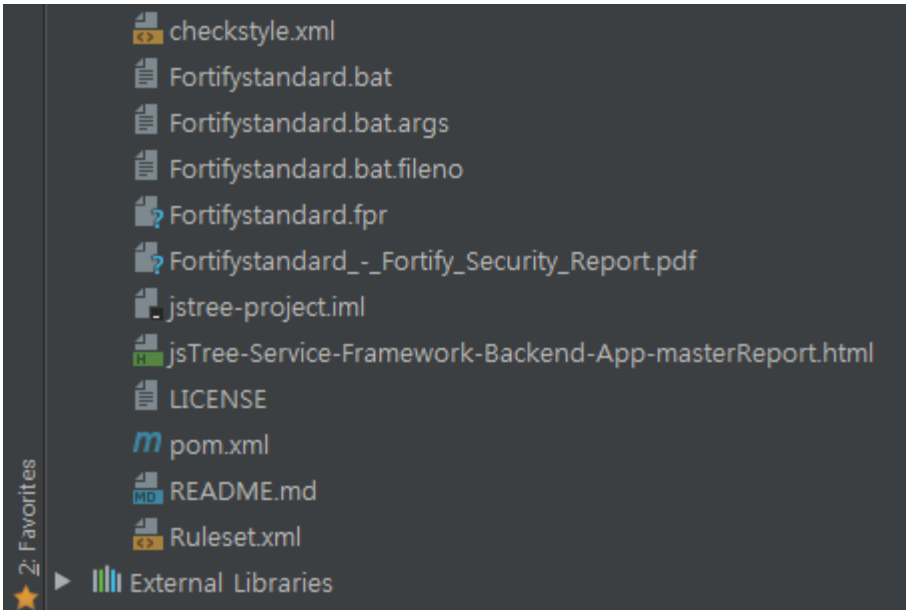
git 설정 대응 파일

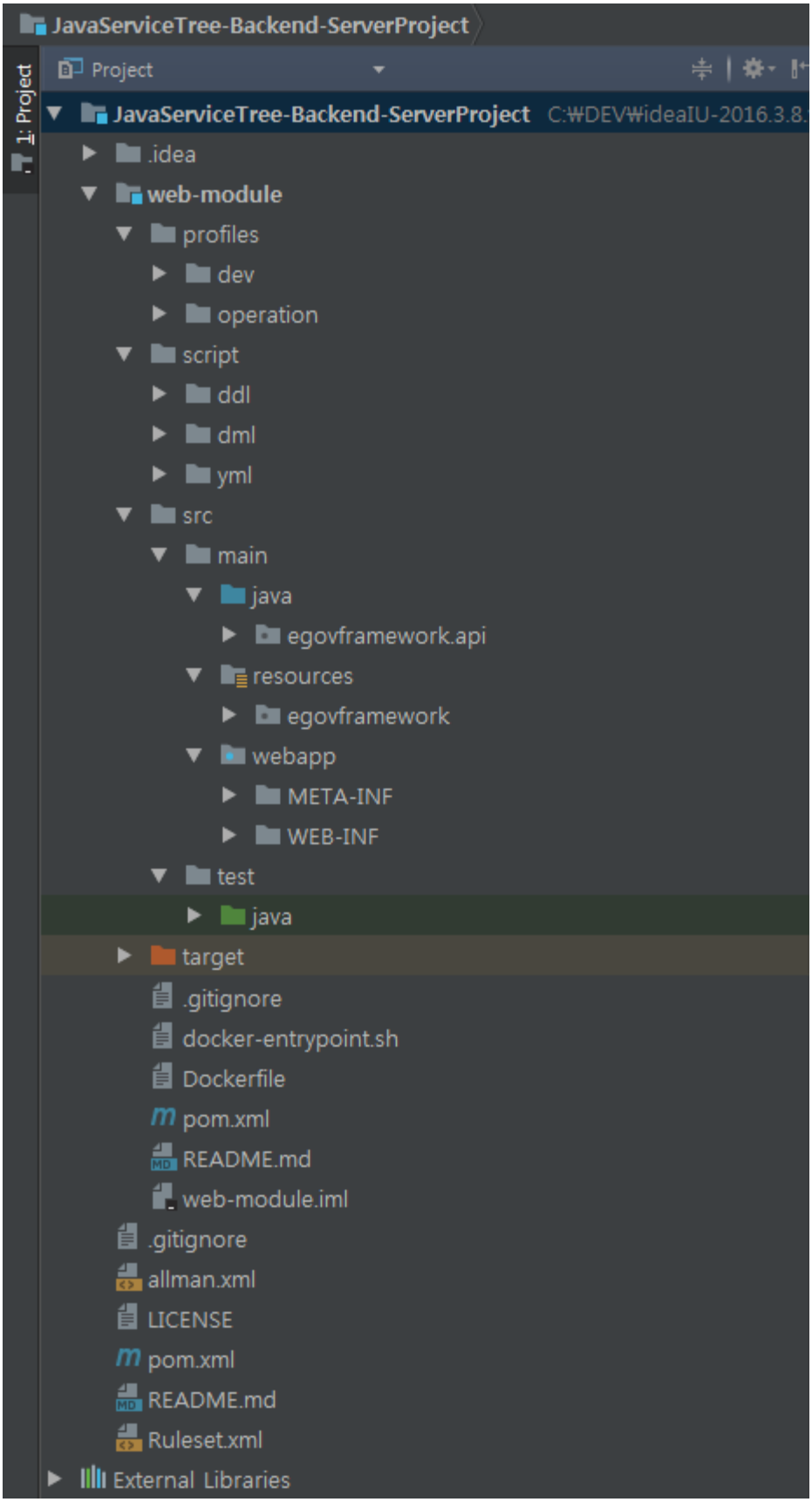
개발 룰에 대한 알림 파일

코드 스타일 검사 기준 파일

포티파이 검증 과 결과 파일

검사 룰 기준 파일


checkstyle.xml
Fortifystandard.bat
Fortifystandard.bat.args
Fortifystandard.bat.fileno
Fortifystandard.fpr
Fortifystandard_-_Fortify_Security_Report.pdf
jstree-project.iml
jsTree-Service-Framework-Backend-App-masterReport.html
LICENSE
pom.xml
README.md
Ruleset.xml
External Libraries
2: Favorites



web-module

profiles → spring active profile 별로 선택적으로 설정 파일 적용

script → database ddl, dml 및 docker swarm monitor install yml

src/main/java → 전통적인 maven 폴더 구조

egovframework.api 패키지 → 관공서 발주 대응 패키지 네임

resource → 비지니스 로직에 추가로 필요한 경우 사용

webapp → 비지니스 로직에 맞는 meta 파일 생성 사용

test/java → TDD 대응

git 관련 파일

docker image 생성 파일

git readme 파일

4.2 빌드 설계

본 프로젝트는 jstree service framework 를 기반하여 개발을 진행하고 있으며,

이미 10년전부터 지속적으로 발전시켜온 Base 이므로, 빌드 및 배포에 있어서 미리 정의된 설정을 공유하는 것으로 갈음하도록 하겠다.

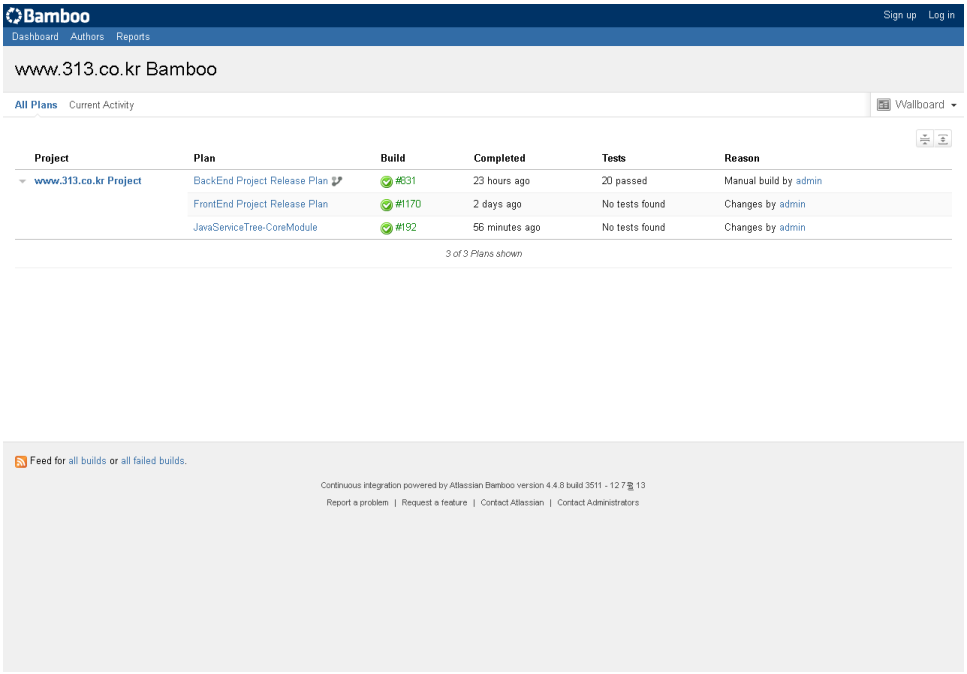
본 빌드 설계는 frontend 는 web server 로 곧바로 (마스터 브랜치 빌드를 기준으로) 배포 될 수 있도록 조치하였고,

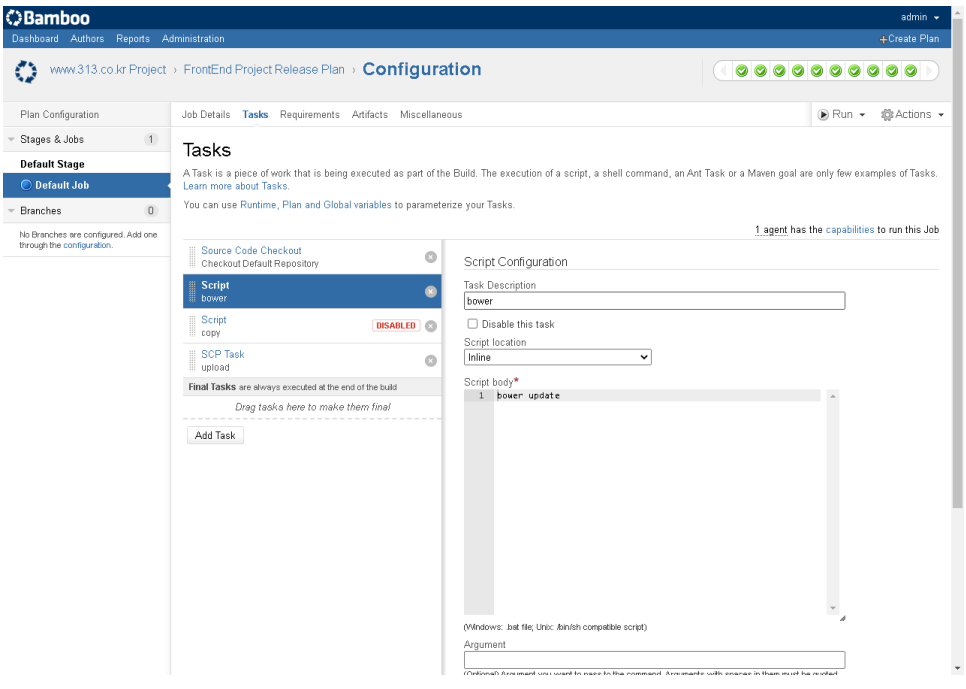
core-module 은 빌드 후 nexus (private artifact repository)로 배포 하도록 설계했다.

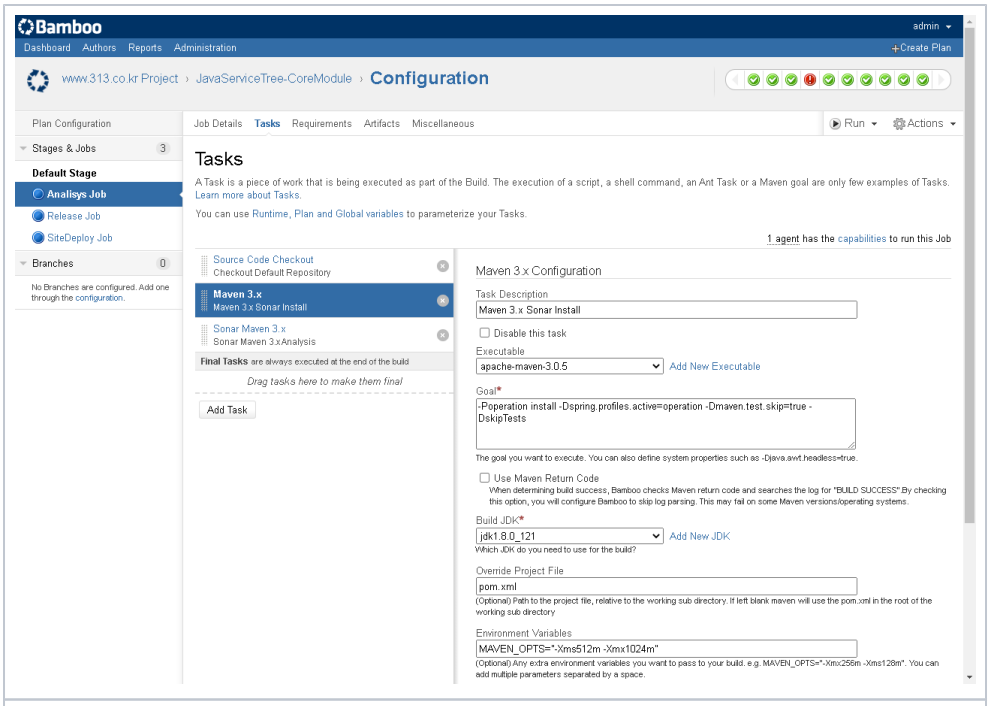
마지막으로 web-module 은 maven의 방식에 따라서 core-module을 버전에 맞도록 내려받은 후 빌드 하고

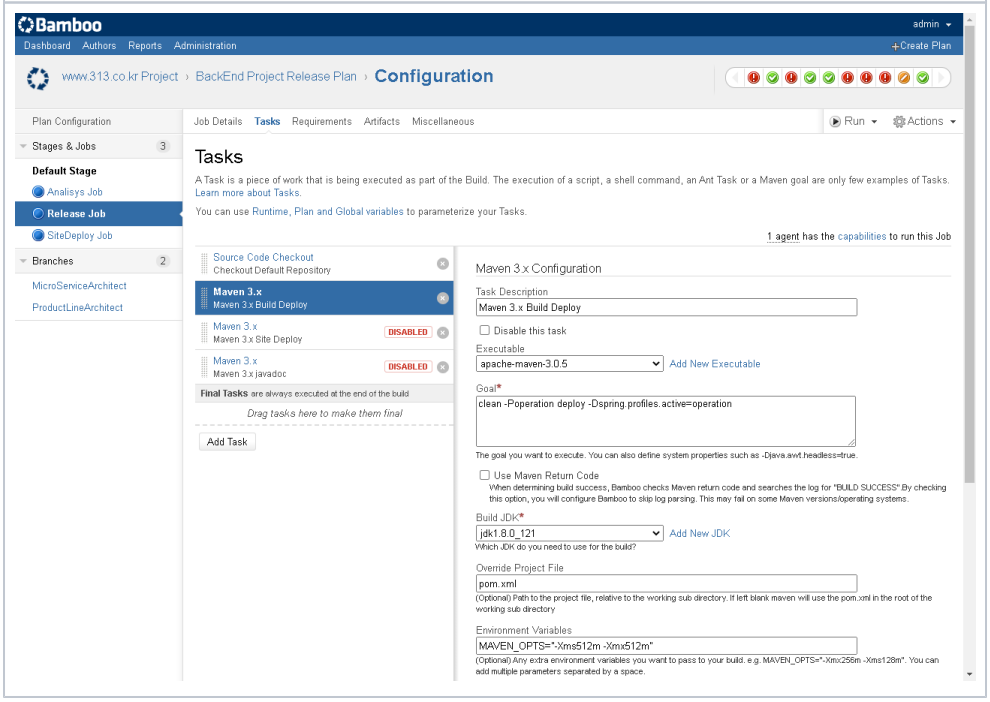
빌드 후 docker image 형태로 구성하여 docker hub에 배포 하도록 구성되어져 있다.

구분









4.3 배포 설계

4.3.1 배포 방법

본 프로젝트의 최종 artifact 는 Docker Image 형태로 배포 되도록 구성되 있으므로,

미리 정의된 docker hub 혹은 artifactory 를 구성하여, 배포 위치를 잡도록 되어 있다.

4.3.2 업데이트 방법

docker image 형태로 구성된 레파지토리로 부터 내려받고 실행하는 구조이지만,

실제 배포 타겟이 k8s 혹은 docker swarm 같은 클러스터 형태라면, jenkins 혹은 spinnaker 를 사용하게 될 것이다.

따라서, 각 환경에 맞도록 파이프라인 툴의 명령을 구성하면 되겠다.