|  |
| --- |
| **2024년 공개SW 개발자대회 결과보고서\_자유과제** |

|  |  |  |  |
| --- | --- | --- | --- |
| **구 분** | **세 부 내 용** | | |
| **팀 명** | 313DEVGRP | **팀 인 원**  **(팀장 포함)** | 9 |
| **부 문** | 일반 | **과제 유형** | 자유 |

**□ 결과보고서**

|  |  |
| --- | --- |
| **프로젝트명** | A-RMS |
| **프로젝트 등록 URL** | <https://www.a-rms.net>  <https://github.com/313devgrp> |
| **시연영상** | *https://youtu.be/intYQUMlqho?si=L32dUDYyNs\_S5KKm* |
| **프로젝트 소개** | **ALM 통합. 요구사항 관리 시스템** |
| **프로젝트 세부 내용** | |
| **개발배경 및 목적** | **프로젝트 개발 배경** :  **“요구사항은 관리의 대상이지만, 업무에 연계되어 정보가 연계되지 않는다”**  **“요구사항을 ALM에 연결해서 업무에 활용할 수 있는 TOOL 을 만들자“**  → 1. 요구사항을 발제 한 사람과 구현하는 사람이 다르다.  → 2. 요구사항 발제자는 제품(서비스)를 기획했지만, 개발자는 프로젝트를 생각한다.  → 3. 요구사항 발제자는 제품(서비스)를 문서로 만들었지만, 개발자는 Issue를 만든다.  → 4. 요구사항 발제자와 개발자간의 이해 간극을 좁히는 도구가 없다. ( 커뮤니케이션 X )  → 5. 요구사항 발제자의 변경 이력을 관리하고 요구사항을 전달하는 도구가 없다. ( 이력 X )  → 6. 요구사항 기반으로 제품(서비스)를 구현하지 않는다. ( ISSUE 는 개발자가 만들기 때문이다 )  → 따라서, 요구사항 기반의 ISSUE 추적성도 확보하지 못하고,  → 따라서, 요구사항 기반의 제품(서비스) 기반의 진척도(가시성)를 확인 할 수 없다.  “요구사항 발제자는 제품(서비스)를 기획했지만, 개발자는 프로젝트를 구현한다.” 이것이 문제다.  (첨부1)은 과거부터 전해오는 요구사항 관리 대장이며, 이 관리는 “관리”에 목적이 있었으며, 요구사항 정의서 및 기술서 관리 대장(첨부2)는 “보고”에 목적이 있는 문서다.  개발자는 이 문서에 의존하여, Issue 를 생성하고 개발 업무를 수행해야 한다.  바꿔야 한다. → 요구사항 기반으로 이슈가 생성되고 관리되고 보고되는 일원화된 체계가 필요하다.      **프로젝트 개요** :  이런 문제를 해결하기 위하여, 이슈를 기반한 요구사항 관리 시스템이 없으니 개발하여, 제품별 요구사항 상태( 추가-채택-변경-삭제 ) 및 이력을 관리하여 추적성을 확보하고 ALM 시스템과 통합하여 제품(서비스) 기반 프로젝트 진척도를 공유 할 수 있는 가시성을 제공하는 RMS (Opensource 기반 Requirement Manage System)시스템을 만들고자 합니다.    **프로젝트 목표 :**  → 1. 제품(서비스)를 등록하고  → 2. 제품(서비스)의 버전을 등록하여,  → 3. 수집된 Issue Tracker ( Jira, RedMine 등 ) 의 Project 와 연결  → 4. 제품(서비스) 기반의 요구사항을 등록하면  → 5. 연결된 Issue Tracker 의 Project 에 요구사항 이슈를 생성  → 6. 개발자는 요구사항 이슈 하위로 Issue 를 생성하여 개발을 진행  → 7. 하위 Issue 를 수집하여 요구사항기반의 진척도를 표현  C:\Users\Administrator\Desktop\이미지 2.png |
| **개발환경**  **(언어, Tool, 시스템 등)** | **프로젝트 개발 언어 :**  - View Part  html+css, Bootstrap, RequireJS, Bower, Grunt, jQuery, React, Vite, Axios  - Server Part  Java  - Framework Part  Spring, iBatis, Hibernate, Spring-integration, Spring-security, Spring-Boot, Spring-Data(JPA), Spring-Batch, Spring-MVC 등  기타 ( 자체 제작 Java Service Framework )  - Tool Part  Quartz, Redis, Apache-Commons, EgovFramework(Component)  - CI Part  Junit, Maven, Jenkins, Nexus, Sonarqube, Spinnaker, ArgoCD, Jira, Fisheye, Crucible, Confluence, Redmine, GitLAB  - Database Part  MySql, Elasticsearch, Hadoop, Sqoop, Spark  - Mobile Part  Android, PhoneGap  - Search Engine Part  Lucene, Elastic Search, Kibana, Logstash, Beats  - Management Part  PMBOK, MicroService, CBD, PLE, Prototype, PMS, ALM  - Virtual Image Part  Docker Swarm Cluster, Kubernetes  - Microservice Part  Netty, Zookeeper, Finagle  - IAC  Ansible,. AWX  **프로젝트 개발 환경 및 아키텍쳐 :**    **프로젝트 개발 툴(TOOL) 시스템 :**  [Requirement Manage] <http://www.a-rms.net>  [Document] <http://www.313.co.kr/confluence>  [IssueTracker] <http://www.313.co.kr/jira>  [GitLab] <http://www.313.co.kr/gitlab>  [Redmine] <http://www.313.co.kr:3000/>  [VersionControl] <https://github.com/313devgrp>  [CodeReview] <http://www.313.co.kr/fecru>  [CICD-Deploy] <http://www.313.co.kr/jenkins>  [BuildManager] <http://www.313.co.kr/spinnaker>  [ArtifactManager] <http://www.313.co.kr/nexus>  [CodeAnalysis] <http://www.313.co.kr/sonar>  [Docker Hub] <https://hub.docker.com/u/313devgrp>  [Kubernetes] <http://www.313.co.kr:31380>  [DockerSwarm] <http://www.313.co.kr/portainer/#!/auth>  [DB Admin] <http://www.313.co.kr/phpmyadmin/>  [S3 Admin] <http://www.313.co.kr/minio/login>  [MEM Admin] <http://www.313.co.kr/redis/>  [NoSql Index] <http://www.313.co.kr/elasticsearch/_cat/indices>  [NoSql Node] <http://www.313.co.kr/elasticsearch/_nodes?pretty=true>  [Kibana] <http://www.313.co.kr/kibana/app/home#/>  [LogStash] <http://www.313.co.kr/logstash/_node/stats?pretty>  [ZipKin] <http://www.313.co.kr/zipkin/>  [ElasticHQ] <http://www.313.co.kr/elastichq/#!/>  [Grafana] <http://www.313.co.kr/grafana>  [Kafka] <http://www.313.co.kr/kafka>  [NAS] <http://www.313.co.kr:5000/webman/index.cgi>  [Mail] <http://www.313.co.kr/mail/>  [Auth] <http://www.313.co.kr/auth/>  [RDP] <http://www.313.co.kr/guacamole/#/>  [DrawIO] <http://www.313.co.kr/drawio>  [DrawDB] <http://www.313.co.kr/drawdb> |
| **시스템 구성 및**  **아키텍처** | **프로젝트 비지니스 아키텍쳐 :**  C:\Users\Administrator\Desktop\이미지 9.png  전체적인 아키텍쳐의 흐름과 기술 구성요소, 그리고 비지니스에서 필요로하는 중심 기능을 차트로 표현하는게 비지니스 아키텍쳐 입니다.  비지니스 아키텍쳐를 기반으로 달성하고자 하는 목적 시스템 구성과 배치를 확인 할 수 있습니다.   * 다음 페이지 참조   C:\Users\Administrator\Desktop\이미지 10.png  **프로젝트 어플리케이션 아키텍쳐 :**  시스템 아키텍쳐를 기반으로 구체화된 어플리케이션 아키텍쳐로 본 장을 갈음하도록 합니다.  어플리케이션 아키텍쳐로 각 시스템의 세부 컴포넌트와 모듈을 표시하고 상호 작용하는 목적이 기술되어 있기 때문입니다.    C:\Users\Administrator\Desktop\이미지 5.png  C:\Users\Administrator\Desktop\이미지 7.png  **프로젝트 데이터 아키텍처 :**  데이터 디자인은 다음과 같이 구성합니다.  어플리케이션 아키텍쳐를 기반으로 구체화된 컴포넌트와 모듈이 사용하는 데이터를 설계하는 데이터 아키텍쳐로 갈음합니다.  ( 소프트웨어 Component 간에 전달되는 자료구조를 설명합니다. )  C:\Users\Administrator\Desktop\이미지 8.png |
| **프로젝트 주요기능 및 구조도** | **프로젝트 주요 기능 구조도 :** |
| **기대효과 및**  **활용분야** | **프로젝트 기대 효과 :**            요구사항 – ALM 의 연관관계를 설정하여, 정보를 연계, 좀 더 명확한 이해와정량적 지표를 기반으로 하는 프로젝트는 성공을 기대할 수 있습니다. |
| **기타**  **(프로젝트 추가 설명 등)** | **프로젝트 설계 목표 :**  Frontend-Web : 모바일까지 지원하는 RMS 웹 프론트엔드 어플리케이션  Middle-Proxy : API 인증, 인가, 라우팅 시스템을 지원하는 Spring Proxy 어플리케이션  Backend-Core : ARMS API 를 처리하고 데이터 적재를 담당하는 어플리케이션  Engine-Fire : 각종 Issue Tracker 에 요구사항 이슈를 전달하고 수집하는 어플리케이션  **프로젝트 개발 아키텍쳐 :**  Java Service Tree Framework를 활용하여 Tree 알고리즘 기준의  Requirement Management System 을 설계합니다.  PLE ( Product Line Engineering ) 기법을 활용하여, 재사용을 극대화한 프로젝트 구조를 적용합니다.  - aRMS는 자체적인 Static Code Analysis ( SonarQube )를 적용하여 코드 품질을 유지하도록 합니다.  - aRMS는 Github을 활용하여 Git으로 형상관리를 적용하도록 합니다  - aRMS는 Frontend 개발과 MiddleProxy, Backend 개발을 분리합니다.  - aRMS는 손쉬운 패키지 업데이트를 지원하도록 Docker 로 배포합니다.  - aRMS는 손쉽게 확장 운영을 하도록 스케일 아웃을 지원합니다.  - aRMS는 Source Write after Deploy 까지 One Shot Flow CI/CD를 제공합니다.  C:\Users\Administrator\Desktop\이미지 4.png                  **모니터링 구성 방안 :**    **요구사항과 ALM 간의 연계에 대한 학술적 근거**  소프트웨어 제품 개발 프로젝트 환경에 기초한 PMS사용이 정량적 프로젝트 관리 지표 관리에 관한 연구 |