

AhnLab, Inc.

# ARMS ( Atlassian RMS )

iQx R&D 신청서

**WARNING**

본 문서 및 문서에 포함된 정보는 안랩의 자산입니다. 안랩의 허가없이 사용/재가공/반출할 수 없습니다. 이 문서의 열람 및 보관할 수 있는 사람은 분실, 도난, 반출에 대한 책임을 자동 승계하는 것에 동의하여야 합니다. 동의할 수 없으면 문서를 열람 및 보관하지 마십시오.

**문서 정보**

파일명	문서 1
문서구분	일반문서
기밀등급	AhnLab Internal Use Only
보존기간	3 년

**문서 변경 이력**

날짜	작성자	버전	리비전	변경 내용

## 목차

1	배경 .....	5
2	내용 .....	10
2.1	기존 기술 .....	10
2.2	추진 과제 .....	11
2.3	기술의 구성 .....	12
3	활용 방안.....	15
3.1	기대 효과.....	15
3.2	한계점.....	15 오류! 책갈피가 정의되어 있지 않습니다.
4	실행 계획 .....	21
5	관련 참고 사항 .....	21

## 과제 요약

과제명	제품명 : ARMS ( Atlassian RMS ) 부재 : 성공과 실패를 결정하는 요구사항 관리 및 이슈 기반 진척도 관리 시스템		
과제형태	<b>어떤 제품인가?</b> 이슈 기반 요구사항 관리 시스템을 개발하여, 제품별 요구사항 상태(추가-채택-변경-삭제 ) 및 이력을 관리하고 ALM 시스템과 통합하여 제품 기반 프로젝트 진척도를 공유 할 수 있는 RMS (for atlassian)시스템 입니다.		
목표설정	목표수준	1 차 : 사내 RMS 적용 및 운영 ( beta ) 2 차 : 상용화 제품 패키징 및 배포	
	기존수준	N/A	
	산정근거	N/A	
예상기간	1 차 범위 : 8 개월 ( Web, Mobile Web, Server Api Server ) 2 차 범위 : 4 개월 ( Mobile Native App )		
최종결과물	<b>어떤 형태의 제품인가?</b> Desktop 웹 서비스용 Frontend Application Mobile 서비스용 Frontend Application ( Web app, Native App ) Server Api 서비스용 Backend Application		
활용계획	하기 기재		
참여자	이름	소속 부서	지분율
	이동민	연구혁신팀	100%

### (※ 순서 )

문제점 요약 > 사내 인터뷰 내용을 통한 문제점 확인 > ALM 기준 문제점 해결 조명 >  
 ARMS 를 사용했을 때 모습 > ARMS 의 시스템 구성 > ARMS 적용 시 해결 효과

## 1 배경

### (※ 문제점 요약)

1. 요구사항 이력 관리가 되지 않는다. ( 등록, 채택, 거절 등 변경이력 )
2. 제품별 요구사항 Mapping 이 어렵다. ( WORKS 는 프로젝트 기반 not 제품 기반 )
3. 제품별 진척상황을 쉽게 볼 수 없다.

### 제안 배경 #1

#### 사내 : 인터뷰 내역

기획을 위한 사내 인터뷰 내역 (QA 팀장님 인터뷰 내용 하기 별첨 )

구 분	기획 내용을 설명 후 RMS 적용에 대한 의견
개발자 A	Version 관리는 이미 잘하고 있다.고 생각한다, JIRA 에서 이슈를 요구사항에 맵핑만 하면 되는거라면 어렵지 않다.
기획자 B	요구사항 픽스 및 변경에 대한 권한관리가 필요하다. ( ex: 요구사항 변경은 기획자 관리 영역 )
영업 담당자 C	docs 에 요구사항을 리뷰 한 후 우리는 일일이 개발팀장에게 진척상황을 물어본다. 매우 필요하다고 생각한다
개발자 리더 D	진척상황에 대한 보고자료를 만드느라 시간을 많이 소모한다. 이 시스템은 매우 필요하다고 생각한다
고위 관리자 F	꼭 필요한 제품이라고 생각한다. 늘 필요하다고 요구했지만 마땅한 대안이 없어서 아쉬웠다.

### 제안 배경 #2

#### 우리의 문제점

( 인터뷰로 보는 ) 우리의 문제점은 이렇습니다.

( 아틀라시안의 문제점이기도 하고, ROSE 의 문제점 이기도 합니다 )

- ① SRS 에 적시되는 요구사항을 누가 작성하나요?  
( > 개발자가 쓰고 있습니다. )
- ② 요구사항을 누가 발의하나요?  
( > 고객이나, 고객과 면대면을 업무로 하는 사람들입니다.  
또는 기획자, 내부 개발자 입니다. )
- ③ 요구사항에 대한 이력은 관리되나요?  
( > 관리되지 않습니다.  
변경이력은 SRS 에 기록하지만 최종 형상만 확인됩니다. )
- ④ 이 요구사항을 발의한 사람들이 요구한대로 잘 만들어지고 있는지 확인할 수 있나요?  
( > 알수 없습니다.  
개발팀장의 해석으로 개발 진행 척도를 전달 받습니다. )

(※ 문제점 요약)

**Atlassian 제품 군에 RMS 를 부재 > 요구 사항 관리 부재**

1. 요구사항 기반 이슈 트래킹 불가능 (추적성 부재)
2. 요구 기반 이슈 트래킹 불가능으로 제품 기반 진척 사항 확인 불가능 (가시성 부재)

**제안 배경 #3****동일환경 : 외부사례**

( 외부 사례로 보는 ) 문제가 무엇인가?

- ① 오픈소스 커뮤니티를 만으로 9 년정도 운영하고 있습니다.
- ② 오픈소스의 원활한 개발을 위하여 지금의 ROSE 구성 초창기 버전인 Atlassian ALM Tool Chain 을 도입하였습니다.
- ③ 나름대로의 개발 프로세스를 오픈소스 커뮤니티에 정착시켰다고 생각했지만,
- ④ 문제가 있었습니다.
- ⑤ 문서도 잘 쓰고, 이슈도 잘 만들고 , 리뷰도 열심히 하고, 빌드도 자동화하고, 코드 품질도 자동화하여 개발 진행은 완벽했지만 요구사항 관리가 되지 않고 있었습니다.
- ⑥ ( 아틀라시안에는 RMS 가 없었고, 다른제품은 독자기능으로 ALM 을 구성하고 있었습니다 - 그동안 요구사항은 대충 문서로 관리? 하고 있었습니다. )
- ⑦ 요구사항이 관리되지 않는 문제와 파생 문제는 상당했습니다.
- ⑧ 요구사항을 기록 및 정리한 내용이 문서로만 머물러 있고
- ⑨ 요구사항 ~ 이슈 mapping 이 되지 않으니, 이 이슈가 왜있지? < 매우 중요합니다.
- ⑩ ( 어느 요구사항 때문에 이 일을 해야 하는구나를 알지못하고 그냥 일(TASK)를 하기 시작합니다 : 아이디어가 없죠 )
- ⑪ 변경된 요구사항은 전파력도 낮았고, 관련자 협의가 되지 않아서 엉뚱하게 개발되는 경우가 많았습니다.
- ⑫ 또한 등록된 이슈들은 요구사항과의 연관관계 없이 나열되어 버전별로 어떤 요구사항을 수렴하여 개발이 진척되고 있는지 알수가 없습니다.
- ⑬ 때문에 " 언제쯤, 어떤 요구사항이 반영된, 제품이 얼마나 진척됐는지" 명쾌하지가 않습니다.

(※ 문제점 요약)

**Atlassian 제품군을 사용시 대부분 겪게 되는 문제점.**

## 제안 배경 #4

## Atlassian 제품 기준

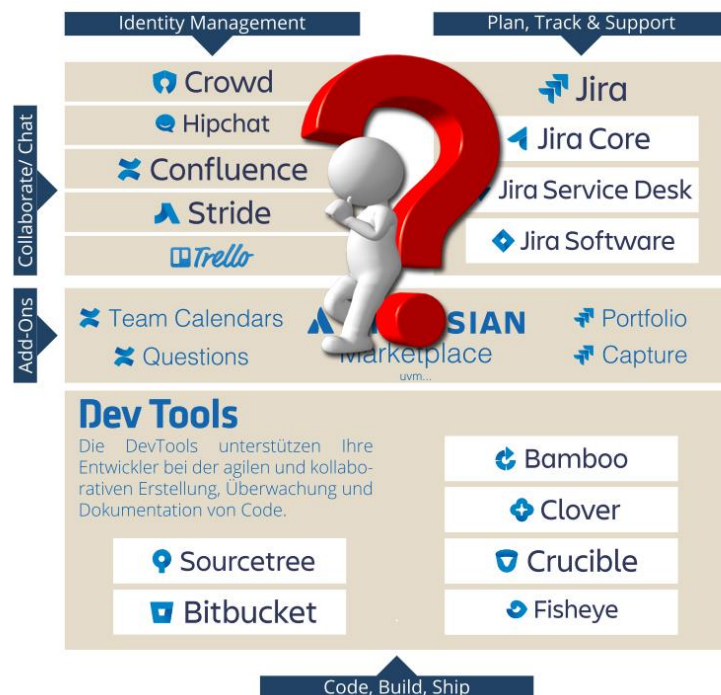
## RMS 부재로 인한

## 문제점 확인

그렇습니다. 아틀라시안 솔루션에는 없습니다.

- ① 요구사항을 관리할 뚜렷한 솔루션이 없습니다.  
( 몇 개의 플러그인이 있지만 요구사항 관련자들이 사용하기 번거롭습니다.  
요구사항을 채택, 변경이력, Jira 는 프로젝트 기준이기 때문. )
- ② 요구사항부터 시작되는 혹은 변경되는 요구관리를 적극적으로 반영할 ALM Point 가 없습니다.
- ③ 따라서, 개발자가 SRS 로 요구사항 리뷰를 받고 난 후, 변경관리는 개발팀장의 커뮤니케이션 빈도에 따라 달라집니다.
- ④ 요구사항을 발의한 사람들은 개발팀장에게 진척상황을 일일이 문의해야 합니다.
- ⑤ 네 그렇습니다. 개발 진척을 확인 할 방법이 마땅히 없습니다.

( ※ Atlassian 제품 라인업 : 출처 : 공식 홈페이지 )



#### • 추적성 부재

요구사항의 시작점 없이 바로 이슈와 문서를 만들어서 개발 Flow 를 진행하는 제품의 흐름 때문에 결국 제품의 요구사항 정보는 추적 불가능.

#### • 가시성 부재

추적성 부재로 인하여, 역으로 제품을 기준하여 ALM 의 여러 정보를 역으로 추적하여 제품의 진척도와 같은 프로젝트 가시성을 확보하지 못합니다.

## ALM 관점에서 본 문제점 확인 Check Point

ALM 정의는 원래 이렇습니다.

- ① ALM 은 어플리케이션 라이프사이클 메니지먼트의 약자입니다.
- ② ALM 의 **제 1 핵심**은 어플리케이션이 개발되는 전체 과정에 대한 **추적성**이 핵심입니다.
- ③ 추적성이란. 요구사항이 채택되고 이슈를 생성하는 것부터 시작해서 리뷰하고 빌드되고 아티팩트가 전달되는 일련의 과정을 확인 할 수 있음에 있습니다.
- ④ ALM 의 **제 2 핵심**은 추적성을 기반으로 프로젝트의 상태를 확인하는 **가시성**이 핵심입니다.
- ⑤ **ARMS 는 ALM 의 시작이고 그 끝입니다**

( ※ ALM 도식 : 출처 : PLM : 폴라리온 )



### • 요구 관리의 중요성

시작이 없는데 아무리 좋은 개발 프로세스가 있더라도 제품의 성공은 보장하기 어렵습니다.

## 타사 사례 분석

- ① 요구사항을 관리하기 위한 대장을 관리합니다.
- ② 채택과정을 명시화 하고, 변경이 발생했을때 이력을 관리합니다.
- ③ 문서로 관리하다보니 작성의 공수가 있고, 제품과 프로젝트의 가시성을 제공하는 자동 시스템이 없습니다. ( 기회 )



번호	위치	요구ID	구분	요청사항	설명
----	----	------	----	------	----

(※ 오그라들지 않아서 미 기수서 관리 대상·추진·조사 )

姓名	学号	院系	姓名	学号	院系
----	----	----	----	----	----

---

나. 사태( 추가-채태-변경-사제 ) 및 이력은 과리하여 추

## 2 내용

### 2.1 기존 기술

#### Atlassian 제품군

#### 요구사항 관리 툴 조사

Atlassian 제품과 적어도 connector 를 제공하는 솔루션.

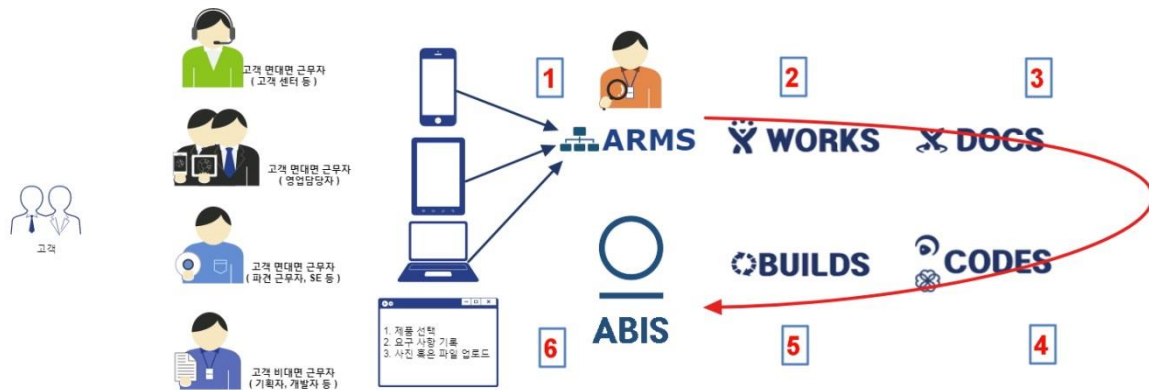
요구사항과 관련한 Atlassian 입장 : [문서 바로가기](#)

솔루션 명	WEB 링크
Modern Requirements	<a href="#">바로가기</a>
Visure	<a href="#">바로가기</a>
reqtest	<a href="#">바로가기</a>
doors	<a href="#">바로가기</a>
jama	<a href="#">바로가기</a>

- ✓ 현재 5 가지 종류의 RMS 는 아틀라시안과 분리된 체 독자적인 제품을 구성하는 경우에 한합니다. ( 당연하겠죠 ).
- ✓ 하지만 이 제품들은 ALM 의 라이프사이클에 녹아들지 못하는 형태로 구성되어, 연관성 및 관리성 그리고 일관성에 문제가 있습니다. 그리고 비쌉니다.
- ✓ 한마디로, 아틀라시안 제품군과는 전혀 다른 UI 및 구성을 가지고 있기때문에 활용성 측면도 현저히 낮은 상태입니다.
- ✓ 추가로, 솔루션에서는 제품 기준 ( 이슈 기반 ) 진척사항을 확인하는 DashBoard 는 제공하지 않습니다.

## 2.2 추진 과제

### ARMS 도식화 모형



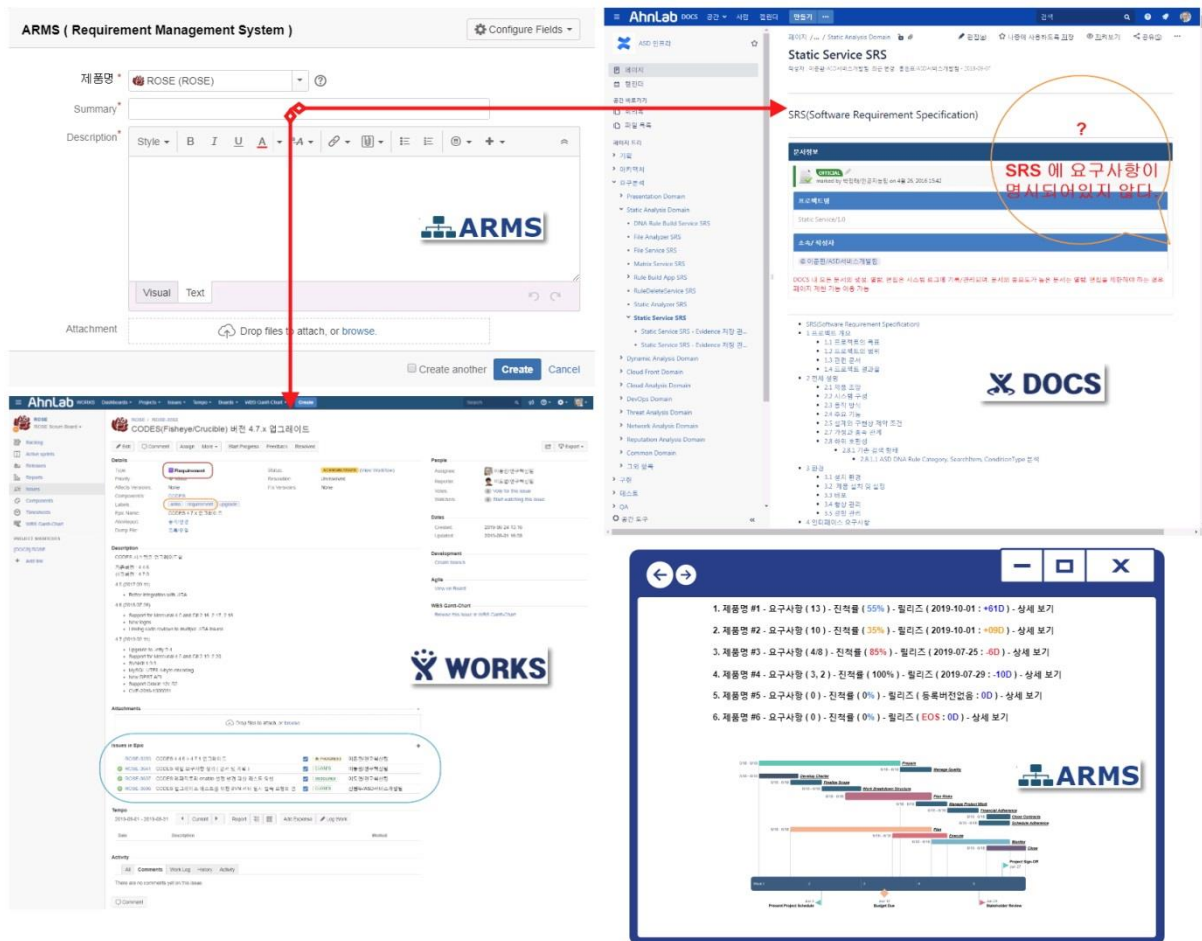
※ ARMS 적용 후 ALM (ROSE) 단계별 설명

구분	단계	설명
1	요구 사항 관리	요구사항을 수집하고 정립 프로젝트 별 이슈 기반 추적 매트릭스를 통한 제품의 가시성 (Dashboard) 관리
2	이슈 관리	요구사항 기반 이슈 생성 및 관리 ex: 요구사항 정립부터 코드 개발에 이르기까지의 전 과정 이슈 추적 관리
3	문서 관리	프로젝트 문서를 온라인 화 및 관리 지식 저장소의 역할
4	개발 관리 코드 품질, 리뷰	온라인 리뷰 관리
5	CI/CD 관리	코드를 빌드하고 (자동화된 Unit Test, 정적 분석 및 자동화된 프로젝트 문서 산출) 아티팩트 생성
6	릴리즈 관리	아티팩트 공유 및 배포 레파지토리 관리

ARMS를 적용 후 ALM (ROSE)은 이런 모습입니다. ( 하기 프로토타입 이미지 첨부 )

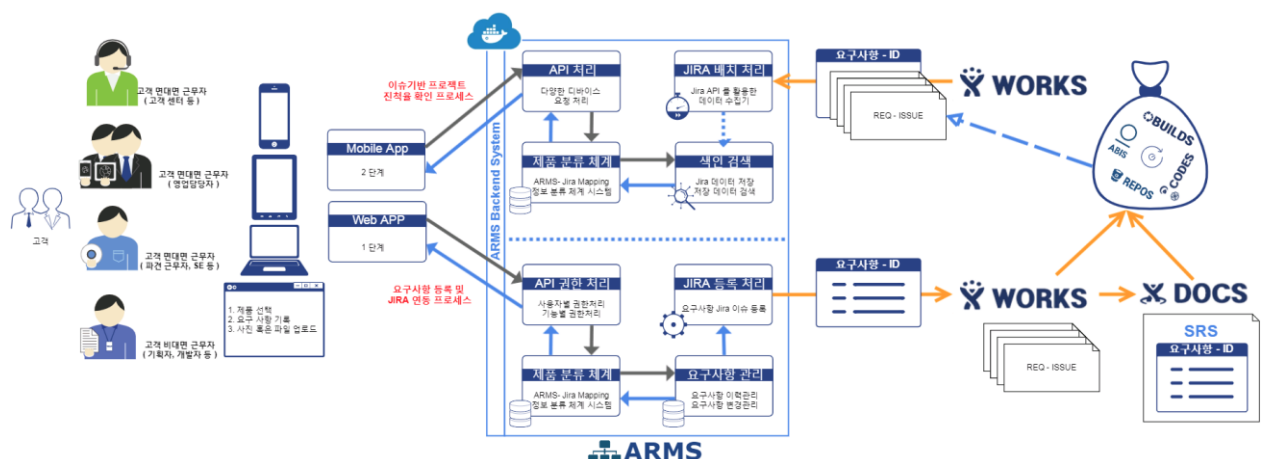
- 제품 이해 관계자들이 요구사항을 쉽고, 단순하게 올릴 수 있습니다  
( 누가 언제 어떤 변화 이력이 있는지는 시스템이 지원합니다 )
- WORKS( Jira)에 특정 label 을 달고 자동으로 요구사항 이슈가 등록되며,
- 선정된 요구사항을 토대로 DOCS ( confluence ) 에 SRS 가 기재되어야 하며  
( SRS 타이틀 밑에 요구사항 ID 가 기록되어야 합니다 :없습니다! )
- ARMS 가 이를 수집하여 현재 상태를 표시합니다. ( 아래와 같은 식으로 말이죠 )
  - 해당 요구사항을 제품에 반영하기 위하여 현재 10 개의 이슈가 등록되어있습니다.
  - 10 개의 이슈 중 3 개의 이슈가 해결되었고, 5 개의 이슈가 진행 중 입니다.
  - 나머지 2 개는 진행이 안되고 있고, 전체 진행률은 55% 입니다.

※ ARMS 적용 후 ALM (ROSE) 프로토타입



## 2.3 기술의 구성

### ARMS 시스템 구성도



## ARMS 시스템 설명

ARMS 프로젝트를 구성하는 시스템은 다음과 같습니다.

- ① 사내, 사외에서 요구사항을 손쉽게 등록할 수 있도록 Device 의 다양성을 대응할 Frontend Application
- ② 다양한 Frontend Application 에 대응하기 위한 표준화된 Backend Application 과 표준 통신 프로토콜 ( JSON ) API 처리 시스템
- ③ 요구사항이 등록되는 대상은 제품, Jira 는 제품을 구성하는 프로젝트 ( 1:1 ~ 1:N ) 이므로 두 시스템을 맵핑 할 수 있는 유연한 분류체계 시스템
- ④ 진척상황(Dashboard)은 실시간 요소가 아니며, Jira 에 추가 부하가 없도록. ( 스케줄러 배치 처리 시스템 )
- ⑤ Jira 에 등록된 정보(API)를 배치처리하여 데이터를 누적하기 위한 정보 저장 및 검색을 지원하는 색인, 검색엔진 시스템
- ⑥ 각 역할별 권한 처리 시스템 ( Role base ) - LDAP, AD, CROWD 인증 연동
- ⑦ 요구사항 이력관리 시스템 ( 분류 체계 시스템의 서브 기능 )
- ⑧ JIRA 등록 처리 시스템
- ⑨ Private Cloud Service 를 위한 Docker 지원 서비스 솔루션 : Kubernetes

## ARMS 기술 요소 특징

추가적인 기술적용의 특징요소는 다음과 같습니다.

- ① ARMS 는 대한민국정부가 인증한 전자정부표준프레임워크 3.6 을 기준한 Framework 를 Backend 로 사용합니다.
- ② 따라서, 누구나 재사용 할 수 있는 아키텍처를 적용하고, 확장할 수 있도록 PLE(Product Line Engineering) 기법을 적용합니다.
- ③ Atlassian UI 구성인 Bootstrap + jQuery 및 CSS 를 채택하여 개발 진행 ( Design 요소가 필요 없습니다 )
- ④ Frontend 어플리케이션과 Backend 어플리케이션을 분리 개발 및 통신 프로토콜을 JSON 표준으로 채택하여
- ⑤ Frontend 개발이 Backend 영향없이 100% 분리하여 병렬 개발 가능합니다.
- ⑥ SonarQube 를 활용한 ( CPD, PMD, Checkstyle 등의 코드 퀄리티 측정 ) Sacle A 등급 코드 유지합니다

- ⑦ UnitTest Ccoverage 를 핵심인 분류체계 시스템에 한정하여 100%를 달성합니다.
- ⑧ 최종 산출 아티팩트를 Docker 로 구성( build automatioin)하고, Kubernetes 를 활용하여 배포 및 서비스 운영을 목표로 합니다.

## ARMS 분류 체계 시스템 제품 – Project mapping

ARMS 의 특징적 기술요소는

유연한 분류체계가 탑재된 서버사이드 시스템 입니다.

- ① 입력된 요구사항이 어느 제품에 요구하는 것인지.  
( 입력값은 적용 희망 제품, 요구사항, option. 파일 첨부 만! )
- ② 해당 요구사항과 JIRA mapping 정보 분류  
( with. JIRA 프로젝트 이슈 정보 )
- ③ 요구사항의 시스템 이력관리 ( 기능 확장의 유연성 )
- ④ 요구사항 자체의 구조관리  
( PM 과 기획자의 역할, 채택과 변경등 ) 때문입니다.

ARMS 분류 체계 시스템 예 (※ 객체의 구조적 활용 중심의 시스템)

The screenshot displays the ARMS classification system interface. On the left, there is a tree view showing a hierarchy of nodes: 'First Child' (drive), 'ASD MSA' (folder), 'Data Architecture' (folder), 'Common Data' (default), 'Cloud Front Data' (default), 'Static Analysis Data' (default), 'Application Architecture' (folder), 'RainMaker' (folder), 'MDSC Dashboard' (default), 'MDSC PathFinder' (default), and 'MDSC DataFront' (default). Below the tree view is a list of methods: '분석 결과enable\_hotkeys', 'init', 'set\_focus', 'is\_focused', 'set\_theme', 'show\_dots', 'show\_icons', 'load\_node', and 'load\_node'. On the right, there is a table with columns: 'c\_id', 'c\_parentid', 'c\_position', 'c\_left', 'c\_right', 'c\_level', 'c\_title', and 'c\_type'. The table shows 12 entries, with the first 10 displayed. The table is titled 'Show 10 entries' and has a search bar. The table data is as follows:

c_id	c_parentid	c_position	c_left	c_right	c_level	c_title	c_type
1	0	0	1	24	0	Root Node	root
2	1	0	2	23	1	First Child	drive
3	70	1	18	19	3	MDSC PathFinder	default
4	2	0	3	14	2	ASD MSA	folder
50	70	0	16	17	3	MDSC Dashboard	default
70	2	1	15	22	2	RainMaker	folder
90	4	0	4	11	3	Data Architecture	folder
110	90	0	5	6	4	Common Data	default
111	70	2	20	21	3	MDSC DataFront	default
112	4	1	12	13	3	Application Architecture	folder

Showing 1 to 10 of 12 entries

Previous 1 2 Next

Node 생성, 변경, 이동, 복사-붙여넣기, 자르기-붙여넣기, 검색, N 차 depth, branch move 기능 등을 지원하는 객체 row data 컨트롤 프레임워크 사용

### 3 활용 방안

#### 3.1 기대 효과

##### ARMS 적용 후 순기능

이렇게 ARMS 가 적용되면 순기능 현상이 생길 것입니다.

- ① 모든 프로젝트는 요구사항 관리에 신경을 쓰기 시작할 것입니다.
- ② 프로젝트 버전관리에 힘쓰기 시작할 것이며,  
( 버전은 결국 릴리즈 날짜를 선정하고,  
어떤 요구사항을 적용했는지에 관심을 가지게 됩니다. )
- ③ 제품의 모든 관련자가 요구사항을 기반으로  
소통할 수 있는 창구가 생길 것입니다.
- ④ 상위 관리자는 현재 진행되는 모든 프로젝트의 진행 척도를  
확인할 수 있고,  
제품의 개발 진척 상황을 확인 할 수 있을 것입니다

##### ARMS 적용 후

##### 역할 별 효과

프로젝트 추진 후 기대효과 ( 순기능을 통한 역할 별 효과 )

구분	효과 설명
고객 면대면 담당자	고객의 요구사항의 실시간 채택여부, 개발 진척 상황, 제품 적용 시점 확인 가능
기획 담당자	요구사항의 이력 관리 및 분류, 구분, 변경에 대한 개발팀과의 소통 창구 개설
상위 관리자	전체 제품 개발 진척 상황 모니터링
개발팀	명시된 요구사항을 근거로 이슈 생성 > 문서 작성 > 개발 > 리뷰 > 코드 분석 > 빌드 > 배포 FM Flow

영업부터 개발까지 제품 명 및 커뮤니케이션 단어 사용에 통일성을 가지게 될 것입니다. ( 안랩의 한목소리 )

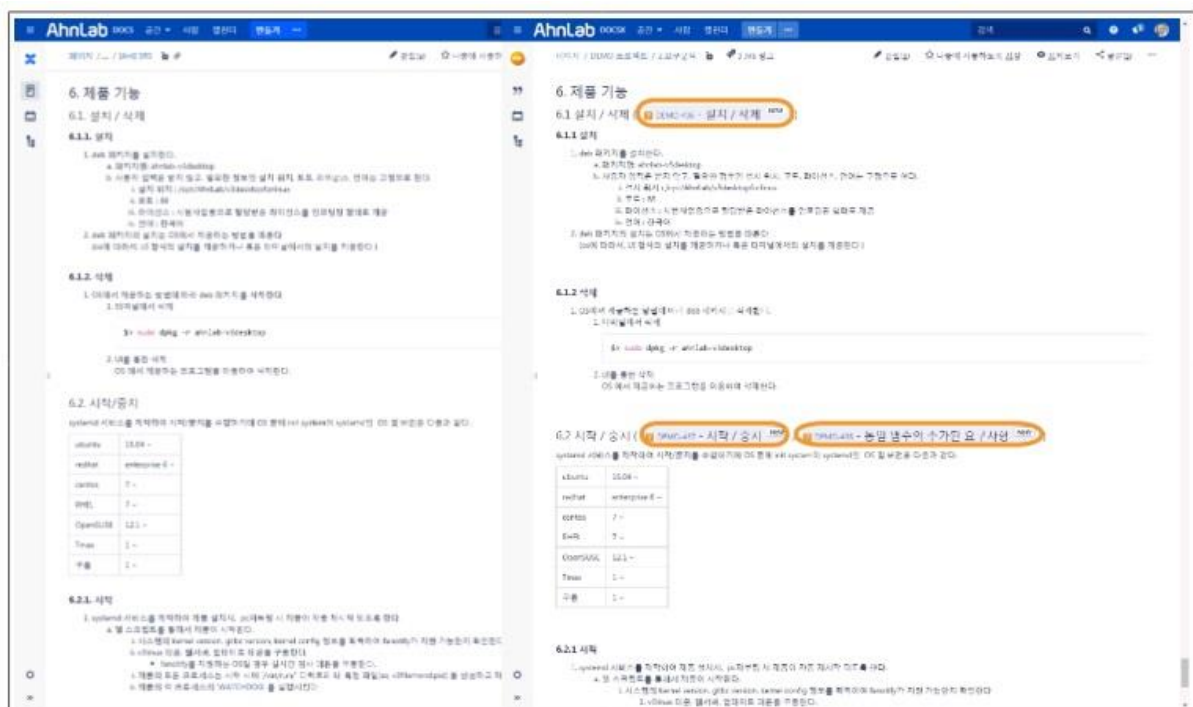
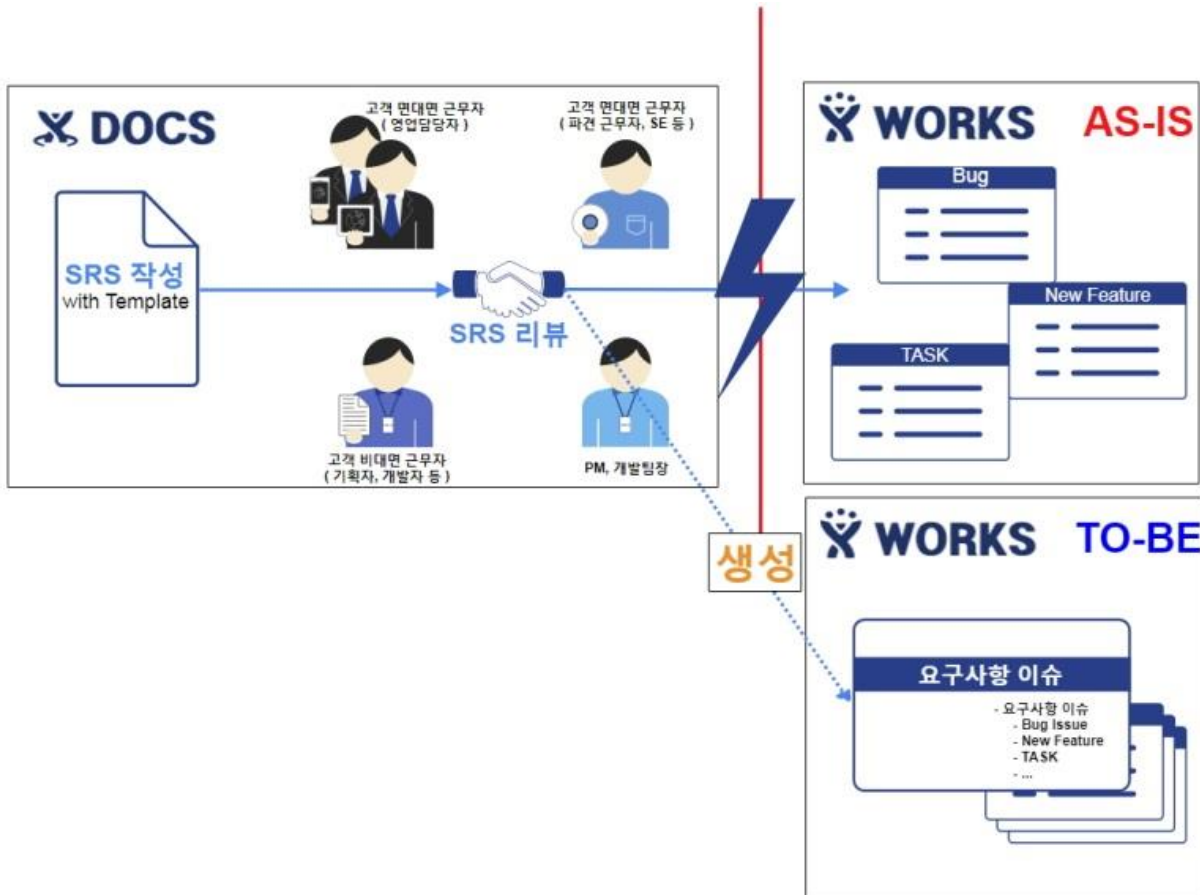
영업부터 개발까지 버전 관리 및 일정 관리, 요구사항 기반 이슈관리에 힘쓰게 될 것입니다.

( DOCS, WORKS, CODES, REPOS, SASS, BUILDS, ABIS ) 를 활용한 개발 및 추적이 가능합니다  
그 시작점이 절실히 요구되는 시점입니다.



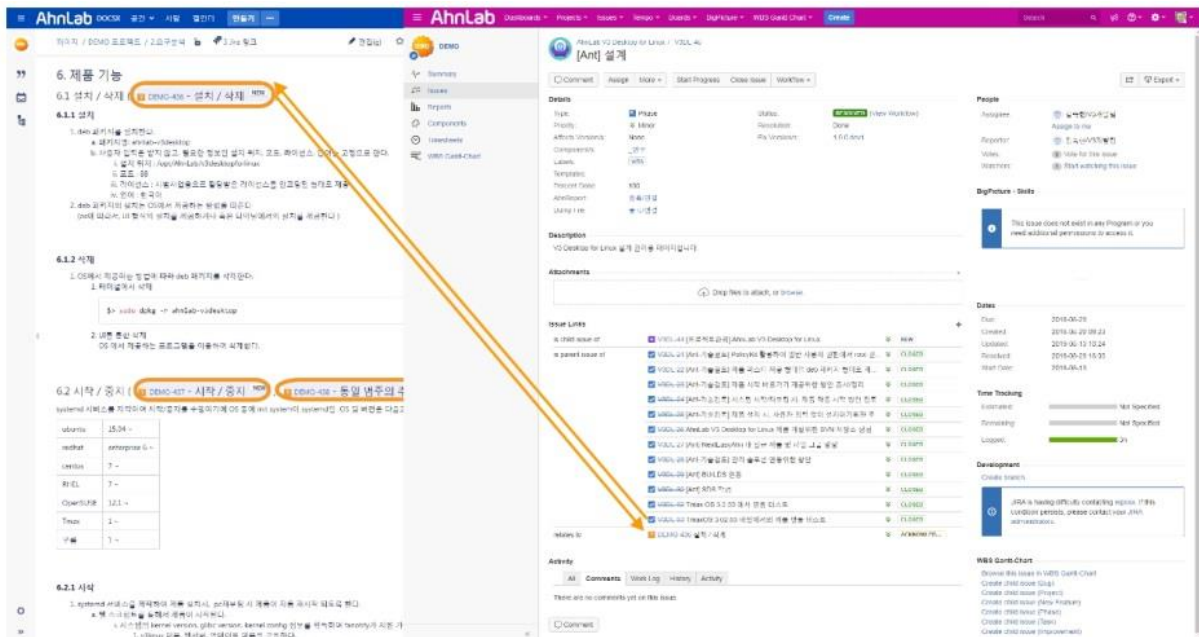
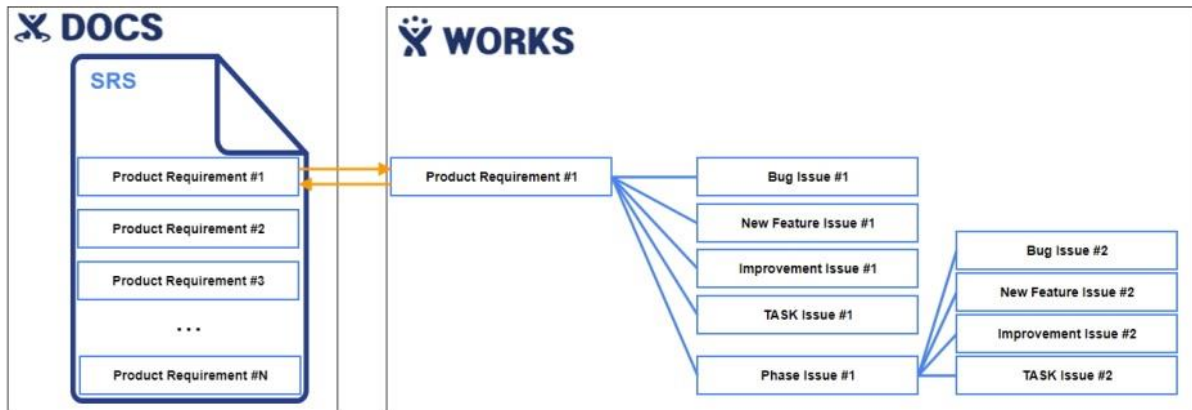
## Mockup Flow

## SRS 개선 ( 변화는 작게 효과는 크게! )



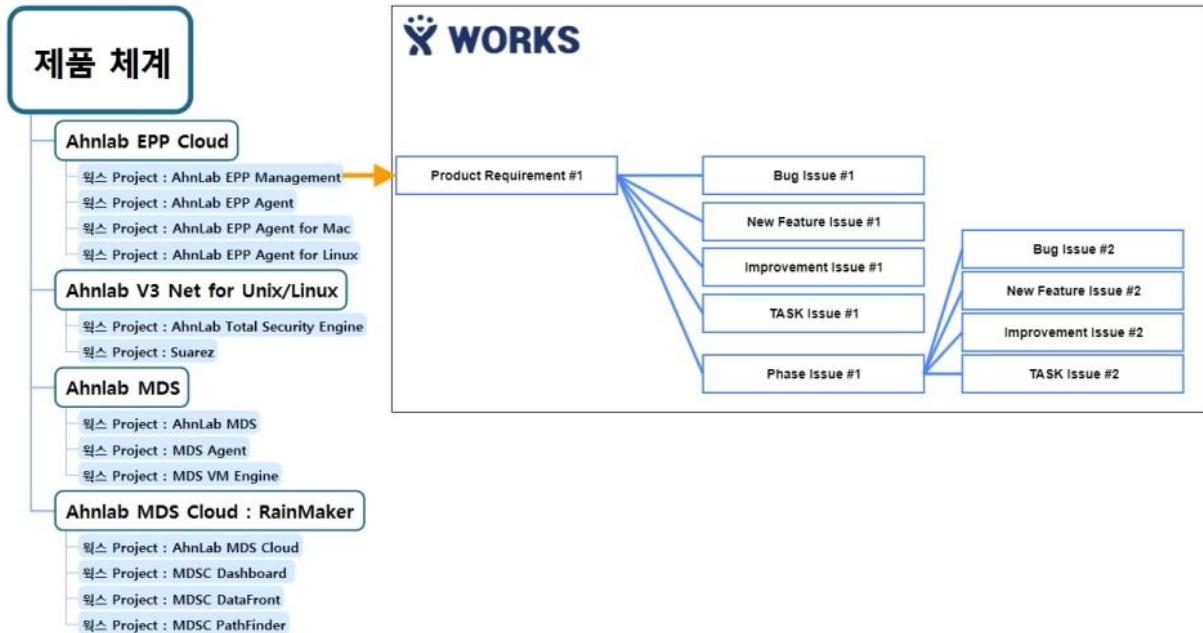


프로젝트 관리 및 이슈관리의 변화는 없다. 단지 SRS 요구사항과 이슈 링크만

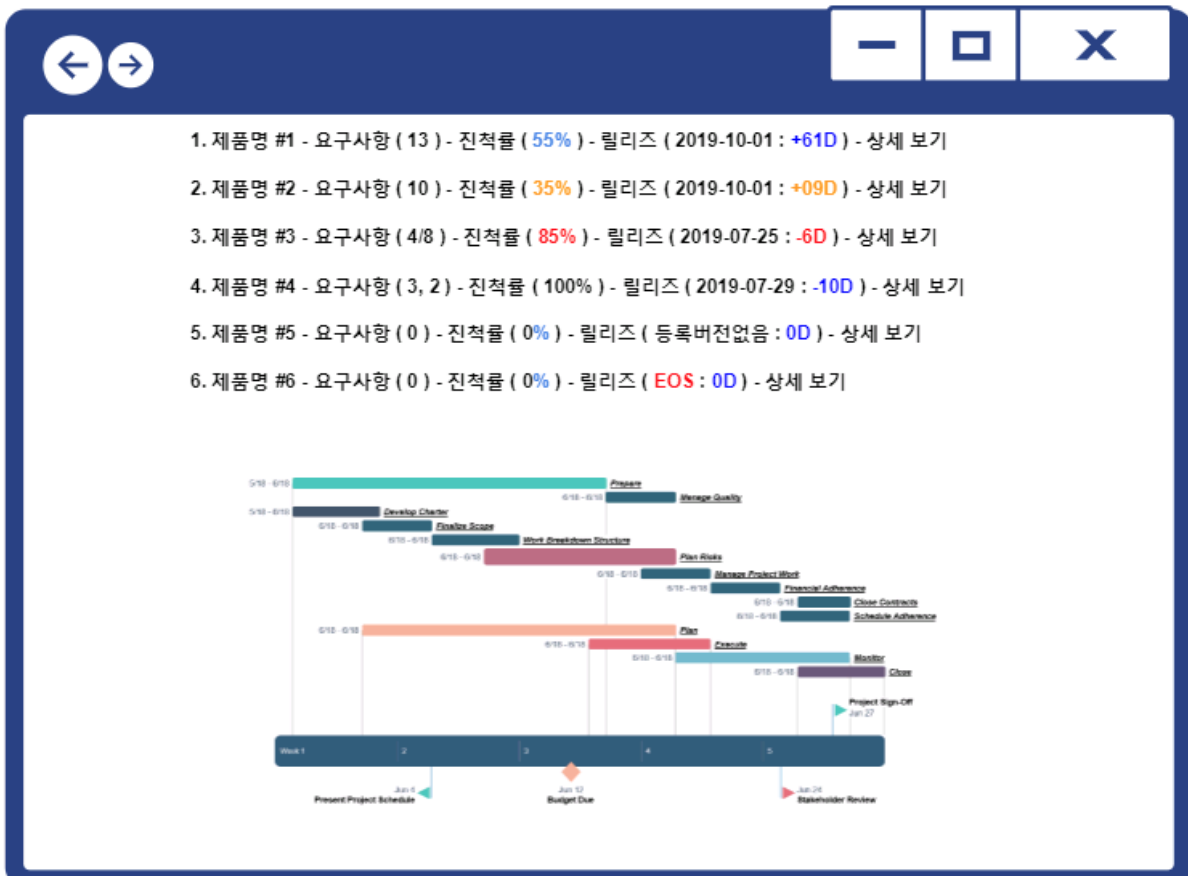


- ✓ 요구사항을 WORKS 에서 먼저 만들어서 관리 후 SRS 에 링크 할 수 도 있고,
- ✓ SRS 에서 대주제 요구사항을 작성 후 요구사항 이슈를 만들 수 도 있다.
- ✓ WORKS 에서는 issue type 이 하나만 추가되는 것이기 때문에 부하가 없다.

제품의 체계는 프로젝트와 1:1+N의 관계이다.



제품별 요구사항을 추적하여 및 데이터를 수집하면



## 제품 체계

### Ahnlab EPP Cloud

웹스 Project : AhnLab EPP Management

[ Current Verison : 1.0.1.1 ] : 요구사항 13개 : 관련 이슈 93개 : **진척률 55%** : 릴리즈 ( 2019-10-01 +51D ) - 상세보기

웹스 Project : AhnLab EPP Agent

[ Current Verison : 2.0.1.3 ] : 요구사항 10개 : 관련 이슈 31개 : **진척률 35%** : 릴리즈 ( 2019-10-01 +19D ) - 상세보기

웹스 Project : AhnLab EPP Agent for Mac

[ Current Verison : 1.0.1.1 ] : 요구사항 4/8개 : 관련 이슈 123개 : **진척률 85%** : 릴리즈 ( 2019-07-25 -6D ) - 상세보기

웹스 Project : AhnLab EPP Agent for Linux

[ Current Verison : 1.0.1.1 ] : 요구사항 3, 2개 : 관련 이슈 93개 : **진척률 100%** : 릴리즈 ( 2019-10-01 -10D ) - 상세보기

### Ahnlab V3 Net for Unix/Linux

웹스 Project : AhnLab Total Security Engine

[ Current Verison : N/A ] : 요구사항 0개 : 관련 이슈 0개 : **진척률 0%** : 릴리즈 ( N/A ) - 상세보기

웹스 Project : Suarez

[ Current Verison : N/A ] : 요구사항 0개 : 관련 이슈 0개 : **진척률 0%** : 릴리즈 ( EOS ) - 상세보기

### Ahnlab MDS

웹스 Project : AhnLab MDS

웹스 Project : MDS Agent

웹스 Project : MDS VM Engine

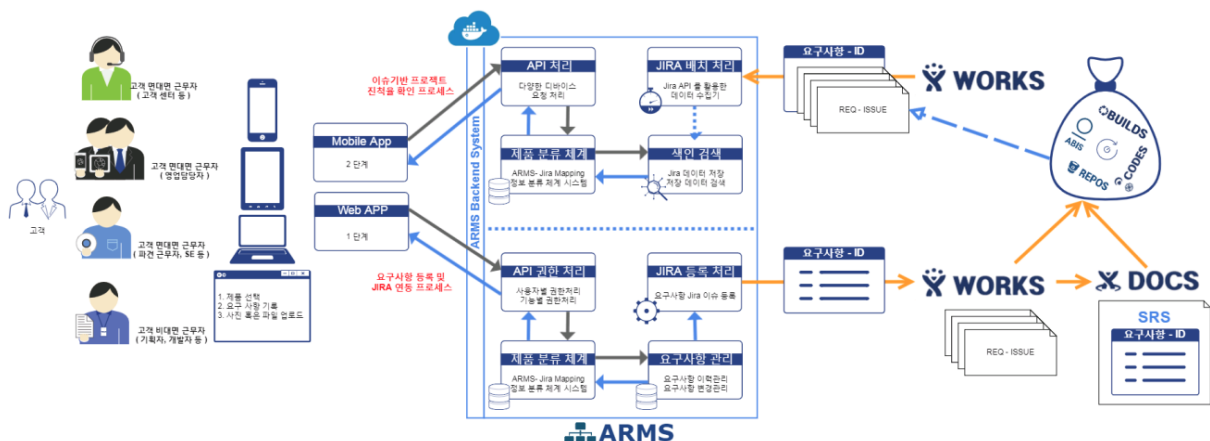
### Ahnlab MDS Cloud : RainMaker

웹스 Project : AhnLab MDS Cloud

웹스 Project : MDSC Dashboard

웹스 Project : MDSC DataFront

웹스 Project : MDSC PathFinder



- ✓ 그렇다면 제품별로 다음과 같이 ARMS 를 통하여 요구사항을 등록 할 수 있다.

**ARMS ( Requirement Management System )** Configure Fields ▾

제품명 \* ROSE (ROSE) ?

Summary \*

Description \* 

Style ▾
B
I
U
A ▾
%A ▾





+
▾

Visual
Text

Attachment  Drop files to attach, or browse.

Create another Create Cancel

- ✓ 그렇다면 ARMS 를 통하여 제품별 요구사항을 관리 할 수 있다.  
( SRS 의 대주제 요구사항을 크롤링 )

프로젝트 요구사항 추적표

**WORKS**

ID	Requirements	DOCS	ABIS	TEST
BS-10	사용자는 특정 기능에 참여할 수 있어야 한다.	BSDRM-1 BSDRM-2		BST-10
BS-11	[Dev] 사용자는 특정 입출력 데이터를 보고할 수 있어야 한다.	BSDRM-1 BSDRM-2	BSDRM-10	BST-8 BST-10
BS-12	사용자는 입출력 데이터를 위해 피드백을 할 수 있다.	BSDRM-4 BSDRM-3		BST-1
BS-13	사용자는 입출력 데이터를 위해 아이디와 비밀번호로 로그인할 수 있다.	BSDRM-4 BSDRM-3	BSDRM-10	BST-2
BS-14	사용자는 아이디를 찾을 수 있다.	BSDRM-4 BSDRM-3		BST-3
BS-15	사용자는 비밀번호를 찾을 수 있다.	BSDRM-4 BSDRM-3		BST-4
BS-16	사용자는 OTP를 통해 로그인할 수 있다.	BSDRM-2 BSDRM-3	BSDRM-7	BST-5
BS-17	사용자는 SMM 인증을 통해 로그인할 수 있다.	BSDRM-4 BSDRM-3	BSDRM-8	BST-6
BS-18	관리자는 사용자를 관리할 수 있어야 한다.	BSDRM-2 BSDRM-3		
BS-20	관리자는 사용자 정보를 수정할 수 있어야 한다.	BSDRM-4 BSDRM-3		

1 / 3 Previous 1 2 3 Next

### 3.2 한계점

물론 ARMS 에는 한계점이 있습니다. ( ARMS 1 차에는 Test 관리 기능이 빠져있습니다 )  
 PMS 관점에서 ALM 은 요구 관리 ~ 배포 ( 전달 ) 까지의 범주 입니다.  
 ( 또한 여기에 테스트 관리가 빠졌음을 이야기 할 수 도 있을 것입니다. 2 차 iQx 예정 )

## 4 실행 계획

	9 월	10 월	11 월	12 월	1 월	2 월	3 월	4 월
SRS 작성								
서버 개발								
프레임워크 개발								
Mobile + Web 시스템 개발								
ARMS 시스템 적용								

## 5 관련 참고 사항

QA 팀장님의 의견.

1. WORKS 중심의 시스템이고, ARMS 에서 역으로 수집해서 진척도 표시는 매우 좋은 기능이지만, 품질 결과도 같이 표시된다면 더욱 좋겠다.
2. Flow 가 꼭 ARMS 에서 시작되어야 하는 것이 아니라 WORKS 에서 요구사항을 만들어도 ARMS 가 수집해서 mapping 되었으면 좋겠다.
3. 현재의 요구사항은 SRS 최초 등록 후 에는 요구사항 ID 를 가져오는 것이 매우 힘들다. 이부분이 ARMS 에서 해결되었으면 좋겠다.
4. Rest API 로 스피라에서 검색이 가능했으면 좋겠다 ( 자동화 관점 )
5. 현재까지 제품과 WORKS 프로젝트를 맵핑하기 위하여 거대한 엑셀파일로 관리하고 있었다.
6. RMS 에 테스트 영역이 빠져있다.