|  |
| --- |
|  |
| Atlassian |
|  |
|  |
| 인턴 김 지 영 |
|  |
| 2018-10-02 |

|  |
| --- |
|  |

**목차**

[**1.** **JIRA** 3](#_Toc526248308)

[**1.** **도입 이유** 3](#_Toc526248309)

[**2.** **기능** 4](#_Toc526248310)

[**3.** **장점&단점** 5](#_Toc526248311)

[**4.** **유사 Tool** 6](#_Toc526248312)

[**5. 도입 사례** 7](#_Toc526248313)

[**2.** **Confluence** 8](#_Toc526248314)

[**1.** **도입 이유** 8](#_Toc526248315)

[**2.** **기능** 9](#_Toc526248316)

[**3.** **장점&단점** 9](#_Toc526248317)

[**4.** **유사 Tool** 11](#_Toc526248318)

[**5.** **도입 사례** 12](#_Toc526248319)

[**3.** **Bitbucket** 13](#_Toc526248320)

[**1.** **도입 이유** 13](#_Toc526248321)

[**2.** **기능** 14](#_Toc526248322)

[**3.** **장점&단점** 14](#_Toc526248323)

[**4.** **유사 Tool** 14](#_Toc526248324)

[**4.** **Bamboo** 16](#_Toc526248325)

[**1.** **도입 이유** 16](#_Toc526248326)

[**2.** **기능** 16](#_Toc526248327)

[**3.** **장점&단점** 16](#_Toc526248328)

[**4.** **유사 Tool** 17](#_Toc526248329)

[**5.** **도입 사례** 17](#_Toc526248330)

[**5. 출처** 18](#_Toc526248331)

# **Atlassian**

소프트웨어 개발자들을 타겟으로 하는 비즈니스 엔터프라이즈 소프트웨어를 만드는 소프트웨어 기업이다. 애자일 팀이 사용할 수 있는 소프트웨어 도구들을 개발, 관리한다.

\*ALM(Application Lifecycle Management) : 요구 사항관리, 소프트웨어 아키텍쳐, 프로그래밍, 소프트웨어 테스팅, 소프트웨어 유지보수, 변경관리, 지속적인 통합, 프로젝트 관리, 릴리즈 관리를 포함하는 말이다.

\*스크럼(Scrum) : 소프트웨어 개발 프로젝트 관리를 위한 상호, 점진적 개발방법론으로 Agile 소프트웨어 공학 중의 하나이다.

\*스크럼 특성 : 특정 언어나 방법론에 의존적이지 않고, 개발 언어는 물론이고 객체 지향 언어와도 관련이 없는 넓은 응용 범위의 개발 기법이다.

* 솔루션에 포함할 기능/개선점에 대한 우선 순위를 부여한다.
* 개발 주기는 30일 정도로 조절하고 개발 주기마다 실제 동작할 수 있는 결과를 만든다.
* 개발 주기마다 적용할 기능이나 개선에 대한 목록을 만든다.

# **JIRA**

Atlassian사에서 개발한 Issue Tracking System 이다. 버그 추적, 이슈 추적, 프로젝트 관리 기능을 제공하는 소프트웨어이다.

## **도입 이유**

프로젝트의 특성상 프로젝트의 범위와 산출물은 초기에 개괄적으로만 정의되었다가 진행하면서 구체화 될 수 밖에 없기 때문에 프로젝트를 진행할 때 초기 계획대로 프로젝트가 진행되는 경우가 거의 없다.

* 이력 관리와 산출물 문제

-산출물 : 소스코드, 매뉴얼 등 공유하거나 확인할 곳을 정해 두어야 한다.

-진철률 : 진척률을 정하는 근거를 확실히 해야한다.

-추적 : 진행중인 일이 환경에 따라 변경될 경우 이력관리를 해야한다.

* 협업과 공유의 문제

-피드백 : 참여자들끼리 피드백을 교환하고 이를 계획에 반영할 필요가 있다.

-구체화 : 실제 진행하면서 구체화되는 내용에 대한 관리가 필요하다.

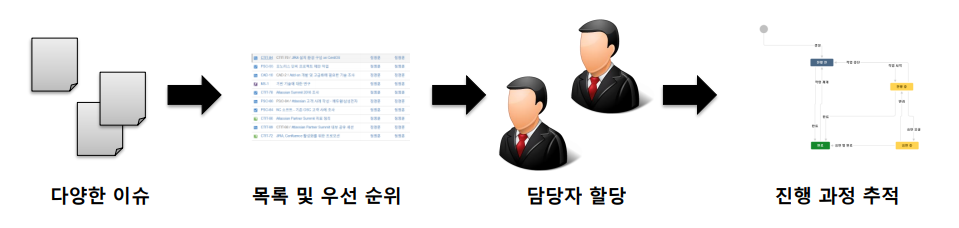
(Excel의 경우 최고의 스프레드시트 프로그램이지만 변화하는 환경에 대응이 어려울 뿐만 아니라 협업과 수정/배포, 추적이 불가능하다. )

이러한 문제점들을 보완하기 위해서 Issue Tracking System을 도입하는 것이 좋다.

## **기능**

**Issue Tracking System**

* 업무(작업), 문제점, 개선 사항 등을 이슈라는 단위로 관리한다.
* 이슈 관리 방법-아래의 내용들을 시스템적으로 관리한다.



[그림1] 이슈관리방법

출처 : https://www.eventservice.kr/atlassian/file/1207\_Atlassian\_busan\_HandsOnLabs03.pdf

* 워크플로우를 관리한다.

-진행 중인 작업의 상태, 업무 과정(테스크 할당, 작업 로그 등)을 명확하게 알려준다.

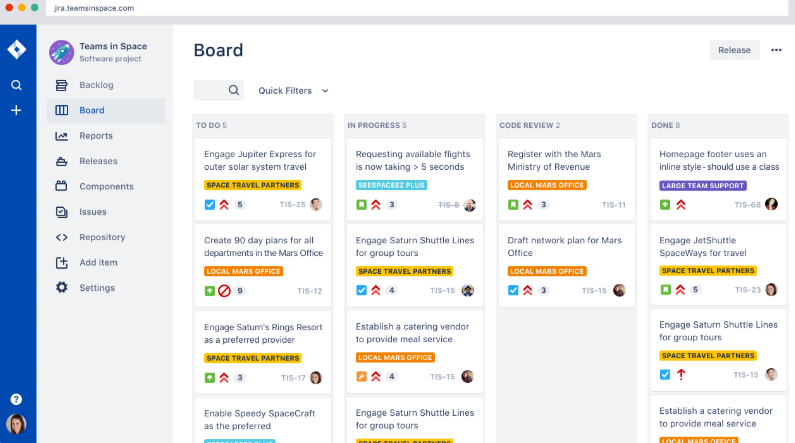
-전체 프로젝트의 상태 파악 및 리소스 계획 수립이 가능하다.

* 통합된 리포트 및 자신이 할당 받은 이슈를 볼 수 있는 대시보드 제공한다.

-JIRA에서 Kanban Software Development, Scrum Software Development를 기본적으로 제공한다.

\*kanban : 포스트잇을 화이트보드에 나열하는 것 같은 보드 스타일.

* 관리하는 데이터를 검색하여 결과를 볼 수 있도록 하는 Filter기능을 제공한다.



[그림2] Jira

출처 : <https://ko.atlassian.com/software/jira>

**Jira Core / Software / ServiceDesk**

* Core: Jira의 기본기능 탑재
* Software: Board 기능 및 Software Type 프로젝트 기능
* Service Desk: 사용자 Portal 및 SLA 관리 기능

\*SLA(Service Level Agreement) : 서비스 수준 협의서. 서비스를 제공함에 있어서 공급자와 사용자간에 서비스에 대한 측정 지표와 목표 등을 정해놓는 협약서 이다.

## **장점&단점**

**장점**

* 이슈에 대한 역할과 임무를 분명이 할 수 있어서 협업할 때 용이하다.
* 작업 현황을 확인하고 스케줄이나 우선순위를 조절할 수 있다.

(특정 이슈를 누가 발견했고, 누가 해결할 것인지, 현재 어떤 상태인지를 한눈에 파악할 수 있다.)

* 개발 과정이나 업무처리에 필요한 다양한 도구와 연동할 수 있다. (Confluence-wiki, Bamboo-CI, fisheye-SCM)
* 워크 플로우 개념 덕분에 개발과 운영 모두 다양하게 사용할 수 있다.

**단점**

* 유료이다.
* 기능이 너무 많고, 화면이 복잡해서 커스터마이징을 할 필요가 있다. 익숙하지 않다면 사용하기 어렵다.
* SCM과 연계하려면 Atlassian의 다른 제품을 추가로 구매해야 한다.

\*SCM(Supply chain management) : 공급망 관리. 부품 제공업자로부터 생산자, 배포자, 고객에 이르는 물류의 흐름을 하나의 가치사슬 관점에서 파악하고 필요한 정보가 원활히 흐르도록 지원하는 시스템이다.

## **유사 Tool**

**Redmine**

장점

* 무료이다.
* 프로젝트 별로 wiki, 게시판, 문서관리 메뉴들이 있다.
* 프로젝트를 tree 구조와 같이 sub project를 계속 만들 수 있다.

단점

* UI가 어려워서 배우지 않으면 사용할 수 없다.
* Visual Workflow 설정 기능이 없다. -> 간단한 추가 기능만 있다.
* Jira와 같이 A 프로젝트에서 B 프로젝트로 이슈를 옮길 수는 없다. ->A 프로젝트와 하위프로젝트 사이에는 이슈를 옮길 수 있다.
* 플러그인 확장이 가능하지만 업데이트 할 때마다 오류가 자주 발생한다.
* 이슈가 많아질수록 속도가 저하된다.
* JIRA에 비해 리포팅 및 관리 기능이 부족하다.

**Mantis (Issue Tracking System)**

장점 : 설치와 사용하기가 쉽기 때문에 단기 프로젝트를 하는 사용자들은 많이 접할 수 있다.

단점 : 버그 트래킹 중심으로 개발되어 전체적인 이슈 관리가 어렵고 다른 툴들과 연동이 어렵다는 단점이 있어 장기 프로젝트에서 사용하기엔 좋지 않다. 또한, UI가 좋지 않아 비개발자들이 쓰기에 좋지 않다.

**Trac (Issue Tracking System)**

장점 : 개발언어가 Python이기 때문에 프로그램을 커스터마이징 하기 쉽다. 또한, 플러그인이 활발하다.

단점 : 설치 및 설정이 어렵다. 또한, UI가 좋지 않아 비개발자들이 쓰기에 좋지않다. 운영부서와의 원활한 의사소통에 방해요소가 될 수 있다.

\*플러그인 :  호스트 응용 프로그램과 서로 응답하는 컴퓨터 프로그램이며, 특정한 "주문식" 기능을 제공한다.

## **5. 도입 사례**

Polaris Office

도입 전 상황

* Grand Open으로 인한 사용자 급증
* 사용자 증가에 따른 고객 요구사항 수집 및 관리에 어려움 발생
* 전화나 이메일만 이용하여 VOC(Voice Of Customer)를 접수 받음
* 사용자 불편 사항 및 요구사항에 대한 빠른 처리 필요

Jira Software(Agile Board) 제품 도입

* Epic(애자일에서 큰 규모의 작업으로 여러 Story들의 집합) 활용
* Agile Board를 통해 기획, 마케팅 등 요구사항을 각 팀에서 직접 등록하여 관리
* Sprint 활용
* Kanban + Scrum 방법론 활용

도입 효과

* 쉽고 빠른 VOC(Voice Of Customer) 접수 및 처리 가능
* 접수된 개선 요청 사항을 즉시 Sprint에 반영 가능
* 각 팀과의 커뮤니케이션 원활, 시장 및 고객 요구사항에 대해 빠른 대응 가능
* 집중해야 할 업무가 무엇인지 쉽게 확인 가능

# **Confluence**

Atlassian사에서 개발한 위키 소프트웨어이다. 다수 팀 간에서의 협업을 도와주고 정보의 발행, 공유, 정리 등을 하는 장소로 사용된다.

\*위키(WIKI) : 불특정 다수가 협업을 통해 직접 내용과 구조를 수정할 수 있는 살아 움직이고 변경되는 정보.

\*위키 소프트웨어(wiki software) : 웹 브라우저를 사용하여 사용자들 여럿이서 웹 페이지를 만들고 편집 할 수 있게 하는 소프트웨어.

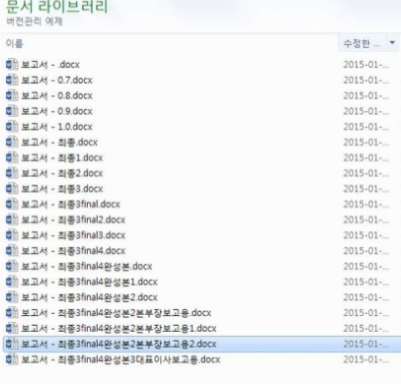
## **도입 이유**

프로젝트를 하면서 작성해야 할 문서가 매우 많기 때문에 그 문서들을 관리하는 것이 매우 중요하다. 또한, 프로젝트가 커질수록 다양한 분야에서의 업무 수행이 요구되고, 다양한 조직간의 협업을 하는 상황도 많다.

기존 협업 도구 : MS Office

공유 방법 : Email 또는 File Server

하지만 Email은 진척 사항 관리 및 이력 관리가 힘들고 검색도 할 수 없다. 수정이나 병합, 배포를 하기에도 매우 어렵다. 따라서 프로젝트를 진행할 때 효율적인 협업을 위해 Wiki software가 필요하다.



[그림3]Email 사용시 문제점

출처 : <https://www.slideshare.net/ssuser800974/atlassian-confluence-wiki>

## **기능**

프로젝트 협업 관리 도구

* 기술 문서 관리용으로 활용할 수 있다.
* 개인 Task를 등록해 개인 일정 관리(Jira활용)뿐만 아니라 팀 일정관리로 활용할 수 있다.
* 주간 업무 보고용으로 활용할 수 있다.
* 고객 서비스용 매뉴얼 페이지로 활용할 수 있다.

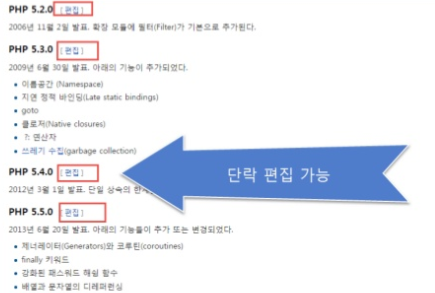
## **장점&단점**

**장점**

* 동시에 편집/수정/병합이 가능하여 협업의 결과물을 실시간으로 남길 수 있다.
* 이력과 버전 관리가 가능하다. -> 언제든지 특정 버전으로 돌아갈 수 있다.
* 모바일 서비스를 지원하여 대부분의 위키 기능들 모바일에서도 수행가능하다. (푸시 알림 설정 가능)
* 모든 팀, 부서 또는 주요 프로젝트에서 지식과 정보를 공유하고 체계를 갖춘 작업이 가능하다. (분류 및 검색이 쉽고 모든 정보의 업데이트 상황을 조회할 수 있기 때문에)
* 인라인, 페이지, 파일 댓글(Comments기능)을 이용하여 작업 자체에 피드백을 남길 수 있다.
* 파일 목록, 프로젝트 계획 또는 제품 요구사항 등을 공유버튼을 이용해 간단히 배포하고 피드백을 수집할 수 있다.
* Confluence 에디터를 이용해 전문가가 아니더라도 쉽게 컨텐츠를 생성할 수 있다.
* 컨텐츠(Space) 별 접근 권한 통제를 할 수 있어 보안을 유지할 수 있다.
* 플러그인이 다양해서 각자에게 필요한 기능을 쉽게 추가 할 수 있다. (MarketPlace)
* 관심있는 페이지가 변경될 경우 알림을 받도록 설정할 수 있다.
* 개발 과정이나 업무처리에 필요한 다양한 도구와 연동할 수 있다.(Jira)

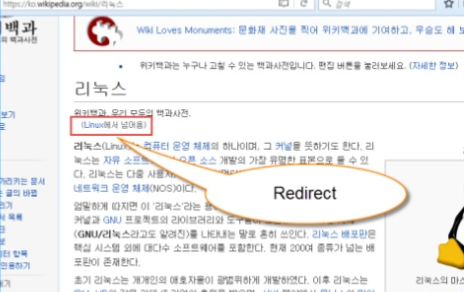
**단점**

* 유료이다.
* 매년 유지보수료를 지불해야 한다. (지불하지 않으면 업데이트와 기술지원이 제한된다.)
* Confluence의 Contents Gardener가 필요하다. (깨진 링크 정리, 중복 페이지 삭제 등)
* 단락편집(Section editing)기능이 없어서 페이지 내용이 많아질 경우 하위페이지로 나눠서 관리가 필요하다.
* Page Redirect 기능이 없기 때문에 페이지 작성시 검색을 고려하여 작성을 해야 한다. (Redirect 플러그인 사용 가능)

\*

[그림4]단락 편집 기능

출처 : https://www.slideshare.net/ssuser800974/atlassian-confluence-wiki

\*

[그림5]Page Redirect 기능

출처:https://www.slideshare.net/ssuser800974/atlassian-confluence-wiki

## **유사 Tool**

**Media wiki**

장점

* Wikipedia에서 사용하는 것으로 기능과 성능이 검증된 제품이다.
* 단락 편집(Section editing) 기능이 있어서 한 페이지에 많은 내용을 넣어서 관리 할 수 있다.
* Page Redirect기능이 있어서 문서가 여러 이름으로 참조되거나, 대소문자, 축약어를 가질 때 유용하다.

단점

* Confluence와 다르게 계층구조를 가지고 있지 않다. Category로만 분류할 수 있다.
* 마크업 언어를 몰라도 컨텐츠를 만들 수 있는 Confluence와는 다르게 문법을 모르면 컨텐츠를 생성할 수 없다.
* 접근 권한 통제가 어렵다.

## **도입 사례**

Polaris Office

도입 전 상황

* B2C사업으로 인해 요구사항 급증, 조직 구조 확대
* 다양한 분야에서의 업무 수행이 필요 하게 됨
* 하나의 프로젝트에 다양한 조직이 참여하여 다양한 조직간의 협업 필요
* 고객 요구사항에 빠른 대응을 하기 위해서 원활한 커뮤니케이션 필요

Confluence 도입

* Jira와 연동하여 사용
* Jira 요구사항 이슈 등록 후 이슈 Link 항목에 Confluence URL 연동
* 기획서, 결과 보고서, 업무 매뉴얼 등을 Confluence를 이용하여 정리

도입 효과

* 요구 사항 및 기획서 버전 관리가 명확
* 문서 형상관리 이슈가 약95% 감소
* 팀 평균 업무 역량 강화
* 자리를 비우거나 결근을 하더라도 업무 수행이 원활하게 진행됨.

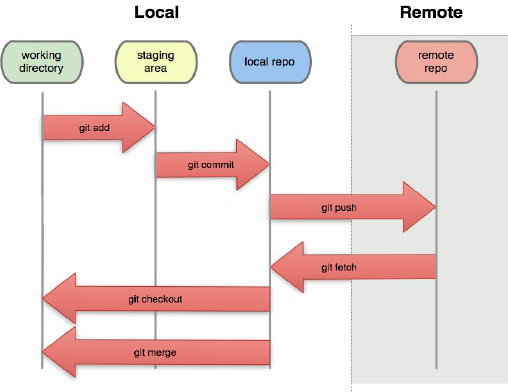
# **Bitbucket**

Atlassian이 소유한 웹 기반 호스팅 서비스로, 소스코드 호스팅 서비스의 일종이다. Mercurial또는 Git 버전 관리 시스템을 지원한다.

\*호스팅 서비스 : 인터넷 접속 서비스 제공자가 제공하는 서버의 보관 서비스.

\*Mercurial : 소스코드 관리 도구로, 사용자들이 소스코드 변경사항을 로컬에서 추적하고 변경 사항들을 다른 사용자들과 공유할 수 있도록 한다.

\*Git : 소스의 이력 등을 확인할 수 있는 버전 관리 시스템으로, 개발자가 프로젝트의 소스들을 편집 할 때마다 그 이력을 기록하고 관리한다. Git은 Local Repository에 commit한 뒤, Remote Repository에 push하는 구조를 가지고 있다.



**Bitbucket**

[그림6] git stage

출처: https://pixelpioneers.co/blog/2017/git-basics-explained-by-designing-a-new-car

## **도입 이유**

깃을 사용하기 더 편리하게 해준다.

* 로컬에서 프로젝트를 관리할 수 있게 해주는 비주얼 인터페이스를 제공한다.
* 웹에서 프로젝트를 버전 관리 간편하게 할 수 있다.

## **기능**

**소스코드 호스팅 서비스**

* Git 기반의 개발 소스 관리 환경을 기반으로 한다.
* 무료로 무제한 private repository를 제공한다.
* Branch권한 및 프로젝트 권한 관리를 할 수 있다.
* 배포 기능이 포함된 Bitbucket Pipelines으로 통합된 CI/CD를 할 수 있다.

\*CI(Continuous Integration) : 빌드, 테스트를 실시하는 프로세스를 말하며, 이러한 통합 프로세스를 상시로 실시해 주는 것이다.

\*CD(Continuous Delivery or Continuous Deploy) : 짧은 주기로 개발중인 소프트웨어를 배포하고 그 과정을 자동화 하는 것이다.

## **장점&단점**

**장점**

* Branch권한 및 프로젝트 권한 관리 기능으로 권한 통제가 가능하다.
* Jira 및 Bamboo 와의 연계를 통해 소스코드의 통합 이력 관리 및 자동화 빌드/배포를 원활하게 할 수 있다.
* 시멘틱 검색을 기본 포함하여 편리한 코드 검색 환경을 제공한다.
* Private Repository는 무료로 제공한다.

**단점**

* 용량 제한이 있고, 기준 인원(5명)을 넘어가면 비용을 지불해야 한다.
* 사용자가 적고 소규모 프로젝트일 때 더 적합하다.

## **유사 Tool**

**Github**

장점

* 가장 많은 오픈소스 저장소를 보유하고 있다.
* 가장 안정적인 서버 상태를 제공한다.

단점

* 용량 제한이 있다.
* 무료로 사용하기 위해서는 무조건 자신의 소스를 공개해야 한다.
* Private Repository로 사용할 경우는 일부 비용을 내야하기 때문에 개인 프로젝트 저장소로는 적합하지 않다.

**Gitlab**

장점

* 설치형 Github를 목표로 나왔지만 Gitlab.com에서 호스팅 서비스 또한 제공한다.
* 가장 합리적인 가격 정책을 가지고 있다.
* Private Repository가 무료이고 인원 제한이 없다.
* 설치형은 무료이고 필요한 기능은 모두 제공한다.
* UI가 좋다.
* 자유롭게 Docker기반의 빌드 머신을 구현할 수 있어서 기업 내에서 빌드 서버를 구축하기 좋다.

단점

* 서버상태가 가끔씩 불안정하다. -6시간 분량 데이터 유실 사태



[그림7]소스코드 호스팅 서비스 비교

출처:https://m.blog.naver.com/dlwhdgur20/221006619001

# **Bamboo**

Atlassian사에서 만든 소프트웨어로, 지속적인 통합, 배포를 위한 제품이다. 자동화된 빌드와 테스트, 릴리즈를 함께 단일 워크플로우에 연결해준다. 대표적인 CI(Continuous Integration) 솔루션이다.

\*CI기능 : 자동화 빌드, 자동화 테스트 및 통지, 커버리지 감시 등

## **도입 이유**

애자일 방법론을 사용하여 개발을 하는 과정에서는 반복적인 빌드, 테스트, 릴리즈가 필요하다. 이러한 반복적인 과정을 자동화 하기 위해 CI 솔루션을 사용할 필요가 있다.

## **기능**

* 빌드, 테스트, 배포를 자동화한다.
* 배포 프로젝트와 함께 배포하고 있는 소프트웨어 프로젝트, 즉 구축 및 테스트를 마친 릴리즈, 릴리즈가 배포되는 환경들을 모두 나열하여 보여준다.
* Fisheye와 연동하여 변경된 부분들을 보여준다.
* JIRA와 연동, 빌드가 실패했을 때나 성공 했을 때 빌드 결과를 메신저로 보내준다.
* 각 프로젝트가 저장소(Repository)와 연동이 되어있다.

\*Fisheye : 코드 변경 사항 검색, 트래킹 및 시각화 기능을 제공한다.

## **장점&단점**

**장점**

* 간편한 UI를 가지고 있어 사용하기 편리하다.
* 자동화된 병렬 테스트로 애자일 개발을 도와주고 버그를 더욱 쉽고 빠르게 잡을 수 있다.
* 릴리즈하는 작업을 각 환경에 맞게 자동화하고 환경 별 권한으로 흐름을 제어할 수 있다.
* 전담 에이전트 기능이 있어서 핫픽스와 중요 빌드를 바로 실행할 수 있다. -> 수정할 중요 버그가 있을 때 무료 빌드 에이전트를 기다릴 필요가 없다.
* 개발 과정이나 업무처리에 필요한 다양한 도구들(Atlassian 제품군들)과 연동하기 쉽다.
* 새로운 릴리즈 배포 전 코드 변경 사항의 전체 롤업과 이전 배포 이후의 이슈들을 확인할 수 있다. -> 운영팀은 생산 시 담당해야 할 배포와 지원을 예상할 수 있고, 릴리즈 관리자는 실시간 상황을 명확하게 파악할 수 있다.

**단점**

* 유료이다.
* Project, plan, stage, task의 개념이 복잡해서 익숙해 지기 위해서는 시간이 필요하다.

## **유사 Tool**

**Jenkins**

장점

* 무료이다.
* Reference 및 사용자가 많아 정보가 많은 편이다.
* 개발속도가 빠르고 플러그인 지원이 좋은 편이다.
* 설치 및 사용이 간편하다.

단점

* Bamboo에 비해 프로젝트 별 보안 및 권한 설정 등이 불편하다.
* JIRA나 Redmine등 Issue Tracking System들과의 연계가 불편하거나 완벽하지 않다.

## **도입 사례**

Polaris Office

도입 전 상황

* 서비스 오픈 후 개인 사용자가 급증하여 다양한 환경 및 기능 개선 요구를 받게 됨
* 기능의 잠재적 Risk 급증
* 외형적인 버그 수정 과정에서 부작용 발생 빈도가 급증
* 잦은 릴리즈를 대응하기 위한, 잦은 빌드 과정에서 빌드 오류가 빈번하게 발생

Bamboo 도입

* Bamboo를 기반으로 한 지속적인 통합(CI)를 구현
* 빌드 오류 관리 품질 측정 자동화까지 확장함.
* Fisheye를 활용하여 형상 관리 정보를 Jira, Crucible과 연동

도입 효과

* 자동 반복적인 빌드 처리에 따른 빌드 작업 가능
* 오류 발견 즉시 담당자에게 공지되어 빠른 처리가 가능
* 배포 과정에서의 오류 및 Risk 감소
* 버그 발생 빈도 개선

# **5.** **출처**

**1) 홈페이지**

➀JIRA

- LinkedIn slideshare 『JIRA를 활용한 업무 공유 및 프로젝트 관리』 :

<https://www.slideshare.net/ssuser800974/jira-98019877>

-오픈소스컨설팅 파일 :

<https://www.eventservice.kr/atlassian/file/1207_Atlassian_busan_HandsOnLabs03.pdf>

-Atlassian홈페이지 : <https://ko.atlassian.com/software/jira/features>

-‘그대 안의 작은 호수’ 홈페이지 : <http://www.smallake.kr/?p=19559>

-‘fightingak’ Daum 블로그 :

<http://fightingaf.tistory.com/entry/JIRA-%EC%99%80-Redmine-%EB%B9%84%EA%B5%90>

-‘개발자스럽다’ 홈페이지 : <https://blog.gaerae.com/2014/05/issue-tracking-system-bug-tracking-system.html>

-‘기록은 기억을 지배한다’ Daum 블로그 :

<http://lesstif.tistory.com/entry/Issue-Tracking-System-%EB%B0%A9%ED%99%A9%EA%B8%B0-1>

-‘아틀라시안 파트너\_인프라웨어테크놀러지’ 블로그 :

<https://m.blog.naver.com/infratech_qm/221343730029>

➁Confluence

-LinkedIn Slideshare 『Confluence WIKI를 활용한 공유와 협업』 :

<https://www.slideshare.net/ssuser800974/atlassian-confluence-wiki>

-Jacks life 블로그 : <http://hadam.tistory.com/32>

-Atlassian 홈페이지 : <https://ko.atlassian.com/software/confluence/features>

-골드피처 블로그 : <http://confluence.goldpitcher.co.kr/pages/viewpage.action?pageId=156107012>

-LinkedIn Slideshare 『Atlassian Solutions Conference 대전 2017』 :

<https://www.slideshare.net/AtlKR/confluence-80156295>

-‘아틀라시안 파트너\_인프라웨어테크놀러지’ 블로그 :

<https://m.blog.naver.com/infratech_qm/221343730029>

-Hyounsoo Kim’s Home Confluence : <https://wiki.studydev.com/pages/viewpage.action?pageId=2982014>

③Bitbucket

-LinkedIn Slideshare 『Atlassian Solutions Conference 대전 2017』 :

<https://www.slideshare.net/AtlKR/git-bitbucket-git-twoseed>

-ZETAWIKI :<https://zetawiki.com/wiki/%EB%B9%84%ED%8A%B8%EB%B2%84%ED%82%B7_Bitbucket>

-조대협의 블로그 : <http://bcho.tistory.com/82>

-[EjongHyuck's 잡다한 세상](https://m.blog.naver.com/PostList.nhn?blogId=dlwhdgur20)  블로그 : <https://m.blog.naver.com/dlwhdgur20/221006619001>

-CLIEN 홈페이지 : <https://www.clien.net/service/board/cm_app/10324558>

-‘Nolboo’s Blog’ 블로그 : <https://nolboo.kim/blog/2013/10/06/github-for-beginner/>

④Bamboo

-Atlassian 홈페이지 : <https://ko.atlassian.com/software/bamboo/features>

-lesstif.com 홈페이지 : <https://www.lesstif.com/pages/viewpage.action?pageId=15269892>

-‘아틀라시안 파트너\_인프라웨어테크놀러지’ 블로그 :

<https://m.blog.naver.com/infratech_qm/221343730029>

-curve 홈페이지: <https://www.curvc.com/curvc/product/atlassian/bamboo>

**2)그림**

➀JIRA

-오픈소스컨설팅일(그림1)

: <https://www.eventservice.kr/atlassian/file/1207_Atlassian_busan_HandsOnLabs03.pdf>

-Atlassian 홈페이지(그림2) : <https://ko.atlassian.com/software/jira>

➁Confluence

- LinkedIn Slideshare 『Confluence WIKI를 활용한 공유와 협업』 (그림 3, 4, 5) :

<https://www.slideshare.net/ssuser800974/atlassian-confluence-wiki>

③Bitbucket

-PIXEL Pioneers 홈페이지 (그림 6) :

<https://pixelpioneers.co/blog/2017/git-basics-explained-by-designing-a-new-car>

-‘EjongHyuck’s 잡다한 세상’ 블로그(그림 7) :

<https://m.blog.naver.com/dlwhdgur20/221006619001>