|  |  |
| --- | --- |
| **Program / Project Name:** | JIRA |
| **Checklist / Template Completed by:** | TBD |
| **Date Completed:** | Click here to enter a date. |

**JIRA Agile**

Revision: A

**Detailed Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev** | **Date** | **Editor(s)** | **Description of change** |
| A | MAY 15, 2020 | Sang-Gu Kang | Initial Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[1 Book - JIRA Agile Essentials 4](#_Toc40445034)

[1.1 Basic of JIRA Agile 4](#_Toc40445035)

[2 Agile 방법론 4](#_Toc40445036)

[2.1.1 Scrum 4](#_Toc40445037)

[2.1.2 XP 5](#_Toc40445038)

[2.1.3 Lean/Kanban 5](#_Toc40445039)

[2.2 Practice 5](#_Toc40445040)

# Book - JIRA Agile Essentials

## Basic of JIRA Agile

JIRA Agile은 다음과 같은 두 가지 애자일 방법론을 지원한다.

* Scrum - 소프트웨어 개발 프로젝트에 적합
* Kanban - 운영 팀에 적합

### Issue and Issue type

JIRA Agile에서 Story나 Epic같은 모든 작업 단위는 지라에서의 이슈다.

* Epic - 더욱 세분화된 요구사항으로 나눠지지 않은 **대규모 사용자 스토리**를 나타낸다. 일반적으로 에픽은 대규모 개발 프로젝트에서의 **모듈**이나 **주요 컴포넌트**뿐 아니라, 여러 스토리들의 **테마**를 정의하는데 사용된다.
* Story - **구현되어야 하는 단일 기능**을 나타낸다. 일반적으로 스토리는 **최종 사용자 측면의 요구사항**을 기술한다. 이러한 이유로 인해 때때로 스토리는 기능의 바람직한 결과에 초점을 두고 비기술적인 언어로 작성된다.
* Technical task(기술 작업) - 기술작업은 서브태스크 이슈 타입으로 스토리를 구현하기 위해 완료되어야 할 필요가 있는 **실제 기술 작업**을 나타낸다.

## JIRA Agile for Scrum

### Scrum

#### Understanding Scrum Process

팀에 의해 구현되어야 할 필요가 있는 **일차적인 기능**과 **요구사항**의 **리스트**인 **백로그(backlog)**를 가진다. 항목의 백로그는 위에서 아래로 우선순위에 따라 나열된다. **제품 책임자**는 **백로그에 대한 담당자**로, 그의 비전에 따라 **우선순위를 정의**하지만, **팀의 모든 사람**은 **백로그에 새로운 항목을 추가**하고, **우선순위에 대해 논의**하고 **구현에 필요한 공수를 추정**함으로써 백로그에 대한 기여가 가능하다.

**스프린트 계획 미팅(sprint planning meeting)** 동안에 팀은 스프린트 범위(scope)를 결정할 것이다. 여기서의 핵심은 스프린트 마지막까지 팀은 완전히 테스트 되고, 잠재적으로 출시 가능한 모든 확정 기능이 포함되는 제품을 만들어야 한다는 점이다.

일반적으로 스프린트가 진행되는 동안 팀은 일일 스크럼 미팅(daily scrum meeting)을 해야 한다.

스프린트의 마지막에 팀은 팀원들이 참석해 그들이 이해 당사자를 위해 어떤 것을 만들었는지 보여주는 **스프린트 리뷰 회의**를 해야 한다. 회의 동안 제품이 형태를 갖춤에 따라 때때로 새로운 변경 사항이 나타나고, 이러한 변경사항들은 백로그에 추가된다.

스프린트의 마지막에 수행되는 또 다른 회의는 **스프린트 회고 미팅(sprint retrospective meeting)**이라 불린다. 회고에서 팀은 그들이 잘한 것은 무엇이고, 잘못한 것은 무엇인지, 그리고 어떻게 개선할 것인지를 함께 토론한다.

# Agile 방법론

### Scrum

가장 일반적인 애자일 방법으로 프로젝트 관리 및 제품 개발에 중점을 둔 소프트웨어 개발 프레임워크이다. 스크럼 프로젝트에서는 스크럼 패턴을 따르는 동일한 기간(보통 30일)의 주기인 sprint(스크럼 팀이 프로젝트를 시간 단위로 나눈 것)로 작업을 분할 한다. 모든 스프린트는 sprint planning 세션으로 시작하면서 스프린트 기간 동안 작업할 범위를 결정한다. 팀은 스프린트를 통해 프로젝트를 진행하며, 날마다 daily scrum이라는 짧은 미팅을 가진다. 스프린트 후반에 팀은 이해관계자들과 함께 sprint review를 실시하면서, 지금까지 만들어온 실행 가능한 소프트웨어를 시연한다.

스크럼 프로젝트에 존재하는 세 가지 핵심 역할

* Product owner: product backlog(아직 구축하지 않은 피처들을 정의한 것)를 관리
* Scrum master: 가시화된 문제점을 처리하기 위해 팀을 돕는다. 팀이 문제점들을 해결하면서 스크럼을 잘 진행할 수 있도록 도와준다.
* Development team members: 모든 개발 팀원

### XP

코드 및 소프트웨어 설계를 중시하는 방법론이다. 잘 설계되고 유지 보수가 쉬운 고품질의 코드를 작성하기 위한 소프트웨어 개발에 초점을 맞춘다.

### Lean/Kanban

소프트웨어 구축 방식을 지속적으로 개선하는데 집중한다.

Kanban은 프로세스 개선에 초점을 둔 애자일 방법론

## Practice

팀이 사용하는 수단이나 기술을 의미한다.

* Refactoring: XP 팀이 코드 구조를 개선하기 위해 지속적으로 수행하는 작업
* User story: 팀이 사용자의 요구사항을 이해할 수 있게 도와주는 것
* Task board: 애자일 계획 지원 도구
* Burndown chart: 프로젝트에 남은 작업량을 추적하는 차트
* Retrospective: 팀이 모여서 그들이 배운 교훈을 정리하는 시간
* Daily standup