

Atlassian과 Redhat이 함께하는 데브옵스 완전정복

품질 향상을 위한 지속적인 코드 인스펙션과 품질 가시화

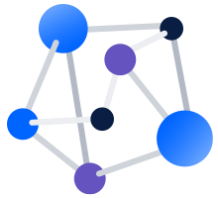
Version	Version 1.0
Date	2017.09.20
Speaker	설진호 이사
Company	커브

Agenda

- 코드 품질 관리의 주요 문제
- 지속적인 인스펙션
- 지속적인 인스펙션의 10가지 원칙
- 지속적인 인스펙션을 위한 솔루션
 - 데모 - SonarLint
 - 데모 - SonarQube
- 개발 프로세스에서 지속적인 인스펙션
 - 데모 - Inspection 자동화
- 지속적인 인스펙션의 효과

코드 품질 관리의 주요 문제

전형적 개발에서 정기 감사(Audit)는 프로세스에 따라 지정된 시간에 이루어지며 이러한 유형의 코드 품질 관리 접근에는 다음과 같은 단점이 있습니다.



너무 적은, 너무 늦은

정시 감사 일정으로 늦은 문제 발견
문제에 따라 프로젝트 일정 변경 필요



개발팀은 무관심

팀 외부에서 생성된 새로운 업무
늦은 스케줄로 개발자의 학습 필요



프로세스 오너십의 부재

개발 프로세스는 개발자
품질 프로세스는 감사자



이질적인 요구사항

다른 요구사항으로 각 어플리케이션 측정
공통적 품질 요구사항 적용이 불가능

지속적인 인스펙션

지속적인 인스펙션(Continuous Inspection)은 소프트웨어 개발 라이프사이클의 완전한 부분으로 소프트웨어 품질을 향상시키기 위해 설계된 코드 품질 관리를 위한 새로운 패러다임입니다.



지속적인 검사

지속적인 통합처럼 지속적인
검사를 수행



품질 가시화

프로젝트 이해관계자를 위한
품질 가시화



조기에 문제 해결

개발 초기부터 문제를 해결하
여 ROI 향상



문제 해결의 적시성

알림을 통한 최대한 빠른 해결
과 개발 훈련

Fortune 100
20,000 Developer
Application 5,000
600,000,000 Code Line

지속적인 인스펙션의 10가지 원칙

- 1 구성원은 소프트웨어 품질에 대한 의미 있는 데이터에 액세스 할 수 있어야 한다.
- 2 소프트웨어 품질 관리는 개발 초기부터 개발 팀의 궁극적인 책임 이어야 한다.
- 3 소프트웨어 품질 요구사항은 개발 프로세스의 일부이다.
- 4 소프트웨어 품질 요구사항은 객관적이어야 한다.
- 5 소프트웨어 품질 요구사항은 모든 소프트웨어 제품에 공통적이어야 한다.
- 6 소프트웨어 품질 데이터는 최신 코드에서 측정된 최신 버전이어야 한다.
- 7 소프트웨어 제품은 지속적으로 인스펙션되어야 한다.
- 8 새로운 품질 결함발생시 이해관계자는 즉시 알림을 받아야 한다.
- 9 개발팀은 소프트웨어 품질 이슈를 제어할 수 있어야 한다.
- 0 새로운 이슈 해결을 위한 명확한 경로와 일정이 지정되어 한다.

sonarlint



IDE에서 조직
품질 요구 검사



사전
이슈 수정



sonarqube



지속적인 코드
인스펙션 활성화



기술 부채의
제어

신뢰성, 보안성, 유지보수성, 코드 중복,
코드 사이즈, 복잡도

Continuous Inspection with Eclipse

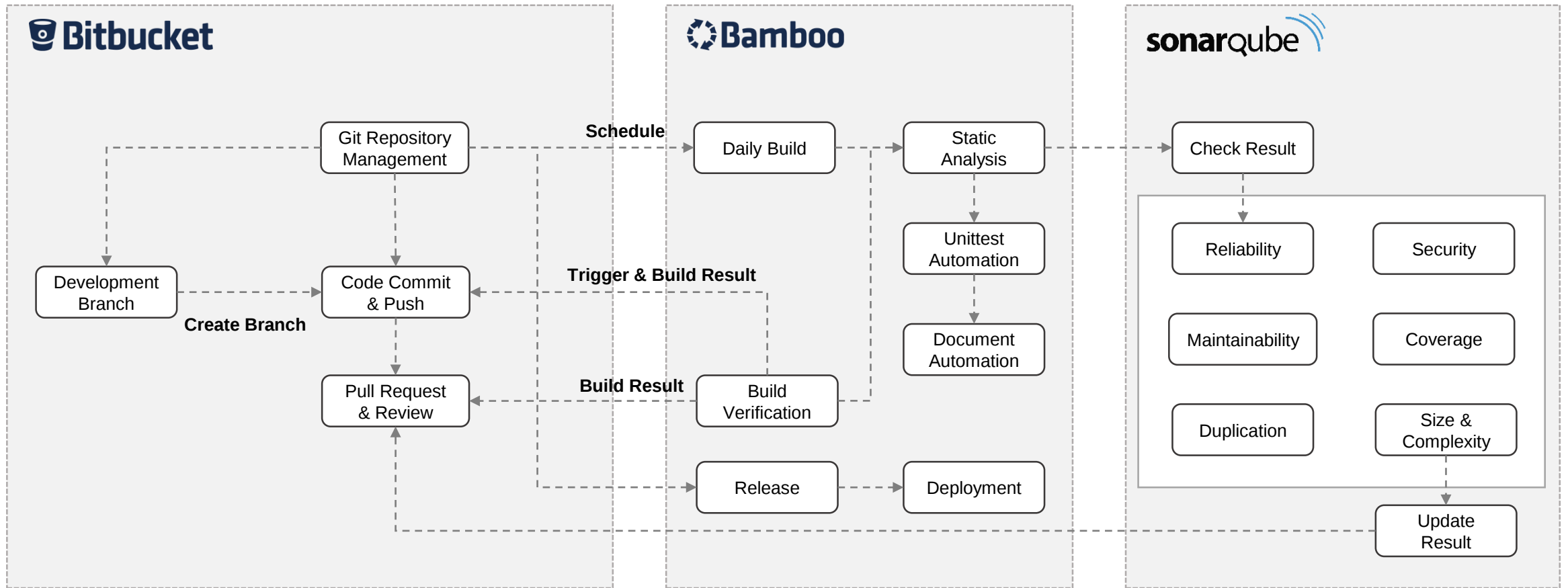
<https://youtu.be/WyrvaMxxinw>

Continuous Inspection with SonarQube

https://youtu.be/yL3_5W7wc7U

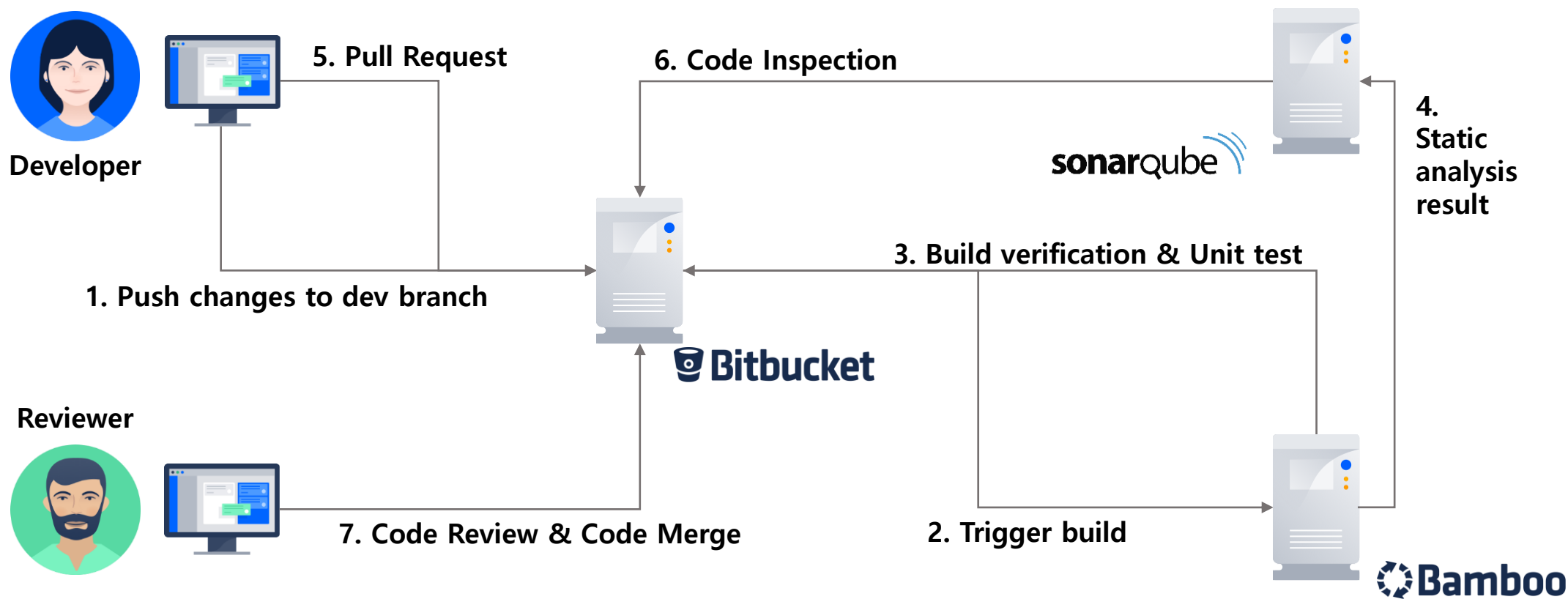
지속적인 인스펙션을 위한 솔루션

지속적인 인스펙션을 통해 고품질의 코드를 유지하기 위하여 Atlassian Bitbucket, Bamboo, Sonarsource의 SonarQube로 구성할 수 있습니다.



개발 프로세스에서 지속적인 인스펙션

다음 그림은 일반적으로 사용되는 품질 게이트키퍼의 주요 활동을 포함한 전체 솔루션의 흐름을 보여주고 있습니다.

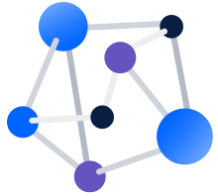


Code Inspection Automation

<https://youtu.be/53XMu921fyU>

지속적인 인스펙션의 기대효과

지속적인 인스펙션은 전형적인 개발에서 정기 감사(Audit)가 가지는 다양한 문제를 보완하고 다음과 같은 다양한 기대효과를 줄 수 있다.



너무 적은, 너무 늦은

- ✓ 팀은 품질에 대한 지속적인 피드백
- ✓ 시간이 지남에 따라 명확한 품질 향상 그림 제공
- ✓ 품질 결함 생성 시 즉각 알림
- ✓ 개발자의 지속적인 교육과 선순환



개발팀은 무관심

- ✓ 품질 수행 계획은 개발 프로세스에 통합
- ✓ 소프트웨어 품질 정보에 실시간 액세스
- ✓ 팀은 더 나은 소프트웨어를 개발할 수 있는 능력



프로세스 오너십의 부재

- ✓ 코드 품질은 개발 팀의 소유
- ✓ 소프트웨어 품질 도구가 조직 전역에 제공
- ✓ 외부 컨설턴트 없이 즉시 이해 가능
- ✓ 지속적인 교육을 통해 장기적으로 품질 향상



이질적인 요구사항

- ✓ 품질 판단은 객관적인 기준에 따라 자동화로 이뤄짐
- ✓ 새로운 코드, 변경된 코드, 전체 코드 기반으로 소프트웨어 품질을 측정
- ✓ 팀은 새로운 문제의 주입을 추적할 수 있음.

감사합니다.
점심식사 맛있게 드세요.

