

2014. 03. 17. [제87호]

# 코드리뷰(Code Review), 과연 유용한 것인가

소프트웨어공학센터 경영지원TF팀

## C o n t e n t s

- ▶ 서론
- ▶ 도구기반 코드리뷰의 차별성
- ▶ 결함이 실제로 수정됐는지 확인하기
- ▶ 결함에 대한 비용에 대해 생각하기
- ▶ 자명한 사실

## Key Message

코드리뷰(code review)는 개발자가 작성한 코드를 다른 개발자가 정해진 방법을 통해 검토하는 일을 말하며, 등위 검사, 제3자 검사라고도 함. 그러나 한 조사결과에 따르면 개발자들이 코드리뷰를 수행하지 않는 가장 큰 이유는 시간부족 때문임. 따라서 과연 코드리뷰는 개발자들이 완료일정의 압박에도 불구하고 수행해야 할 가치가 있는 것인지 또는 다른 대응요소가 존재하는 것인지에 대한 견해를 제시함.

## ▶ 서론

- 미 IT전문지에서 600명 이상의 개발자들을 대상으로 수행한 최근 조사에 따르면 코드리뷰를 수행하지 않는 가장 큰 이유는 개발자들에게 시간이 없다는 것임
- 오늘날 모든 현대인들이 시간에 대해 압박을 받고 있으며, 모든 개발자들에게도 가장 중요한 문제 중 하나임
- 코드리뷰에 시간이 많이 걸리는 것이 문제라기보다는 기업 또는 개발팀에서 코드리뷰에 드는 시간을 할당할 여유가 없다는 것이 문제임
- 일부에서는 형식적이고 수동적인 코드리뷰에는 많은 시간이 소요되고, 대부분의 사람들은 의심의 여지없이 코드리뷰에 시간을 들이는 것은 비생산적인 것이라고 주장할 수 있음
  - 여기서 “형식적이고, 수동적인 코드리뷰”란 개발자들이 코드를 검토하는 회의를 의미함
- 물론 다른 유형의 코드리뷰도 있는데 오버더숄더(Over-the-shoulder) 코드리뷰<sup>1)</sup>가 도움이 될 수 있지만, 일반적으로 관련된 두 사람에게 제한된다는 단점이 있으며, 개발자가 다른 개발자에게 피드백을 위해 코드를 이메일로 보내는 이메일 기반의 코드리뷰도 마찬가지로 할 수 있음
- 그렇다고 코드리뷰가 유용하지 못하는 답을 경솔하게 내릴 수는 없으며 다음의 내용을 통해 코드리뷰의 유용성 여부를 판단하도록 함

## ▶ 도구기반 코드리뷰의 차별성

- 도구기반의 코드리뷰는 기존 코드리뷰와의 차별성을 확보할 수 있다는 점을 확인

1) 오버더숄더(Over-the-shoulder) 코드리뷰: 코드개발자가, 또 다른 개발자가 코드를 변경하는 작업을 지켜보면서 키보드와 마우스를 이용해 다양한 파일을 열고 변경된 부분을 가리키고 설명하면서 리뷰를 진행하는 것을 말함

할 필요가 있음

- 도구기반 코드리뷰를 통해 해당 개발자들은 피드백을 위해 팀 전체에 코드를 전송하여 각각의 코드리뷰 결과들을 확인하고 공유할 수 있음
- 이런 협력은 기존의 이메일/오버더숄더 리뷰 등 서로 학습하는 두 사람 뿐만이 아니라, 전체 팀을 향상시킬 수 있는 방법임
- 그러나 이러한 방식은 너무 많은 시간을 요구하는 것은 아닌지 즉 코드를 리뷰하는데 걸리는 시간에 대한 보상가치가 있는지를 질문해봐야 하며 이런 질문에 답하기 위해서는 우선 코드를 리뷰하는데 걸리는 시간을 다음의 사례를 통해 살펴봐야 함
- 시스코(Cisco)에서 실시한 코드리뷰 조사에 따르면 최적의 효율성을 위해서는 개발자가 한 번에 200~400 사이의 코드 라인(LOC)을 검토해야 하는 것으로 나타남
  - 200~400 사이의 코드 라인을 넘어가면, 결함을 발견할 수 있는 능력이 감소함
- 200~400 사이의 코드 라인을 60~90분을 넘지 않는 속도로 검토하면, 70~90%의 수확을 얻을 수 있음
  - 다시 말해, 10가지 결함이 존재하는 경우 7~9가지 결함을 찾을 수 있다는 것임
- 시스코사는 10개월간 모니터링 후에 이 조사를 토대로 다음과 같은 결론을 내림
  - 위에서 제시한 빠른 속도로 수행한 경량 코드리뷰는 기존의 형식적인 코드리뷰 만큼이나 효율적이면서도, 실행에 있어서 실질적으로 더 빠르고 불편이 덜함
  - 경량 리뷰들은 형식적인 리뷰보다 적은 평균 6시간 반 정도 걸리지만, 더 많은 버그들을 발견할 수 있었음

## ▶ 결함이 실제로 수정됐는지 확인하기

- 버그를 발견하기 위해 코드를 검토하는 모든 문제에 있어서, 확실히 버그들을 수정하는 것이 중요함
- 대부분 검토하는 동안 발견된 결함들을 추적하고 검토를 마치기 전에 실제로 버그들이 수정되었는지 확인하는 좋은 방법을 코드를 검토하는 많은 팀들은 가지고 있지 않음
  - 기존 이메일 또는 오버더숄더 리뷰들에 있어서 결과들을 확인하는 것은 특히 어려움
- 버그들은 코드가 QA(Quality Assurance)에게 배포되기 전에 발견되기 때문에, 이러한 버그들은 일반적으로 버그 추적 로그에 입력되지 않음을 명심해야 함
  - 코드가 “모두 삭제”명령을 하기 전에, 결함이 수정되었는지 확인하는 좋은 방법을 찾아야 함
- 리뷰에서 발견된 결함을 추적하기 위해 버그 추적 시스템과 통합되는 우수한 공동 검토

소프트웨어를 사용할 것을 권장함

- 적합한 도구를 가지고 검토자들은 버그를 기록할 수 있으며, 필요에 따라 제작자들과 함께 버그에 대해 논의할 수 있음
- 제작자들이 문제들을 수정하고 검토자들에게 통보한 다음, 검토자들은 각 문제들이 해결되었는지 확인해야 함
- 도구사용에 있어서 주의해야할 점은 검토 중에 발견된 버그를 추적하고, 모든 버그가 수정된 것으로 검토자에 의해 확인되거나 이후에 해결하기 위한 별도의 작업 항목으로 추적되기까지는 검토를 완료하는 것을 금지해야 한다는 것임

### ▶ 결함에 대한 비용에 대해 생각하기

- IT전문지인 SmartBear는 여러 해 동안 결함에 대한 비용에 대해 단계에 따라 매우 단순한 측정법을 제시했으며, 측정법은 일반적으로 다음과 같음
  - 요구사항 동안 발견된 결함들 = \$250
  - 설계 동안 발견된 결함들 = \$500
  - 코딩과 테스트 동안 발견된 결함들 = \$1,250
  - 출시 후 발견된 결함들 = \$5,000
- 물론 이러한 수학적 제시는 회사나 프로젝트 규모에 따라 매우 다를 수 있다는 것을 인지하고 있으나 중요한 것은 출시 후 발견된 결함 수정에 대한 비용은 기하급수적으로 증가한다는 것을 확인할 수 있음
  - 개발이 출시에 도달하면 5,000달러까지 증가되지만, 요구 사항 단계에서 발견된 결함의 원래 가격은 250달러임
- 출시 전에 발견한 모든 결함은 결과적으로 높은 수익을 발생시킨다고 할 수 있음
  - 즉 출시 후 발견된 결함이 5000달러의 비용을 발생시킨다고 가정한다면, 과정 중에 더 일찍 발견된 모든 결함은 훨씬 더 비용을 절감시킴

### ▶ 자명한 사실

- 높은 수준의 소프트웨어는 적절한 개발프로세스 없이는 얻기 어려움
- 높은 수준의 소프트웨어는 결함이 최소화되고 계획된 품질을 보장해야 한다는 측면에서

코드리뷰는 매우 유용하다는 것을 발견할 수 있으며, 그 유용성의 확보는 시간 절약 측면에서 도구기반 코드리뷰를 수행하는 것이 좀 더 효율적이라는 것을 알 수 있음

- 결론적으로 코드리뷰는 개발제품의 품질향상을 위해 가치가 있으며 이를 위해 소요되는 시간은 유용하다는 것임

#### 참고 자료

1. <http://devblog.wesmcclure.com/posts/the-value-of-code-reviews>
2. <http://travisthetechie.com/2013/03/the-value-of-code-reviews.html>
3. <http://www.ibm.com/developerworks/rational/library/11-proven-practices-for-peer-review/>
4. <http://support.smartbear.com/resources/cc/book/code-review-cisco-case-study.pdf>