코드의 바다에서 정적 분석이라는 등대를 찾다

최예원

2022.09.09

Abstract

본질적으로 프로그램은 특정한 규칙을 기반으로 하며, 규칙을 도입함으로써 오류 혹은 보안 문제와 같은 프로그램이 일으킬 수 있는 비의도적인 결과를 예방할 수 있다. Facebook은 정적 분석 도구인 Infer와 Zoncolan을 개발하고 도입하였다. Infer 및 Zoncolan을 개발 흐름에 포함하여 코드 작성에 제한을 더함으로써 오류를 빠르게 감지하고 수정하는 개발 문화를 구축했다. 또한 Infer와 Zoncolan 자체의 성능을 발전시켜 높은 정확성과 속도, 큰 규모를 만족하도록 하여 일반적인 코드에 사용할 수 있을 정도의 성과를 이룩했다. 이를 통해 방대한 코드 내에 잠재한 오류 및 보안 취약점을 효과적으로 찾고 해결한다는 목적을 성공적으로 달성하였다. 이처럼 정적 분석과 같은 규칙을 도입하여 생산성을 증대하고자 한다면 상황과 이루고자 하는 가치를 고려하여 외부, 내부의 동작을 적절하게 고안할 필요가 있다.

개발자에게 자유와 규칙 중 무엇을 선호하는지 묻는다면, 언뜻 보기에 대부분은 자유를 택할 것으로 보인다. 가능한 한 자율성을 부여하는 것이 이상적이므로, 개발자가 더욱 용통성 있게 작업할 수 있도록 돕는 것이 규칙을 지키도록 강제하는 것보다 중요할 것만 같다. 그러나 실제로는 그렇지 않다. 많은 개발자들이 Javascript에 불편을 느끼고 Typescript를 사용하며, 안전한 시스템 프로그래밍 언어로서 주목받은 Rust에는 소유권 개념을 도입함으로써 이룩한 철저한 메모리 안정성과 철저한 타입 규칙이 있다.

이와 같이 규칙 속에서 편안함을 느끼는 데는 프로그램이 실질적으로 몇 가지 규칙 안에서 움직이기 때문이다. 통사 구조(syntax) 와 의미 구조(semantic)는 프로그래밍 언어를 정의하며, 이를 기반으로 프로그램을 작성한다는 것은 결국 인간의 비정형적인 사고를 유한한 규칙의 집합으로 변환하는 것이다. 프로그램을 자유롭게 작성할수록 변환 과정에서 오류가 발생할 가능성이 높아지며, 오히려 개발자의 사고를 특정 프로그래밍 언어의 구조 안에 제한함으로써 프로그램의 정확성을 보장한다. 언젠가 자연어 수준의 통사 구조 및 의미 구조를 가진 프로그래밍 언어가 등장할 것인가? 불가능에 가깝다고 여긴다. 이와 같이 규칙은 프로그래밍을 성립시키는 요소 중 하나이며, 적절한 규칙은 프로그램의 정확성을 높이는 데 도움을 준다.

프로그램 분석의 종류인 정적 분석 역시 결과적으로 프로그램 작성에 제한을 주는 방법으로 볼 수 있다. 작성한 프로그램에서 통사구조 위반 뿐 아니라 프로그램 실행 시 나타날 수 있는 잠재적인 오류 또는 보안 취약점을 찾아냄으로써 프로그램의 의도에 맞추어 수정하도록 지침을 제공한다. 정적 분석은 개발 초기 단계에서 개발자를 번거롭게 만드는 대신, 출시 시점에서 발견되면 큰 타격을 주는 문제들을 사전에 감지하여 위험을 경감한다.

정적 분석 도구인 Infer와 Zoncolan를 이용한 Facebook의 성공을 이러한 맥락에서 이해한다면, 개발 절차에 새로운 규칙을 도입하여 생산성을 증대한 것으로 볼 수 있다. 주 개발 흐름과 별개로 분리되어 있던 정적 분석을 코드 리뷰 과정의 일부로 만들어 개발자가 일과의 하나로서 정적 분석 결과를 검토하도록 만들었다. 엄밀히 말해 강제한 것은 아니지만, 전적으로 개발자의 자율에 맡기던 방식을 실패한 후 새로 적용한 이 개발 절차는 개발자가 문제를 빠르게 인지하고 해결하도록 유도했다.

그러나 과도한 규칙은 오히려 발목을 잡는 결과로 이어질 수 있다. 개발 절차가 개편되었어도 막상 정적 분석 도구가 유용하지 않았다면 오히려 생산성을 저하시키는 문제를 낳았을 것이다. Infer와 Zoncolan의 사례에서는 두 정적 분석 도구의 외부적인 요소 뿐만이 아니라 정적 분석 기술 자체와 같은 내부적인 요소 역시 훌륭하게 고안되었다. 개발자가 코드 수정을 제출하는 변경 시간(diff-time) 마다 프로그램 분석을 수행하는 대신, 개발 속도를 과도하게 저해하지 않도록 구성성을 준수하였다. 하위 요소에 대한 분석을 합쳐 상위 요소를 분석하는 방식을 취함으로써 전체 요소가 아닌 변경된 요소에 대한 분석이 수행되도록 하고 병렬 분석을 가능하게 하여 높은 속도를 보인다. Infer는 또한 동시성 오류, 함수 간(interprocedural) 오류와 같은 일반적으로 탐지하기 어려운 오류를 보고할 수 있도록 발전하였다. Infer는 현재 오픈소스로 공개되어 일반적인 코드를 위한 정적 분석 도구로서 사용되고 있다. 보안 취약점을 잡아내는 Zoncolan은 변경 시간 분석에 더하여 전체적인 분석을 주기적으로 수행하며, 개발자와 정적 분석 기술자와의 긴밀한 되먹임을 통해 높은 정확도와 효과적인 문제 보고를 이루었다.

규칙은 개발자에게 올바른 코드로 가는 길을 알려주는 등대와 같다. 자유 속에서 개발자는 오히려 길을 잃고 파멸적인 결과를 초래하는 방향으로 나아갈 수 있다. 정적 분석이라는 규칙을 보다 적극적으로 도입함으로써 오류를 해결한 Facebook의 사례와 같이 적절한 규칙을 효과적으로 적용하여 개발자가 안정적인 코드를 작성하도록 도울 수 있다. 그러나 또한 과도한 규칙이 개발을 방해하지 않도록 유의해야 한다. 생산성을 증가시키기 위한 최적의 규칙을 찾아내고 적절하게 도입하는 방법을 찾아내는 것은 끝없는 과제로 남아있다.