Ciągła integracja

Większości deweloperów zna jest dobrze znane pojęcie „ciągłej integracji" (Continuous Integration, w skrócie CI). Jak wspomnieliśmy w rozdziale 2, termin ten oznacza, że przed każdą operacją ewidencjonowania (*check-in*), kod jest automatycznie kompilowany i weryfikowany przez automatyczne testy. Zmiany ewidencjonowane są tylko wtedy, gdy wszystkie testy kończą się sukcesem. Proces, który kompiluje kod i uruchamia testy, zwykle działa na jednym lub kilku dedykowanych serwerach kompilacji, a nie na maszynie dewelopera. Pozwala to na scentralizowane sterowanie tym procesem, a także umożliwia wykonywanie innych zadań na maszynie dewelopera w czasie działania tego procesu.

POMNIEJSZE WARIANTY CIĄGŁEJ INTEGRACJI

Choć powyższy opis ciągłej integracji (CI) jest obecnie najbardziej typowy, istnieją również inne warianty tego pojęcia. Warianty te były częściej stosowane w przeszłości, przy czym dzisiaj nadal są one dosyć powszechne. Ponieważ są one również prostsze, czasem korzysta się z nich w mniejszych zespołach.

1. Zamiast kompilować kod i uruchamiać testy *przed* wprowadzeniem zmian do głównego repozytorium kontroli kodu, procesy te wykonywane są *po* zakończeniu tej operacji. Ponieważ w takim wypadku niemożliwe jest powstrzymanie operacji check-in, proces kompilacji (który obejmuje kompilację i testy) raportuje jedynie, czy proces zakończył się sukcesem, czy niepowodzeniem. Często rezultaty te są również wysyłane automatycznie do odpowiednich osób poprzez e-mail, a głównie do dewelopera, który dokonał operacji check-in. Jeśli kompilacja zakończy się niepowodzeniem, dobrą praktyką jest, aby deweloper jak najszybciej naprawił ten błąd, zanim ktokolwiek inny będzie mógł zaewidencjonować inne zmiany.
2. Drugi wariant stosowany jest zwykle w bardzo małych zespołach lub gdy nikt nie posiada umiejętności pozwalających na zainstalowanie i skonfigurowanie serwera kompilacji. Choć wariant ten jest zwykle mniej preferowany, gdyż opiera się on bardziej na samodyscyplinie niż na narzędziach, to nadal wyraża on istotę i ideę stojącą za pojęciem CI. W wariancie tym każdy programista pobiera najnowszy kod, kompiluje i uruchamia testy *lokalnie*, i wykonuje operacje check-in tylko wtedy, gdy wszystkie testy kończą się sukcesem.

Przejście z kompilacji nocnych do CI może stanowić pewne wyzwanie. Ale ostatecznie uzyskiwane w ten sposób korzyści przewyższają te problemy. Więcej informacji na temat właściwego sposobu dokonania tego przejścia można znaleźć w rozdziale 15.

Tworzenie oprogramowania sterowane testami akceptacyjnymi

Choć ciągła integracja pozwala odpowiedzieć na pytania dotyczące tego kto, kiedy i jak powinien *uruchamiać* testy, nie daje ona odpowiedzi na pytania: kto, kiedy i jak powinien *pisać* testy. W rozdziale 3 udzieliliśmy odpowiedzi na pytanie „*kto* powinien implementować testy". Ponadto w rozdziałach 2 i 3 mówiliśmy o tym, dlaczego lepiej jest pisać testy podczas opracowywania funkcji, a nie po ich ukończeniu. W rozdziale 16 omawiamy ten