

# Continuous Integration

# Co to jest Continuous Integration

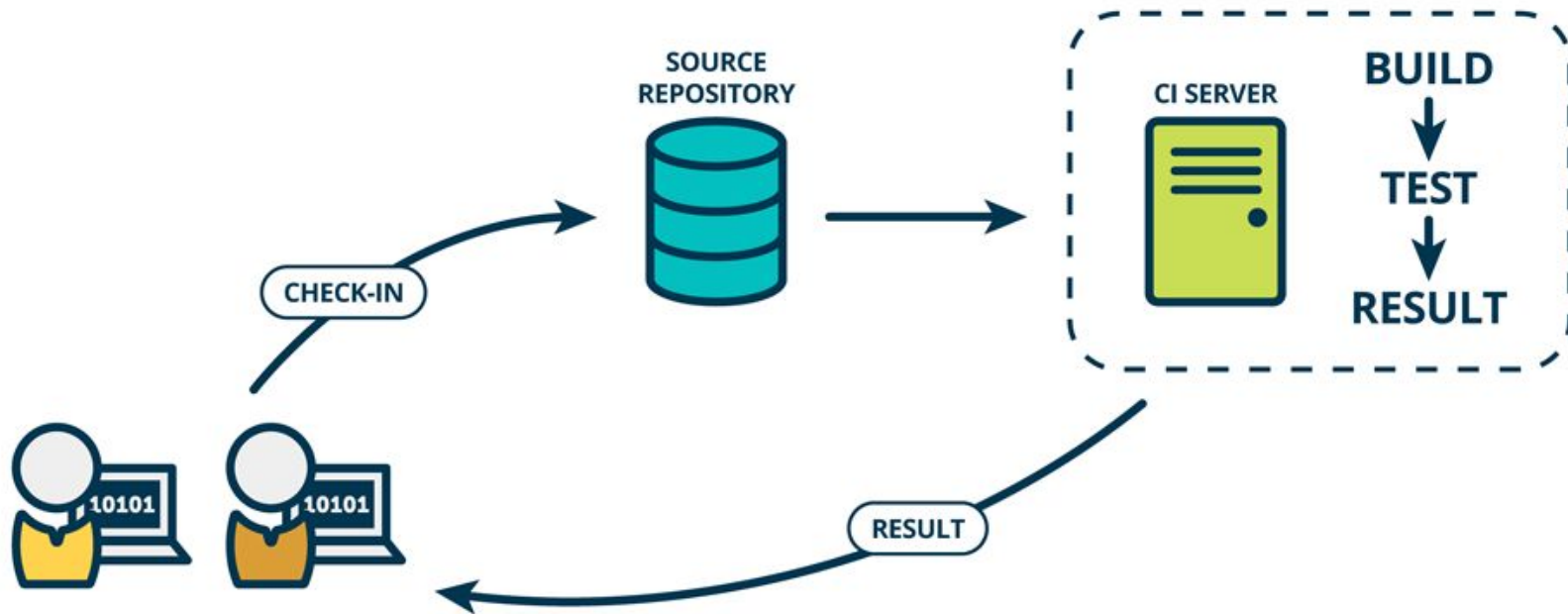
Ciągła Integracja

Ciągłe dostarczanie

„CIĄGŁA INTEGRACJA NIE ELIMINUJE BŁĘDÓW, ALE SPRAWIA, ŻE ICH ZNALEZIENIE I  
USUNIĘCIE JEST DRASTYCZNIE ŁATWIEJSZE.”

*Martin Fowler*

# Ciągła integracja - podstawowy schemat



# Piekło...

## Integration Hell

W tradycyjnych metodykach (np. Waterfall) integracja była jednym z ostatnich etapów tworzenia produktu. W takim wypadku połączenie wielkich części kodu bywa bardzo kłopotliwe i pracochłonne, prowadzi do opóźnień projektu.

- jak efektywnie przetestować wielkie części kodu 'nagle' ze sobą połączone?

# Czy można coś zrobić?

- praca na wspólnym repozytorium kodu
- integracja jak najczęściej i jak najmniejszych kawałków kodu
- ciągłe testowanie
  - różne poziomy testów
    - testy modułowe
    - testy integracyjne
    - smoke testy
    - testy systemowe
    - analiza statyczna kodu
    - testy akceptacyjne

Założenie: każdy commit może prowadzić do wersji produkcyjnej

# Podstawowe założenie

## BUILD FAST

Commitujemy drobne części kodu

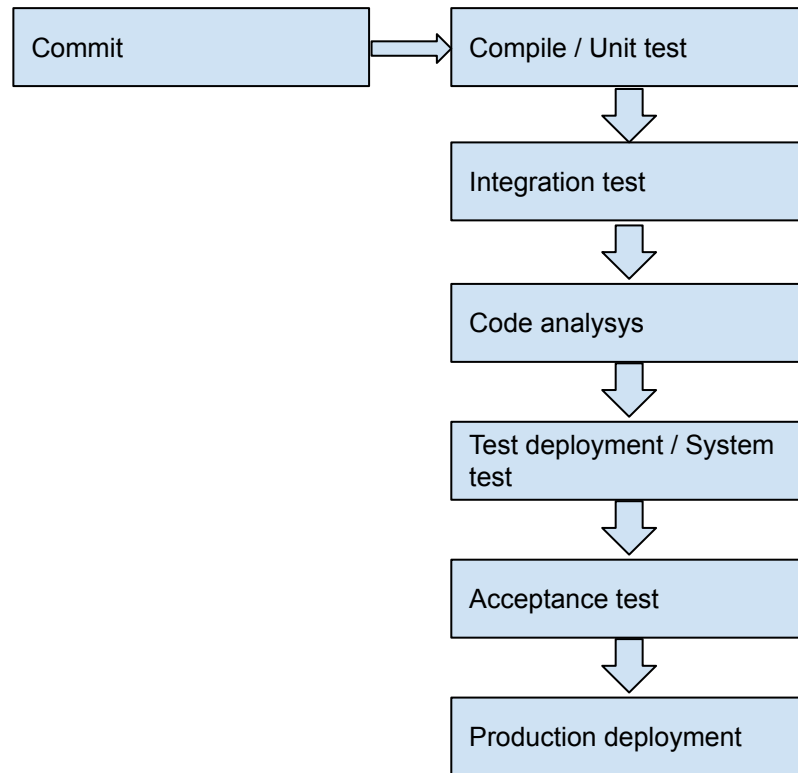
Następuje proces automatycznego budowania i testowania

## FAIL FAST

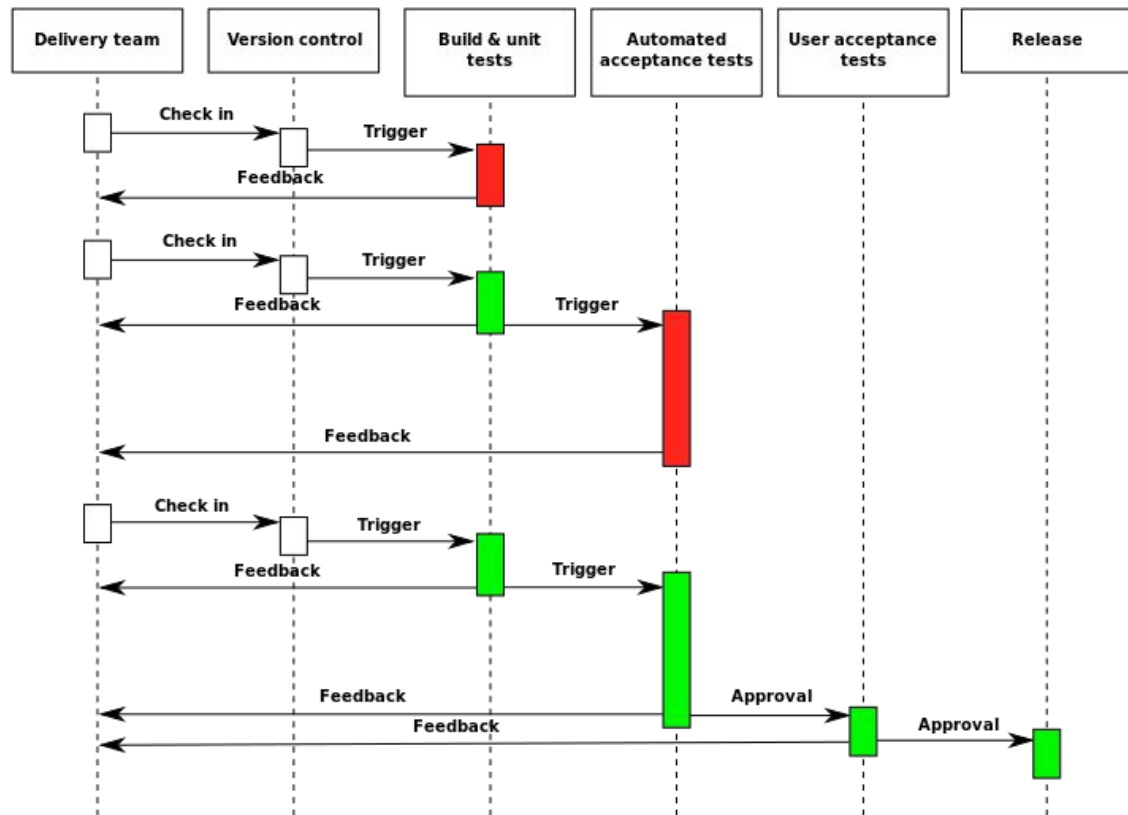
Jeżeli jest błąd, system powinien zwrócić go jak najwcześniej

# Build pipeline

- Każdy commit uruchamia cały łańcuch zdarzeń
- Wszystko (z wyjątkiem fazy akceptacyjnej) powinno odbywać się w sposób automatyczny



# Build pipeline - inny widok





# Potrzebujemy narzędzi

- Wspólne repozytorium kodu
  - GIT, GitLab, GitHub
- Narzędzia do testów automatycznych
  - Unit testy - specyficzne dla języka, NUnit, JUnit, dotCover
  - Testy integracyjne
  - Testy UI - Selenium
  - Testy wydajnościowe - JMeter
  - Analiza statyczna - SonarQube, Lint
- Narzędzia do deploymentu
  - Octopus, Docker

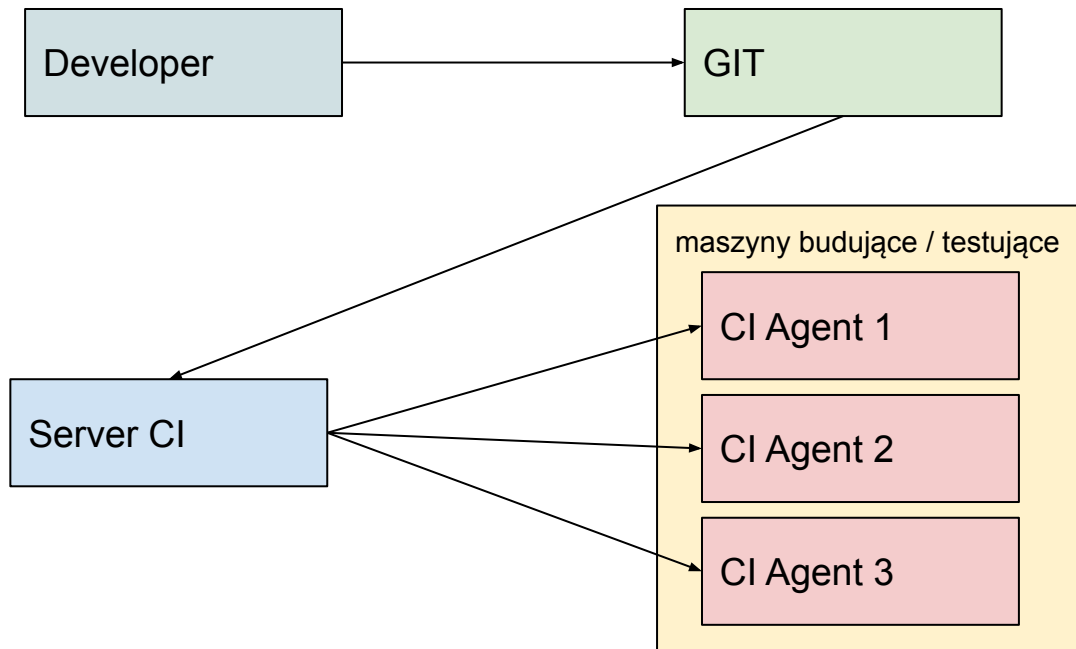
# The One To Rule Them All

System Continuous Integration



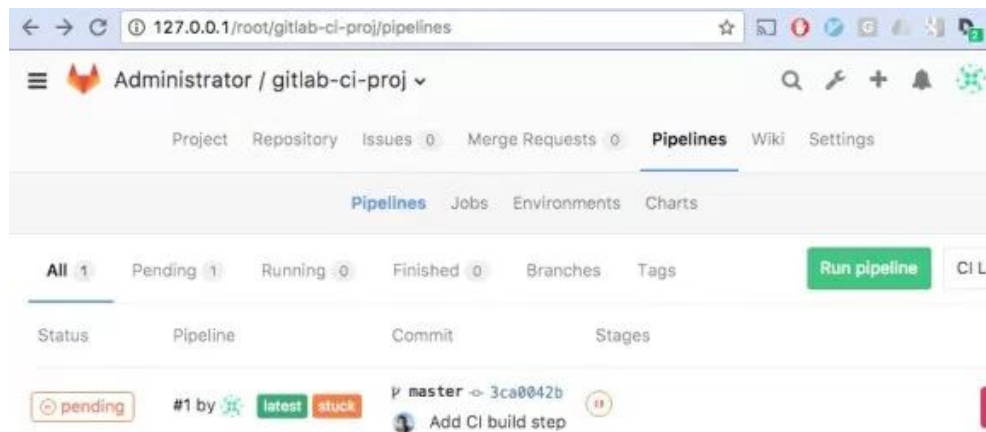
# Jak zbudowany jest system CI

- Serwer sprawdza zmiany w repozytorium kodu
- Przydziela zadania agentom - budowanie systemu, testowanie
- W przypadku błędu wysyła powiadomienie



# GitLab

- w porównaniu do dedykowanych serwerów CI daje proste możliwości konfiguracji
- konfiguracja odbywa się za pomocą plików YAML
  - + łatwe przechowywanie i wersjonowanie konfiguracji
  - dość wysoka bariera wejścia, sporo nauki na początek



# Przykład pliku yml

```
stages:
  - compile
  - test
  - package

compile:
  stage: compile
  script: cat file1.txt file2.txt > compiled.txt
  artifacts:
    paths:
      - compiled.txt

test:
  stage: test
  script: cat compiled.txt | grep -q 'Hello world'

package:
  stage: package
  script: cat compiled.txt | gzip > packaged.gz
  artifacts:
    paths:
      - packaged.gz
```

- definiujemy kroki
- opisujemy, co te kroki mają robić


# Widok pipelines

The screenshot displays the GitLab interface for the 'dvwa' project, specifically the 'Pipelines' view. The left sidebar shows the project structure with links to Overview, Repository, Issues, Merge Requests, CI/CD, Pipelines, Jobs, Schedules, Environments, Clusters, and Charts. The main content area shows a list of pipelines with their status, commit hashes, and execution times.




Status	Pipeline	Commit	Stages	Duration	Time Ago
passed	#11 by latest	master -> a643195e Unrelated changes	✓	00:00:27	less than a minute ago
passed	#10 by	master -> 02606f81 Proper fix	✓	00:00:28	less than a minute ago
failed	#9 by	master -> a4355eaf Fix attempt	✗	00:00:28	less than a minute ago
failed	#8 by	master -> cf821756 New feature	✗	00:00:31	less than a minute ago
passed	#7 by	master -> 8ec4fc20 Use tag	✓	00:00:33	a day ago

# Pipeline

Gitlab Org / Gitlab Ce



This project Search



Project

Activity

Repository


Pipelines

Graphs

Issues 7


Merge Requests 7

Wiki



Pipelines Builds Environments Cycle Analytics

 running

Pipeline #42 triggered 8 months ago by  ernstvn

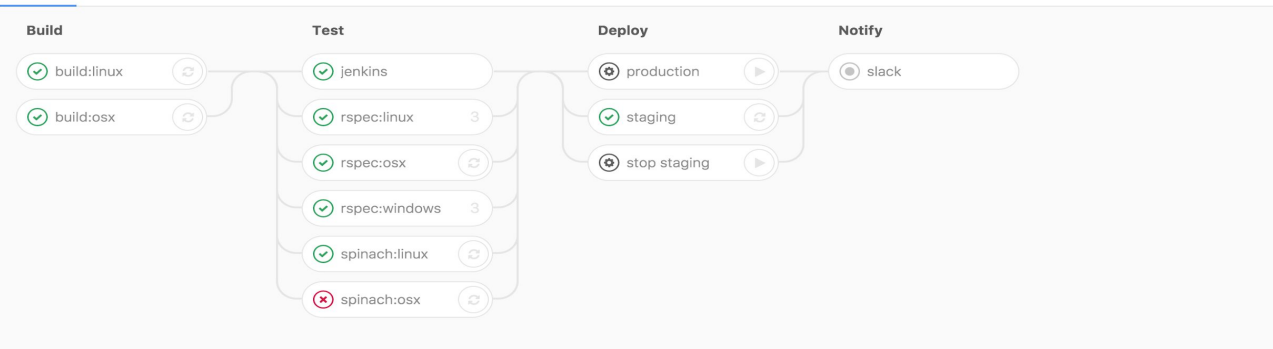
Retry failed

## remove need for schedules

 17 builds from docs/membership-refactor (queued for 7 seconds)

 e3c12332  

Pipeline Builds 17



# Wykresy

Informacje statystyczne nt przebiegów build pipeline - ile było prawidłowych, ile nieprawidłowych, itp.

