howest
hogeschool

Continuous Integration Basics

Graduaat Programmeren

howest
hogeschool

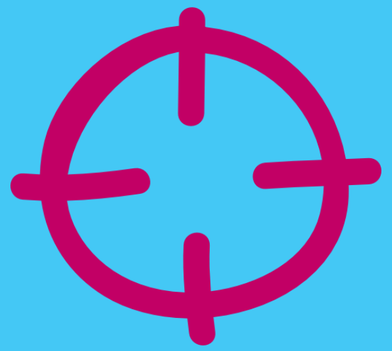
Wie is wie?

Kleine introductie

howest
hogeschool

DOEL?

Waar werken naartoe?



Doel

- Basis van Continuous Integration
- Gebruik van git aanleren
- Toepassen van git onder de knie krijgen
  - Lokaal (init – add – commit)
  - Remote (clone – push – pull)
  - Branches
  - Gitignore
  - Git fixes (revert – reset)
  - Conventies
- Git in combinatie met GitHub *(GitHub Classrooms)*

Opdrachten / Labo’s

- Labo’s **verplicht** te maken + **aanvullende Leho toets**
- 6-tal labo’s
- **40%** van je totaalscore ***(60%** examen)*

Evaluatie(s) voor de eerste examenkans

Moment	Vorm	%	Opmerking
examenperiode 1 (1e sem) binnen examenrooster	examen: andere vorm of combinatie van vormen	60,00	
examenperiode 1 buiten examenrooster	permanente evaluatie: andere vorm of combinatie van vormen (Permanente evaluatie)	40,00	Een gemiddelde score van 8/20 op de permanente evaluatie is vereist om te kunnen slagen voor deze module in EK1.

Evaluatie(s) voor de tweede examenkans

Moment	Vorm	%	Opmerking
examenperiode 3 (augustus/september) binnen examenrooster	examen: andere vorm of combinatie van vormen	100,00	

howest hogeschool

howest
hogeschool

NUT?

Het nut van Continuous Integration

Innovative Project

Workplace Experience

Programming Innovation

Programming Integration

Workplace Simulation

Programming Expert

Web Backend

Programming Basics / Advanced

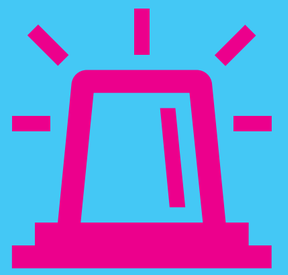
Web Frontend Basics / Advanced

Continuous Integration

howest
hogeschool

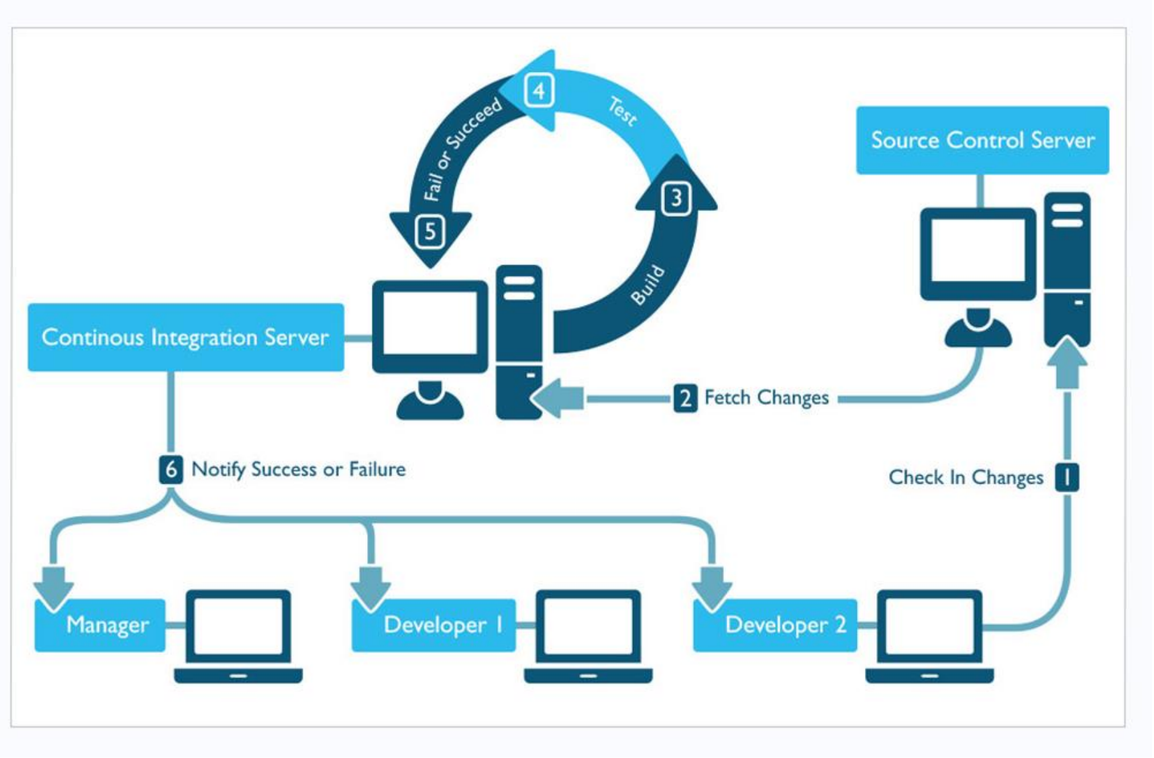
Continuous Integration

Wat is Continuous Integration?

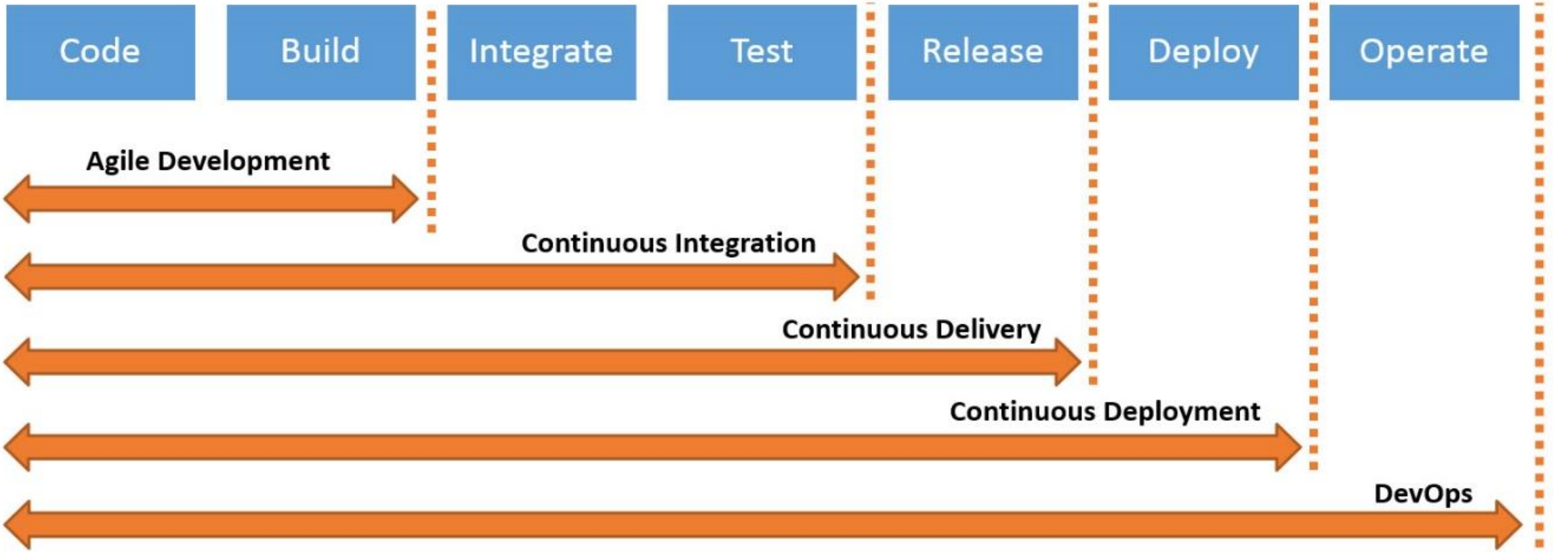


Overzicht

- Wat is CI?
- Plaats van CI in softwareontwikkeling
- Verschil tussen
  - Continuous Integration
  - Continuous Delivery
  - Continuous Deployment



Continuous Integration, Delivery, Deployment



howest hogeschool

Continuous Integration - Definitie

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.

- Martin Fowler https://www.martinfowler.com/

Continuous Integration – Stappen softwareontwikkeling

1. Haal de nodige bestanden op vanuit de source code repository
2. Breng wijzigingen aan in de code
3. Klik op ***Build*** in het Visual Studio-menu *(en hoop dat alles wordt gecompileerd)*
4. Ga terug naar **stap 2** *(je had waarschijnlijk een of meer compilatiefouten)*
5. Voer **Unit Tests** uit en hoop dat alle vinkjes groen kleuren *(we gaan ervan uit dat er Unit Tests zijn)*
6. Ga terug naar **stap 2** *(een of meerdere Unit Tests falen, er dient nog iets aangepast te worden)*
7. Wijzig je code, zodat deze begrijpelijker/leesbaarder wordt, ga vervolgens terug naar **stap 5**
8. Maak een update van je code naar de source code repository

Continuous Integration – Stappen CI

1. Haal de nodige bestanden op vanuit de source code repository
2. Breng wijzigingen aan in de code
3. Klik op ***Build*** in het Visual Studio-menu *(en hoop dat alles wordt gecompileerd)*
4. Ga terug naar **stap 2** *(je had waarschijnlijk een of meer compilatiefouten)*
5. Voer **Unit Tests** uit en hoop dat alle vinkjes groen kleuren *(we gaan ervan uit dat er Unit Tests zijn)*
6. Ga terug naar **stap 2** *(een of meerdere Unit Tests falen, er dient nog iets aangepast te worden)*
7. Wijzig je code, zodat deze begrijpelijker/leesbaarder wordt, ga vervolgens terug naar **stap 5**
8. Maak een update van je code naar de source code repository

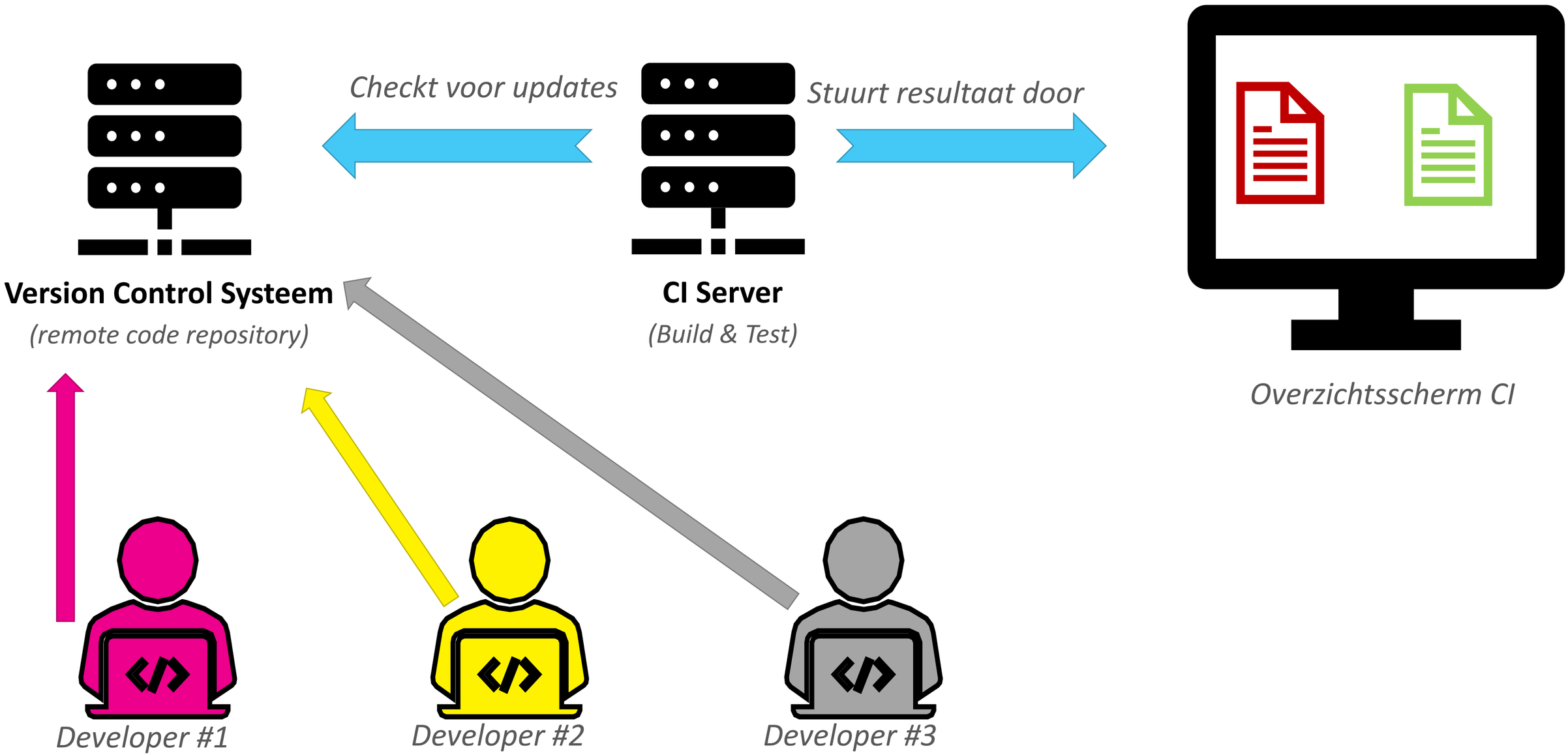
howest hogeschool

Continuous Integration – Stappen CI

Stap 1 t.e.m. 8, **aangevuld** en **ondersteund** door:

9. Een **geautomatiseerd systeem** houdt je code repository in de gaten en als er updates gemaakt worden haalt deze de laatste versie van de code op
10. Het geautomatiseerd systeem **build vervolgens de code**
11. Na een **succesvolle build** zal dit systeem ook alle **Unit Tests** gaan uitvoeren
12. Na deze stappen stuurt dit systeem de nodige **feedback** naar de developer/het development team. Op deze manier is iedereen op de hoogte van de huidige status van het softwareproject.

howest hogeschool


Checkt voor updates
Stuurt resultaat door
Version Control Systeem
(remote code repository)
CI Server
(Build & Test)
Overzichtsscherm CI
Developer #1
Developer #2
Developer #3

Nut van Continuous Integration

- Kleinere/beheersbare wijzigingen in de code
- Snelle isolatie van fouten in de code
- Sneller in staat een fout op te lossen
- Betrouwbaarheid van de testen verhoogt
- Kleinere backlog *(kleine defecten komen naar boven d.m.v. testen)*
- Verhoging van transparantie en verantwoordelijkheid in het team *(rapportage)*
- Verlaagde kost *(automatisering verlaagt de kans op fouten)*

howest
hogeschool

git

Van Start!

Een Distributed Version Control System