

A decorative graphic on the left side of the slide consisting of several overlapping geometric shapes. There is a blue parallelogram, a light green parallelogram, and a dark grey parallelogram, all slanted at an angle. The background is a dark navy blue with subtle diagonal lines.

Test-Driven Development

Integrations and deployments

Continuous integration, the CI in CI/CD

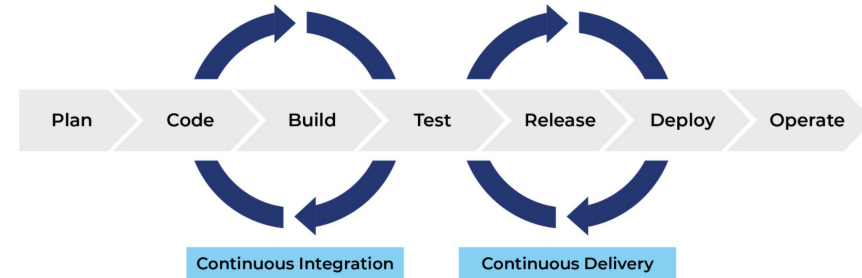


Continuous integration, the CI in CI/CD

Continuous integration innebär att nya kod förändringar regelbundet byggs, testas och “mergas” i ett gemensamt repository.

Exempel:

1. Utvecklare A skapar upp en ny branch med sina förändringar, som sedan byggs och testas. “A” skapar en pull request mot en gemensam develop/master branch.
2. Utvecklare B godkänner förändringarna och pull requesten. Förändringarna mergas sen till en gemensam branch



Continuous delivery, the CD in CI/CD

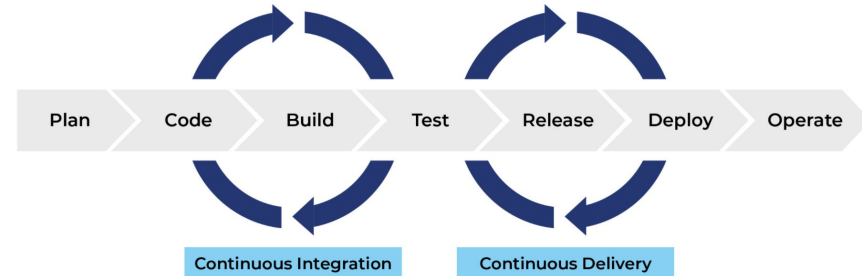


Continuous delivery, the CD in CI/CD

Continuous delivery innebär att ny kod regelbundet uppdateras till en gemensam test miljö.

Fortsättning på tidigare CI exempel:

1. När koden mergas till develop branchen, påbörjas ett automatiskt byggjobb som distribuerar den nya koden till den gemensamma develop/test miljön.



Continuous deployment, the other CD in CI/CD

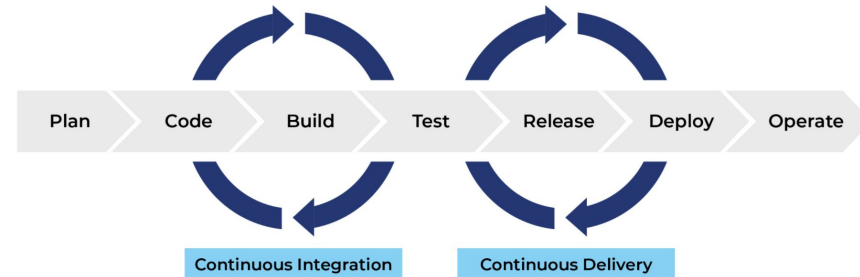


Continuous deployment, the other CD in CI/CD

Continuous deployment är ett annat alternativ, där koden då distribueras direkt till en produktionsmiljö. Detta är en ovanligare tillvägagångs metod som kräver mycket bra mekanismer för distribueringen. Exempel på en vanlig sådan metod är “Canary deployments”

Fortsättning på tidigare CI exempel:

1. När förändringarna mergas till den gemensamma master branchen, skapas ett automatiskt bygge som distribuerar koden till produktionsmiljön.



Build a CI/CD Pipeline with AWS CodePipeline

<https://www.youtube.com/watch?v=E-YanBx38Cs>



Kursutvärdering

https://docs.google.com/forms/d/181gptcVjfaGoh_Q1gd3j3TX1Y-jbNp77KpfUmHy_zq88/edit?ts=691c1848



Strategier för distribuering

Hur distribuerar vi våra webb applikationer till en produktionsmiljö?



Strategier för distribuering

Hur distribuerar vi våra webb applikationer till en produktionsmiljö?

- Big bang deployments
- Rolling deployments
- Blue-Green deployments
- Canary deployments



Big Bang deployments

Beskrivning: Hela systemet uppdateras och ersätts på en gång.

Fördelar: Enkel och snabb att utföra om ingen annan strategi är möjlig.

Nackdelar: Hög risk på grund av förändringar på en gång, potential för betydande driftstopp och svåra återställningar.

Bäst för: Applikationer där driftstopp är acceptabelt och infrastrukturändringar inte är möjliga stegvis.



Rolling deployments

Beskrivning: Instanser ersätts gradvis i omgångar, med trafik dirigerad till de nya versionerna när de kommer online.

Fördelar: Lägre risk och stilleståndstid jämfört med Big Bang, håller kostnaderna nere genom att inte kräva dubbla miljöer.

Nackdelar: Kan vara komplicerat att hantera och kan ge användare olika versioner under lanseringen.

Bäst för: Tillämpningar med många instanser och där ett balanserat förhållningssätt till risk och kostnad krävs.



Blue-Green deployments

Beskrivning: Två identiska miljöer upprätthålls: "Blue" (den nuvarande produktionen) och "Green" (den nya versionen). När den nya versionen är klar ändras trafiken från blått till grönt.

Fördelar: Erbjuder noll stilleståndstid och omedelbar återställning genom att helt enkelt byta tillbaka trafiken till den gamla miljön.

Nackdelar: Kräver dubbla infrastrukturresurser, vilket kan vara dyrt.

Bäst för: Verksamhetskritiska applikationer där driftstopp är oacceptabelt och snabba återställningar är avgörande.



Canary deployments

Beskrivning: Den nya versionen släpps först till en liten, slumpmässig undergrupp av användare. Trafiken flyttas gradvis över till den nya versionen när förtroendet växer.

Fördelar: Möjliggör verkliga tester i liten skala, minimerar effekten av buggar och ger enkla återställningar.

Nackdelar: Kan vara långsammare och mer komplex att implementera.

Bäst för: System med hög trafik, kontinuerliga leveranspipelines och när du vill samla in riktig användarfeedback om nya funktioner innan en fullständig release.



Build a CI/CD Pipeline with AWS CodePipeline

<https://www.youtube.com/watch?v=E-YanBx38Cs>



Kursutvärdering

https://docs.google.com/forms/d/181gptcVjfaGoh_Q1gd3j3TX1Y-jbNp77KpfUmHy_zq88/edit?ts=691c1848

