UKE 13 - DevOps

IN1030 - Gruppe 2

Hva skal vi i dag?

- DevOps
- Oblig 5
- Ukesoppgaver
- Hjelp til oblig 5?

Menti

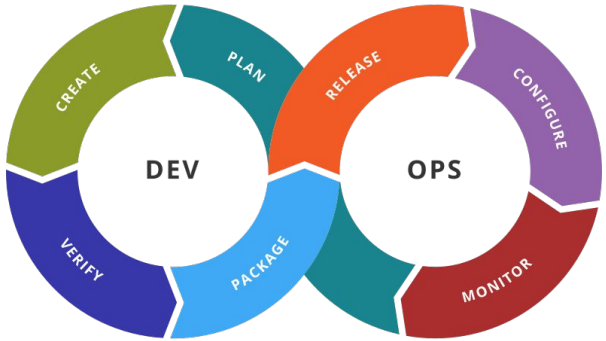
Læringsmål

...kjenner du til **ulike faser og aktiviteter** som inngår i systemutvikling.

...kan du anvende metoder og teknikker for kravhåndtering, utføre modellering ved hjelp av UML, og **vurdere fordeler og ulemper ved forskjellige metoder og teknologier for systemutvikling**.

DevOps

Development & Operations
HVORFOR DET?



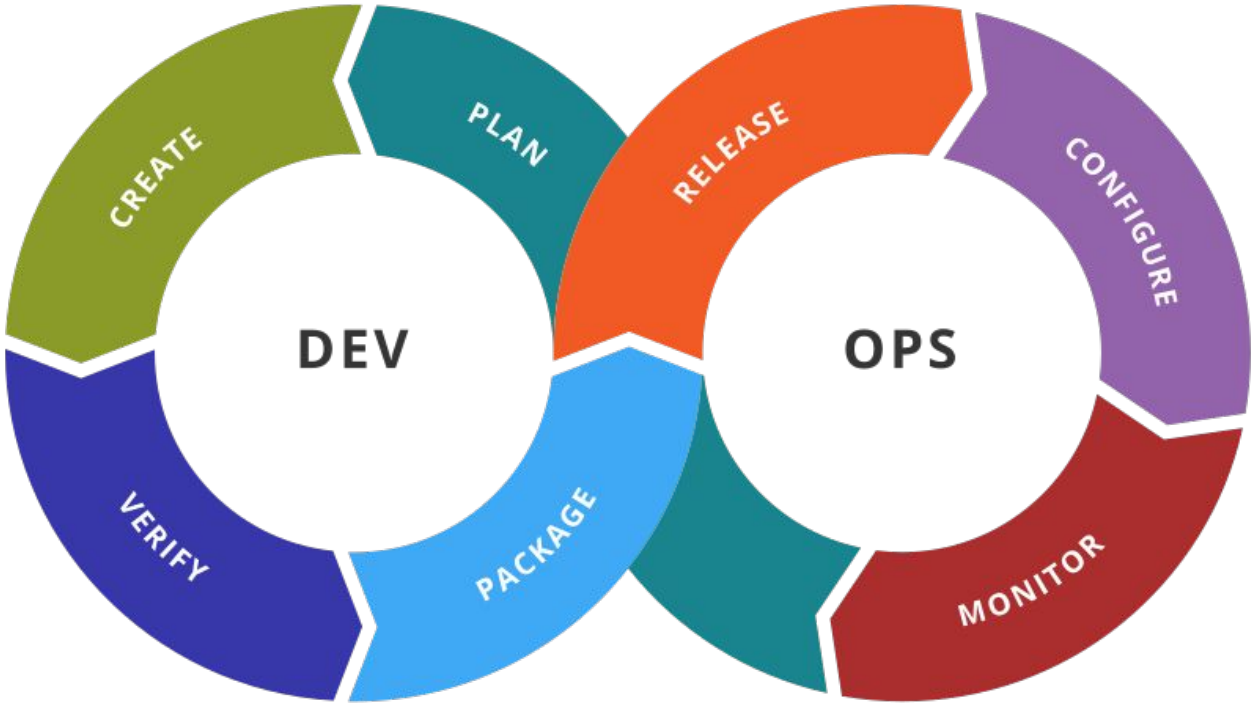
“Tradisjonelt” - Release and support

- Vanligvis har produktutvikling og vedlikehold bestått av tre team:
  - Utvikling, Utgivelse, Support, Vedlikehold (om den jobben ikke er gjort av utviklingsteamet)
- Modellen tar høyde for at de forskjellige teamene som oftest ikke har forståelse av problemene et annet team har møtt på som de må håndtere, som resulterer i en lenger periode før ny utgivelse kan komme ut

Derfor...

DevOps - litt teori

DevOps prinsipp	Forklaring
Alle ansvarlige for alt	Alle i teamet har delt ansvar for utvikling, utgivelse og vedlikehold/support av programvaren.
Alt som kan bli automatisert burde bli det	All testing-, utgivelse- og supportsaktiviteter bør bli automatisert når det er mulig. Legg opp til minst mulig manuelt arbeid med utgivelsen av programvaren.
Mål først, endre etterpå	DevOps burde bli drevet av målingsprogram hvor du samler data om systemet og operasjonene. Avgjørelser som angår endring i DevOps prosessen og verktøy, bør ta med denne dataen som grunnlag.


CREATE
PLAN
RELEASE
CONFIGURE
DEV
OPS
VERIFY
PACKAGE
MONITOR

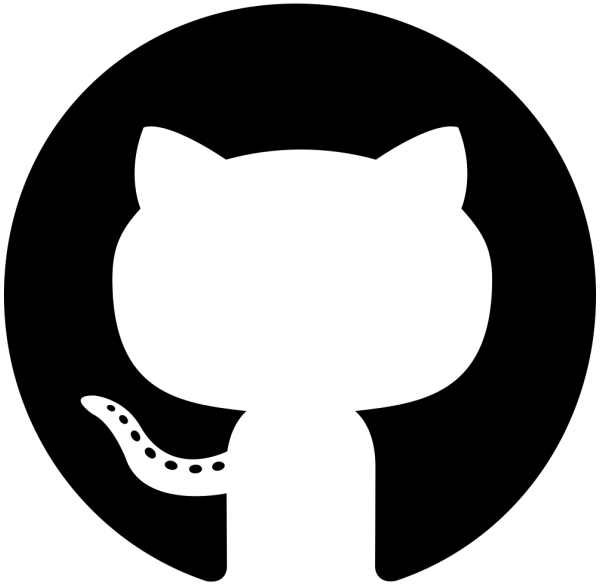
Fordeler

- **Raskere utgivelse/leveranse (Deploy)**
  - Fordi: kommunikasjons delay mellom teamene er blitt dramatisk redusert
- **Redusert risiko**
  - Fordi: inkrementet i hver leveranse er mindre, og dermed enklere å lokalisere kilden til problemet
- **Raskere reparasjon**
  - DevOps teamene jobber sammen for å få programvaren oppe å går
  - Man trenger heller ikke å finne ansvarlig team og vente på at de fikser det dersom et problem oppstår
- **Produktive team**
  - Teamene blir gladere og mer produktive fordi de jobber sammen og ikke på separate områder

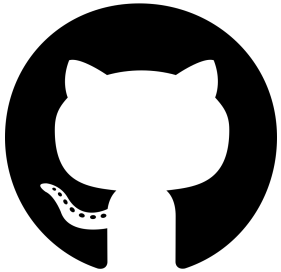
Kodeadministrasjon/ Code management

Sett med programvare-støttende praksiser for å administrere en inkrementelt voksende kodebase.

- Kodeoverføring
  - jobbe lokalt → kodeadministrasjonssystemet
- Versjonslager og henting
  - Spesifikke versjoner kan bli hentet
- Merging/branching
  - Paralell utviklingsgrener kan lages for å jobbe samtidig → merge endringene til samme løsning
- Versjonsinformasjon
  - Kontroll over endringer som blir gjort



Automatisering



Jenkins

Kontinuerlig integrasjon - Continuous integration

- kodeendring blir gjort → testing av endring implementert skjer
- enklere lokalisering av eventuelle feil
- systemet blir tilgjengelig for alle i teamet ved jevnlige oppdateringer
- Kodehåndtering - Github

Kontinuerlig levering - Continuous delivery

- Overordnet strategi
- Korte sykler som sørger for at programvaren alltid er funksjonell og klar for å slippes. Hurtig og ofte

puppet

Kontinuerlig utgivelse - Continuous deployment

- Oppdatering når master branch endres (Jenkins)
- Automatisk test → automatisk slipp

Diskuter i grupper

Hvilke fordeler får vi ved å automatisere og gjøre integrering, levering og utgivelse automatisk?

Fordeler med kontinuerlig utgivelse

1. **Redusert kostnad:** kontinuerlig utgivelse er hurtigere og mer effektivt enn manuell - som også ofte feiler.
2. **Hurtigere problemløsning:** problem påvirker kun en liten del, fordi ting oppdateres jo kontinuerlig, enklere å finne årsak til feil.
3. **Hurtigere kunde feedback:** identifisere forbedring gjennom å stadig gi ut nye funksjonaliteter.
4. **A/B testing:** noen bruker gammel og andre ny versjon - kan sjekke hva som fungerer bra / best.

Mål

DevOps - ønske om å kontinuerlig forbedre seg for å oppnå raskere utgivelse og bedre kvalitet på programvaren. Da trenger man å måle litt for å se hvordan ståa er…

Prosessmetrikker:

Gjennomsnittstid på gjenopprettelse

Prosentandel feilutgivelser

Utgivelsefrekvens

Endringsvolum

Forsinkelse fra utvikling til utgivelse

Diskuter i grupper

Hvilke tilpasninger kan man gjøre for å redusere gjennomsnittstid for gjenopprettelse eller utgivelsesfrekvens?

Mål

Prosessmetrikker:

Gjennomsnittstid på gjenopprettelse

Prosentandel feilutgivelser

Utgivelsefrekvens

Endringsvolum

Forsinkelse fra utvikling til utgivelse

Prosessendringsforslag:

Revurdere de automatisere testene

Sjekke kodehåndtering

Ansette flere

Revurdere målsetting

Mål

Tjenestemetrikker:

Prosent økning i sluttbrukere

Antall klager

Tilgjengelighet

Ytelse

Diskuter i grupper

Hvilke tilpasninger kan man gjøre for å redusere klager? Eller for å øke prosentandel nye sluttbrukere?

Mål

Tjenestemetrikker:

Prosent økning i sluttbrukere

Antall klager

Tilgjengelighet

Ytelse

Tjenesteendringer:

Markedsføring

Universell utforming

Brukerundersøkelser

Kode for robusthet

Endre systemarkitektur

Diskusjon

Hva kan være dumt med kontinuerlig utgivelse av ny programvare? Gi gjerne eksempler.

Kontinuerlig utgivelse - obsobs

Ukomplette funksjonalitet - lekkasje av geniale ideer.

Irriterte kunder som ikke oppdaterer fordi de må gjøre det så ofte.

Eks.: Mac-oppdatering, Chrome-oppdatering.

Synkronisering av bedriftenes ønsker og dine utgivelser - husk på det.

Eks.: lønnssystem til UiO

For å oppsummere:

DevOps kan bidra til innovativ og raskere utvikling. Målet er økt samarbeid mellom operasjoner og ansatte som vil bedre kvalitet. Dette kan bidra til at man er mer responsiv til behov og krav.

Ved å implementere DevOps så kan man trenger ny mindset, skills og verktøy.

Spørsmål?

Hvordan skal dere løse obligen (Oppgave 4)?

Oppgave 4 - Smidig & DevOps

Teamet får beskjed om at den planlagte systemutviklingsprosessen vil ta for lang tid.

a) Hvordan kan bedriften korte ned tiden fra ide til produkt-slipp og samtidig være sikker på systemets kvalitet? Begrunn.

b) Hva er forskjellen på “continuous integration”, “continuous delivery” og “continuous deployment”?

Spørsmål?

Nyttige linker:

IN2030 intro til versjonskontroll og Git

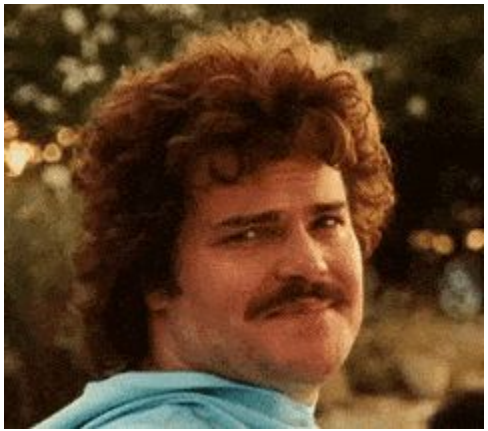
https://www.uio.no/studier/emner/matnat/ifi/IN2030/h20/timeplan/versjonskontroll.mp4

DevOps:

https://www.youtube.com/watch?v=_I94-tJlovg

Ukesoppgaver

Talk but write



UKESOPPGAVER

Fasit

emmatv@uio.no