

UKE 14 - DevOps

IN1030 - Gruppe 2

Hva skal vi i dag?

- DevOps
- Hjelp til oblig 5?

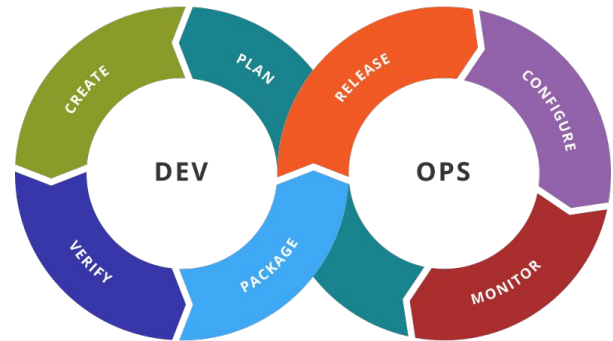
Læringsmål

...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

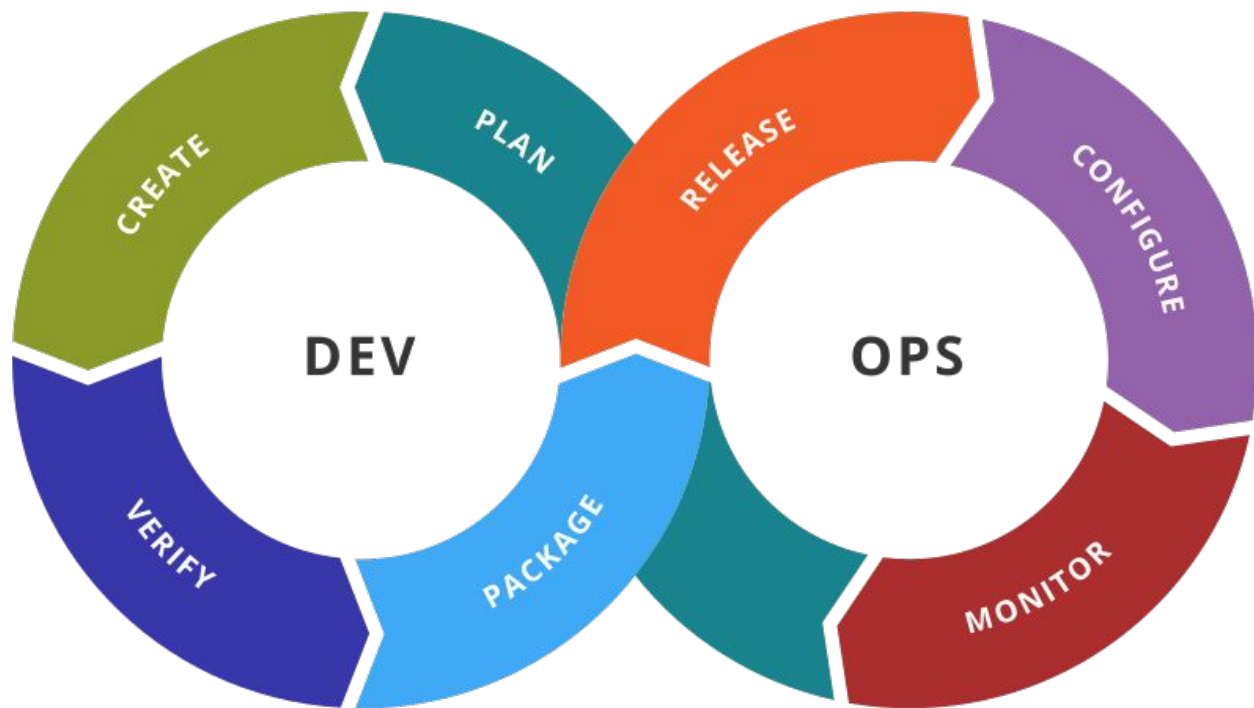
Derfor...

DevOps, tilnærming til programutvikling

- Fokuserer på:
 - samarbeid
 - kommunikasjon
 - automatisering mellom utviklings- og driftsteamene
 - kontinuerlig forbedring
- Målet er å:
 - minske forsinkelser → øke hastigheten
 - øke kvaliteten
 - øke pålitelighet

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.



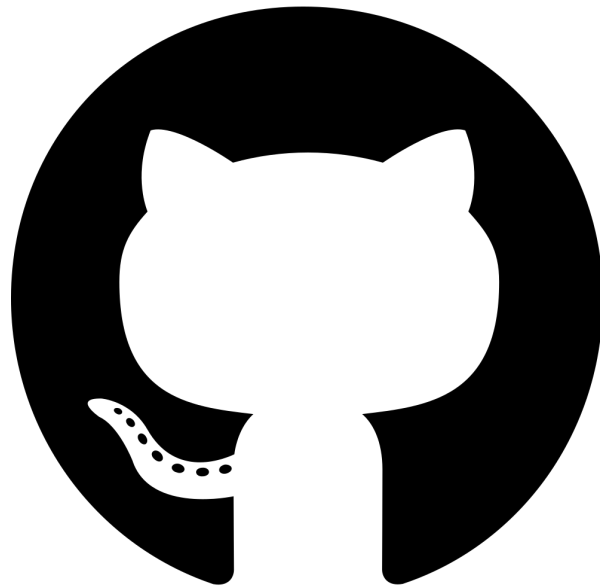
Fordeler

- **Raskere utgivelse/leveranse (Deploy)**
 - Fordi: kommunikasjons delay mellom teamene er blitt dramatisk redusert ++ automatisering
- **Redusert risiko**
 - Fordi: inkrementet i hver leveranse er mindre, og dermed enklere å lokalisere kilden til problemet ++ automatiserte tester og kvalitetskontroll
- **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
 - Bedre kommunikasjon, mindre misforståelser og bedre samarbeid for problemløsning
- **Bedre tilpasningsevne**
 - Med kontinuerlig tilbakemelding blir man bedre på å tilpasse seg endringer og behov

Kodeadministrasjon/ Code management

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

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



Automatisering: ulike nivåer

Kontinuerlig integrasjon - Continuous integration

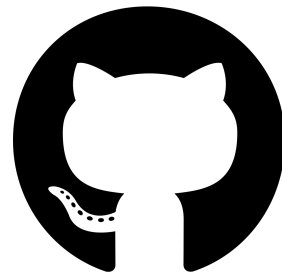
- Automatisere bygg- og testprosessene
- Kodeendring blir gjort → testing av endring skjer
- Enklere lokalisering av eventuelle feil
- Redusere risiko for problemer ved å synliggjøre feil tidlig

Kontinuerlig levering - Continuous delivery

- Automatisert utrulling av programvare til produksjon eller testmiljø
- Korte syklener som sørger for at programvaren kan leveres raskt og pålitelig

Kontinuerlig utgivelse - Continuous deployment

- Endringer som passerer CI og CD blir automatisk utrullet til produksjonsmiljø
- Automatisk test → automatisk slipp



Jenkins



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...

Prosessmetriker:

Gjennomsnittstid på gjenopprettelse

Prosentandel feilutgivelser

Utgivelsesfrekvens

Endringsvolum

Forsinkelse fra utvikling til utgivelse

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

Prosessmetriker:

Gjennomsnittstid på gjenopprettelse

Prosentandel feilutgivelser

Utgivelsefrekvens

Endringsvolum

Forsinkelse fra utvikling til utgivelse

Prosessendringsforslag:

Revurdere de automatisere testene

Sjekke kodehåndtering

Ansette flere

Revurdere målsetting

Ønsker å se nedgang i:

Antall klager

Ønsker å se økning i:

Tilgjengelighet

Ytelse

Antall sluttbrukere

Måler derfor tjenestemetrikker som:

Prosent økning i sluttbrukere

Antall klager

Tilgjengelighet

Ytelse

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

Basert på tjenestemetriker:

Prosent økning i sluttbrukere

Antall klager

Tilgjengelighet

Ytelse

Kan vi gjøre tjenesteendringer som:

Markedsføring

Universell utforming

Brukerundersøkelser

Kode for robusthet

Endre systemarkitektur

**Hva kan være dumt med kontinuerlig utgivelse av
ny programvare?**

Kontinuerlig utgivelse - obs

Ukomplette funksjonalitet - lekkasje av geniale ideer?

Risiko for nedetid - tilbake til ønske om tilgjengelighet

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.: lønssystem 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.

Hvordan henger dette sammen?

Scrum legger også vekt på:

- øke samarbeidet mellom team
- levere høykvalitets programvare raskere
- kontinuerlig forbedre prosessen

Fokus på prosess: å levere verdi til kunden i små, inkrementelle trinn

Kanban legger også vekt på:

- kontinuerlig forbedre prosessen
- optimalisere arbeidsflyten
- visuelle metoder for å oppdage/unngå problemer tidlig

DevOps har en mer operasjonell tilnærming som ønsker å sikre at programvaren er pålitelig og skalerbar og at endringer kan implementeres raskt og pålitelig. Dette med fokus på verktøy, automatisering og måling.

Ulemper med DevOps

- **Kostnader**
 - Kan kreve store investeringer i tid, ressurser og teknologi å implementere DevOps
- **Kompleksitet**
 - DevOps tar i bruk mange verktøy og teknologier som kan øke kompleksitet og gjøre det vanskelig å holde oversikt
- **Avhengig av teknologi**
 - DevOps bruker mye verktøy for automatisering, hvis noe streiker kan dette medføre store problemer. + kan man være sikker på at alle testene kjører riktig?
- **Kulturelle utfordringer**
 - DevOps krever endringer i organisasjonskulturen og samarbeid som kan kreve mye tid og innsats på å tilpasse
- **Endring av roller og ansvar**
 - Endring i roller og ansvar mellom utviklings- og driftsteamet kan føre til usikkerhet og motstand fra ansatte
- **Stress for brukere**
 - Ofte oppdateringer kan bli stress/plagsomt for brukere

Hvordan skal dere løse obliken (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”?

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=_l94-tJlovg

emmatv@uio.no