# IN2000 - DevOps

The world is how we shape it

sopra steria


**Martin Løtveit Brox**

Enterprise Agile Coach

# Hvem er jeg?

- **Bachelor of Science - Informatikk fra NTNU**
- **Har jobbet som konsulent siden 2007**
- **Utvikling, arkitektur og smidig coaching**

- **39 år gammel. Bor i Asker med kone og to barn**
- **Hobbykokk**
- **Ølbrygger**

https://www.linkedin.com/in/martin-løtveit-brox-30955027



sopra steria

# Mål for forelesningen

Etter denne forelesningen håper jeg dere skal være i stand til å:

Beskrive hva DevOps er

Forklare DevOps gjennom bruk av «The Three Ways»

Kjenne til effekten DevOps kan ha, og kunne de fire viktigste målingene for ytelse

01

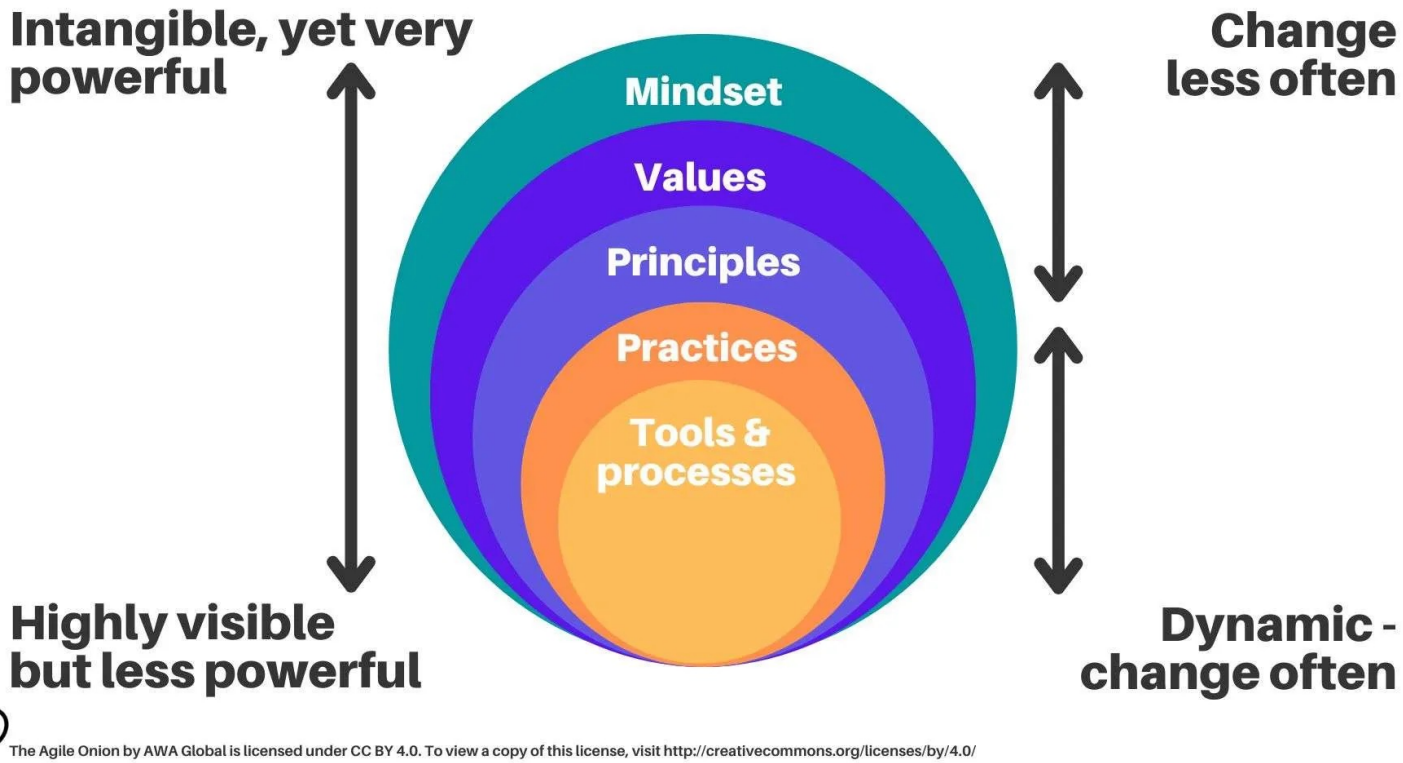
# Begreper

# Smidig og DevOps



sopra steria

# The Agile Onion






Kilde: https://www.adventureswithagile.com/powers-definition-agile-mind-set/



sopra steria

# The Agile Mindset

**Kompleksitetstanken**

- Velg tilnærming etter kompleksiteten til problemet som skal løses. Noen problemer kan løses med planlegging og bruk av ekspertise og kjente mønstre. Andre, mer komplekse og uforutsigbare problemer krever en smidigere tilnærming med eksperimentering og læring.

**Mennesketanken**

- Behandle alle med betingelsesløs positiv anseelse.
- Dette er troen på at mennesker er motiverte for å gjøre sitt beste, og at med riktig motivasjon og riktig miljø (trygghet, respekt, mangfold og inkludering) så kan tillit og selvorganisering oppnås.

**Proaktivitetstanken**

- Dette er den endeløse jakten på forbedring

Kilde: https://norge.adventureswithagile.com/hva-er-det-smidige-tankesettet/

# Hvor kommer DevOps inn?

- Mange ser på DevOps som en naturlig forlengelse av smidig.
- Som med smidigløken er det verktøy og praksiser som er mest synlig
- Og hvor de største effektene hentes ut først med kontinuerlig læring, kulturendringer, og nye måter å tenke og jobbe sammen på.

--> DevOps er kultur for å lage bedre programvare raskere

# DevOps

Og fjerningen av siloer

- DevOps som begrep kom i 2008.
- Men praksis og prinsipper vi forbinder med DevOps har vært til stede lenger.
- Grunnlaget for begrepet er sammenføringen av Development og Operations, hvor siloer brytes ned for å levere verdi sammen.
- Med siloer mener vi at separate enheter innad i en organisasjon har hatt ulike mål. Tradisjonelt sett har utviklingsteam fokusert på å lage ny funksjonalitet, og levert denne fra seg til driftsteam som sørger for stabilitet og pålitelighet på kjørende kode.

- Konsekvensen av siloene har vært mange. Som:
  - Treghet fra manglende samarbeid
  - Transaksjoner med ventetid og informasjonstap
- Uklare ansvarsforhold, blame game og svekket sikkerhet
- Disse siloene finner vi fortsatt overalt!
- Samt andre siloer, som separate avdelinger for informasjonssikkerhet (DevSecOps anyone?)

# Hvordan føres så Dev og Ops sammen?

- Siloene brytes ned gjennom samhandling.
- Og samhandling får vi når det jobbes sammen mot samme mål.

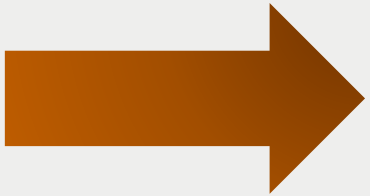
- Idealet i DevOps er at utvikling og drift gjøres i ett og samme team.

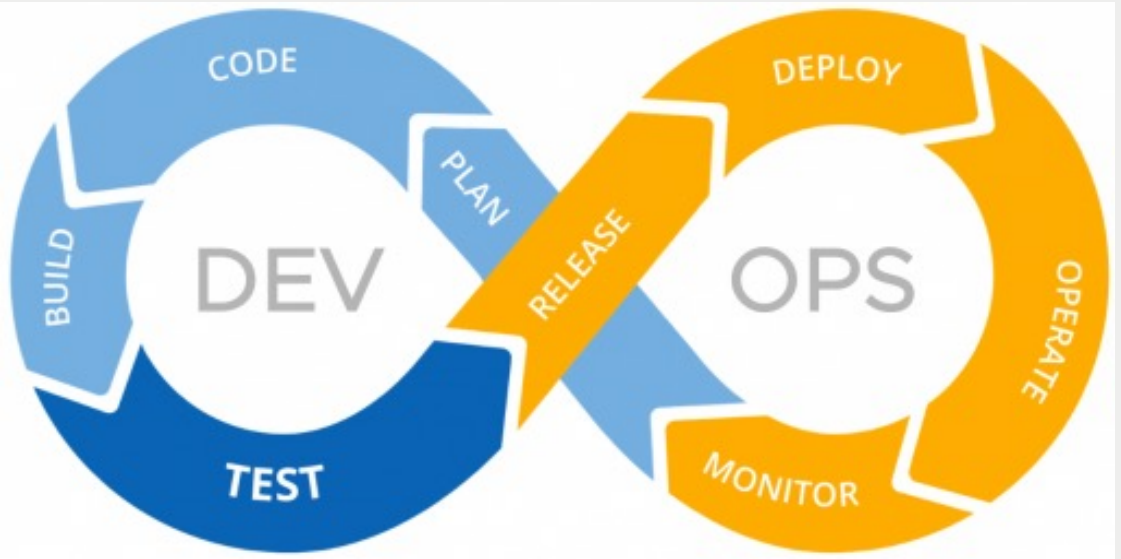
«You Build It, You Run It»

- Og et DevOps-Team måles gjerne med metrikker for både flyt og stabilitet.

# DevOps beskrives som sirkulært

I motsetning til den lineære tanken for levering av kode (vannfall)






sopra steria

02

# The Three Ways



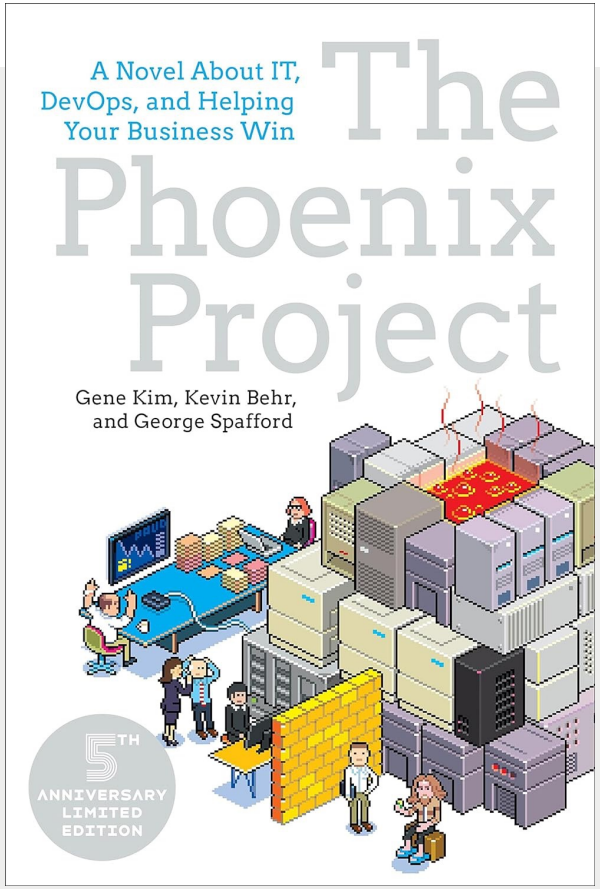
sopra steria

# DevOps beskrevet med The Three Ways

Grunnprinsipper som alle DevOps patterns kan utledes fra

- Kjent fra bøkene «The DevOps Handbook» og «The Phoenix Project»

- The First Way: Flow/Systems Thinking
- The Second Way: Amplify Feedback Loops
- The Third Way: Culture of Continual Experimentation and Learning






sopra steria

# The First Way: Flow/Systems Thinking

The whole is greater than the sum of its parts

**Aristoteles**

# The First Way: Flow/Systems Thinking

Som betyr:

- Å forstå flyten av verdi slik den leveres til kunde
- Og å se på alle involverte som et system
  - Ta beslutninger basert på forståelse av alle dimensjoner
  - Optimaliser for flyten til systemet som helhet

- Kjente praksiser og prinsipper:
  - Redusere work in progress
  - Lean
  - Kanban

# The Second Way: Amplify Feedback Loops

- The Second Way handler om å korte ned og forsterke feedback loops.
- Slik at velkvalifiserte justeringer kan gjøres raskt.
- Og at feil kan oppdages og rettes nærmest mulig kilden.

- Kjente praksiser og prinsipper:
  - Testautomatisering
  - Stoppe kode som ikke tilfredstiller ønsket kvalitet
  - Metrikker
  - SRE

# The Third Way: Culture of Continual Experimentation and Learning

- The Third Way handler om å bygge kultur for kontinuerlig læring
- Innovasjon krever eksperimentering, og eksperimentering involverer risiko
  - Denne risikoen må aksepteres for at nyskapning skal finne sted

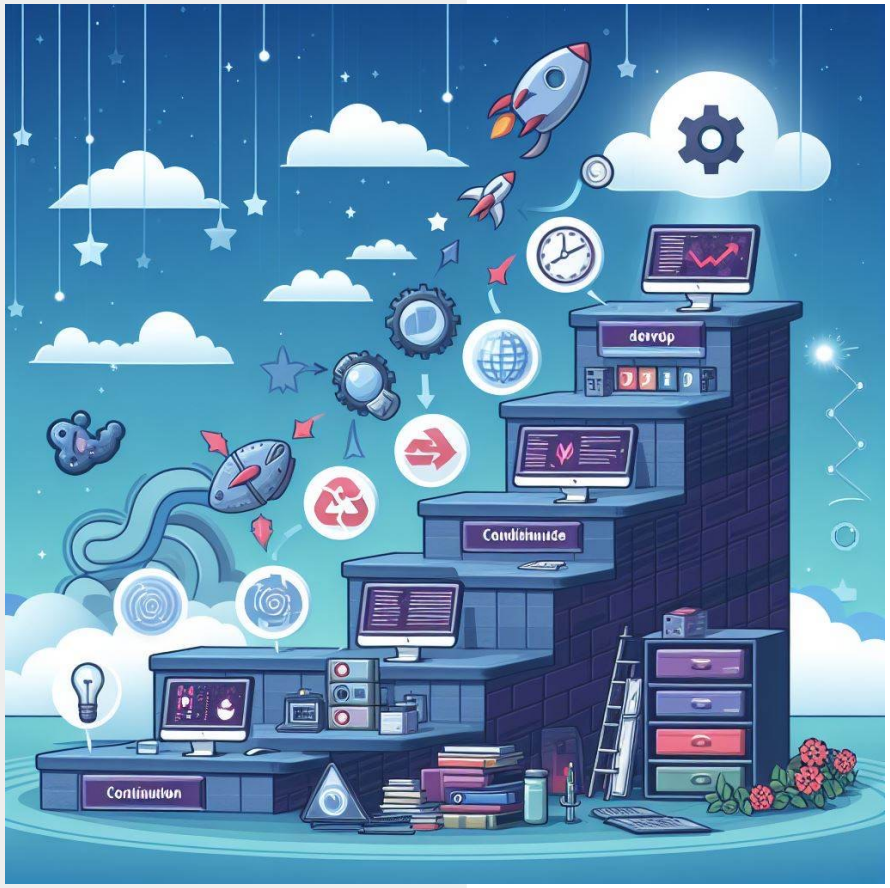
- Kjente praksiser og prinsipper:
  - Refaktorering
  - Experiment Canvas
  - Ritualer som retrospektiv, "blameless post mortem"
  - Bygge et nivå av trygghet og tillit som gjør det mulig å snakke om feil
  - Chaos Engineering

# Tilnærming

03

# Hvordan?

Tilpass praksiser -> Continous Everything




- Continous Integration
- Continous Deployment
- Continous Testing
- Continous Delivery
- Continous Monitoring & Operation

Automatiser alle gjentakende oppgaver slik at tid og krefter kan rettes mot utvikling av kode som gir verdi

# Continous Integration

- Continous Integration er å kontinuerlig ta inn kode fra deltakere inn i et delt repository

- Gode praksiser
  - TDD
  - Byggautomatisering
  - Trunk Based Development
    - Holder løsningen i en tilstand som kan deployes «når som helst»
    - Alle utviklere må jobbe på kortlevde kodebrancher
    - Løsningen skal alltid kunne produksjonssettes fra master-branch

# Continous Deployment

- Kontinuerlig utrulling av kode

- Gode praksiser:
  - Miljøer on demand
    - Selvbetjent provisjonering
    - Infrastruktur som kode
    - Miljøene slettes når de ikke trengs
  - Feature toggles
    - Kan skille deployment fra release, og deploye kontinuerlig til produksjon med styring på når vi releaser til hvilke kunder
  - Containerteknologi
    - Utrulling uten nedetid
    - Når vi vil, og så ofte vi vil

# Continous Testing

- Gode praksiser:
  - Generering av testdata
  - Verdikjedetesting
  - Ytelsestesting
  - Sikkerhetstesting
    - SAST
    - DAST
  - A/B testing

# Continous Delivery

- Kulmineringen av alt vi automatiserer og gjør kontinuerlig
- Levert = Gjort tilgjengelig for kunde
- Mål gjerne at funksjonaliteten faktisk lar kunden oppnå det den skal, og dermed gir verdi
  - Ellers er vel ingenting levert?

- Etterstreb å levere kode ofte, og i små batcher

# Continous Monitoring & Operation

- Når funksjonalitet er levert til kunde starter den kontinuerlige jobben med å sørge for at den er tilgjengelig og gir verdi til kunde
- Hva som ofte overvåkes:
  - Nettverk og infrastruktur
  - Informasjonssikkerhet
    - Kjenn kundenes risikotåleranse
  - Forbruk (på skytjenester)
  - Kundesentriske – Får kundene gjort det de skal?
- Etabler også rutiner for varsling og håndtering av hendelser

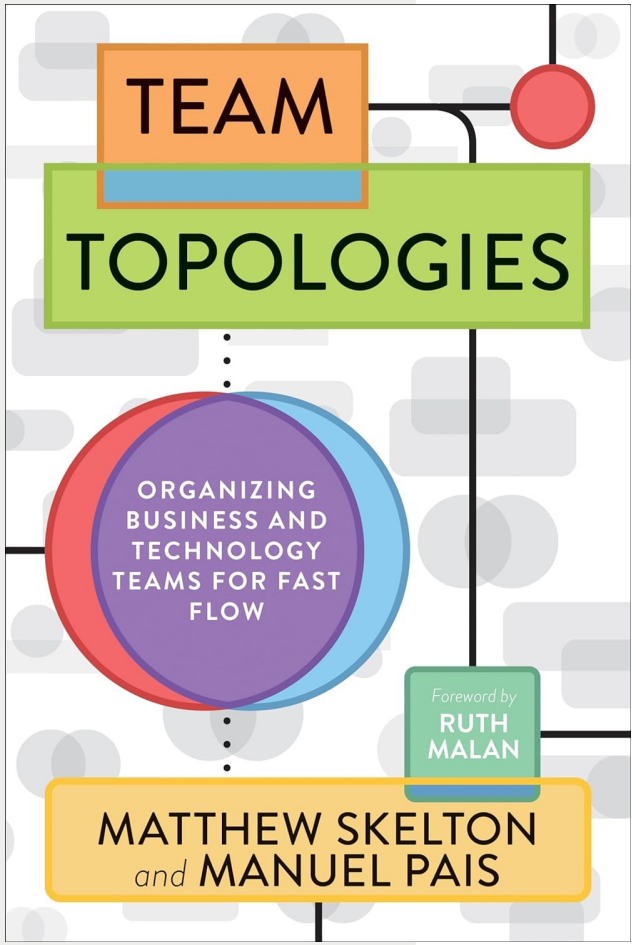
# Platform Engineering

*"Platform engineering is the discipline of designing and building self-service capabilities to minimize cognitive load for developers and to enable fast flow software delivery. Platform teams deliver shared infrastructure platforms to internal users"* (Puppet | State Of DevOps Report 2023)

- Baserer seg på nevnte, kjente praksiser, men leveres som standardisert støtte til utviklere for skalering i større miljøer.
- Er bygget på automatisering, samhandling og verdi for bruker.
- Hjelper organisasjoner med innføring av DevOps i en større skala, med økt realisering av gevinster.

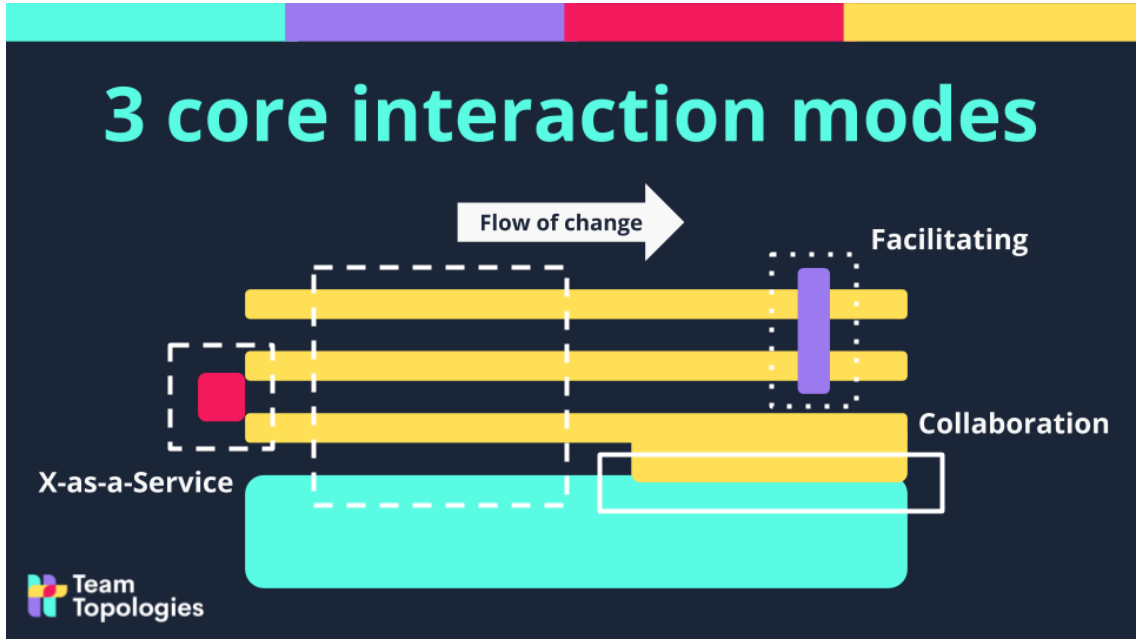
# Team Topologies

Matthew Skelton og Manuel Pais, 2019




- For å organisere seg for flyt og verdiskaping definerer de
  - 4 teamtyper
  - 3 samhandlingsformer mellom team
- Disse er alt du trenger, også i store organisasjoner

# Teamtyper og samhanddlingsformer


Kilde: https://teamtopologies.com/key-concepts



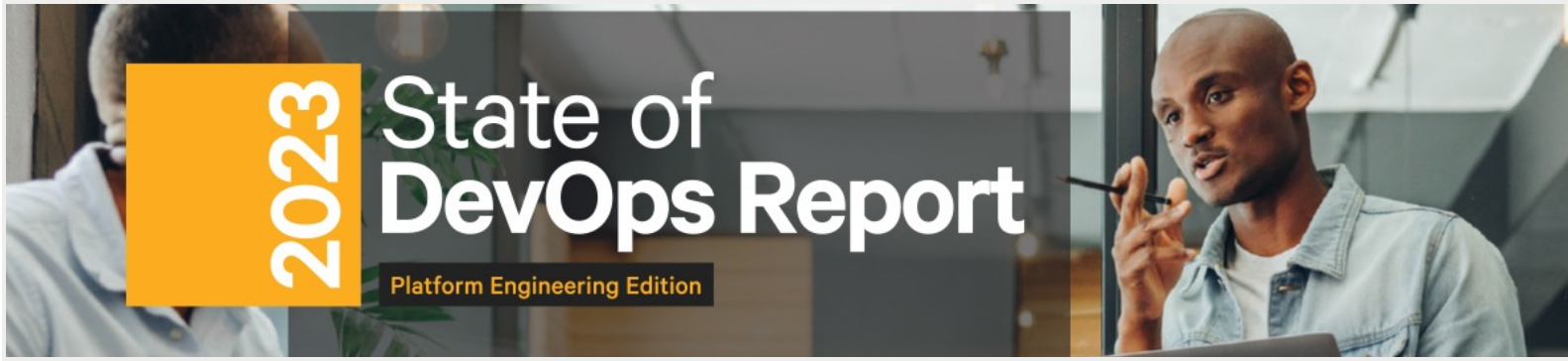
sopra steria

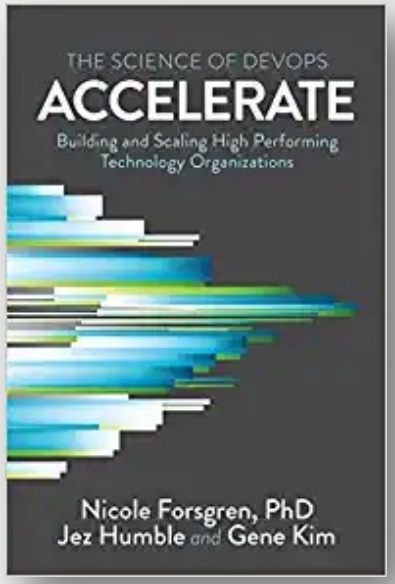
# Puppet og DORA publiserer årlige State of DevOps rapporter

Store spørreundersøkelser med mange like spørsmål fra år til år

*Accelerate* er basert på Puppet's undersøkelser

4 standardiserte nøkkelmålinger







sopra steria

# The four key metrics

De viktigste målingene for ytelse ved levering av programvare

| Software delivery performance metric | Low | Medium | High |
|---|---|---|---|
| **Deployment frequency**<br>For the primary application or service you work on, how often does your organization deploy code to production or release it to end users? | Between once per month and once every 6 months | Between once per week and once per month | On-demand (multiple deploys per day) |
| **Lead time for changes**<br>For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)? | Between one month and six months | Between one week and one month | Between one day and one week |
| **Time to restore service**<br>For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)? | Between one week and one month | Between one day and one week | Less than one day |
| **Change failure rate**<br>For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)? | 46%-60% | 16%-30% | 0%-15% |

- Legg merke til at det er 2 målinger for flyt, og 2 for stabilitet
- Prøv selv!
  - https://dora.dev/quickcheck/

Kilde: Accelerate | State of DevOps 2022

# Fordeler ved DevOps og Platform Engineering

- Økt effektivitet
- Raskere leveringstid
- Økt pålitelighet
- Innovasjon
- Økt Robusthet

- Fornøyde brukere
- Fornøyde utviklere

# Mer om boka Accelerate

- Som bokens navn indikerer, og tallene fra undersøkelsene viser, akselereres ytelse med modenhetsnivået på DevOps.
- Binder teorien sammen på en god måte, og er beviset på at det er mer enn verktøy som skal til for å oppnå høy ytelse.
- Boka beskriver andre dimensjoner:
  - Organisasjonskultur
  - Informasjonsflyt
  - Ledelse

- Tar oss tilbake til mennesketanken i smidig
  - Og at høy ytelse kan oppnås gjennom å frigjøre potensialet i motiverte menneskers evne til å løse komplekse problemer sammen

# Oppsummering

Hva har vi vært gjennom?

- Smidig
- DevOps som begrep
- The Three Ways
- Continous Everything
- Platform Engineering
- Team Topologies
- Måling av ytelse ved levering av programvare

# Takk for meg!

Gi meg gjerne din tilbakemelding

Join at menti.com | use code 1126 0982