



id: SI\_002

model: mistralai/mistral-large-2512

seeds:

04\_risiko: sårbarhet

03\_relasjon: samarbeid

05\_handling: utforsk

01\_perspektiv: strategisk

02\_tidhorisont: historisk

## Instans SI\_002

**Seeds:** strategisk (langsiktig kontroll) | historisk (lærdom fra tidligere autonomi-fiaskoer) | samarbeid (grenseflater mellom menneske og maskin) | sårbarhet (hvor systemet kan knele) | utforsk (begrenset pilot som læring)

### Strategisk: Ikke bygg en autonom skyskraper på et sandfundament

Agentisk KI er ikke et teknisk problem, men et organisatorisk eksperiment i autonomi. Historien viser at halvautonome systemer (f.eks. NASAs Mars Climate Orbiter, 1999) feiler når grensesnittene mellom menneske og maskin er uklare. Dere står foran en lignende arkitektonisk paradigmeskifte: fra deterministiske systemer til probabilistiske agenter. Den strategiske feilen ville være å tro at dere kan “sikkerhetsdesigner” dere ut av dette på 6 måneder.

#### Handlingspunkt:

- **Definer autonomi-nivåer** (f.eks. OODA-loop-tilnærming: Observe-Orient-Decide-Act). Begrens agentene til nivå 2 (Decide) i piloten – la mennesker håndtere Act for irreversible handlinger.
- **Bygg en “autonomi-risikomatrise”** med aksene handlingsfrihet (lav/høy) vs. konsekvens (reversibel/irreversibel). Prioriter systemer med lav konsekvens først (f.eks. forslag til saksbehandling, ikke fakturabetaling).

### Historisk: Lær av andres fiaskoer – agentisk KI er ikke ny

I 2016 slo Microsofts Tay-chatbot tilbake mot sine skapere etter 16 timer, mens Knight Capital mistet \$460M på 45 minutter da en algoritme gikk amok. Fellesnevneren?



Manglende “kill switches” og dårlig integrasjon med legacy-systemer. Dere har en fordel: eldre fagsystemer er ofte mindre sårbare fordi de er mindre tilgjengelige – men også vanskeligere å overvåke.

#### Handlingspunkt:

- **Implementer “safety envelopes”** (inspirert av robotikk): Agentene får kun handle innenfor forhåndsdefinerte grenser (f.eks. maks 10 API-kall/time, ingen direkte database-skriv). Bruk Open Policy Agent (OPA) for dynamiske policyer.
- **Lag en “fiasko-simulator”** som tester hvordan agentene oppfører seg når:
  - Et API returnerer korrupte data.
  - En ekstern tjeneste er utilgjengelig i 48 timer.
  - En agent misforstår en brukerinstruksjon (f.eks. “slett alle dokumenter fra 2023” → “slett alle dokumenter”).

---

### Samarbeid: Menneskelig oversikt er ikke en flaskehals – det er en nødvendighet

Autonome agenter krever mer menneskelig innsikt, ikke mindre. Problemet med dagens IAM (RBAC) er at det forutsetter statiske roller, mens agentisk KI trenger dynamiske tillatelser. Løsningen ligger i samarbeidsmodeller som kombinerer maskinens hurtighet med menneskelig kontekstforståelse.

#### Handlingspunkt:

- **Design en “agent-menneske-loop”** med tre stadier:
  1. Foreslå (agenten genererer handlinger).
  2. Godkjenne (menneske vurderer høyrisiko-handlinger via en decision dashboard).
  3. Lære (menneskelig feedback trenes inn i agentens policyer via reinforcement learning).
- **Bruk “break-glass”-mekanismer** for kritiske handlinger (f.eks. krev to-faktor-godkjenning fra to ulike roller for å reversere en agenthandling).

---

### Sårbarhet: De tre mest undervurderte risikoene

1. **Integrasjons-kaos:** Agentene vil oppdage skjulte avhengigheter mellom systemer (f.eks. at et API-kall til system A utilsiktet trigger en batch-jobb i system B). **Løsning:** Kjør en avhengighetskartlegging før piloten.
2. **Data-lekkasje via eksterne APIer:** Agentene kan sende sensitiv data til eksterne tjenester (f.eks. ved å inkludere interne IDer i prompts til OpenAI).



**Løsning:** Bruk data tokenization (f.eks. erstatt personnummer med UUIDer før eksterne kall).

3. **Logging-gap:** Agentene vil generere uforutsigbare loggformater. **Løsning:** Krev at alle agenthandlinger logges i et standardisert format (f.eks. OpenTelemetry) med korrelasjons-IDer på tvers av systemer.

---

## Utforsk: Piloten er et eksperiment - ikke en implementering

Dere har ikke tid til å løse alt. Derfor må piloten designes som en kontrollert eksplosjon – med klare grenser for hvor mye som kan gå galt.

### Handlingspunkt:

- **Begrens piloten til ett fagsystem** (f.eks. saksbehandling i sosialtjenesten) med maks 3 agenter og maks 5 handlingstyper.
- **Mål suksess i “læringsrater”**, ikke effektivitet\*\*:
  - Hvor mange ganger må en agent eskalere til menneskelig godkjenning?
  - Hvor mange ganger misforstår agenten en instruksjon?
  - Hvor lang tid tar det å rulle tilbake en agenthandling?

---

## Konkrete neste steg

1. **Kjør en “autonomi-risiko-workshop”** med teknisk og juridisk team for å identifisere irreversible handlinger i deres systemer. Bruk resultatet til å definere pilotens grenser.
2. **Implementer en “policy-sandbox”** (f.eks. med OPA) som tester agenthandlinger mot regler som:
  - “Ingen agent kan slette mer enn 10 poster per time.”
  - “Alle eksterne API-kall må godkjennes av en menneskelig rolle.”
3. **Bygg en “agent-decision-dashboard”** som viser:
  - Hvilke handlinger agenten har utført (med korrelasjons-IDer).
  - Hvilke policyer som ble brutt (og hvorfor).
  - Hvilke handlinger som ble eskalert til mennesker (og utfallet).

---

**SI\_002 avslutter:** Agentisk KI er ikke et teknisk problem – det er en organisatorisk prøveballong. Dere vil mislykkes delvis. Det er meningen. Lær raskt, begrensk skaden, og bygg kontrollen etter hvert som dere forstår systemets sårbarheter.