4. CORBA Portable Object Adapter (POA)

- Baggrund
- Teknisk opbygning (herunder policies)
- POAManager
- Alternativer

Baggrund

I 1990 specificerede OMG en BOA (basic), der dog var for primitiv i forhold til internettet – den var underspecificeret, hvilket resulterede i, at alle ORB udbydere havde sin egen implementation. Derfor specificerede de senere POA (portable), som er portable imellem ORB udbydere.

POA'en befinder sig på serveren, hvor der er en RootPOA med nogle child-POAs under sig.

- Står for at lokalisere og håndtere objekter på serveren.
- Sørger for at server-HW udnyttes optimalt ved at deaktivere servants, der ikke er i brug. Gør at der kan skaleres heftigt.
- Genererer og fortolker referencer.
- Mapper objekt referencer til de korrekte servants.
- Sørger for at metodekald udføres.

Teknisk opbygning (herunder policies)

POA'en er et objekt på serveren, som indeholder en tabel (**Active Object Map**) med ObjectId'er og pointere til deres tilhørende servants.
Man kan konfigurere en POA ved at sætte policies på den. RootPOA'en har en standard-policy, der ikke kan ændres. Derimod kan child-POAs ændres → vi har en masse forskellige POA'er.

- **IdUniquenessPolicy:** Der findes to forskellige mapninger imellem ObjectId'er og Servants:
  - *én-til-én* (default) eller
  - *én-til-mange* (ved fx statiske (tilstandsløse) objekter).
- **LifeSpanPolicy:** Bestemmer om object referencen (ikke objektet selv) skal overleve en slukning af serveren.
  - *Transient (standard) - kort:* Object referencen virker ikke mere efter servergenstart.
  - *Persistent - lang:* Object referencen virker stadig (dog skal man på en måde pesistere objektet server-side, for at det giver mening at bruge denne tilstand).
- **ThreadPolicy:** Single- eller multi-threaded (default).

POAManager

Metodekald på CORBA objekter routes gennem POAManageren, som finder den korrekte POA, som derefter finder den servant, der er tilhører ObjectId'et fra CORBA objectet. Denne står for at videresende requests til POA'er – kontrollere flowet ind. Kan være i 4 forskellige tilstande:

- **Holding:** Når POA'er endnu ikke er startet op holdes alle request tilbage.
- **Active:** Normal tilstand. Requests sendes videre til respektive POA'er.

- **Discarding:** Bruges ved en presset server – der udføres ingenting, men svares med en exception.
- **Inactive:** Afviser alle requests. Bruges typisk lige inden serverslukning, da man ikke kan komme ud af denne tilstand igen.

Laves på samme tid som RootPOA.

Alternativer

Det eneste reelle alternativ er at droppe POA'en (da BOA ikke bruges mere). Så skulle objektet i stedet refereres direkte, så man kan tilgå det over netværket. Hvert objekt skal så bruge en port på serveren (da vi ikke længere har en gatekeeper i form af en POA). Så er vi tilbage til noget socket-programmering, og vores system er ikke længere et CORBA-system.

SE KODE

Vi har brugt RootPOAen – den bruger som udgangspunkt transiente objekter, hvilket betyder at Doctor-objektet dør ved nedbrud. Her burde vi have lavet en ny POA med en seperat policy, der angiver brugen af persistente objekter (dette kræver dog, at man gemmer state på objektet).