BÆRUM KOMMUNE
IKT-AVDELINGEN

Case 3 for Semicolon-prosjektet

Pilot for søknad om SFO-plass
Bærum Kommunes Proof of Consept på mellomvare

Dato: 15.01.2009

Indeks

Case 3 for Semicolon-prosjektet 1
Innledning 3
Forprosjekt Virksomhetsarkitektur i Bærum 3
Proof of Concept 4
Elektronisk arbeidsprosess fra skjema via integrasjonslag 4
  Mål for Proof of Concept: 5
  AS-IS kartlegging 5
  TO-BE kartlegging 6
  Dokumentasjon 7
SOA (Tjenesteorientert arkitektur) 8
  Fordeler med SOA 8
  SOArkitektur 10
  Kriterier for valg av komponenter 12
  Kriterier for valg av komponenter - Bærum spesifikt 14
  BK – SOArkitektur 15
  Delvis introduserte komponenter 17
Erfaringer 18
  Teknologi 18
  Semantikk 19
  Fri programvare 20
  Gjennomføring 21
  Leverandører 21
  Juss 21
  Eierskap og ansvar 21
  Arkitektur og forvaltningsprinsipper 22
  Suksessfaktorer for implementering av SOA 22
Sluttføre arbeidet med SFO prosess – fra pilot til drift 23
Nytt prosjekt – oppstart i januar 2009 23

Innledning

Bærum kommune startet å jobbe med IKT-arkitektur i 2004 for å kunne oppfylle både interne og eksterne krav. Vi hentet bl.a. informasjon om hva Danmark har gjort på dette området, som trolig er ledende på arbeid med virksomhetsarkitektur innenfor offentlig virksomhet.

I 2008 ble vi invitert til å delta i Semikolon-prosjektet som en del av Case ”Standardisert integrasjon mellom IKT-systemer i kommunesektoren”. Dette Case skal ha 2 Piloter, hvorav det ene skal være basert på mellomvare for integrasjon, gjerne også fri programvare. Asker kommune har kjørt sitt prosjekt for barnehagesøknad som den ene piloten, og Bærum bruker sitt prosjekt for automatisering av SFO-søknadsprosess som pilot på bruk av mellomvare og fri programvare.

Forprosjekt Virksomhetsarkitektur i Bærum

I 2006 startet vi et forprosjekt som skulle vurdere ulike akritektur-modeller for synkronisering av virksomheten og IT, med den hensikt å oppnå:

- Forretningsmessig vekst
- Bedre kvalitet på tjenestene som produseres
- Nye tjenester som følge av teknologisk utvikling og innovasjon
- Økt forretningsmessig fleksibilitet
- Bedre bruk av virksomhetens ressurser
- Lavere IT kostnader

Føringer forprosjektet

- Digital kommunikasjon mellom kommunen og innbyggere
- Effektiv tjenesteproduksjon
- Digital kompetanse i skolen – kunnskapsløftet
- Effektiv IT produksjon

Hovedaktiviteter

- Kartlegge tjenesteproduksjon og applikasjoner
- Analyse av integrasjonsgrad mellom systemer
- SWOT analyse
- Utarbeide nye målbilder for IKT infrastruktur basert på et betydelig sterkere tjenestefokus.
- Anbefaling om videre arbeid

Konklusjon

- Bærum kommune bør etablere en virksomhetsarkitektur, inkludert
  - Metodikk / maler / roller
  - Forvaltningsregime
  - Arkitekturprinsipper
- Det er nødvendig å vurdere styringsmodell for IKT
- Det anbefales å etablere en tjenesteorientert arkitektur (SOA)
- Tjenestekartlegging bør gjennomføres på de gjenstående områdene

Kriterier for valg av tjenester (skjema på nett)

- Utgangspunkt tjenestekatalogen til Bærum
- Utvalg på bakgrunn av
    - Ikke personsensitiv informasjon
    - Har skjema på BK's nett i dag
    - Nytte for innbyggerne
    - Kompleksitet i realisering

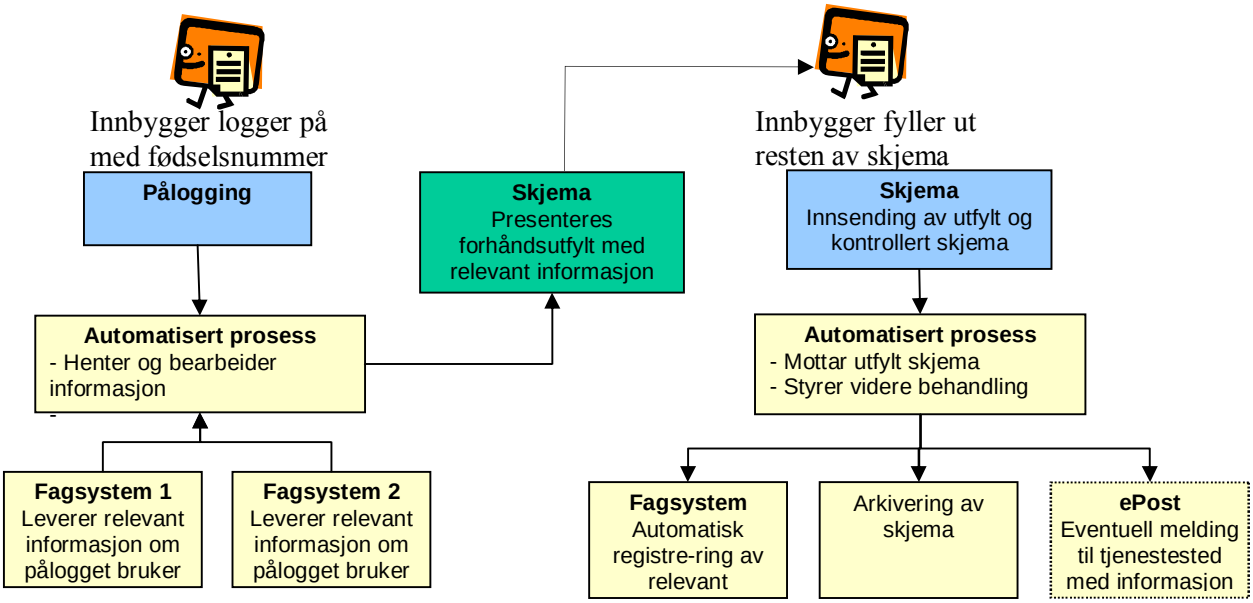
Proof of Concept

SFO-søknadsprosess ble valgt som Case for å implementere SOA-infrastruktur og teste om valgte produkter kunne være hensiktsmessig i kommunens virksomhetsarkitektur.

Elektronisk arbeidsprosess fra skjema via integrasjonslag

I sin dialog med innbyggerne skal kommunen der det er hensiktsmessig ta i bruk elektroniske arbeidsprosesser via integrasjonslag (SOA-infrastruktur). Dette skiller seg fra vanlige skjema på nett i 2 viktige punkter:

- Brukeren må autentisere seg for å kunne bruke tjenesten, og kommunen vet dermed hvem brukeren er.
- Skjema pre-utfylles med relevant informasjon som kommunen har om den påloggede brukeren.



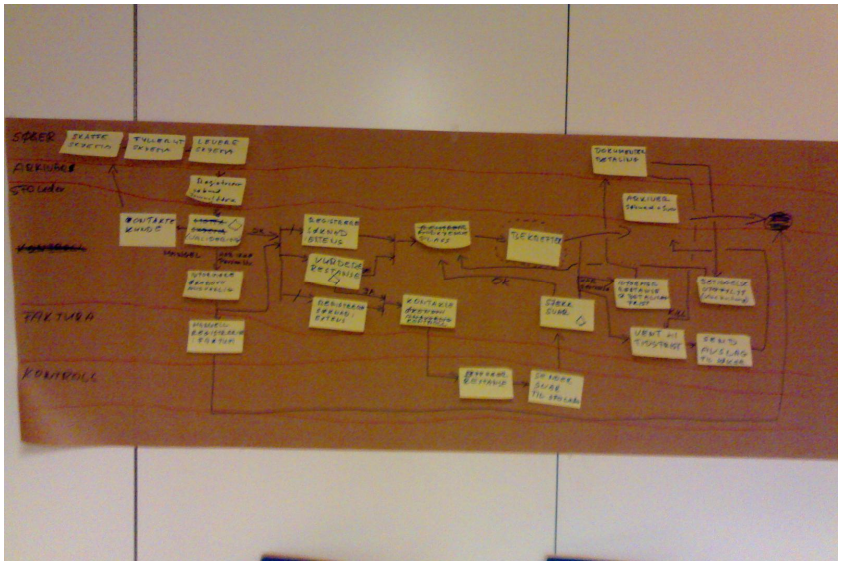
Mål for Proof of Concept:

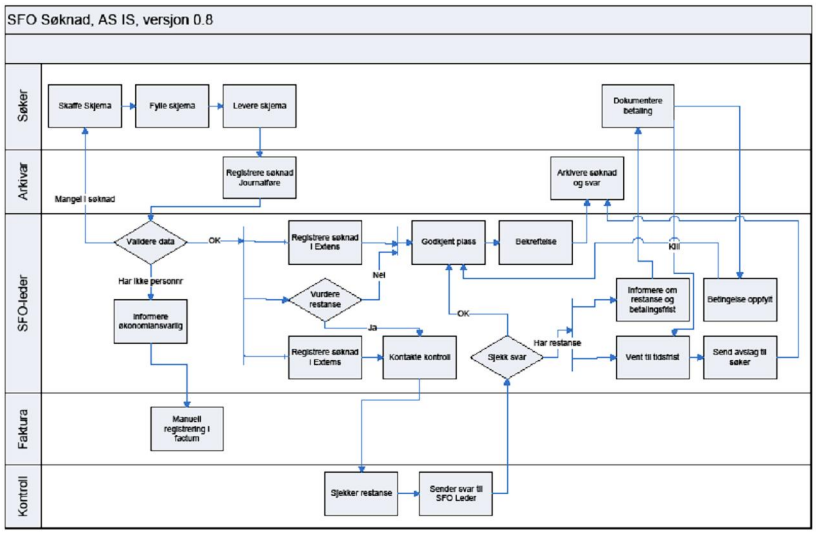
- Designe og implementere automatisert løsning for SFO-innmelding så nært opp til produksjon som mulig
- Implementere SOA-plattformen
- Etablere arkitektur og forvaltningsprinsipper for bruk av SOA-plattform
- Etablere sikker pålogging med Kobling til Min Side

AS-IS kartlegging

Først identifiseres dagens arbeidsprosess:

- Hva som er input til prosessen
- Hvilke aktører som er involvert
- Alle aktiviteter
- Alle beslutningspunkter
- Ønsket resultat

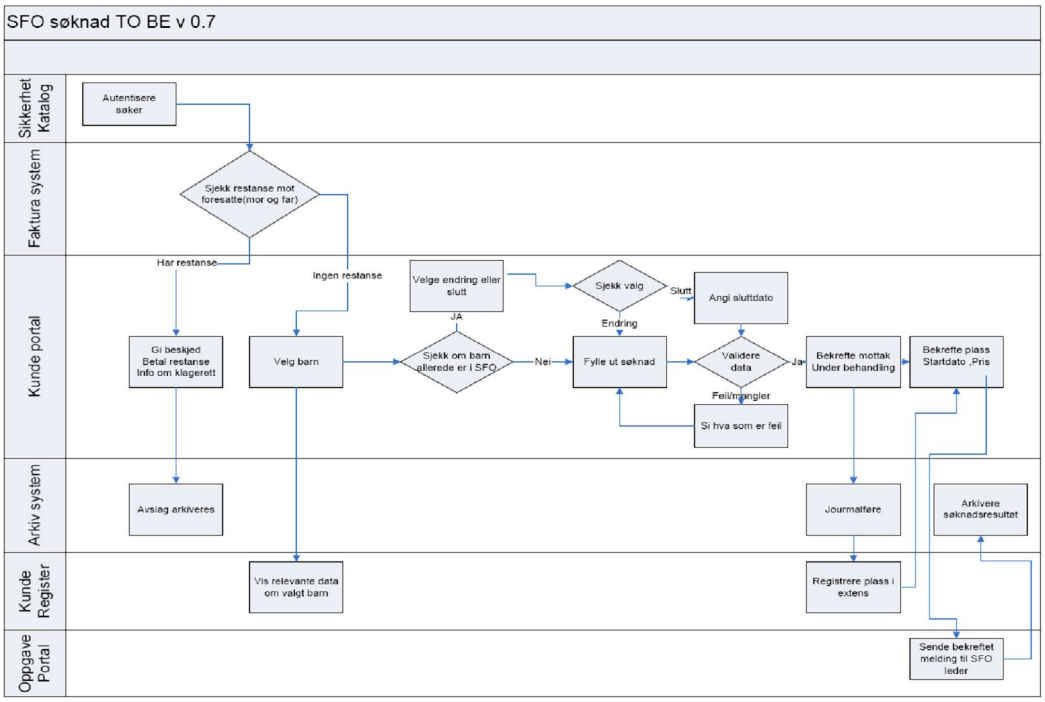




TO-BE kartlegging

Deretter lages en enkel og lett forståelig modell av ønsket prosess med fokus på

- Aktiviteter som skaper verdi for innbyggere
- At prosessen skal øke tjenestens leveranse og kvalitet
- At alle lover og regler etterleves



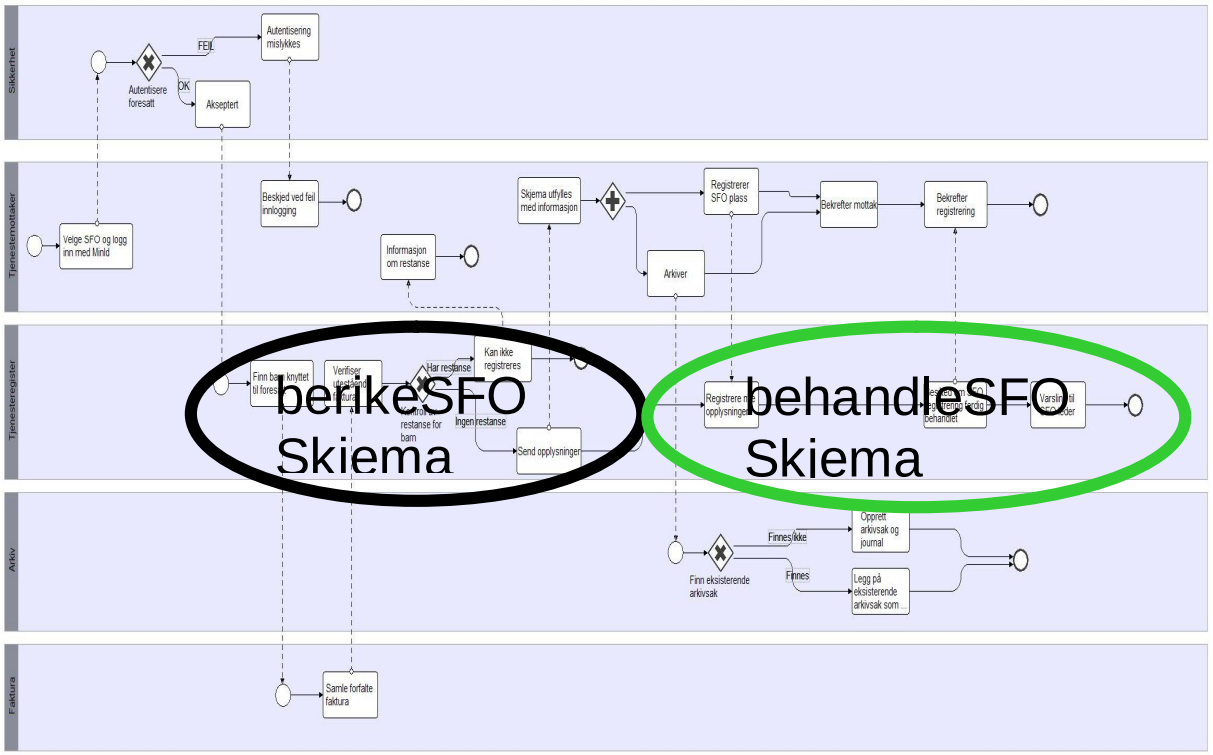
Dokumentasjon

Alle tjenester spesifiseres i 2 nivåer

- **Funksjonell Tjenestebeskrivelse**
  hensikt, eier, beskrivelse av tjeneste, funksjonell [prosess] design, egenskaper, bruksmønstre, funksjonelle avhengigheter og –avvikshåndtering

- **Tjeneste Arkitekturbeskrivelse**
  implementasjon design (BPMN og/eller Use case), informasjon modell, interaksjon med klient (metoder), krav til drift og sikkerhet, teknisk avvikshåndtering, avhengigheter til valgt teknologi, SLA avtale

Utifra denne dokumentasjonen implementeres prosessen med tilhørende tjenester på SOA-infrastrukturen. Prosessen brytes ned i mindre deler som defineres i prosessmodelleringsverktøy.



SOA (Tjenesteorientert arkitektur)

Fordeler med SOA

- **SOA muliggjør bruk av automatiske prosesser.** Automatiserte prosesser kan effektivisere organisasjonen ved å ta over oppgaver som tidligere måtte utføres manuelt. I tillegg vil automatiske prosesser ofte kunne øke kvaliteten på tjenestene som leveres. Et relevant eksempel kan være skjema på nett. Kvaliteten på leveransen kan økes ved at prosessen henter relevant informasjon fra fagsystemene og forhåndsutfyller felter i skjemaet. Ved å implementere forretningsregler kan behandlingen av skjemaet helt eller delvis automatiseres slik at ressurser frigjøres til andre oppgaver.

- **Synkroniserer IT og organisasjon.** Det er ikke uvanlig at organisasjoner har måttet tilpasse sine rutiner til begrensninger i IT systemene som brukes. SOA leverer IT tjenester som i større grad tilfredsstiller organisasjonens behov for smidighet og hurtige endringer.

- **Forenkler og forener kommunikasjon mellom alle tekniske nivåer**. En viktig komponent i SOA er mellomvarelaget. De ulike andre komponentene i brukergrensesnitt, prosesser og fagsystem kommuniserer med hverandre gjennom mellomvaren, som da kan tilby et felles og enhetlig grensesnitt for all form for kommunikasjon.

- **Gjenbrukbare tjenester.** I en organisasjon vil det finnes mange grunnleggende IT tjenester som mange andre tjenester og prosesser er avhengige av. Når alle disse tjenestene tilbys med et felles grensesnitt gjennom mellomvaren, kan de gjenbrukes uten at spesialtilpasninger og frittstående integrasjoner er nødvendig.

- **Man kan innføre SOA gradvis i organisasjonen.** De grunnleggende SOA komponentene kan legges til organisasjonens eksisterende IT arkitektur uten at endringer eller større investeringer er nødvendig. Når man utvikler de første tjenestene og prosessene i SOA, legger man til de integrasjonspunktene man trenger. Etter hvert som flere tjenester innføres, vil flere av disse kunne gjenbrukes av nye og endrede applikasjoner og prosesser. I starten er også de maskinressurser som trengs i en SO Arkitektur minimale med mulighet for å skalere ved behov knyttet til økt trafikk, flere integrasjonspunkter og antallet nye prosesser.

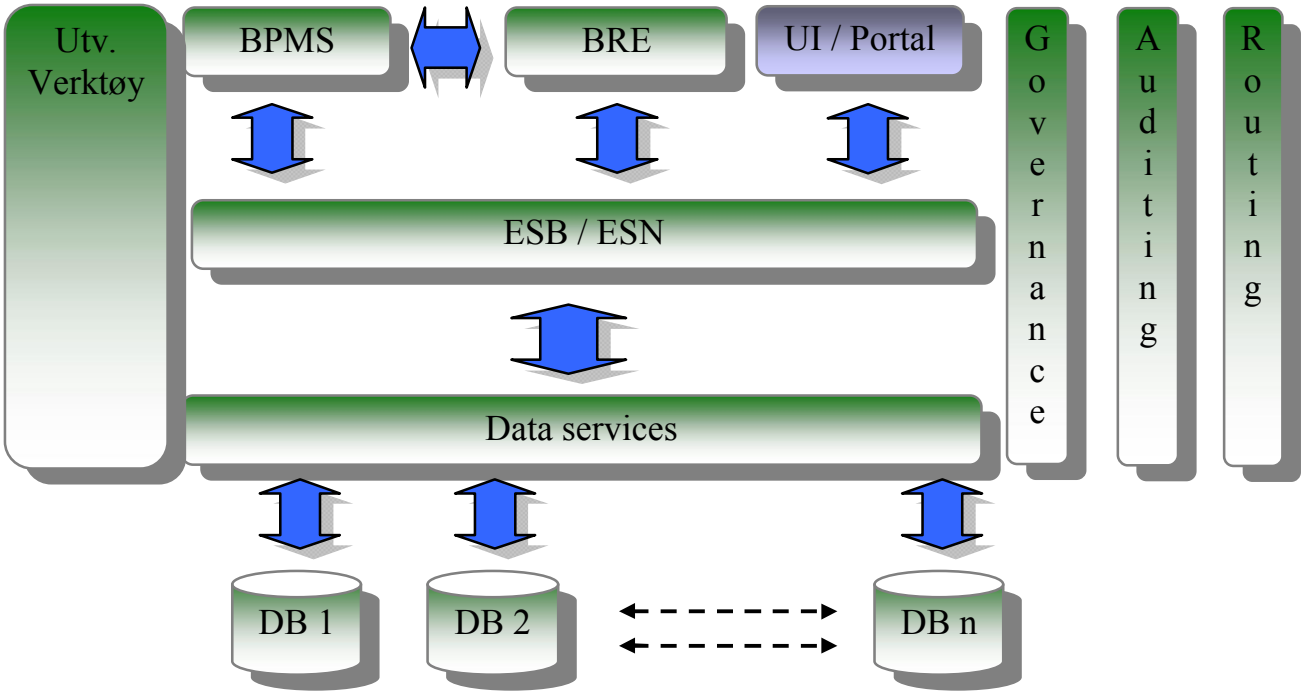
- **Forenkler samlet overvåknings og logging strategi.** Når en komponent integreres i SOA kan også overvåkning og logging integreres mot et felles grensesnitt i mellomvaren. På sikt vil all overvåkning og alle logger kunne samles og behandles av ett system med felles grensesnitt mot systemoperatører og komponenter introdusert i arkitekturen.

- **Enhetlig IT strategi senker kostnader.** SOA begrenser behovet for spesialtilpasninger og reduserer antallet integrasjonspunkter mellom IT systemene. Samtidig forenkles drift og overvåkning og man oppnår god effektivisering ved implementasjon av automatiserte prosesser.

- **Muliggjør fokus på nyutvikling fremfor vedlikehold.** Ettersom komponenter i en SOA kommuniserer med et felles grensesnitt gjennom mellomvaren, frigjøres de ulike komponentene fra hverandres interne endringer. Hvis en komponent endres eller skiftes ut, vil endringene bare påvirke denne komponentens kobling mot mellomvaren. Man slipper å videreføre endringene til andre komponenter. Dette frigjør spesielt ressurser som vanligvis allokeres av aktiviteter som for eksempel forvaltnings- og vedlikeholds oppgaver og gjør det mulig å flytte fokus til utvikling av nye tjenester for organisasjonen.

- **Kan bygges på eksisterende fysiske nettverk.** SOA baserer seg på åpne og standardiserte kommunikasjonsprotokoller og krever ingen eller minimale endringer i eksisterende infrastruktur utover maskinressurser til selve SOA komponentene.

SOArkitektur



Generelle komponenter i SOA

- **Utviklingsverktøy.** I SOA benyttes fortrinnsvis standard utviklingsverktøy som Java, .NET og Delphi.

- **BPMS (Business Process Management System).** Et verktøy hvor man kan designe og kjøre automatiserte prosesser. Prosessene orkestrerer tjenester som tilbys av andre komponenter i SOA.

- **BRE (Business Rule Engine).** Lar deg implementere automatisk behandling av forretningsregler. Forretningsreglene benyttes i behandlingen av automatiske prosesser.

- **Brukergrensesnitt/Portal.** Brukergrensesnitt for igangsetting og interaksjon med automatiske prosesser. Typisk eksempel er skjema for offentlige tjenester på nett.

- **ESN/ESN (Enterprise Service Bus/Enterprise Service Network).** Mellomvaren er den mest sentrale komponenten I SOA. Alle andre komponenter og systemer kommuniserer gjennom mellomvaren. Mellomvaren tilbyr et felles enhetlig grensesnitt som alle andre komponenter forholder seg til. Mellomvaren kan også oversette der andre komponenter er låst til sine proprietære format, sørge for last balansering eller videresende forespørsler til erstatningssystemer ved nedetid.

- **Data services.** Data services er et samlebegrep for tjenester i de ulike fagsystemene. De henter data fra og skriver data til de ulike databasene i organisasjonen.

- **Governance.** Forvaltning av SOA komponenter og integrasjonspunkter. Holder rede på konfigurasjoner, definisjoner og andre artifakter som hører til hvert enkelt integrasjonspunkt og eventuelt ulike versjoner av de enkelte integrasjonspunkter. Forvaltningsverktøyet brukes også til å rulle ut nye eller endre eksisterende integrasjonspunkt.

- **Auditing.** Overvåkningsregime for SOA. Her får man tilgang til statistikker og logger til både enkeltstående integrasjonspunkt og for hele SOA. Det er også mulig å sette opp alarmer ved gitte hendelser eller tilstander, slik at nøkkelpersoner kan varsles hvis det oppstår feil, problemer eller uventede situasjoner i SOA.

- **DB.** Databaser som inneholder dataverdiene i organisasjonen.

Kriterier for valg av komponenter

- **Storingsmelding nr. 17 (2006-2007).** Anbefaler at kommuner skal forsøke å finne egnete produkter under fri kildekodelisens fremfor kjøpevare der det er mulig.

- **Ikke versjon 1.0 eller bare beta utgivelse.** Det er en stor fordel at et produkt har hatt mulighet til å "vokse av seg barnesykdommer" før man tar det inn i en så sentral rolle som en SOA komponent. Vi anbefaler derfor at man fortrinnsvis velger produkter som har vært ute i markedet en stund og som har modnet gjennom et par versjoner.

- **Mange installasjoner.** Antallet installasjoner av et produkt, spesielt hvis disse installasjonene finnes hos større, etablerte aktører, er ofte en god indikasjon på kvalitet.

- **Aktivt miljø.** Aktivitet på diskusjonsfora og ellers i miljøet rundt et produkt er en god indikasjon på hvorvidt produktet faktisk er i aktiv bruk. Spesielt for fri kildekodeprodukter er også slike fora et av de beste stedene for å få hjelp og støtte.

- **Mulighet for bistand / kurs i Norge.** Det er viktig å være klar over at det fort kan påløpe store kostnader hvis mange brukere må sendes på kurs i utlandet for opplæring av forretningskritiske produkter. Det samme gjelder ved eventuell teknisk bistand av fagpersonell fra utlandet eller innleie fra utlandet for opplæring lokalt i organisasjonen. Mange konsulentselskaper i Norge har spesialisert seg på kurs og bistand for mange fri kildekodeprodukter, så det er viktig å avklare tilstedeværelsen av slike tilbud i Norge for de enkelte produkter som vurderes.

- **Skal primært bruke åpne standarder i grensesnittene / tilgangspunktene.** Åpne standarder gjør integrasjon mot systemer fra andre leverandører enklere. Det finnes typisk også mye mer og bedre dokumentasjon på åpne enn proprietære standarder.

- **Støtte for multiple og standard kommunikasjonsprotokoller**. Når man skal lage integrasjonspunkter som virker på tvers av organisasjonens IT systemer er det viktig at alle de vanligste protokollene (fil, ftp, http, https, osv) støttes. Mange av organisasjonens eksisterende komponenter eller løsninger kan være bundet opp til en eller noen få åpne protokoller.

- **Multiple plattformer (Linux, Windows).** Ettersom SOA komponentene på sikt vil distribueres over mange forskjellige maskiner på grunn av f.eks. last balansering, soner osv., er det en fordel om komponentene kan kjøres på mer enn bare en bestemt plattform. Dette øker også sannsynligheten for at

komponentene kan benyttes på de plattformer som allerede finnes i organisasjonen.

- **Ukompliserte /effektive rammeverk.** Rammeverket som SOA komponentene utgjør bør være enkle å sette opp og drifte. De bør heller ikke være unødvendig ressurskrevende. I tillegg vil opplæringstiden for videreutvikling av nye eller eksisterende komponenter i rammeverket være minimal.

- **Støtte for òstandardò sikkerhetsregimer (LDAP, Acegi, WS-S etc.).** SOA komponentene må kunne bruke samt kommunisere med de standard sikkerhetsregimene som allerede eksisterer i organisasjonen.

- **Enkelt å programmatisk utvide / tilpasse komponenter.** Ingen SOA rammeverk vil være komplette ut i fra behovene til en større organisasjon med mange eksisterende IT systemer. Det er derfor viktig at SOA komponentene enkelt lar seg utvide og tilpasse med andre moduler.

Kriterier for valg av komponenter - Bærum spesifikt

- **Enkel integrasjon mot eksisterende fagsystemer og nye integrasjonspunkter.** For Bærum kommune var det spesielt viktig at de SOA komponentene som skulle anskaffes enkelt lot seg integrere mot alle de eksisterende fagsystemene vi har i vår organisasjon (WS, API, direkte DB oppslag, fillesning etc.).

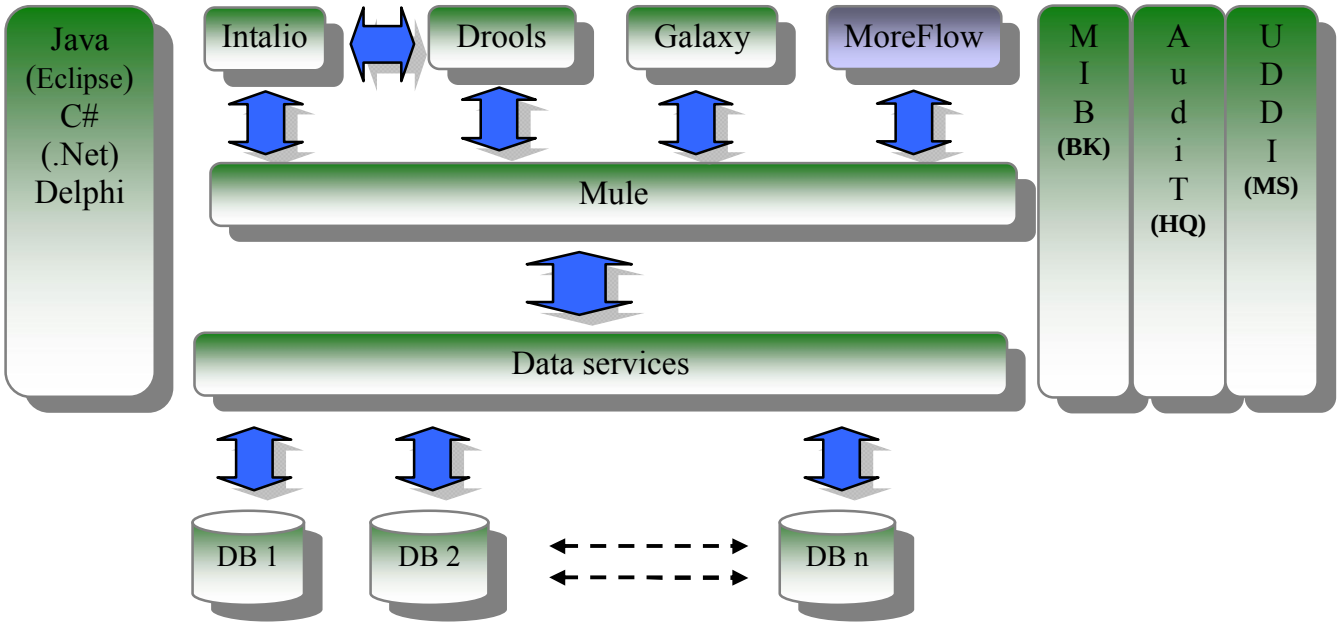
- **PoC => Infrastruktur.** Bærum kommune ønsket å gjennomføre en PoC med nøyaktig de samme komponentene som vi senere ville innarbeide i vår SOA. Vi ønsket ikke å skifte ut de komponentene som ble introdusert i PoC når SOA infrastrukturen skulle etableres.

- **SOA forvaltning bør fungere parallelt med eksisterende forvaltningsregime.** Bærum kommune har allerede etablert gode fungerende forvaltningsregimer for våre eksisterende fagsystemer. Nye forvaltningskomponenter som introduseres grunnet innføring av SOA må derfor kunne fungere parallelt eller innunder de eksisterende regimer.

- **SOA overvåkning bør fungere parallelt med eksisterende overvåkningsregime.** Bærum kommune har allerede etablert gode fungerende overvåkningsregimer for våre eksisterende fagsystemer. Nye overvåkningskomponenter som introduseres for SOA infrastrukturen må derfor kunne fungere parallelt eller innunder de eksisterende regimer for monitorering og logging.

- **Utviklingsverktøy.** Bærum kommune driver utvikling i programmeringsspråkene Delphi, C# og Java. Eventuelle utvidelser eller nyutvikling relatert til SOA komponenter og rammeverk bør derfor støtte minst et av disse språkene.

BK ï SOArkitektur



Komponenter i Bærum kommunes SOA infrastruktur

- **ESN/ESN (Enterprise Service Bus/Enterprise Service Network).** Som ESB/ESN har Bærum kommune valgt fri kildekode produktet Mule (http://www.mulesource.org).

- **Utviklingsverktøy.** Ettersom Mule er utviklet i Java, foregår utviklingen av utvidelser til Mule også i Java. I tillegg har vi en del utvikling av dataservices og nettside relaterte applikasjoner (siter) i Delphi og C#.

- **BPMS (Business Process Management System).** Som BPMS benyttes Intalio (http://www.intalio.org). Intalio består av en BPMN designer for prosess diagrammer. Med denne kan man generere eksekverbare prosesser som kjøres i en tilhørende BPML server.

- **BRE (Business Rule Engine).** For eksekvering av forretningsregler har vi valgt regelmotoren Drools (http://jboss.org/drools). Drools samarbeider for tiden med Intalio og det forventes at det vil bli mulig å lage og endre forretningsregler direkte fra designverktøyet til Intalio i fremtiden.

- **Brukergrensesnitt/Portal.** I PoC ble brukergrensesnittet presentert i KF-Skjema fra Kommuneforlaget (http://www.kommuneforlaget.no), med produktet MoreFlow som er utviklet av More (http://www.more.no).

- **Data services.** Bærum kommunes dataservices er enten produkter levert av eksterne leverandører/aktører eller utviklet internt av BKB Data-Systemavdelingen.

- **Governance.** Til forvaltning av SOA komponenter og integrasjonspunkter har Bærum kommune valgt Mule Galaxy (http://www.mulesource.org). Galaxy har mye innebygget funksjonalitet tilpasset Mule, men støtter også andre artifakter.

- **Auditing.** Som overvåkningsregime for SOA komponentene vil Mule HQ introduseres. Mule HQ bygger på Hyperic (http://www.hyperic.com). Som Galaxy har HQ mye innebygget funksjonalitet tilpasset Mule, men støtter også overvåkning av operativsystemet sine bestanddeler som minneforbruk, ledig diskplass etc. I tillegg kan andre programvarekomponenter, f. eks. ActiveMQ og .Net, overvåkes såfremt de benytter overvåkingsprotokoller alà JXM.

- **DB.** Bærum kommune bruker fortrinnsvis MS-SQL, men har også andre databaser servere som Oracle og MySQL.

Delvis introduserte komponenter

- **Galaxy**
  - Manglet mulighet for å lagre forvaltningsinformasjonen i egen database / predefinert destinasjon. Vi ønsker å kunne lagre forvaltningsinformasjonen i en eller flere databaser der vi allerede har forvaltningsinformasjon av eksisterende systemer / løsninger lagret.

  - Kan ikke assosiere properties-filer med konfigurasjonsfiler. Mule støtter muligheten for å trekke for eksempel maskinspesifikk informasjon ut fra konfigurasjonsfilene til egne properties filer. Galaxy støtter dessverre ikke noen form for kobling mellom properties filene og korrekt konfigurasjons fil.
- **HQ**
  - Er utenfor scopet av selve PoC'en, men testes for fremtidig bruk.

Erfaringer

Teknologi

Under Poc og i startfasen av virksomhetsarkitektur designet er det erfart at modenheten av open source programvare er veldig variabel. I de underliggende kapitler beskrives i kortform de hovedmomenter som krevde mest ressurser for å oppnå tilfredsstillende resultat.

Mule 1.4 ⇔ Intalio 5.1
Det var store grensesnittproblemer mellom Intalio 5.0 og Mule 1.4. Mule 1.4 støtter kun Axis 1 og Xfire, hvorav Intalio 5.0 er designet for bruk av Axis 2.

- Axis 1 fungerer dårlig i grensesnittet mot Intalio
- Axis 2 er støttes ikke av Mule 1.4.
- XFire fungerer best av de eksisterende alternativene, men skaper problemer da den hele tiden legger til ”ArrayOf” på alle listetyper definert i XSD.
- Intalio med Axis 2 er orientert mot ”Contract first” WSDL utvikling, mens Mule med XFire er mer XSD orientert.

Mule ⇔ Fagsystemer
Mule med XFire mot Delphi 2007 WS og .Net 2.0 WS fungerer uten problemer når Mule er satt opp som proxy.
Ved alle situasjoner hvor det skal skje en eller annen form for transformasjon ble det standardisert på CXF (ver. 2.0.5) i en egen komponent.
Et alternativ til CXF er å utvikle egne transformatorer for XML binding (med JiBX, JAXB eller lignende).
For å generere C# klasser i .Net (xsd.exe) er det nødvendig å lage globale element definisjoner i XSD’ene da klassegeneratoren til .Net ikke er beregnet på å generere klasser kun ut fra complex- og simpletype definisjoner.

Systemspesifikt grensesnitt
Som i vanlig lag orientert (n-tier) arkitektur ble det designet to typer kontekster for å bistå informasjonsflyten i hele arkitekturen.

Innkontekst
- Korrelasjons id
- Annen prosess spesifikk informasjon

Utkontekst
- Resultat av forespørsler
- Korrelasjons id
- Statusinformasjon (Feilkoder og beskrivelser)

I statusinformasjonen skilles det på forretningsorienterte feil og teknisk orienterte feil.

I tillegg fanges unntak (Exceptions) som sendes til et eget feil håndteringssystem. Denne type feil logges og kan eventuelt videreformidles for eksempel per epost til den som er ansvarlig for drifting av systemet. Til den kallende prosess sendes det en melding om teknisk feil i et objekt av typen utkontekst.

Semantikk

- **Forretning ⇔ Teknisk**
  - Semantikk er et viktig begrep når man arbeider med SOA. Det er viktig å være bevisst på at man må forholde seg til semantikk på to helt adskilte nivåer.

    - For å få SOA til å fungere må man ha fokus på teknisk semantikk. Det vil si at det må defineres klare regler for hvordan grensesnitt skal utvikles og beskrives, slik at alle komponenter og integrasjonspunkter blir enhetlige og "snakker samme språk". Det er også viktig å opparbeide seg et felles typebibliotek som alle integrasjonspunkter benytter seg av slik at samme type har samme navn, format og innhold på tvers av arkitekturen. På samme måte må samme navn referere til samme type i alle integrasjonspunkter.

    - På et helt annet nivå er det viktig å arbeide med semantikk på et mellommenneskelig nivå. For at SOA skal kunne fungere er det viktig at tjenesteeiere fra forretningsområdene og teknisk personell fra IT avdelingen samarbeider. Det er da viktig å bli enige om en del felles begreper og betydningene av disse, slik at man opparbeider seg et felles samtaleunivers. Hvis ikke vil man ofte kunne oppleve at ord og uttrykk kan ha svært forskjellige betydninger i de ulike avdelingene i organisasjonen.

  - SOA krever andre type bestiller og utfører roller med tilhørende kompetanse enn hva som er har vært tradisjonelt ved programvareutvikling. For å få smidige prosesser som best tjener forretningen bør deltagere både fra forretningssiden og tekniske utviklere samarbeide og møtes jevnlig gjennom hele prosjektet. Det bør dessuten finnes ressurser med SOA som spesialkompetanse. Disse ressursene må spesielt kjenne til eksisterende tjenester for best mulig gjenbruk og nedsatte retningslinjer ved all form fr videreutvikling.

  - SOA prosjekter har typisk mer fokus på tekniske beskrivelser med XSD og XML etc. enn "vanlige" utviklingsprosjekter. Dette er viktig for å bibeholde semantikk og enhetlige grensesnitt. Grensesnittene mellom alle komponentene som inngår i prosjektet bør i størst mulig grad defineres i en meget tidlig fase av prosjektet (muliggjør contract first), slik at komponenter kan utvikles mest mulig individuelt og med minst

mulig innbyrdes avhengighet.

- PoC prosjektet i Bærum kommune vurderte muligheten for å benytte allerede eksisterende klassifiseringssystemer, men kom til at dette ville kreve for mye ressurser av en PoC som var rettet mot kompetanse og erfaringsoppbygging.

- Et felles samtaleunivers ble bygget opp med begreper mer tilhørende den forretningsorienterte siden enn den tekniske.

- Kommunikasjon mellom forretning og teknisk ble beskrevet i to typer dokumentasjon.

    - Funksjonell tjeneste dokumentasjon beskriver prosessen ut i fra forretningsmessige hensyn. Det fokuseres på forretningsregler, aktører nødvendige data og ønskede resultater.

    - Teknisk tjenestedokumentasjon blir utledet fra den funksjonelle tjeneste dokumentasjonen og detaljerer de ulike elementene presist på teknisk nivå.

- For å oppnå best mulig konsishet av funksjonell og teknisk tjeneste dokumentasjon var det fornuftig at denne oppgave ble tildelt til en og kun en ressurs. Ut fra et organisatorisk og kunnskapsmessig synspunkt ble oppgaven satt til rollen; informasjonsarkitekten.

Fri programvare

- Fri programvare er ikke gratis, men koster mindre!
- Fri programvare til bruk i sentrale løsninger og i infrastruktur medfører
- Åpenhet, standardisering og mye tilgjengelig kompetanse
- Må ha betalbare tilleggsprodukter utover community-versjon – og det finnes masse valg
- Må ha avtale for support- og versjonsgarantier
- Applikasjoner basert på fri programvare
- Finnes få løsninger for kommunal anvendelse i dag
- Vi er svært godt fornøyd med valgene som stort sett fungerer som forventet. Selv om begge bruker samme åpne standarder er det noen tilpasninger for å få til samhandling mellom de ulike lagene.
- God og tilgjengelig norsk/lokal kompetanse/support/kursing på MULE i tillegg til på nettet
- Litt mindre for Intalio, men der er det derimot svært mange kunder

Gjennomføring

Samhandling mellom virksomhet og teknisk er utfordrende, men helt nødvendig og følgende hovedpunkter pekte seg ut som de mest ressurskrevende:

- Mye tid brukt til å vente på leveranser fra leverandører, eller gi gjentatte tilbakemeldinger.
- Vanskelig å implementere infrastruktur, tjeneste og rammeverk samtidig.
- Prosjektet ble mye mer teknisk enn antatt.
- Utfordrende å systematisere dokumentasjon.
- Ting tar tid!

Leverandører

Løsninger/produkter levert av flere leverandører er ofte teknologiske løsninger med foreldet arkitektur og er ennå ikke modne for leveranser med SOA orientert grensesnitt eller tankegang. Forretningsstrategien er i stor grad salg av ”silo” produkter og ikke tjenester som passer inn i en SOA. Leverandørene er lite fleksible for nye typer leveranser som kreves i en SOA.

Juss

Vanligvis er applikasjonsdomenet sterk knyttet til en eller flere applikasjoner med tilhørende data og eies samt forvaltes av en systemeier. Fra en ”Silo” tankegang til SOA vil fagsystemer som eksponerer tjenester benyttes av flere tjenestebrukere og skape nye juridiske problemstillinger. F. eks. Hvem er ansvarlig for oppetiden av en tjeneste? Hvem skal varsles ved endringer i tjenesten? Hvem skal betale for endringer? Etc.

Eierskap og ansvar

- **Hvem bør eier tjenesten?**
  - Den som er ansvarlig for prosessens sluttresultat og derfor motivert for å gjennomføre kontinuerlig effektivisering.
  - Eierskap må også innebære full og detaljert forståelse av tjenesten med tilhørende prosess

- **Viktig å avklare eierskap til regelverket**
  - Ingen rom for ”personlige tolkninger” av regelverk i automatisert prosess
  - Automatisering betyr ofte endrede oppgaver for folk

- **Hvem har ansvar for forvaltning av prosessen og regelverket etter implementasjon?**

- **Hvem skal betale for utviklingskostnadene relatert til eksponering av tjenester?**
  - Systemeier?

  - De som ønsker at systemet eller deler av systemet skal eksponeres og tilbys som tjenester?
  - Begge?
- **Drift**
  - Hvem er ansvarlig for drift av tjenesten?
  - Krav til oppetid? Forventes det 24/7 oppetid?
  - SLA kontrakter; Er de i eksisterende kontrakter dekkende nok?

- **Forvaltning og videreutvikling**
  - Hvem er ansvarlig for forvaltning og videreutvikling av tjenesten?

Arkitektur og forvaltningsprinsipper

Dette arbeidet ble minst ferdig av de oppgavene som var planlagt. Dette skyldes til en viss grad nødvendigheten av å få på plass en infrastruktur for å kunne teste løsningen, samt at prosjektet stort sett ble drevet av IKT-ressurser. For å få dette på plass er det nødvendig med en sterk forankring (se punktet under) og involvering fra virksomheten. Resultatet av dette arbeidet vil trolig få innvirkning på kommunens styringsmodell innen IKT, noe som er en stor prosess å starte. For å gjennomføre SOA og virksomhetsarkitektur kreves det en sterk styring og forvaltning.
OIO rammeverk ble delvis gjennomgått, men dette var for omfattende til å klare å implementere samtidig med å få på plass en teknisk løsning. Det er trolig heller ikke lurt å overføre alle OIO-elementer direkte, men lage en egen av-art av OIO for Bærum.

Suksessfaktorer for implementering av SOA

- Prosjektet må ha forankring oppover i kommunen
- Må inkludere BKB Data sin kompetanse og ressurser på området
- Må være åpne for at prosjektet medfører omstrukturering av IKT-oppgaver i kommunen.
- Viktig å få lagt oppgavene i riktig rekkefølge
- Dedikerte ressurser må ha tilstrekkelig avsatt tid, må samordnes med fremdriftsplan
- Prioritere fremtidige prosjekter i forhold til hvor bruk av SOA-arkitektur vil gi størst nytteverdig (volum, gevinstpotensiale)
- Forventningsavklaring
- Samarbeid og erfaringsutveksling med andre virksomheter

Sluttføre arbeidet med SFO prosess ï fra pilot til drift

- Sluttføring test og implementering av Galaxy som SOA governance i parallell med eksisterende forvaltningsløsninger i kommunen.
- Sluttføring av HQ i parallell med Galaxy. Det legges opp til at HQ skal benyttes som auditing verktøy for både SOA løsninger og eksisterende løsninger der det er hensiktsmessig.
- Drools er prosjektert for testing desember 2008 – januar 2009.
- UDDI –Q1 2009 (bruk av både statiske og dynamiske WSDL'er).
- Bedre organisasjons assemblering i prosjektstart med utgangspunkt i OIO eller lignende metodikk.
- Videreutvikle virksomhetsarkitektur og infrastruktur i hele kommunen.
- Kartlegge applikasjonsdomener og potensielle tjenester felles for hele kommunen

Etter dette arbeidet vil Bærum kommune ha en tilnærmet komplett SOArkitektur i drift, og kan implementere nye prosesser og tjenester på denne.

Nytt prosjekt ï oppstart i januar 2009

Et nytt prosjekt vil bli startet i januar 2009 for å innføre virksomhetsarkitektur i Bærum kommune. Rammene for prosjektet er ikke helt klare ennå, men følgende oppgaver er allerede identifisert etter arbeidet med implementering av SOA:

- Gjennomgå rapport fra for-prosjekt for videre oppfølging
- Skaffe status på arkitekturen (teknisk-, informasjons-, tjeneste-)
- Definere hvilke stamdata vi trenger å ha kontroll på
  - Fellesdata (kontoplan, person…)
  - Datakvalitet
  - Sikre eierskap og antall registreringer
  - Sikre enhetlig gjenbruk
- Fullføre arkitekturprinsipper
- Etablere arkitekturforum (del av styringsmodell) for å sikre virksomhetsfokus
- Identifisere roller og ansvar (BK-OIO?)
- Etablere forvaltningsregime (retningslinjer)
- Kartlegge og bestemme eierskap til systemer, data og infrastruktur
- Tegne hovedprosessen(e) i kommunen