INF3290

Obligatorisk innlevering 4

Sluttrapport: Gruppe 6

Madelen H. Ljunggren

Daniel Lange

Kathrine Tangård

Jonathan Rykkja Ibsen

Abdinor Mahamed

Innholdsfortegnelse

Innledning

Vi valgte FINN.no som et prosjekt for å analysere deres kompleksitet i informasjonsinfrastrukturen, og hvordan de håndterer sine legacy-systemer, integrasjoner og standardisering. Vi ønsket å tilegne oss teoretiske perspektiver for å analysere og forstå kompleksiteten knyttet til informasjonssystemene til bedriften. Vår motivasjon til å jobbe med FINN er fordi vi har selv sett hvordan de har ekspandert, alt fra torget til reiser og småjobber. Det er også Norges største markedsplass for kjøp og salg mellom privatpersoner og små og store bedrifter.

Arkitekturen er kompleks ettersom systemutvikling sjeldent skjer fra bunn, det er en installert base der og en legacy-database som er utdatert. Arkitekturen til Finn har tidligere vært monolittisk arkitektur, men for to år siden så ønsket de å bytte til mikrotjenester, API og domenedrevet design. Finn har derfor satt seg noen mål om hvordan de skal løse kompleksitet. De har valgt å bruke arkitekturen til å adressere kompleksiteten ved å unngå store endringer i infrastrukturen og minimalisere risikoen og opprettholde kvaliteten over tid.

Kapittel 1: Bruk av informasjonssystemer

1.1 Om Finn.no

FINN.no ble etablert i mars 2000 og er Norges største markedsplass. De har spesialisert seg på tjenester og annonser for kjøp og salg mellom privatpersoner, samt store og små bedrifter. Schibsted Media Group eier 90,01% av FINN AS og Polaris Media eier 9,99% (Finn.no AS, u.å.). Schibsted Media Group er Norges største mediekonsern og en av Skandinavias store medieaktører med ca 6800 ansatte og en brukerbase på rundt 200 millioner mennesker. Schibsted er representert i omkring 30 land i hele verden med digitale markedsplasser (Schibsted Media Group, u.å.). Schibsted prøver nå å samle produkt, teknologi og organisasjon på tvers av alle landene. Dette krever standardisering og skaper en del utfordringer, som vi vil adressere senere. (Verheughe, 2016).

FINN har lokaler på Grensen i Oslo. De har for øyeblikket omkring 400 fast ansatte medarbeidere. Disse ansatte har blant annet stillinger som kundebehandlere, systemarkitekter, utviklere, sikkerhetsrådgivere, systemutviklere med mange fler. De har rundt 120 utviklere, leverer opp til 250 tjenester, og utgir endringer til produksjon omtrent 150 ganger om dagen. Dette fører til en omsetning på nærmere 1,4 milliarder.

FINN er delt inn i 12 avdelinger, og hver avdeling har ansvar for hvert sitt marked av de 12 markedene de har å tilby. De 12 markedene er; Eiendom, Bil, Torget, Jobb, MC, Båt, Småjobber, Reise, Oppdrag, Nyttekjøretøy, Kart og Møteplassen. De har også mange flere underavdelinger, noen eksempler er HR, utvikling, support, selger og innsiktsavdeling. Markedene Jobb, Bil og Eiendom var de tre første markedene på FINN, etablert i år 2000. Videre kom Torget i 2003, Reise i 2006 og Oppdrag i 2011.

FINN samarbeider med eksterne partnere som blant annet Politiet, Toll, Forbrukerrådet osv., men dette er bare når det kommer til svindel eller annen ulovligheter. De samarbeider også med Proffcom AS i Kristiansund, som er et av Norges mest ledende Callcenter. FINN gir

salgs- og kundeserviceoppdrag til Proffcom, som oftes jobber hos FINN i helgene med kundeservice.

1.2 Gjennomføring

I dette kapittelet har vi valgt å fokusere på kundebehandleren som bruker av forskjellige IT-systemer innad i Finn, og hvordan kundebehandleren tar i bruk disse systemene for å løse sine arbeidsoppgaver. I første trinn i gjennomføringen av prosjektet har vi intervjuet en kundebehandler som jobber med henvendelser fra kunder og bedrifter på telefon, e-post og chat. Intervjuobjektet ble informert om hvilke temaer vi ønsket å snakke om på forhånd, samt at vi introduserte prosjektet og emnet "Store og komplekse informasjonssystemer" ved begynnelsen av intervjuet. Som god forsknings- og intervjuetikk fikk intervjuobjektet dette også skriftlig, på et samtykkeerklæringsskjema som begge parter skrev under på. Her ble det også informert om at det ble tatt opp lydopptak, at dataene fra intervjuet var anonymt, og at vedkommende kunne trekke seg når som helst også i ettertid.

Intervjuet med vedkommende var ca. 50 minutter langt og alle gruppemedlemmene fra vår prosjektgruppe var tilstede. Vi tar litt selvkritikk at det var kanskje litt overkant mange fra vår side (5 personer) som skulle intervjue en person, selv om bare 2 personer fungerte som intervjuere og resten observerte. Heldigvis tok vedkommende dette bra, men vi tar til lærdom at neste gang bør vi være maks forhold 3:1 på et slikt intervju, for at det blir en litt løsere stemning. Vi vil påstå at i denne omgang tok vårt intervjuobjekt dette fint, og det ble god stemning - men ikke alle intervjuobjekter vil muligens ta det like bra og dermed påvirke validiteten av dataene som blir samlet inn.

Intervjuet semi-strukturert med både åpne og lukkede spørsmål. Flesteparten av spørsmålene bygget på svar fra hovedspørsmålene. *Intervjuguide er lagt til som egen fil i innleveringen.*

1.3 Brukerperspektiv

Systemene og verktøyene FINN bruker: Telefon, E-post, chat, Zendesk, Admin, Agresso, CRM og Ambassador, og Slack og InneHos internt.

Telefon, E-post og chat

Kundebehandleren vi intervjuet bruker flere verktøy for å løse arbeidsoppgavene sine. Alt av henvendelser fra kunder tar hun via telefon, e-post og chat innenfor alle felt på FINN.no (Reise, jobb, Torget osv.). Chat er det eneste verktøyet hvor man kan hjelpe flere kunder samtidig. Samtaler fra disse verktøyene skal logges i Zendesk slik at man enkelt kan finne tilbake til den senere, f.eks hvis kunden ønsker samme kundebehandler eller en annen kunde har det samme problemet. For å være mer produktiv, logger ofte kundebehandleren samtalen fra chatten samtidig som hun sitter i telefonen med en ny kunde.

Zendesk

Zendesk er et av hovedsystemene kundebehandleren bruker. Det brukes til loggføring, chatting, mottak av e-poster og se tidligere kontakt og samtaler med kunder. Alle henvendelser til FINN loggføres i forskjellige kategorier med en logg eller sammendrag, her har kundebehandleren litt ekstra arbeid ved telefonsamtaler og chat, som det manuelt må skrives et lite sammendrag fra telefonsamtaler og henvises i en annen logg for chat. For å finne tilbake til tidligere samtaler, må kundebehandleren bruke en “FINN-kode”, som er knyttet til brukeren, via Admin. Annonser som brukere av FINN.no rapporterer, kommer som e-post til Zendesk.

Admin

Admin er administratorpanelet for kundehandlere på nettsiden, deres “backstage”. Her kan kundebehandlere og andre som har admin-tilgang endre og slette annonser, finne “FINN-koder” og hjelpe kunder med det de synes er vanskelig. Dette kan ikke gjøres i Zendesk, kun Admin.

Agresso, CRM og Ambassador

I Agresso kan kundebehandleren finne informasjon om brukeren og/eller bedriften, blant annet navn, e-post og hvilke betalinger som er gjort. Internt brukes Agresso også til registrering av ferie og fravær hos ansatte.

CRM (Customer Relationship Management) er et eget fakturasystem for bedrifter. Her finner de e-post- og fakturaadresser.

Ambassador er et system som sjekker betalingsinformasjon og hva som har kommet inn av betalinger fra bedriftskunder, siden det er eneste personer som betaler med faktura.

Slack og InneHos

Slack er et chattesystem som brukes internt i FINN. Det brukes ofte til å spørre andre ansatte om hjelp, og gjør det lettere for de å få kontakt med FINN-kontorer andre steder i Norge. InneHos er også et system som brukes internt. Her får de ansatte alt av beskjeder, som for eksempel hvordan man søker ferie, opplysninger om endringer og når det er julebord.

Arbeidsprosess - En typisk dag for kundebehandleren

På starten av arbeidsdagen, får hver ansatt i et team utdelt en timeplan med oversikt over områdene de har ansvar for den dagen, som for eksempel chat og e-post. Det er aldri noe krav til hvor mange kunder kundebehandleren må hjelpe i løpet av dagen. Kundebehandleren vi intervjuet begynte med å ta imot telefoner kl. 8 og gikk inn på chat kl.9. Hun logger seg inn i Zendesk og Admin, som er i bruk hele arbeidsdagen. De andre systemene logger hun seg inn i hvis det er nødvendig. På en typisk arbeidsdag kan kundebehandleren få en telefon fra en som trenger hjelp til å markere annonsen sin som solgt. Da må kundebehandleren få FINN-koden til annonsen/brukeren det gjelder, for og så gå videre til den via Admin og markere den som solgt. Når det er gjort, må kundebehandleren loggføre samtalen med brukeren manuelt i Zendesk, med et kort referat, hvilket område det gjaldt og hva problemet var.

Systemene kundebehandleren bruker brukes først og fremst for primærbruk (for å løse arbeidsoppgavene), som for eksempel at hun bruker Zendesk for logge og ta imot mail, eller svarer på chat og telefon for å hjelpe brukere. Informasjonen som samles inn til

sekundærbruk er hovedsaklig loggen i Zendesk, som blant kan brukes som "historie" til en bruker.

Samarbeid med andre

Arbeidet kan i de fleste tilfeller utføres alene av kundebehandleren uten noe behov for hjelp av andre. Kundebehandleren mottar en sak og behandler denne direkte overfor kunden.

Dersom kundebehandleren ikke kan besvare spørsmålet kan den hente inn hjelp fra forskjellige interne avdelinger. Det er en egen avdeling på FINN som tar seg av saker relatert til sikkerhet og svindel, utvikleravdelinger for de forskjellige tjenestene - eller så kan det være behov for å få hjelp av en annen kundebehandler. I alle disse tilfellene vil saken som oftest bare bli tildelt til en ny person i Zendesk. I tilfeller hvor det kun trengs en rask avklaring på et spesifikt spørsmål kan kundebehandleren også bare spørre den relevante ansatte via chattetjenesten Slack som benyttes internt.

Brukeren vi intervjuet var en del av et team på fem, som overlappet et annet team med de samme arbeidsoppgavene. Akkurat denne brukeren koordinerte arbeidet sitt sammen med ulike brukere, men mest med kollegaene som jobbet på samme team. De hadde et system for de ulike arbeidsoppgavene der de byttet på regelmessig slik at arbeidsoppgavene varierte. Som tidligere nevnt var hovedoppgavene telefon, mail og chat. Til de ulike systemene så hadde alle private brukere som de logget inn med, mens telefonen var et fysisk verktøy som ikke krevde noe innlogging. Brukeren koordinerer også arbeidet sitt med andre avdelinger.

Zendesk er et system som blir brukt til chat, mail og loggføring av chat-henvendelsene. Slik at andre kan få tilgang til tidligere henvendelser, og snakke med samme kundebehandler hvis ønsket. Arbeidet til de ulike brukerne påvirket hverandre i liten grad, med mindre det var feil i loggføringen. Videre var det Admin-siden til FINN.no som ble hyppig brukt sammen med andre brukere for å administrere annonser, finne FINN-koder og hjelpe kunder med det nødvendige. Agresso, Ambassador og CRM er også systemer som blir brukt sammen med andre for finne informasjon om kunder, fakturaer, betalinger epost osv. Siden

arbeidsoppgavene til brukeren vår gikk ut på å ut innhente informasjon så påvirket ikke de ulike brukerne hverandre. For å kommunisere internt brukte de Slack som er et chatteprogram og innhos.finn.no som er del av nettsiden. Her var det spørsmål og interne henvendelser som gjaldt.

Brukeren koordinerte arbeidet sitt også utenfor systemene. Det var en bruker som var ekspert på Zendesk som de fleste tok kontakt hvis det var noe som ikke stemte eller lurte på systemet. Dette foregår ved muntlig kommunikasjon. Dette gjør at frustrasjon og ventetiden blir mindre, samt kommer det også en mestringsfølelse.

IT spiller en stor rolle ettersom mye av arbeidet foregår på maskiner med ulike systemer. IT-systemene har gjort det enklere å dele/redigere/slette informasjon. Systemene kommuniserte i liten grad med hverandre, dette førte ofte til at brukeren måtte logge inn og ut av flere systemer iløpet av en vanlig arbeidsdag. Det fører til kompleksitet når du har mange systemer å forholde deg til samtidig. Få systemer og lite samhandling fungerer bra for FINN nå, men framover kan det bli mange systemer og lite samhandling. IT vil spille en større rolle for koordineringen etterhvert som FINN vokser og designet av informasjonsinfratrukturen vil komme tydeligere fram.

Kapittel 2: Beskrivelse av informasjonssystemer

2.1 IT-systemer og kommunikasjonskanaler

FINN har mange ulike teknologier i bruk. Av de ulike komponentene i informasjonsinfrastrukturen var brukerne i like stor grad i fokus som selve teknologien. For å kunne fortsette å vokse i riktig vei har FINN fokusert mye på å snakke med sine ansatte som bruker disse ulike systemene, det var dette systemarkitekten hadde som oppgave. De har delt opp de ansatte i 5 områder som har cirka 4 domener hver og ikke kun delt de ansatte etter domener som i den tekniske løsningen, dette for å skape fleksibilitet for de ansatte. De deler domene etter hvor nært relatert de er til hverandre. Dette er basert på Domenedrevet Design som FINN har satset på de to siste årene. Noe av det de jobbet med nå var hvordan domene skulle eie funksjonaliteten helt ut til brukeren, men det er ikke tilfelle nå. Da de ønsket å endre dette laget de 2 domener som de kaller plattformweb som andre skal kunne levere på. Tanken er at andre kan eie web applikasjonen, men ikke funksjonaliteten. FINN ønsker å eie helt fra databasen og helt ut til brukeren og begrunner det med å gjøre dette for å unngå flaskehals.

m.finn.no som er hovedsiden til FINN var blant de mest omfattende fordi det var nettopp her mange parter skulle ha tilgang og muligheten til å gjøre endringer. De hadde et program som leverte en nøye oversikt over alle systemene og alle domene i grafer og ulike moduler slik at sjefs arkitekten og alle utviklerne kunne følge forandringene veldig nøyaktig.

Kommunikasjonkanalene til FINN var ulike etter behovet, noen hadde kun behov for kjappe spørsmål eller svar over chatteprogrammet Slack, men videre hadde de også en intern side innehos.finn.no for mer organisatoriske saker. Som nevnt fokuserte systemarkitekten mye på snakke med de ansatte og der kunne de fortelle om ulike løsninger de ønsket eller om noe som var vanskelig i de ulike systemene. Epost og muntlig kommunikasjon var også i bruk.

2.2 Overordnet systemarkitektur

FINN er bygget opp av mange såkalte "mikrotjenester". Alle tjenestene utfører små enkeltvise oppgaver og utveksler akkurat nok informasjon med hverandre for å utføre det arbeidet de er tiltenkt.

Mikrotjenestene ligger under fagmessige domener. FINN.no som vi kjenner det er delt opp i 16 domener. Ett domene kan eksempelvis være å legge inn en annonse. Målet er at domenene kun skal representere en spesifikk arbeidsoppgave på FINN (se annonse, lage annonse osv), men det er også et domene de kaller plattformweb som omfatter drift av selve grunnplattformen alle de andre tjenestene leveres på. Hvert domene har et tilhørende team som har ansvaret for det domenet, og sitter på både den tekniske men også bedriftsnyttige kompetansen tilhørende det fagfeltet. Domenene sorterer igjen under 5 forskjellige områder.

Endringer i koden på FINN.no rulles ut kontinuerlig ut over dagen ettersom endringene ferdigstilles, et eksempel kan være endring av design eller fiks av bugs. Intervjuobjektet vårt estimerer at rundt 200 mindre endringer i koden rulles ut separat daglig på FINN.no. For å gjøre dette har FINN utviklet et eget system med navn "Pipeline". Pipeline inneholder alle utviklingsprosjekter som gjøres på FINN og har egne løp for å automatisk teste, bygge og rulle ut endringer som gjøres.

Når en programmerer har gjort en endring på "git" som er en versjonskontrollering av kode, fanges dette opp av Pipeline, knyttes til riktig prosjekt, bygges, kjøres automatiske tester på - før en får valget om å automatisk rulle det ut på FINN. Noen typer endringer rulles ut uten videre, mens andre krever manuelt oppsyn av utvikler før det rulles ut. Utrullingen skjer til fysiske maskiner driftet av IBM. Maskinene har et lag med virtualisering, men på sikt er målet til FINN å bygge på med et nytt abstrahert lag for å få en skyløsning basert på Kubernetes fra Google hvor man får en konteinerbasert infrasturktur.

Tidligere hadde FINN serverene driftet igjennom Schibsted, men har med tiden gått bort fra dette og satt dette ut til IBM. I løsningen for FINN.no er det lagt opp muligheter for å utføre

testing av nye funksjoner i produksjonsmiljøet uten at dette blir synlig for alle brukerene. Til dette benyttes et åpent kildekodeverktøy med navn "Unleash" som lar utviklerne velge og slippe ut nye funksjoner for spesifikke brukere, ansatte, halvparten av alle besøkende (A-B testing) og andre parametere.

Alle bilder og annet statisk materiale serveres igjennom en CDN (content delivery network) slik at FINN med dette får vekk en stor kompleks faktor fra serverdriften sin og kan konsentrere seg om den ovennevnte utviklingen i stedet. Det er mange ting som kan gå galt på et så omfattende nettsted som FINN uten at det nødvendigvis produserer noen synlige feilmeldinger. Det kan være alt fra en viktig knapp som ikke lenger vises på en side til linker som videresender feil. For å fange opp denne og andre typer feil som kan oppstå har FINN bygget opp et avansert system for å grafe og fange opp metrikker fra alle mulige funksjoner på nettsiden. Metrikkene måler alt fra trafikk, til hvordan salgstallene forløper seg, responstid, antall klikk, feil som oppstår, besøksstatisitkk m.m. Basert på disse metrikkene blir dermed FINN raskt i stand til å fange opp en eventuell uventet oppførsel et sted blant applikasjonene for så å etterforske det.

2.3 Aktører

Det er flere involverte aktører hos FINN. De utvikler i hovedsak det meste av sin markedsplass helt selv, men her tar de også hensyn til at Schibsted ønsker i større grad å bruke felles komponenter mellom sine bedrifter. Det har seg slik da Schibsted har kjøpt opp flere lignende markedsplasser og relevante bedrifter rundt i verden, og dermed har de innført sine egne interne standarder og retningslinjer. Dette gjør at flere deler av FINN kan brukes i andre underbedrifter av Schibsted. Meldingstjenesten som benyttes av brukere for å kommunisere mellom hverandre på FINN er for eksempel utviklet av et oppkjøpt selskap i Barcelona, og ikke på FINN sitt hovedkontor.

FINN har også egne domeneeksperter som er ansatt hos dem. De jobber dedikert med det tekniske bak det nye domenebaserte designet til selskapet og deres IT-systemer. Det leies også inn kundeservice fra Proffcom utenfor ordinær åpningstid i hverdager og i helgene.

Tidligere delte de servere med Aftenposten, men har nå gått over til IBM som serverleverandør. Selve driften gjør FINN fremdeles selv. For å gjøre siden kjappest mulig, bruker også FINN Fastly som sin CDN-løsning for å optimalisere kode, levere bilder og statisk innhold til sine nettsider, som f.eks. JavaScript, CSS og lignende. Dette er en beslutning av FINN at de finner det mer lønnsomt at en profesjonell aktør gjør dette for dem, enn at de skulle ha gjort dette selv.

Det er også brukere på FINN som selger og kjøper varer, annonsører, og ikke minst kundeservice som bruker FINN-rammeverket til å håndtere henvendelser de får inn, slik som de bruker sin eget system Admin for å se og redigere annonser, og ZenDesk for logging av henvendelser.

2.4 Standarder og lovverk

Et sentralt lovverk og en standard som FINN følger er universell utforming knyttet til deres IKT-løsninger. Universell utforming, handler om å utforme omgivelsene slik at vi tar hensyn til variasjonen i funksjonsevnene hos mennesker, inkludert personer med nedsatt funksjonsevne (Direktoratet for forvaltning og IKT, u.å.). De er derfor nødt til å passe på at IKT- løsningene deres er tilpasset alle type mennesker. Før universell utforming ble lovpålagt 1.juli 2014 hadde FINN allerede noen initiativtakere som var veldig engasjert i akkurat dette kravet. Derfor har FINN kommet veldig bra igjennom alle tester som har blitt gjort relatert til universell utforming, men de uttrykker at det å designe for alle type mennesker er et evolverende arbeid, det er ikke noe de blir ferdige med. FINN mener at innføring av universell design i løsningene sine er en fordel fordi det fører til i at flere mennesker kan benytte seg av tjenestene deres. De ser også at ved å tilrettelegge for mennesker med nedsatte funksjonsevner, som for eksempel svaksynte, resulterer i at løsningene deres blir mer intuitive.

FINN har også hatt stort fokus og jobbet mye med strenge reguleringer som ha kommet i forhold til personvernloven. De kjørte et helt eget prosjekt, som gikk på tvers av alt, hvor de

gjorde de viktigste tiltakene som måtte gjøres. En av de tingene de jobbet med i dette prosjektet var for eksempel strenge reguleringer som har kommet i forhold til hvilke data de kan ta vare på fra brukerne og hvordan de må behandle den dataen. De har aldri hatt en policy på å måtte slette data før, de har alltid tatt vare på all data, men dette ble et problem i forhold til de nye reguleringene. FINN får ikke lenger lov til å ta vare på personidentifiserbare data over tid uten av brukeren godkjenner det. De har derfor startet med å slette unødvendig data, et eksempel på unødvendig mener de kan være for eksempel brukerloggen til en bruker fra flere år tilbake i tid. Men en ting de har valgt å ta vare på er ”anbefalte annonser” som er basert på søkeloggen til en bruker i en viss periode, dette tar de vare på for å gjøre tjenesten lettere å bruke. Men de har ikke lov til å ta vare på denne i evig tid.

Et annet problem som prosjektet løste var at alle sidene på FINN måtte ha personvernerklæring lett tilgjengelig. Dette løste de ved å ha en personvernerklæring nederst på alle sidene til FINN.no (se bilde 1).

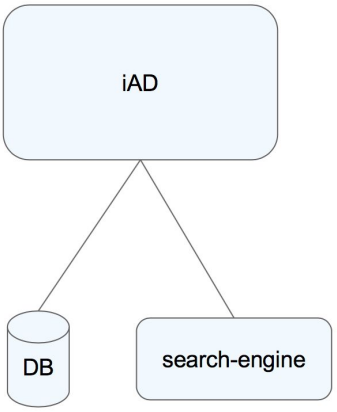
Bilde 1: Screenshot av den nederste delen av nettsiden til FINN.*no med rød markering rundt “Personvernerklæring”.*

FINN har ansatt en person til å jobbe med personvern på heltid. Det er frivillig å ansette en slik person, det er ikke lovpålagt for et selskap å ha dette men det er anbefalt av datatilsynet.

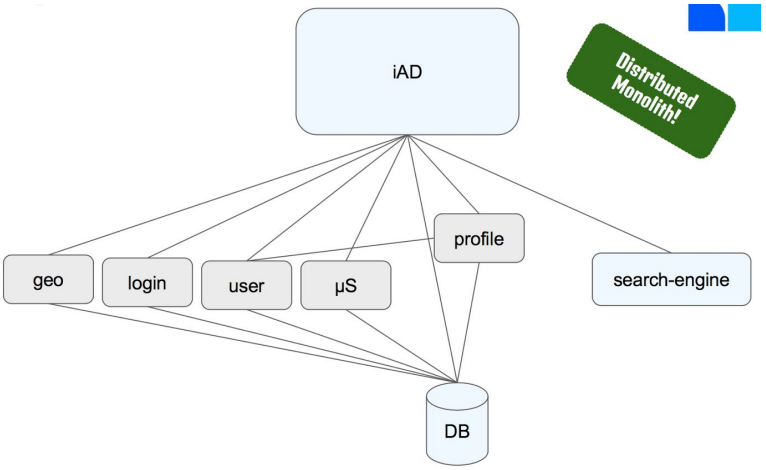
2.5 Utvikling

FINN.no har utviklet seg mye siden den første versjonen av hjemmesiden ble lansert. Det har vært en rask utvikling hvor mange valg har blitt tatt, og hvor disse har blitt sentrale og viktige hendelser for FINN. I begynnelsen hadde de kun fem utviklere som jobbet på ett domene

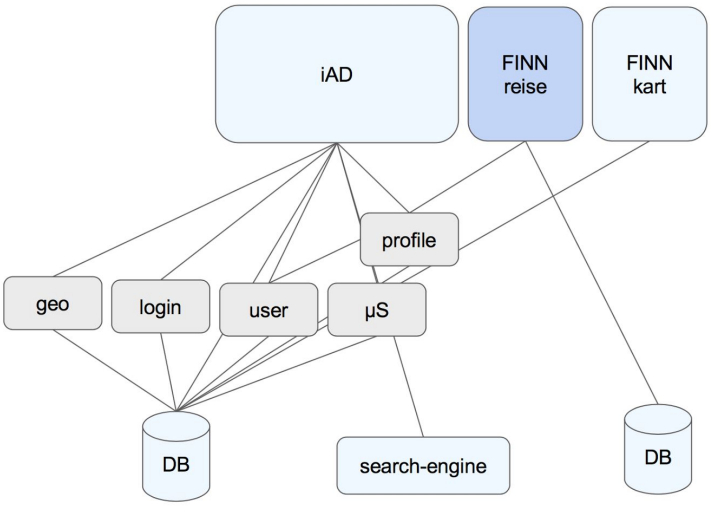
lagret i én database. I dag har de 16 domener, med ca. 120 utviklere og andre ansatte som hele tiden tester nye funksjoner og tjenester, og er opptatt av å gjøre alt mye mer brukervennlig. FINN bruker idag Domenedrevet Design for å håndtere kompleksiteten under utvikling, og det er denne tilnærmingen de håper å kunne bruke videre fremover tid. Det gjør det enklere for utviklerne å finne ut nøyaktig hvor en feil ligger, og det har blitt mye enklere å bruke tjenestene hver for seg, men også gjort hele systemet og utviklingen enda mer kompleks.



Dette avsnittet gir et raskt overblikk over utviklingen til FINN, med enkle modeller for å forstå fremgangen. Utviklingen vil beskrives dypere i kapittel 3.

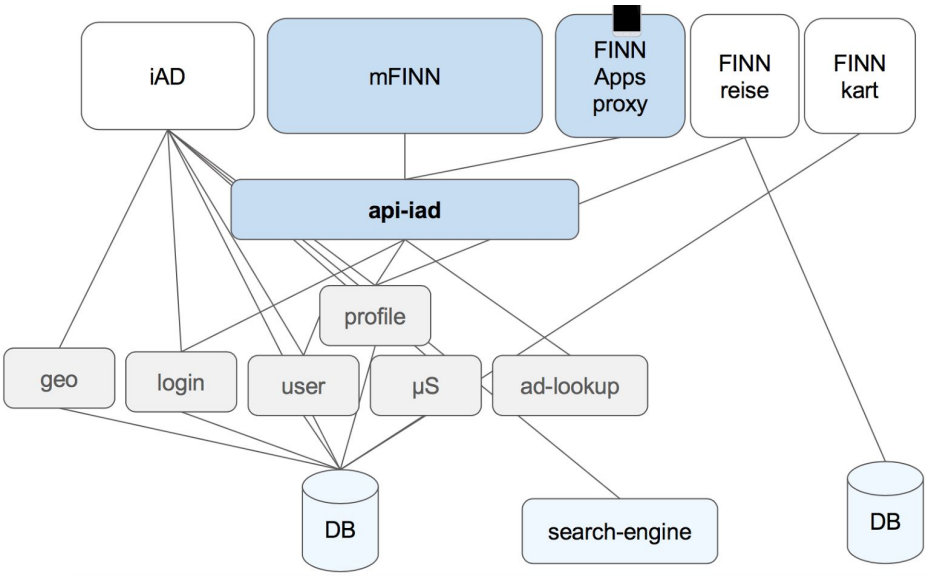


Da FINN.no startet opp i mars 2000 hadde de en monolittisk arkitektur. De hadde en løsning/et domene, en database og fem utviklere som jobbet med dette. Bildet til høyre viser FINN.no sin modell av den monolittiske arkitekturen ("The monolith")



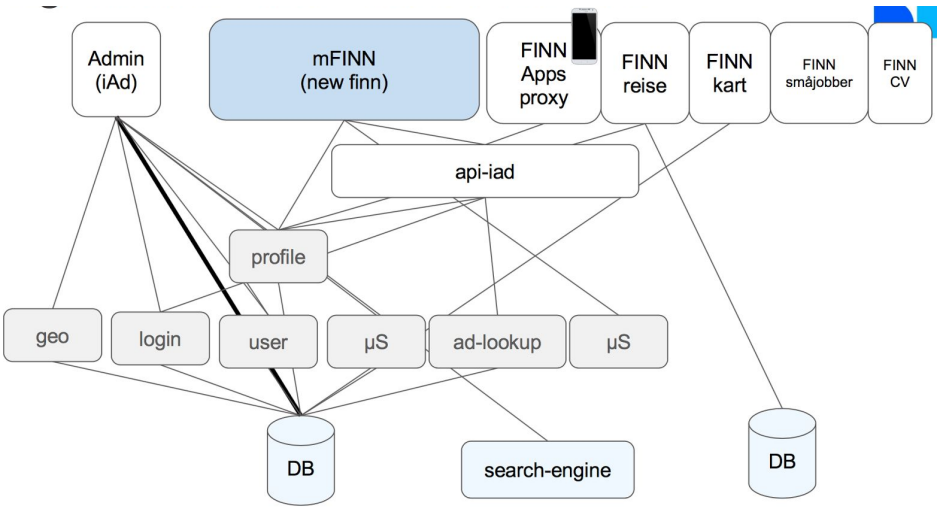
Ikke lenge etterpå fant de ut at de ønsket å bli mer fleksible og bestemte seg derfor for å utvide tjenesten sin med mikrotjenester. De lagde blant annet tjenestene geo (kart), login, bruker og profil. Bildet til høyre viser FINN.no sin modell av endringen ("Distributed Monolith" med mikrotjenester). Etter denne endringen utvidet de igjen med to nye markeder, som var FINN Reise og Kart. Dette var isolert fra de andre, da det ikke passet inn med de andre tjenestene, og fikk sin egen database. Profil ble en løsning som snakket med alle de andre mikrotjenestene. En bruker kan f.eks logge inn, bruke kart og lage profil. Bildet til høyre viser hvordan de nye markedene ble implementert i arkitekturen.

Videre utvidet de til å ha både FINN.no på pc og mobil. De opprettet mFINN og en egen FINN apps proxy. FINN "apps proxy" er en server som står mellom client og hovedserver som begrenser brukerens tilgang til serveren for å øke sikkerheten. De opprettet den for å kunne mappe dataene til applikasjonen.



De opprettet også et eget api-iad slik at de kunne bruk løsninger fra de tjenestene de allerede hadde opprettet, og ble også brukt som public API som vises til partnere. Her begynner FINN å bli mer komplekst.

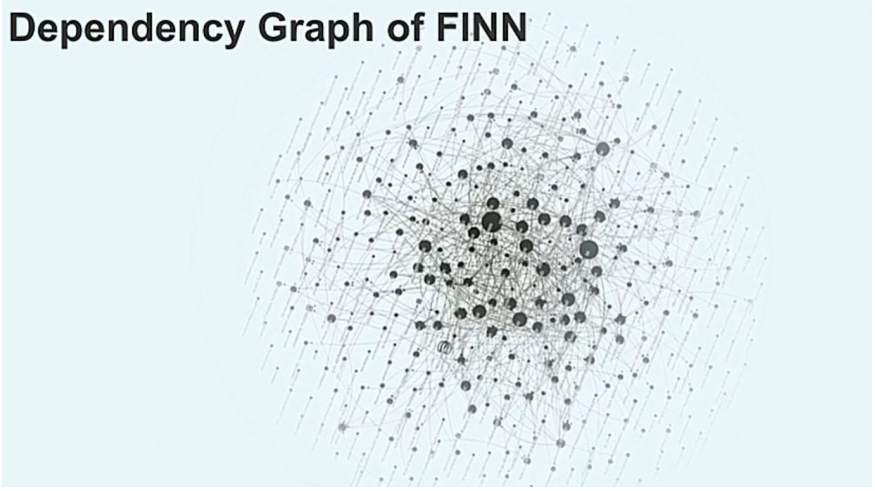
FINN ville ikke ha to forskjellige versjoner, en for pc og en for mobil, så de bestemte seg for å gå helt over på mobil. Da ble finn.no til m.finn.no, og iAd,



desktop-versjonen av FINN, brukes hovedsaklig av admin, som

er administratorpanelet for ansatte hos FINN. Bildet til høyre viser en modell av hvordan arkitekturen så ut etter at de valgte å gå over på mobil (‘Legoland - én versjon’)

Her (bildet til høyre) har FINN brukt et program som lager en graf som viser alle domenene og mikrotjenestene, og hvordan de snakker sammen på FINN inn.no idag. Ut fra denne kan man med en gang se at utviklingen bak nettsiden kan være en ganske kompleks prosess.

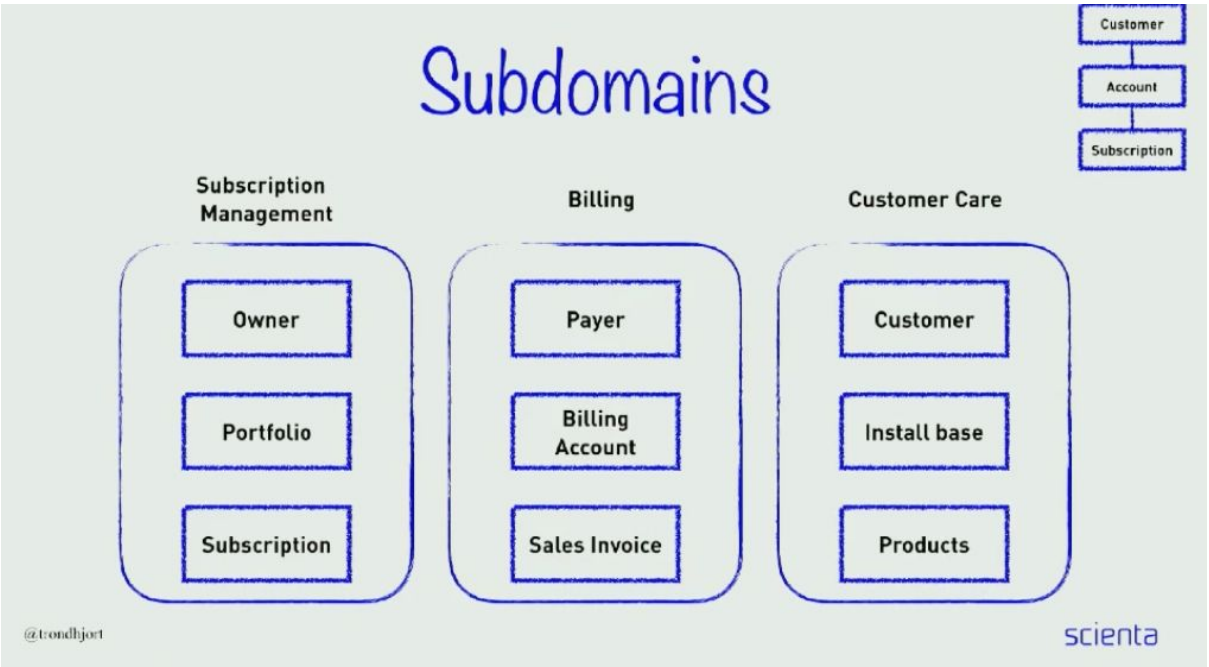


Kapittel 3: Endringsstrategier for informasjonsinfrastrukturen

3.1 Domenedrevet design

For å kunne forstå hvorfor FINN har valgt å gå over til domenedrevet design (DDD), er det viktig å forstå hva det er, hvordan det gjøres og hvorfor DDD er bedre for bedriften. FINN vil bruke DDD sammen med "The Inverse Conway Maneuvre" for å kunne trekke ut deler av legacy koden på en strategisk måte. "The Inverse Conway Maneuvre" er en teknikk som hovedsaklig går ut på at man setter sammen team og endrer organisasjonsstrukturen for å oppnå ønsket arkitektur, og hvor det ideelle er at teknologiarkitekturen speiler organisasjonsarkitekturen.

DDD er en strategi/tilnærming for å håndtere mye av den kompleksiteten man kan ende opp i. Kompleksiteten i FINN har noen syndere som er legacy-databasen, annonsemodellen og organisasjonshierarkiet. Systemutvikler og designer Eric Evans var den første som introduserte begrepet. DDD er ikke en teknologi eller metode, men en tilnærming til programvareutvikling for å løse komplekse behov, hvor man kobler gjennomføringen til en modell i utvikling. Det innebærer flere ting, blant annet at: Bedriftens hovedfokus skal være på kjernedomenet (problemdomenet) og domenelogikk, komplekst design skal baseres på en modell og det skal inngås et samarbeid mellom designere, tekniske eksperter og domeneeksperter for å stadig kunne komme nærmere til selve hjertet av problemet. Kjernedomenet er som regel det som er spesielt og eget for akkurat den bedriften det gjelder, det er unikt. Når et samarbeid mellom ekspertene inngås, er det viktig at alle har en felles forståelse om hva bedriften driver med, hva som går under et domene og at de har et felles språk. Felles språk kan for eksempel være felles programmeringsspråk og at alle forstår hva "Bruker" betyr for akkurat den bedriften. Når alle har en felles forståelse, kan de videre dele domenet inn i sub-domener, som på en måte blir klassene eller komponentene som inngår i domenet. En "Bruker" kan for eksempel ha sub-domener som profil, kommunikasjon, venneliste osv. Domenet brytes ned bit for bit, for å komme så nærme problemet og løsningen som mulig.

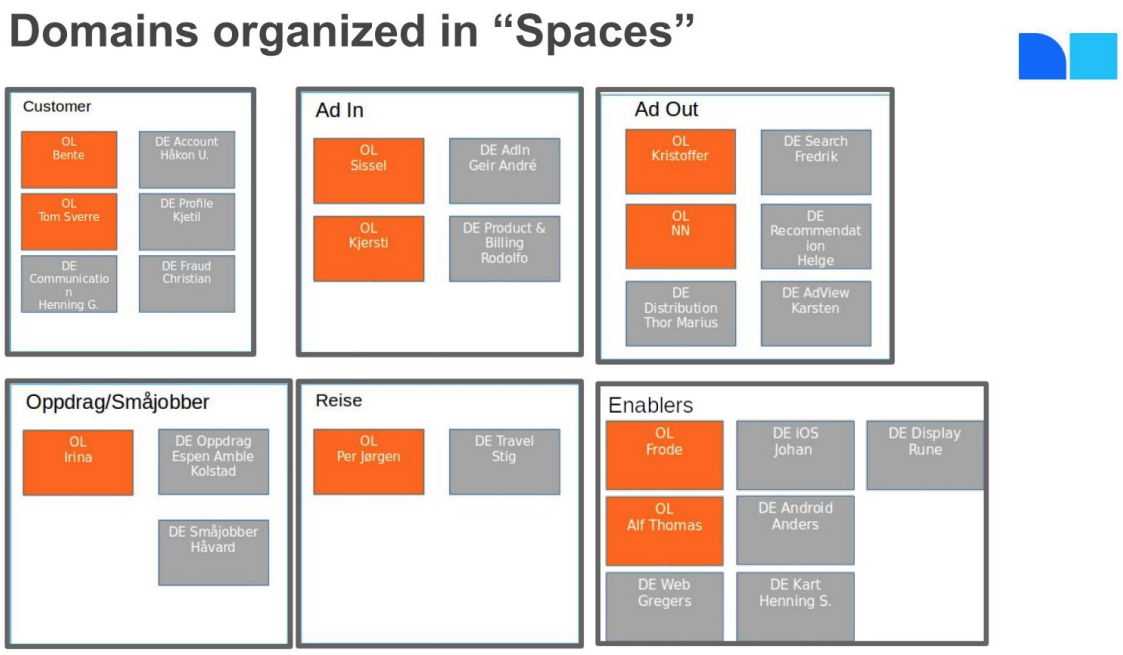


Eksempel på inndeling i subdomener (Trond Hjorteland, Javazone, Scienta)

I FINN har de 16 domener, hvor domenene deles inn i områder. Et område kan for eksempel være "Kunde". FINN oppdager kanskje at det er noe feil med meldingstjenesten deres, og at noen meldinger ikke kommer frem. Hovedfokuset rettes da mot kjernedomenet, som her blir "Kunde". FINN har egne faste ansatte tekniske eksperter og domeneeksperter som allerede kjenner til bedriften og domenene godt. Begge har derfor et felles språk og en felles forståelse av domenet og hva som inngår i kjernedomenet. De lager en modell hvor de lister opp alle sub-domener innenfor kjernedomenet "Kunde", som blant annet bruker, profil, kommunikasjon og svindel. Ekspertene vet at det er et problem med meldingstjenesten og skjønner at det må ligge under "kommunikasjon". Videre finner de det nøyaktige problemet og designer en bedre løsning. Problemer som ligger i legacy-databasen til FINN, som annonsemodellen og organisasjonshierarkiet, er de problemene FINN anser som vanskeligst å løse. Hvis en ny løsning skal designes for et problem i legacy-databasen, lønner det seg å lage den utenfor denne.

FINN har valgt domenedrevet design for å håndtere og løse problemer som fører til mer kompleksitet. For FINN er dette en bedre tilnærming enn den tidligere fordi det blant annet gir de mer effektivitet ved at utviklerne har et bestemt område de driver ekspertise, og

dermed er det enklere å gjøre relative små endringer ofte, fordi det tar mye tid å sette seg inn i hele FINN sin løsning, grunnet dens kompleksitet.



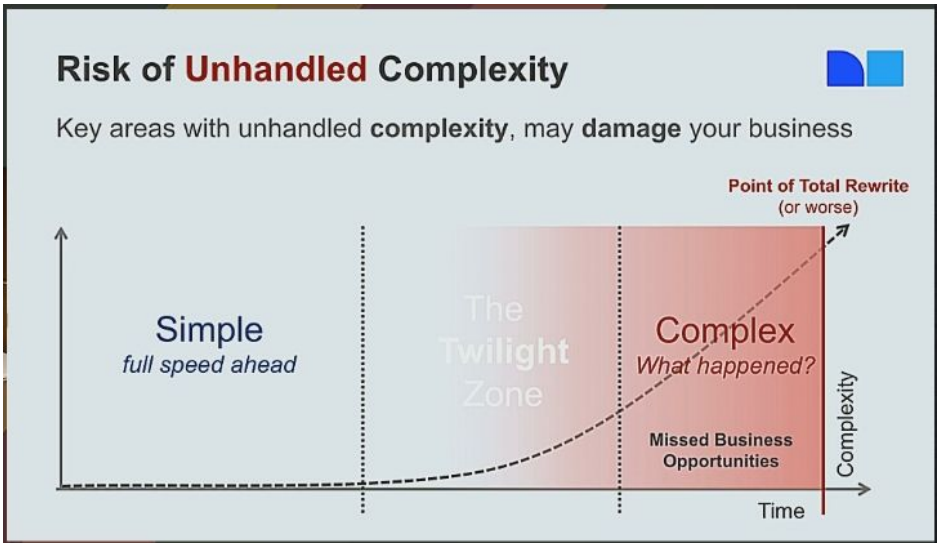
Eksempel på noen av FINNs domener organisert i områder

Med tanke på hva som inngår i begrepet kompleksitet, kan DDD sammen med The Inverse Conway Maneuvre være en løsning på flere punkter i definisjonen av Hanseth og Lyytinen, som sier at kompleksitet er “*et resultat av et stort antall sosio-tekniske komponenter, disse komponentenes heterogenitet, relasjonene mellom dem og deres dynamiske uforutsigbare interaksjon”*. Når FINN tar i bruk DDD-tilnærmingen tar de inn sosio-tekniske komponenter som en del av tilnærmingen, hvor sosio-delen er teamet som blir satt sammen, med designere, tekniske- og domeneeksperter. Den tekniske-delen består av domener og en heterogen brukergruppe som vedlikeholder de forskjellige domenene.

3.2 Kompleksitet

Vi har også her valgt å se på kompleksitet i FINN i forhold til Hanseth og Lyytinens definisjon av kompleksitet, som defineres i avsnittet over.

Kompleksitet er noe som kommer, men det er ikke alltid planlagt. Så den bedriften som takler kompleksitet på en bra måte har et fortrinn. Det er fremdeles noen utfordringer knyttet til arkitekturen til FINN, som alltid vil det være der fordi bedriften stadig vil være i utvikling og som oftest ønske å ekspandere. Større installert base, flere mikrotjenester og flere tredjeparter er noen av utfordringene arkitekturen til FINN må regne med å takle i fremtiden. FINN har også en relativt stor installert base som også gjør det vanskelig å gjøre store endringer i systemet.



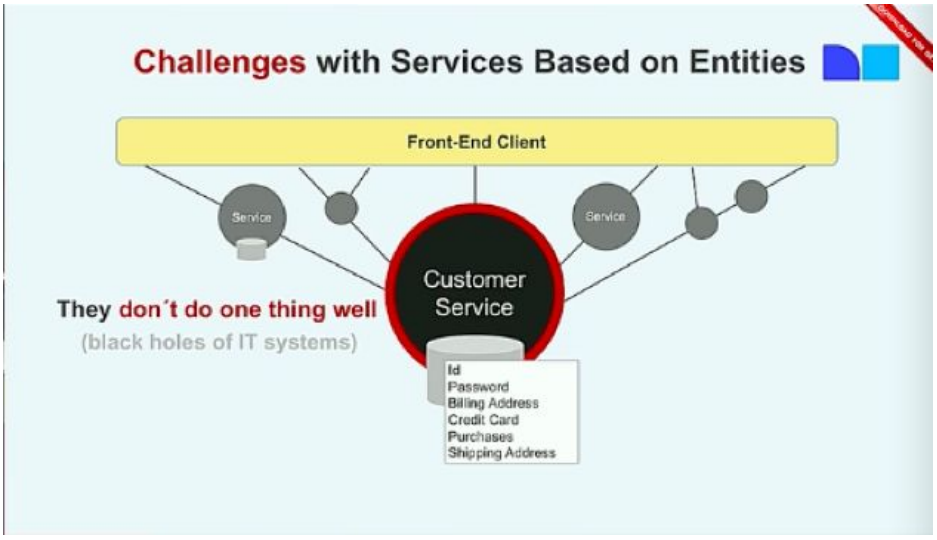
Eksempel på hvordan kompleksitet oppstår og er uventet

FINN har valgt å være strategiske når det kommer til kompleksitet, hvordan de kan være effektive og unngå store omskrivninger av systemer samt opprettholde informasjonsinfrastrukturen. Det finnes som oftest et system som allerede er der, det er en legacy og allerede her starter kompleksiteten for mange. Så deler av strategien har vært å lære seg og leve med de gamle systemene som ikke kan byttes ut. Selv om det optimale er å lage alt fra bunnen av er det veldig tidskrevende, omfattende og ikke minst veldig dyrt. Samt så er behovene og kravene stadig i endring, og da er systemene også nødt til å være i endring hele tiden. Som tidligere nevnt ruller FINN ut til produksjon omtrent 150 ganger om dagen for å imøtekomme behovene. Det var noen få sentrale strukturer som skapte høy kompleksitet, annonsemodellen og organisasjonshiearkiet var nevnt som de største synderne. Som tidligere nevnt FINN har valgt mikrotjenester som er veldig godt integrerte og sterkt knyttet til

tjenestene, men det er fremdeles vanskelig å gjøre store endringer. Dette diskuterer vi lengre ned.

3.3 Utfordringer ved arkitekturen til FINN

Selv om FINN har gått over fra monolittisk arkitektur til mikrotjenester har de fremdeles utfordringer knyttet til arkitekturen. Monolittisk arkitektur er et enkelt system som ikke samhandler med andre systemer, men de kan ofte være veldige gode til det systemet er laget for (Verheughe, S., 2016). Som nevnt tidligere så har FINN utfordringer med noen strukturer, annonsemodellen og organisasjonshierarkiet var de store synderne, disse to er sentrale strukturer med høy kompleksitet i den form av at de består av mange sosio-tekniske komponenter som skal samhandle over tid. Samt at endringer i legacy er veldig vanskelige og vil påvirke andre systemer og kan skape utilsiktet eller uønsket atferd. De har også utfordringer med høy kompleksitet i mikrotjenestene der det ble veldig lange kjeder med hopp og disse lange kall-kjedene skaper noen problemer når det kommer til ytelse og oppetid (uptime). En tredje utfordring er at selvom FINN nå har innført en tjenestebasert arkitektur så sitter de igjen med problemet/spørsmålet; *hva skal de gjøre med front-end?* Front-end er stedet hvor alt aggregeres og de hadde derfor problemer med å unngå en monolittisk arkitektur akkurat her. For å løse dette problemet valgte FINN å bruke en partisjoneringsstrategi. De prøvde først å se hva som skjedde hvis de delte opp entitetene:



Viser oppdeling av entititer Front-End i kundemodellen.

En av utfordringene som oppstod med denne delingen var at en entitet ikke gjorde en ting godt nok. Tjenester som ligger i entiteter støtter vanligvis flere forskjellige problemer, og de blir derfor fort komplekse. Det vil oppstå mange avhengigheter på tvers av hele systemet, noe som gjør endringer veldig vanskelig. Man vil også se at selv når man prøver å løse nye forretningsutfordringer så vil gjenbruk av disse entitetene føre til at disse kommer til å vokse mer og mer og bli ganske monolittiske over tid. FINN fant det vanskelig å få svar på hva denne tjenesten gjorde godt, så de valgte heller å dele opp i forskjellige modeller for de forskjellige forretningsbehovene som FINN ønsker å løse.

Viser kundetjenesten splittet opp i forskjellige kundemodeller for de forskjellige forretningsbehovene.

FINN mappet tjenestene sine til forretningsmål. Det de fant ut av å gjøre dette var at de fikk mindre koblinger og at de fikk tjenester som gjorde en ting godt. Det løste forretningsbehovene. De fikk flere tjenester som var mer spesialiserte, og det var dette de først ønsket. Videre ønsket de at tjenestene i FINN skulle speile de forretningbehovene eller domenene som de prøvde å løse, ikke entitetene som er involvert i disse forretningsbehovene. Det er derfor FINN nå har valgt følgende arkitekturretning; logiske grenser med domener.

3.4 Endring av bedriftsbehov

Etterhvert som arkitekturen til FINN endret seg ble også en stadig mer kompleks forretningslogikk innført. Det ble et økt voksende antall områder å ha kjennskap til og ønskene på det tekniske laget samsvarte som oftest ikke med det forretningsmessige laget. FINN så seg derfor nødt til å endre eierskapsmodellen de tidligere hadde benyttet. Målet var å oppnå at en utvikler ikke bare hadde eierskap til det tekniske ved et problem/mål men også kjente til den forretningsmessige logikken bak et eventuelt valg. Ved en slik løsning unngår en at utvikleren blir for opphengt i det tekniske laget men at utviklingen heller sentreres rundt bedriftsbehovene. En annen gunstig effekt er også at utvikleren tar ansvar for applikasjonen både i utviklingsløpet men også etter produksjonssetting.

FINN er med bakgrunn i dette derfor delt opp i domener der domenene representerer den forretningsmessige logikken ved FINN. Eksempelvis å sende en melding, kjøpe en annonse m.m. Alle domenene sorterer så igjen under områder. En utvikler er en del av et domene, men også ett område. Ett område er forskjellige paraplyer som domenene sorterer under.

Det FINN ønsket å unngå var nettopp hva "Conways law" uttrykker. "*organizations which design systems (...) are constrained to produce designs which are copies of the communication structures of these organizations*" (Conway, Melvin E. April 1968). Oversatt går det ut på at løsningene en produserer speiler seg i organisasjonsstrukturen og grensene i ett selskap. FINN har snudd på det og ønsker at være vare på å se både sammenheng mellom arkitektur og organisasjon samtidig med ønske om å basere seg rundt de forretningsmessige logikkene.

Ønsket er at dersom FINN organiserer seg på en viss måte vil teknologien og arkitekturen følge etter. Når det kommer til spørsmålet om hvilke forretningsmessige mål en skal arbeide mot har FINN for å løse dette introdusert en annen interessant modell. Det samles en oversikt over de forskjellige målene selskapet ønsker å oppnå, hvor de forskjellige avdelingene får utdelt et gitt antall lekepenger/sjetonger som alle representerer en gitt mengde arbeidstimer.

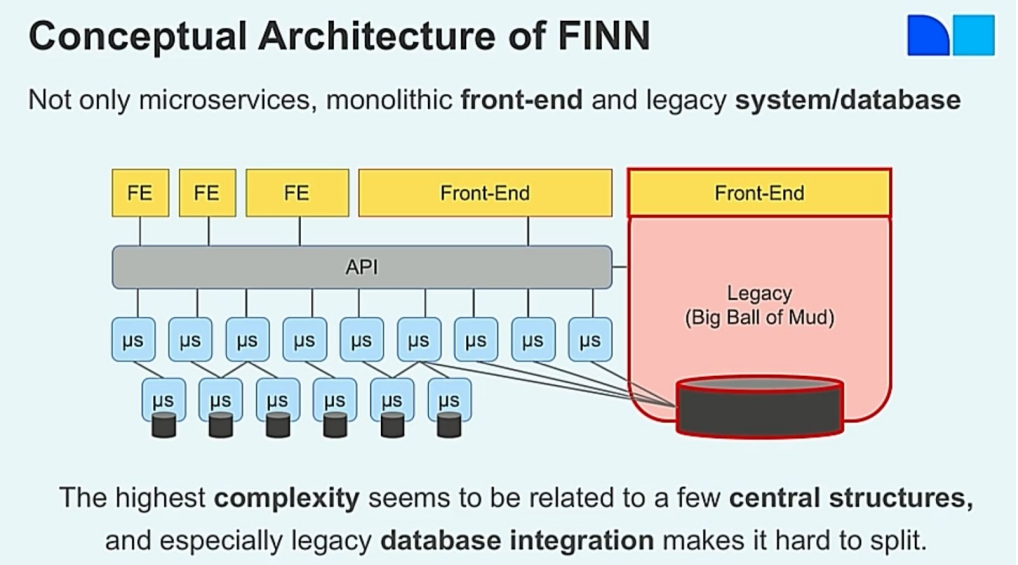
Det blir dermed opp til avdelingene å putte "penger" på de målene som det fra deres synspunkt anses som mest viktig å prioritere.

Videre sitter hvert team igjen med tildelte konkrete mål for sitt område/domene som de skal tilnærme seg til. Teamene eier prosessen og tar selv de nødvendige beslutninger underveis for å oppnå dette - uten at det må gå på lange runddanser høyere opp i organisasjonen.

Når det kommer til de konkrete endringene som gjennomføres har FINN erfart at det er mye mer fordelaktig å ta beslutninger rundt eksisterende systemer fremfor fremtidige løsninger som enda ikke finnes. Ved å heller ta beslutninger på et nivå en kan og innføre endringer gradvis vil en hele tiden kunne arbeide for å levere "Fastest minimum viable capability" som altså tilsvarer det minste/raskeste steget som må til for å løse/levere forretningen. Ikke bare hoppe på en ny stor endring, men heller arbeide for å forsøke å forstå hva en egentlig prøver å oppnå.

Med denne arbeidsmetoden vil en hele tiden arbeide for å løse akkurat den nødvendige konkrete endringen som forretningslogikken krever. Samtidig vil en da over tid forbedre hele systemet på de områdene der det faktisk har en mening med en endring tatt businessen i betraktning.

3.5 FINN nå



FINN er nå godt på vei til å drive en domenedrevet struktur med mikrotjenester og API som kobler disse sammen på tvers, men det er fremdeles en del legacy databasestrukturer og front-end som må konverteres og tilpasses før de er helt i mål. Den nye informasjonsstrukturen fungerer via et API (Application programming interface) som sender forespørsler til riktig mikrotjeneste, samt henvendelser tilbake. Et API kan forklares som et kontaktpunkt hvor man autentiserer seg, og dermed får tilgang til forskjellige deler av systemet som man kan bruke på tvers av APIet. Slik unngår man silobaserte strukturer som krever en egen løsning per annen løsning den ønsker å snakke med. APIet blir som en terminal som ruter trafikken til riktig sted, og ikke minst på tvers av mikrotjenester - slik at det ikke blir et nytt silosystem.

FINN sin API-struktur og løsning kan sammenlignes med den tekniske infrastrukturen til flyselskapet Norwegian, som også er bygget under samme prinsipper. En slik løsning gir fordelene ved at man hurtig ekspandere tjenestene til FINN, men samtidig en økt risiko for at en mer teknisk kompleksitet kan true fleksibiliteten. Samt at utviklere som jobber med elementer har en dypere forståelse for en slik infrastruktur. (Bygstad, B., Aanby, H., 2009)

Bruken av denne infrastrukturen har også sine utfordringer. Det er veldig viktig å at alle utviklere jobber med APIet, og ikke frister seg selv til å jobbe rundt denne f.eks. for å spare tid eller kostnader. Et slikt valg kan på sikt lage semi-silobaserte programmer som vil raskt øke i kompleksitet og skape kaos i strukturen. Dette er en utfordring enhver bedrift bør adressere og gjøre tiltak mot at snarveier ikke blir tatt.

Kapittel 4: Oppsummering og begreper

4.1 Kompleksitet

Definisjon

I følge Hanseth & Lyytinen (2010) er kompleksitet et resultat av et stort antall sosio-tekniske komponenter, disse komponentenes heterogenitet, relasjonene mellom dem og deres dynamiske og uforutsigbare interaksjon. Store og komplekse informasjonssystemer består av mange forskjellige komponenter som henger sammen. Sosio-teknisk betyr at det er forskjellige sosiale og tekniske aspekter som henger sammen og former og påvirker hverandre gjensidig. Sosiale aspekter er at det er avhengigheter og sammenhenger mellom ulike personer og arbeidsoppgaver/prosesser i organisasjonen. Tekniske aspekter er at det er mange, ulike og sammenkoblede informasjonssystemer.

Eksempel fra prosjektoppgaven

FINN har utfordringer med kompleksitet i systemarkitekturen, grunnet at de har flere domener, områder og løsninger som skal snakke sammen sosio-teknisk. De har en del kompleksitet som ligger igjen som legacy database, som noen utvalgte tjenester bruker, men dette er planlagt å fases ut på sikt gjennom en evolverende prosess. Det er også utfordringer i annonsemodellen og organisasjonshierarkiet. Endringene fra monolittisk system til et komplekst informasjonssystem fører til stadig nye brukerbehov fører til stadig endringer og utfordringer. FINN har valgt domenedrevet design og mikrotjenester som sin tilnærming som en løsning på kompleksitet.

Eksempel fra kurslitteraturen

Et godt eksempel på kompleks informasjoninfrastruktur er Rolland og Monteiros artikkel om en maritim klassifiseringsbedrift (MCC) “Balancing the local and the global in infrastructural information systems” Denne bedriften utførte undersøkelser hver gang et skip la til. Det etablerte systemet bestod av 74 ulike papirbaserte sjekklister for de ulike undersøkelsene. Sjekklistene var skreddersydd i henhold til ulike sammenhenger og miljøer. Så dermed var det ingen standard representasjon eller felles bruk av terminologi. Dette skapte irrelevante problemstillinger, merarbeid, mye tilleggsinformasjon og usynlige arbeidsoppgaver. Dette

skapte kompleksitet i høy grad og da må det endringer til for å takle kompleksiteten. (Rolland & Monteiro, 2002).

4.2 Delt

Definisjon

Hanseth & Lyytinen's (2010) definisjon av delt er *"Shared by an increasingly growing number of heterogeneous user communities, designers, regulators and other social actors"*.
Med *delt* menes det at hvem som helst kan bruke og hvem som helst kan videreutvikle, men dette med begrensninger. Det at informasjonsinfrastrukturen er delt har noen konsekvenser som f.eks at endringer blir veldig vanskelig siden det påvirker alle og at de ikke kan splittes opp i mindre deler. Et veldig godt eksempel som ofte blir nevnt er internettet, som kan sammenlignes med en veldig stor og kompleks informasjonsinfrastruktur.

Eksempel fra prosjektoppgaven

FINN sitt system er delt, men med begrensninger som i de aller fleste store og komplekse informasjoninfrastrukturer. De er opptatt av å dele koden og strategiene sine, og velger å legge ut mye på GitHub, Javazone og Slideshares. Andre kan bruke koden deres til eget bruk (open source), men det er kun en definert utviklergruppe innenfor FINN som kan endre på FINNs nettside. FINN deler også noen tjenester og funksjoner med andre som f.eks Proffcom som tar imot kundehenvendelser utenfor åpningstidene til FINN, hvor det er en personlig innlogging til alle systemene de deler. FINN har også noen funksjoner som er mer delte enn andre, noen funksjoner tillater f.eks alle brukere å legge ut annonser og kontakte personer eller bedrifter.

Eksempel fra kurslitteraturen

I artikkelen *"Designing work oriented infrastructures"* skrevet av Hanseth og Lundberg (2001) analyserer og identifiserer de noen problemer knyttet til implementeringen av PACS og RIS. Her diskuteres det hvordan disse to systemene er delte. Denne artikkelen er basert på hypotesen om at den høye frekvensen av feiling blant prosjektene som sikter på implementeringen av PACS-systemet i radiologiavdelinger er på grunn av variasjonen og kompleksiteten rundt arbeidspraksisen på sykehusene, og avhengigsforholdene mellom

teknologier som støtter arbeidspraksisene. Når de ønsket å tilnærme seg informasjonsinfrastrukturen så fokuserte de på 4 aspekter hvor en av dem var delt.

4.3 Åpen

Definisjon

Hanseth & Lyytinen (2010) definerer begrepet *åpen* som en egenskap et system kan ha som gjør det mulig at nye komponenter kan legges til og integreres med hverandre på uventede måter og i ulike kontekster. I tillegg så er det heller ingen klare grenser mellom de som bruker informasjonsinfrastrukturen og de som ikke gjør det, samt om de er i stand til å designe IIet. Man bør derfor ikke anta begrensninger i form, innhold, kapabilitet eller omfang.

Begrepet "åpen" henger tett sammen med begrepet "delt", definisjonene for begrepene er ganske like. Åpenhet er en forutsetning for at store komplekse informasjonssystemer blir delt, og det er gjennom denne delingen at de fremstår som åpne. At et system er åpent vil si at det er en stor variasjon av brukere som kan styre utviklingen og bruken av systemet, med andre ord så er det et system med få begrensninger.

Eksempel fra prosjektoppgaven

Åpenhet rangeres etter ulike grader, det er sjeldent et system enten er helt åpent eller helt lukket. FINN sine systemer er til en viss grad åpne. De er åpne i den form av at koden til FINN kan brukes av alle, den blir offentliggjort, men det er kun utviklerne i FINN som kan gjøre endringer i koden og legge til funksjonaliteter på FINN.no, den er derfor åpen men med begrensninger.

Eksempel fra kurslitteraturen

Hanseth & Lundberg (2001) viser til et eksempel om åpenhet i en artikkel som beskriver utfordringene ved å implementere Picture Archiving and Communication Systems (PACS) and Radiological Information Systems (RIS) som vist til tidligere i avsnittet om begrepet "delt". Hanseth & Lundberg (2001) forklarer at den radiologiske informasjonsinfrastrukturen er en del av en stor og åpen infrastruktur for hele sykehuset, samt at de har en felles

infrastruktur for kommunikasjon mellom alle Healthcare enheter. Det vil derfor være en grad av åpenhet i systemet PACS som skal brukes av ansatte ved alle Healthcare enheter, hvem som helst kan ikke bruke systemet men det er en stor variasjon av brukere som kan det.

4.4 Heterogen

Definisjon

En *heterogen* informasjonsinfrastruktur påvirkes av åpenhet, ved at den har flere forskjellige aktører og komponenter med forskjellige egenskaper. Graden av heterogenitet øker etter hvert som flere typer teknologiske komponenter inkluderes, men først og fremst komponenter av forskjellig natur, som f.eks brukere, operatører, standardiseringsorganer, lover og standarder osv. I en heterogen II er standarder og fleksibilitet viktig for å få komponentene til å snakke sammen (Hanseth & Lyytinen, 2010). Aktørene kan også bli mer forskjellige over tid, som følge av hvordan de blir inkludert og hvordan systemer brukes.

Eksempel fra prosjektoppgaven

Selve nettsiden FINN.no har mange forskjellige aktører og brukergrupper som inkluderes i informasjonsinfrastrukturen på forskjellige måter, både på front-end og back-end/ "backstage". Brukere av nettsiden på front-end er de som kun bruker den til kjøp og salg og de ansatte i FINN som bruker Admin (administratorpanelet). De som bruker Admin, bruker også andre teknologiske verktøy, som f.eks Zendesk og telefon. På back-end jobber blant annet utviklerne med koden som ligger bak hjemmesiden med egne utviklerverktøy. Heterogeniteten til Finns II øker ved at alle disse aktørene er en del av den samme II, hvor forskjellige teknologiske komponenter tas i bruk av disse. Jo mer heterogent, jo mer komplekst kan det også bli.

Eksempel fra kurslitteraturen

I artikkelen av Hanseth og Lundberg (2001) brukes begrepet heterogentitet om Internett som er delt inn i flere sub-infrastrukturer, som for eksempel det globale IP nettverket, e-post, nyheter og andre infrastrukturer på nett. Disse kan ses på som individuelle infrastrukturer, men er også eksempler på type infrastrukturer som ofte integreres i andre informasjonsinfrastrukturer, noe som gjør de heterogene. De er også heterogene fordi de

består av forskjellige komponenter, både menneskelige aktører og teknologi. Det er for eksempel et stort antall supportpersonell bak Internett og en e-posttjeneste. Uten disse ville det ikke fungert slik som det gjør. En sub-infrastruktur fremstår her som en delt kilde mellom heterogene brukergrupper.

4.5 Evolverende

Definisjon

Evolverende er et begrep som er definert i kurset slik: "*(...) the evolution of 'infrastructure' is fixed in modular increments, not all at once or globally*" (Hanseth & Lyytinen, 2010). Et eksempel på evolusjon er for store komplekse systemer, hvor man kan ikke gjøre radikale endringer. Blant annet fordi slike systemer har ofte en installert base som må også endres inkrementelt. Dermed må de aller fleste systemer endres inkrementelt over tid, og endringene må forbli relevante og endre seg etter brukerbehovet. Det er helst ønsket en kultivering av hva som eksisterer, enn en kontrollert endring.

Eksempel fra prosjektoppgaven

Da FINN så at trafikken til deres nettsider på mobil ble veldig stor, og ønsket en og samme nettside å forholde seg til, flyttet de desktopversjonen av nettsiden til mobilsiden, som da er skalert responsivt, slik at siden passer på alle enheter. Dette henger sammen med FINN sin nye mikrotjeneste-struktur som er diskutert tidligere.

FINN er stadig under utvikling og de publiserer 150-200 små og store endringer på nettsiden deres per dag. De jobber etter arbeidsmetoden "Fastest minimum viable capacity", som tilsvarer et minste eller raskeste steget som må til for å løse og levere forretningen. FINN er veldig opptatt av brukervennlighet og mye av endringene går på dette punktet, de evolverer løsningene sine gradvis og gjerne tester det på et utvalg av kunder før de lanseres for hele basen.

Eksempel fra kurslitteraturen

Et eksempel på en evolverende informasjonsstruktur er flyselskapet Norwegian sin API arkitektur som la opp til inkrementell utvikling av de forskjellige tjenestene, veldig lignende

FINN sin mikrotjenestestruktur og domenedrevet design. En slik løsning gir større rom for inkrementell endring av tjenestene og ikke minst nye tjenester kan oppstå uten å direkte påvirke de eksisterende. (Bygstad & Aanby, 2009)

4.6 Installert base

Definisjon og beskrivelse av begrepet "Installert base"

Hanseth & Lyytinen (2010) definerer installert base som *"... a set of IT capabilities and their user operations and design communities"*.

Begrepet *installert base* omhandler de systemene som alt finnes i en organisasjon. Systemene er knyttet opp med brukere og ulike innarbeide bruksmønstre. I arbeidet med systemutvikling er dette en viktig faktor som må tas med i våre vurderinger og arbeidspraksis. Når man arbeider med en eksisterende organisasjon kan man aldri starte helt fra scratch, det er behov, systemer, bruksmønstre m.m som må tas hensyn til.

Disse bruksmønstrene og systemene kan være både ulogiske og dårlige men man må fortsatt ta hensyn til de noe som også former våre valg i nye utviklingsprosesser da både gammelt og nytt må kunne snakke sammen.

Eksempel fra prosjektoppgaven

I FINN sitt tilfelle møter de i stor grad dette problemet i form av at de alt har et stort installert system som må tas hensyn til i utviklingen deres. Denne basen inneholder alt størsteparten av FINN sitt businessområde og har eksistert siden oppstarten deres. Dette har bygget opp både en rekke avhengigheter i tillegg til at systemet har fått en såpass innviklet logikk at det derfor ikke bare uten videre erstattes. FINN må i stedet hele tiden arbeide for å bygge seg rundt og oppå denne basen hvorpå de sakte men gradvis arbeider seg over i en ny base ved å erstatte mindre komponenter av gangen. Dette er en både tidkrevende og komplisert prosess, men er en naturlig del av systemutviklingsprosessen.

Eksempel fra kurslitteraturen

Hanseth & Lundberg (2001) viser til et eksempel om utfordringene ved å jobbe med en installert base i en artikkel som beskriver utfordringene ved å implementere Picture Archiving and Communication Systems (PACS) and Radiological Information Systems (RIS) som vist til tidligere i avsnittene om begrepet "delt" og "åpen". I dette tilfellet nevnes utfordringen med analoge systemer benyttet av radiologene for å analysere og ta røntgenbilder samtidig som behovet for å knytte seg opp mot nye digitale pasientbehandlingsverktøy.

Avslutning

Vi har underveis i oppgaven fått helt nye øyne på hva som angår kompleksiteten bak informasjonsinfrastruktur i praksis. Spesielt de innblikkene FINN har gitt oss i sin utviklingsprosess over tid har vært med til å belyse dette. FINN har dessuten overrasket oss med det som utenfra ser ut til å være en meget smart måte å organisere utviklingsteamene sine på, nettopp for å hanskes med denne kompleksiteten.

Intervjuet med kundebehandleren og systemarkitekten fra FINN var meget interessante opplevelser for oss. Vi fikk et nyttig innblikk i hvordan FINNs mange avdelinger arbeider sammen for å levere det som egentlig er en stor portal med et bredt utvalg av tjenester som strekker seg fra flybilletter til salg av hus og godt brukte sofaer. Vi lærte mye om kompleksiteten i systemene og hvor stort FINN egentlig er. De har mange systemer, som i stor grad er heterogene, noe som gjør det enda litt mer komplekst. På kundeservice er det eksempelvis lett å la seg imponere av hvordan det kun er en avdeling som tar seg av henvendelser for alle disse tjenestene. FINN har rukket å bli et av landets mest brukte nettsider og møteplass for salg og kjøp av brukte og nye varer. FINN har også mottatt priser for å være den beste arbeidsplassen og ha de mest fornøyde kundene, dette sier mye om hvordan FINN har klart å ta vare på sine ansatte og kunder. Informasjoninfrastrukturen i bedriften kan være en av grunnene til suksessen. Åpenheten om infrastrukturen, koden og strategier samt inspirasjon fra andre informasjonsinfrastrukturer som f.eks Spotify og Norwegian sin har også vært noen av grunnene til suksessen.

Referanser

Bygstad, B., Aanby, H. (2009) *ICT infrastructure for innovation: A case study of the enterprise service bus approach* [Internett]. Tilgjengelig fra: <https://www.uio.no/studier/emner/matnat/ifi/INF3290/h16/artikler/bygstadaanby2009.pdf> [Lest 24. oktober].

Direktoratet for forvaltning og IKT (u.å.) Kva er universell utforming? [Internett]. Tilgjengelig fra: < https://uu.difi.no/kva-er-universell-utforming > [Lest 12. oktober].

Domain-Driven Design Community. (2007) *What is Domain-Driven Design?* [Nettside]
Tilgjengelig fra: <http://dddcommunity.org/learning-ddd/what_is_ddd/>
[Lest 25.oktober]

Conway, Melvin E. (1968) "How do Committees Invent?", *[Internett]*. Tilgjengelig fra:
< http://www.melconway.com/Home/Committees_Paper.html >
[Lest 25.oktober]

Evans, Eric. (2003) *Domain-Driven Design: Tackling Complexity in the Heart of the Software* [E-bok/pdf]
Tilgjengelig fra: <http://dddcommunity.org/wp-content/uploads/files/books/evans_pt01.pdf > [Lest 20.oktober]

Finn.no AS (u.å.) Litt Fakta [Internett]. Tilgjengelig fra: <https://hjemmehos.finn.no/no/om_oss/litt_fakta/ >
[Lest 26. september].

Hanseth, O and Lyytinen, K. (2010). *Design theory for dynamic complexity in information infrastructures: The case of building Internet. Journal of Information Technology,* [Internett]. Tilgjengelig fra:
<https://www.uio.no/studier/emner/matnat/ifi/INF3290/h16/artikler/hansethlyytinen2010.pd>
[Lest 25.oktober]

Hanseth & Lundberg. (2001) "*Designing Work Oriented Infrastructures"*
<https://www.uio.no/studier/emner/matnat/ifi/INF3290/h16/artikler/hansethlundberg2001.pdf> [Lest 5. november]

Hjorteland, T. (2016) *Mikrotjenester uten domenedrevet design er risikosport* [Internettvideo]. Tilgjengelig fra:
<https://2016.javazone.no/program/mikrotjenester-uten-domenedrevet-design-er-risikosport> [Lest 24. oktober].

Rolland & Monteiro. (2002) "*Balancing the Local and the Global in Infrastructural Information Systems"*

<https://www.uio.no/studier/emner/matnat/ifi/INF3290/h16/artikler/rollandmonteiro2002.pdf> [Lest 1. november].

Schibsted Media Group (u.å.) Land [Internett]. Tilgjengelig fra: <http://www.schibsted.com/no/Om-Schibsted/Land/> [Lest 26. september].

ThoughtWorks. (2015) *The Inverse Conway Maneuver* [Nettside]. Tilgjengelig fra: <https://www.thoughtworks.com/radar/techniques/inverse-conway-maneuver> [Lest 25.oktober].

Verheughe, Sebastian. (2016) S*trategic design by Architecture and Organisation* [Internettvideo]. Tilgjengelig fra: <https://2016.javazone.no/program/strategic-design-by-architecture-and-organisation-finn-n> [Sett 24. oktober].