

Ut med desktop, inn med webtop

Dette er den andre av to artikler som introduserer konseptet *webtop*, det nye årtusenets erstatning for vår tradisjonelle *desktop*. Første artikkel sto i Mellvik-Rapporten nr. 80.

Det er en tilsnikelse å kalle 'webtop' et nytt begrep. Både Netscape, Sun og andre leverandører har brukt det i årevis – på ulike områder og med forskjellig betydning. Det finnes sågar et nettsted med navnet www.webtop.com, som primært er en søketjeneste. Ikke minst derfor er det behov for en aldri så liten begrepsoppydding.

Et nytt vindussystem?

Vi introduserte begrepet i forrige utgave – i artikkelen “Webtop: Taran-tella angriper MetaFrame”, og kan syntetisere definisjonen ned til følgende: **Webtop er et Java-basert arbeidsmiljø innenfor rammene av en nettleser.** At Java er et viktig grunnlag, er uinteressant for den gjengse bruker, mens det – som vi skal se – har betydning for oss som skal evaluere teknologi og produkter. For å klargjøre definisjonen ytterligere, og legge et grunnlag for å kunne vurdere *webtop* mot andre alternativer, er følgende observasjoner nyttige:

- ✓ *Webtop* er en moderne utgave av *desktop* – og betegner først og fremst omgivelsene vi møter på skjermen: Vinduer, knapper, rammer, farger, ikoner, administrative funksjoner – og mye mer.
- ✓ Det viktigste argumentet for forandringen – fra *desktop* til *webtop* – er å eliminere avhengigheten mellom operativsystem og applikasjoner på klientsiden.
- ✓ Som konsept er *webtop* konsistent med sentraliseringstanken. I forhold til et tradisjonelt PC-miljø, trengs det praktisk talt ingen programvare på klienten – utover nettleseren og et effektivt bufferlager.

Webtoppens raison d'être er med andre ord frikobling og forenkling, plattform- og lokasjons-uavhengighet. Dette er vel og bra, ser flott ut på papiret og ikke minst på enkelte leverandørers produkt- og teknologi-presentasjoner. Spørsmålet vi naturlig stiller oss, er om dette samsvarer med virkeligheten. Hvordan ser en *webtop*-løsning ut i praksis? Har vi oppnådd en reell gevinst eller kun flyttet på et sett problemer? Hva er medaljens bakside, og ikke minst: Hvor er produktene?

Svar på disse spørsmålene får vi ved å først gjennomgå en ideell *webtop*-arkitektur, og deretter sette sammen en realistisk arkitektur tilpasset dagens virkelighet med hensyn til tilgjengelig og eksisterende utstyr, applikasjoner og pålitelighetskrav. Deretter kan vi sammenligne dem, evaluere avstanden fra den ene til den andre, i tid og teknologi, og til slutt trekke noen konklusjoner med hensyn til hva som er optimale valg, når og under hvilke omstendigheter.

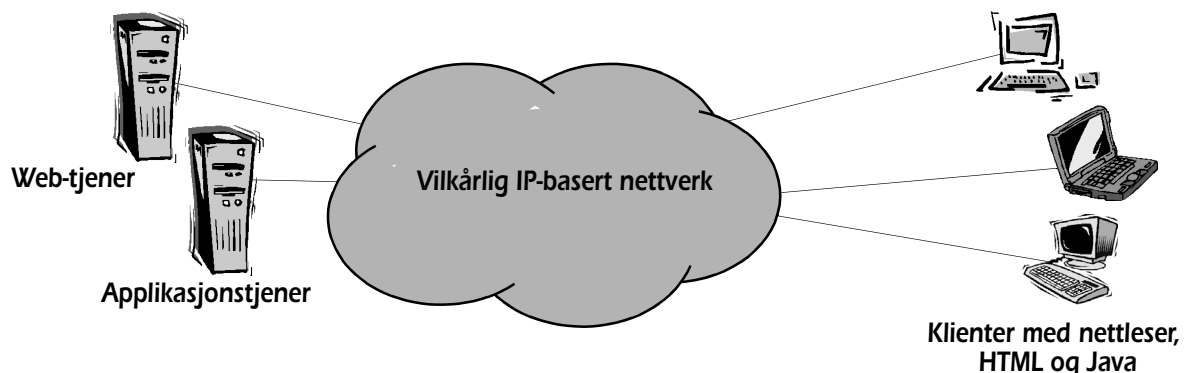
En WEBTOP-arkitektur

Når vi graver i leverandørens presentasjonsmateriale, pressemeldinger, produktspesifikasjoner og andre dokumenter, finner vi først og fremst fremtidsvisjoner. Alle ser for seg en utvikling, ingen føler seg helt sikre på hvor dette bærer, men innser at den som er tidlig ute har størst mulighet til å påvirke utviklingen. Vi har i løpet av det siste halve året fått servert Microsofts .NET-konsept, Suns ONE (*Open Net Environment*) og HPs *e-services*, alle med samme målsetting: Å bli ledende på et område de ser vil bli viktig om 2 til 4 år. Og fellestrekkene stopper ikke der: Alle tre – og andre som har meldt seg på scenen til denne forestillingen – kombinerer eksisterende teknologier med en samling etablerte og nye standarder, legger til et antall nye som de håper skal slå an – og vips, så mener de å ha en vinnende strategi for fremtiden.

Mye skrik, lite ull

Her er fremtiden et nøkkelord: Fint lite av dette kan anvendes til noe som helst i dag. Vi får servert spesifikasjoner, grensesnitt og verktøy – samt en del halvfunksjonell programvare som er mer eller mindre ferdige byggeklosser til en endelig løsning. For eksempel er Suns StarOffice og StarPortal nå blitt nøkkel-elementer i ONE, mens Microsoft har dratt med seg det meste av sine *backoffice*-tjenester inn i .NET.

Vi kan synse rundt disse planene og deres substans, men det bringer oss lite mat med hensyn til vår nære fremtid. For eksempel legger Microsoft – etter forliket med Sun angående Java – inn tungt jern for å få all verdens utviklere til å snu seg mot .NET-spesifikke verktøy i stedet. Mens ingen overraskes av dette, er det verdt å minne om at det tok 6 år fra Java kom på banen til det ble en faktor å regne med i det generelle markedet. At Microsoft tror de kan klare det samme på et år eller deromkring, virker i beste fall naivt.



Figur 1 En enkel – og ideell – klient/tjener arkitektur anno 2001: Klientsiden tar vare på brukergrensesnittet, applikasjoner kjører i sin helhet på mer eller mindre dedikerte tjenere. Web-tjeneren fungerer som bindeledd, og leverer nødvendige APPLETS (brukerdelen av applikasjonene) til nettleseren hos klienten når behovet dukker opp.

Grunnelementer

Når vi konsentrerer oss om nåtid og nær fremtid, finner vi at en *web-top*-arkitektur må bestå av følgende grunnleggende elementer:

- ✓ Klient med nettleser som støtter Java og HTML – jo tynnere, desto bedre så lenge funksjonelle og ytelsesmessige behov er dekket
- ✓ Applikasjonstjenere
- ✓ Et nettverk som er tilpasset de behov denne arkitekturen dikterer
- ✓ En samling verktøy/applikasjoner som er '*web-enabled*', med et klart skille mellom de brukersentriske elementene og de funksjons-/behandlingssentriske delene

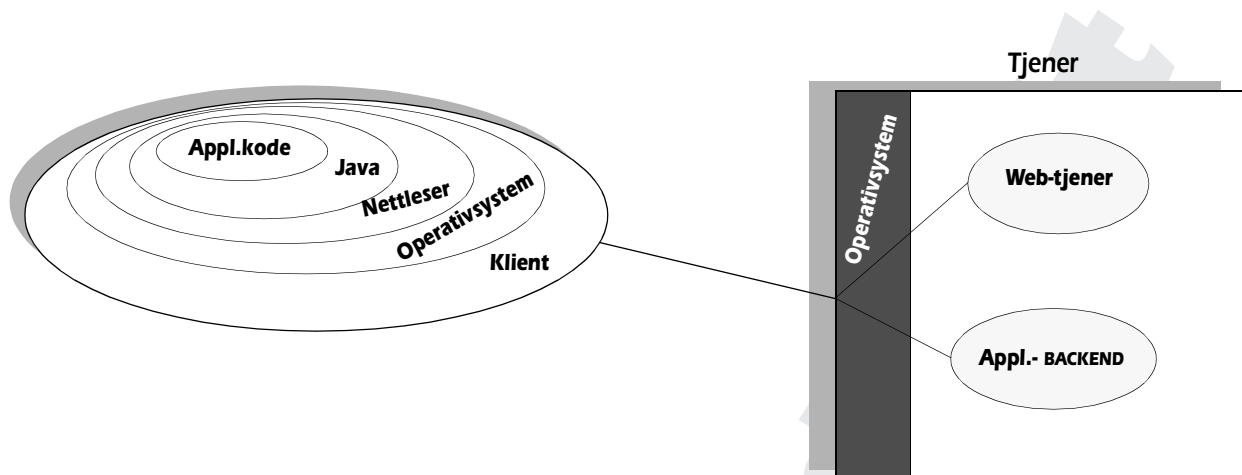
Fellestrekkene med en tradisjonell klient/tjener-arkitektur er påtagelige – med god grunn: Dette er nettopp en klient/tjener-arkitektur – tilpasset krav, behov og muligheter i 2001.

På søken etter applikasjoner

Mens de tre første punktene er enkle å tilfredsstille – enten med dagens utstyr, eller med nye produkter tilpasset disse behovene (se for eksempel *The New Internet Computer* på side 30), er det langt mellom alternativene på applikasjonssiden. Riktignok er det blitt utviklet hundrevis av oppgavespesifikke løsninger etter denne modellen (figur 2) i løpet av de siste årene, men de har lite med generelle databehandlingsverktøy å gjøre. Her ser vi en annen parallell til tradisjonell klient/tjener-teknologi: Bølgen som rullet over IT-verden i første halvdel av 90-årene, ble dempet av mangel på applikasjoner bygget etter 'riktige prinsipper'. Til slutt døde det hele ut uten egentlig å ha tatt av, fordi en annen bølge – WWW og Internett – veltet inn over verden og feide til side alt som ikke hadde fått ordentlig feste.

Heller ikke *webtoppen* kan påstås å ha fått ordentlig fotfeste i markedet per dags dato. Utviklingsverktøy eksisterer, applikasjonsnisjer finnes og mange av oss gjør mer eller mindre bevisst bruk av slike løsninger når vi handler på Internett eller søker etter informasjon hos leverandører og søketjenester: Små Java-programmer finner veien til vår nettleser og utgjør brukergrensesnitt mot omfattende applikasjoner på tjenersiden.

Veien derfra til å plassere brukere flest overfor en *webtop* i stedet for en *desktop* er imidlertid lang. Generelle verktøy – kontorautomasjon, salgsverktøy, CRM- og ERP-løsninger er med få unntak fraværende. Et av unntakene som bekrefter denne regelen, bærer navnet 'Anywhere 2.0 Application Server' – en ombygget utgave av den ikke helt ukjente integrerte kontorpakken ApplixWare, som har vært på markedet for Linux og Unix siden 80-tallet. Funksjonelt havner Anywhere 2.0 i kategorien Office-klone – en odde betegnelse i og med at 'opphavet' var 10 år gammelt da Microsoft Office så dagens lys. I dag er imidlertid MS Office målestokken for de fleste, og dermed det naturlige sammenligningsgrunnlag.



Figur 2 Selv en tynn klient trenger et omfattende system av grunnlags-programvare for å være funksjonell. Kravene til operativsystemet er imidlertid dramatisk forenklet i forhold til hva vi er vant med fra PC-verden. Dermed kan operativsystemet realiseres med noen få prosent av koden vi finner i Windows, hvilket ikke bare gir en enorm effektivitetsgevinst, men også gjør det mulig å oppnå både stabilitet og sikkerhet.

Mens konseptet dermed enn så lenge fortjener stryk for applikasjons-tilfang, har vi mer enn nok til å evaluere potensialet i arkitektur og konsept. Hvor store er Java-programmene, hvor lang tid tar det å laste dem ned, hvordan er responsen, og hvordan virker det på brukerne? Produktet fra Applix¹ gir oss glimrende erfaringsmateriale i så måte – gjennom praktiske prøver via Internettet – eller lokalt dersom vi vil gjøre jobben med å laste ned og installere produktene.

Som vi nevnte i forrige artikkel, har flere forsøk på å lage nettopp hva Applix her viser, strandet fullstendig, og årsaken er interessant: Windows- (og Mac- for den del) applikasjoner har en monolittisk arkitektur der applikasjonslogikk, datastrukturer og brukergrensesnitt er tett sammenvevd med hverandre og operativsystemet. Her er det fint lite som kan bringes over til en arkitektur der brukergrensesnittet er skilt fra applikasjonslogikken. Kun unntaksvis finnes det veldefinerte grensesnitt å skjære etter når en slik oppsplitting skal finne sted. Dermed må praktisk talt hele applikasjonen skrives på nytt, en formidabel oppgave – for å si det mildt.

Verktøy som kommer fra Unix/Linux-siden (egentlig XWindows-siden) har et ganske annet utgangspunkt: De har denne modellen innebygget allerede,² og forandringene begrenser seg til selve brukergrensesnittet. Det meste av applikasjonslogikken kan beholdes – eller benyttes som utgangspunkt hvis den er mangelfull i forhold til behovene. Observasjonen er interessant fordi den indikerer hvem som står overfor hvor store utfordringer i forbindelse med denne overgangen. Det er en skjebnens ironi at Microsoft, i godt selskap med applikasjonsleverandører på PC-plattformen generelt, på denne måten får servert konse-

¹ Leverandøren kaller seg VistaSource, og er et underbruk av Applix Corp.

² Se artikkelen "X11 - misforstått teknologi skaper revolusjon" i Mellvik-Rapporten nr. 6, tilgjengelig via vår Web-tjeneste.

kvensene av sin tette kobling mellom OS og vindussystem. Etter å ha gjort nødvendigheten av tett integrasjon til et hovedargument i monopol-rettsaken, må det smake vondt å få demonstrert det eksakt motsatte for åpen scene kort tid etterpå.

Her finner vi hovedårsaken til at Anywhere 2.0 fungerer så bra som den gjør. Systemet er langt fra prikkfritt, verken funksjonelt eller bruksmessig, men i forhold til hva markedet har vendt seg til å akseptere, er dette en glimrende start. Fonter kan fra tid til annen se grove ut, og dekorasjoner (vindusrammer, trykknapper, menyer etc.) er ikke alltid like estetisk attraktive som vi er vant med. De mest spennende parametrene, som ytelse, respons og tiden det tar å komme i gang (størrelsen på Java-programmene som må lastes ned ved første gangs bruk), er akseptable på en ISDN-linje og gode når båndbredden er høyere. Dermed har vi fått verifisert at ideene lar seg realisere.

Hvilke klienter?

Det neste naturlige spørsmål er om klienten virkelig blir enklere, billigere og bedre. Må vi ikke ha inn et operativsystem som nettleseren kan 'kjøre under' i alle fall, og forsvinner ikke vinningen dermed i spinningsen? Og hvordan får vi tatt vare på behovene for lokalt periferiutstyr – printere, scannere, kameraer etc.?

Svaret er produktavhengig: Kjører klienten Windows NT/2K/9x/me, blir nettleseren et vindussystem inne i et vindussystem, en smør-på-flesk løsning som aldri kan bli spesielt effektiv, og heller ikke tilfredsstillende grunnleggende krav til pålitelighet og korte oppstarttider.³

Angripes utfordringen uten historisk ballast, blir forholdet et helt annet. Operativsystemet kan flyttes tilbake dit det hører hjemme, godt ute av syne og med oppgave å disponere hardware-ressursene på en optimal måte, inklusive nettverksgrensesnitt og -trafikk. QNX, Linux og flere andre alternativer har det som skal til for å ta vare på slike oppgaver.

Tilsvarende gjelder for nettleseren. Mens mastodontene fra Netscape og Microsoft representerer ytelsesmessige og stabilitetsmessige utfordringer selv for dagens raskeste prosessorer, demonstrerer for eksempel Opera at det er mulig å levere gode resultater uten å være elefantsyk. Støtte for lokalt periferiutstyr er teknisk sett trivielt, uten at vi dermed skal hevde at det er problemfritt. En USB-port dekker de fleste tenkelige behov, mens programvaren som skal besørge sømløst samspill med applikasjoner på tjenere i nettverket, er en utfordring som krever oppmerksomhet.

Kort sagt er situasjonen under kontroll på klientsiden: Elementene finnes, og en rekke leverandører har produkter klare på startstreken. I det øyeblikket markedet løsner, vil de bli tilgjengelige i løpet av få uker – til priser som får en CD-brenner til å virke kostbar.

³ Det er rimelig å forlange at en klientarbeidsplass skal være operativ på mindre enn 10 sekunder.

Hva så med det utstyret vi allerede har? Er det ikke meningsløst å kaste utstyr som for mindre enn et år siden kostet NOK 15.000 til fordel for et som i dag koster 3.000? I de fleste tilfeller er det nettopp det, og det er ingen grunn til å lage et problem av noe som slett ikke er det. En fet klient kan gjerne brukes som om den var tynn, om den er aldri så overkvalifisert. Med mindre operativsystemet byttes ut eller lokalt masselager fjernes, vil den imidlertid fortsatt være driftsmessig kostbar, energikrevende og støyende. Det kan lages et regnestykke, som riktignok må baseres på betydelige estimater, men som vil gi gode indikasjoner med hensyn til hva som er riktig tidspunkt for å kvitte seg med fedmen til fordel for det slanke og enkle.

WEBTOP anno 2001

Startskuddet for det vi kan kalle 'den ekte *webtop*' lar imidlertid vente på seg; pistolen kan knapt sies å være ladet enda. Like lite som en ny generasjon løsninger står klare til å overta, er markedet modent for en omveltning i stor skala. Pålitelighet er viktigere enn å spare noen kroner, det vi har er tryggere enn det vi ikke kjenner, og det vi kan – eller tror vi kan – er vår livsforsikring i jobb-sammenheng.

Derfor ville revolusjonen tatt sin tid selv om alle brikkene var på plass, og av samme grunn er investeringene i nettopp de brikkene som mangler, altfor små. Av skade blir man klok, og overoptimisme har sendt mang en gründer og mangt et godt produkt til de evige jaktmarker av feil årsaker.

Det betyr imidlertid ikke at vi kan legge *webtoppen* på hylla og la den hvile i fred til noe dramatisk skjer. Når vi har akseptert at forandringen blir evolusjonær i stedet for revolusjonær, har vi også innsett at løpende små forandringer vil være en del av hverdagen i uoverskuelig tid fremover. Forandringene som om noe tid vil gjøre *webtoppen* til en selvfølge, har tikkert av gårde i flere år allerede. Vi diskuterte en del av dem i forrige artikkel, hvor vi også trakk en viktig konklusjon:

Webtoppen representerer et reelt alternativ allerede i dag. Vi kan når som helst velge å flytte alle tenkelige applikasjoner over på en tynn klient etter de spesifikasjonene vi har presentert ovenfor, mens vi fortsetter å bruke de samme applikasjonene som før, på Windows – med Windows Terminal Server, og på Unix/Linux via XWindows. Limet som får det hele til å henge sammen er produkter som Tarantella Enterprise, Citrix MetaFrame og NCD ThinPath.

Fordelen ved å ta konsekvensen av denne virkeligheten snarest, har lite med å posisjonere seg for fremtiden å gjøre, selv om nettopp det følger med på kjøpet. Det viktigste er å få arkitekturen – og dermed kostnadene – under kontroll.

www.tarantella.com
www.citrix.com
www.ncd.com

Teknologi- visjoner

Blant leverandørene vi nevnte innledningvis – og de fleste andre som er aktive innen ulike varianter av Web-teknologi, blir XML presentert som vidundermedisinen som skal forandre verden – i nær fremtid. Vi har hørt de samme historiene år etter år siden XML kom på banen,

men fint lite har skjedd. De siste månedene har vi sågar fått høre at avledninger av XML skal utradere Java.

Det aller meste av disse gyrasjonene høre hjemme i kategorien 'spin'. XML er særdeles interessant, og utviklingen har på ingen måte stått stille de siste årene, men teknologien spiller fortsatt en relativt ubetydelig rolle i hverdagen. Likeledes finnes det grenser for hva et 'utvidbart markeringsspråk' eger seg til. Hva som er mulig er én ting, og det er et faktum at regelrett utrolige ting er mulig med XML. Hva som er smart, er en annen sak, og vi minner igjen om at det tok 6 år for Java å komme til dagens modenhetsnivå – teknologisk og sikkerhetsmessig.

Videre blir vi tutet ører og øyne fulle av hvor intelligent fremtidens nettverk skal bli. Det talende kjøleskapet som verserer på TV-reklamen i disse dager, er kun en sped begynnelse: Små programsnutter skal på egen hånd traversere nettverkene og ikke bare gjenkjenne oss når vi logger inn, men omtrent kunne gjette hva vi ønsker før vi får sagt et ord. Dette er interessante visjoner, og realistiske om vi ser tilstrekkelig langt frem i tid. Når vi velger å konsentrere oss om det som er matnyttig på kort sikt, blir det mest underholdning – som vi kan like, men som ikke bør forstyrre vår fokusering.

Konklusjon

Den viktigste gevinsten ved overgangen fra *desktop* til *webtop* er overlappende med hva vi har observert rundt tynne klienter de siste årene. Faktum er at *webtoppen* er katalysatoren markedet har ventet på for at gevinsten skal la seg realisere på bred basis. MetaFrame og Windows Terminal Server har på mange måter lagt grunnlaget, gjennom å demonstrere et tjenersentrisk konsept for en generasjon som aldri har sett verken stormaskiner, minimaskiner eller dumme terminaler.

Det er urovekkende å oppdage den teknologiske ironien i dette: I etterpåklokskapens grelle lys er det bortimot flaut å ende 90-årene med en løsning som overfører skjermbilder fra, og museklikk til en tjener som primært er laget for å levere filer over nettet. Ideen ville blitt mottatt som en dårlig vits på 80-tallet, og er en nødløsning overmoden for avløsning.

Webtop er et viktig trinn i denne avløsningen. Dessuten – som vi kommer tilbake til i artikkelen på side 21, blir *webtoppen* et viktig element i videreutviklingen av ASP-konseptet! ■