Forord

Statlig forvaltning blir i stadig større grad ensrettet i sine valg av IT-produkter. Statskonsult ønsker å påpeke hvilke ulemper en for stor ensretting kan medføre, og vurdere alternativer til de eksisterende løsningene.

Denne rapporten vurderer et spesielt fenomen innenfor programvareutvikling som betegnes som ”Open Source Software”, eller åpen programvare som vi velger å kalle det. Et eksempel på åpen programvare er operativsystemet Linux.

Rapporten har vurdert om åpen programvare kan gi forvaltningen større leverandøruavhengighet, og om den kan redusere de totale IT-kostnadene.

Rapporten har blitt skrevet over et lengre tidsrom samtidig som vi her i Statskonsult har drevet egen utprøving av ulike typer åpen programvare. De fleste i IT-avdelingen i Statskonsult har bidratt samt intern IT-drift og interesserte i andre avdelinger. I tillegg har vi hentet inn synspunkter fra andre deler av forvaltningen. Prosjektleder har vært seniorrådgiver Endre Grøtnes.

Oslo, mars 2001

Jon Blaalid

1 Sammendrag

Pressen, fagmiljøer og politikere har i den senere tid fokusert på Linux som et alternativ til Windows. Av denne grunn, og fordi Statskonsult er opptatt av å utrede forhold som kan redusere leverandøravhengigheten og se nærmere på løsninger som kan redusere kostnadene ved anskaffelse av programvare, har den foreliggende rapporten blitt til.

Linux er et åpent, fritt tilgjengelig operativsystem som er utviklet som "Open Source"-programvare, eller *åpen programvare*. Åpen programvare betegner programvare der kildekoden er fritt tilgjengelig, og programmet kan brukes, endres, forbedres, utvides og redistribueres av alle som måtte ønske det.

Statskonsult har gjennom innhenting av informasjon, samtaler med aktører og egen utprøving vurdert anvendeligheten av Linux og annen åpen programvare. I tillegg til nå å kunne si noe nærmere om hvor anvendelig åpen programvare er i forvaltningen, har vi dokumentert hva åpen programvare betyr for utviklingen av IT-infrastrukturen.

Statskonsult mener at Linux og annen åpen programvare har et stort potensiale. Det er i hovedsak på områder som utvikling av infrastruktur, utarbeidelse av standarder og i undervisning og opplæring. Det er foreløpig ikke et tilstrekkelig tilbud av åpen sluttbrukerprogramvare, slik at forvaltningen kan erstatte sin eksisterende programvare med disse. Det finnes for eksempel ingen regnskaps- eller virksomhetsstyringssystemer for Linux.

Åpen programvare framstår i dag som en viktig bidragsyter til videreutviklingen av Internett-infrastrukturen. De grunnleggende standardene for overføring av data mellom maskiner, sending av e-post og oversetting av adresser på Internett er alle tilgjengelig som åpen programvare. Standardene er også utviklet og spesifisert gjennom en åpen prosess.

Anbefalingene om åpen programvare kan oppsummeres i følgende punkter:

- Bruk av åpen programvare for å redusere kostnader vil være hensiktsmessig i virksomheter der de totale IT-kostnadene i hovedsak er knyttet til kjøp av lisenser, og kostnadene ved opplæring av brukerne på "ny" programvare er relativt liten.
- Linux er velegnet som tjeneroperativsystem, også for offentlig sektor. Vår vurdering er at åpen sluttbrukerprogramvare ikke per i dag kan erstatte de kommersielle alternativene når det gjelder funksjonalitet og integrasjonsmuligheter.
- Det offentlige bør oppmuntre til bruk av Linux og annen åpen programvare innenfor skole- og utdanningssystemet.
- Det offentlige bør støtte utviklingen av åpen programvare. Støtten kan komme gjennom forsknings- og utviklingstiltak. Ved tildeling av forsknings- og utviklingsmidler kan det være et krav at programvare som utvikles, gjøres tilgjengelig som åpen programvare.

2 Innledning

2.1 Bakgrunn

Gjennom undersøkelsen "IT i staten 1999"[1] fikk Statskonsult fram følgende tall om Microsofts dominerende posisjon som brukermiljø i statlig forvaltning.

- 96 % benytter MS Word som tekstbehandler.
- 91 % av virksomhetene hadde en Windows NT tjener, og Windows NT utgjorde 60 % av alle tjenermaskinene.
- Over 60 % av alle klientmaskiner benytter en eller annen versjon av Windows som operativsystem[2].
- Outlook og MS Mail utgjør 44 % av antall e-postklienter[3].

Den avhengigheten statlig sektor, ifølge tallene i undersøkelsen, har til én leverandør, samt de stadig økende kostnadene til IT generelt, gjør det naturlig å vurdere mulige alternativer til Windows/Office plattformen.

I den senere tid har det i pressen, fagmiljøer og blant politikere blitt fokusert på Linux som et alternativ til Windows. Linux er et åpent, fritt tilgjengelig operativsystem som er utviklet som "Open Source" programvare, eller *åpen programvare*[4] som vi velger å betegne konseptet i denne rapporten.

Bakgrunnen for at Statskonsult har utredet åpen programvare er de antatte mulighetene denne programvaretypen har til å

- redusere leverandøravhengigheten i statlig sektor
- redusere kostnadene til anskaffelse av programvare i statlig sektor

For at forvaltningen skal ta i bruk åpen programvare, bør programvaren være enkel å anskaffe, enkel å installere og enkel å drifte. I tillegg må det være et programtilfang for operativsystemet, slik at man for eksempel kan benytte programvaren til saksbehandling eller annen oppgaveløsning. Åpen programvare beregnet på sluttbrukere, som tekstbehandlere og e-postklienter, må også ha den samme grad av ”brukervennlighet” som proprietære[5] produkter for å kunne bli tatt i bruk i forvaltningen.

[1] Statskonsultrapport 2000:8. IT i staten 1999.
[2] IT i Staten 1999 spurte ikke direkte om hvilket klientoperativsystem som ble benyttet, men ut fra svarene om bruk av tjeneroperativsystem og antall tilknyttede PC-er/terminaler har vi utledet dette minimumstallet.
[3] Dette var før Fts overgang til Outlook.
[4] Vi vil benytte betegnelsen åpen programvare på den typen programvare som blir omhandlet i denne rapporten. Engelske betegnelser omfatter blant annet "Free Software og "Open Source Software". Se for øvrig kapitlet om "åpen programvare" for mer utførlig beskrivelse.
[5] Vi vil i denne rapporten benytte betegnelsen proprietær programvare på all programvare som ikke er åpen programvare.

Statskonsult er ansvarlig for standardisering av datakommunikasjon med tilhørende infrastruktur innenfor offentlig forvaltning. Vi har derfor også vurdert hvordan åpen programvare kan benyttes for å legge forholdene bedre til rette for felles standarder for data- og informasjonsutveksling innenfor det offentlige og ut mot omverdenen.

2.2 Leverandøruavhengighet

En av grunnene til at Statskonsult utreder åpen programvare er for å vurdere om større bruk av åpen programvare kan redusere leverandøravhengigheten i offentlig sektor. Dette avsnittet vil beskrive de ulempene for stor leverandøravhengighet kan medføre.

Begrepet leverandøruavhengighet oppfattes gjerne som reell konkurranse, det vil si at vi som kjøpere kan velge mellom flere leverandører og produkter når vi skal kjøpe en vare eller tjeneste. Avhengighet av en leverandør kan oppfattes som en monopolsituasjon eller mangel på konkurranse og er noe som ikke er ønsket. Hele innkjøpsregelverket for offentlig sektor er basert på prinsippet om flere leverandører, og at det er en reell konkurranse mellom leverandørene.

Monopolsituasjoner kan føre til saktere produktutvikling, dårligere tjenester, høyere priser og større sårbarhet fordi kun et bestemt produkt benyttes. I NOU 2000:24 ”Et sårbart samfunn” står det: *”Det har vist seg at noen leverandører har oppnådd tilnærmet monopol for utvikling av programvare. Ulempene ved slikt monopol ligger ikke bare i markedsmakten og økte kostnader i mangel av konkurranse, men også avhengighet og stort nedslagsfelt for negative konsekvenser dersom noe skulle gå galt.”*

Produktutvikling. Det er generelt antatt at konkurranse fører til raskere produktutvikling og produkter som er bedre tilpasset sluttbrukernes behov. Innenfor IT-området er det en stor diskusjon om sammensetting av produkter fra flere leverandører eller om alle produkter fra samme leverandør gir den beste totale løsningen for brukerne. Vi vil i denne rapporten anta at flere leverandører gir brukeren større valgmuligheter og dermed bedre muligheter til å finne produkter og tjenester som best er tilpasset brukeren.

Tjenester. På dette området er det meget viktig med konkurranse, slik at brukeren får riktig tjeneste til en fornuftig pris. Med monopol på tjenester må brukerne bare ta det de får. På de fleste områder innenfor tele- og dataområdet har man opphevet tidligere monopoler for å få bedre konkurranse på tjenester. Et eksempel er opphevelsen av telemonopolet, som har ført til at alle kan tilby teletjenester, og en markert nedgang i priser på teletjenester. I denne rapporten antar vi at konkurranse på tjenester er et ubetinget gode for brukerne.

Monokultur. Ved utstrakt bruk av et bestemt produkt oppstår det vi kaller monokulturer. Monokulturer er svært utsatt hvis det oppstår feil i produktet. Eksempler på dette fra IT-verdenen er feilene som har oppstått i Intels prosessorer i den senere tid, noe som har ført til at datamaskinleverandørene

ikke har kunnet levere så mange maskiner som kundene vil ha. Intel har tilnærmet monopol på tilvirkningen av prosessorer til PC-er.

Statskonsult anser åpen programvare som leverandøruavhengig. Grunnene til dette er blant annet:

- Flere leverandører kan levere det samme produktet siden alle spesifikasjoner, all dokumentasjon og selve kildekoden er tilgjengelig for alle.
- Det finnes ingen patenter eller deler av programvaren som andre ikke kan bruke.

2.3 Mål og gjennomføring

Målet med utredningen har vært å vurdere hvor egnet forskjellige former for åpen programvare er for offentlig sektor.

I utredningen har vi spesielt lagt vekt på å vurdere mulige bruksområder for åpen programvare i forvaltningen, de økonomiske konsekvensene ved å velge åpen programvare og hvordan åpen programvare kan skape større leverandøruavhengighet.

Vi har testet ulike distribusjoner av operativsystemet Linux, e-posttjeneren Sendmail og web-tjeneren Apache når det gjelder åpen programvare. Utredningen har i hovedsak fokusert på bruk av åpen programvare på tjenermaskiner, men også klientsiden ble vurdert. Vi har også vurdert bruk av åpen programvare som et standardiseringstiltak.

Det er innhentet erfaringer fra noen sentrale miljøer og brukere av åpen programvare i offentlig sektor. Noen eksempler på bruk av åpen programvare er beskrevet i rapporten. I tillegg har vi hatt kontakt med miljøer som tilbyr ulike tjenester innenfor området åpen programvare.

Under arbeidet med utredningen har vi åpnet opp for og mottatt en rekke henvendelser, kommentarer og bidrag fra eksterne aktører. Vi har så langt som mulig tatt hensyn til disse ved utarbeidelse av rapporten.

2.4 Om rapporten

Kapitlene 3-5 beskriver hva åpen programvare er, ulike økonomiske modeller for åpen programvare og de mest utbredte åpen-programvare-produktene.

Kapittel 6 gir eksempler på bruk av åpen programvare i offentlig sektor. Kapittel 7 beskriver kort de åpen programvareproduktene og tjenestene som tilbys i Norge i dag. Kapittel 8 tar spesielt for seg åpen programvares mulighet som standardiseringstiltak, mens kapittel 9 vurderer anvendeligheten til forskjellig åpen programvare. Kapittel 10 oppsummerer de konklusjonene og anbefalingene vi har kommet fram til.

3 Åpen programvare

INFORMATION, NO MATTER HOW EXPENSIVE TO CREATE, CAN BE REPLICATED AND SHARED AT LITTLE OR NO COST.
— Thomas Jefferson

Å dele og kopiere informasjon er i dag enda enklere enn da Thomas Jefferson levde. Det er verdt å merke seg at (kildekoden til) programvare også er informasjon, og at sitatet like gjerne kan benyttes om programvare. Det er muligheten til enkel spredning av informasjon, til store brukergrupper over lange avstander[6], som er årsaken til at konseptet åpen programvare har fått den betydningen det har.

3.1 Hva er åpen programvare?

Åpen kildekode programvare, eller åpen programvare som vi vil benytte i dette dokumentet, er en norsk oversettelse av begrepet "Open Source Software".

Det er ikke enkelt å beskrive hva åpen programvare er med noen få ord. Grunnen til dette er at det finnes flere varianter og definisjoner av konseptet. Vi vil i dette avsnittet kun gi en overordnet beskrivelse, ikke noen definisjon, før vi i neste avsnitt beskriver hva vi forstår med åpen programvare gjennom å presentere hovedprinsippene i konseptet.

Åpen programvare betegner programvare der kildekoden til programmet er fritt tilgjengelig, og at programmet fritt kan brukes, endres, forbedres, utvides og redistribueres av alle som måtte ønske det.

Det finnes klare definisjoner av både "Open Source Software"[7] og "Free Software"[8]. Vår bruk av betegnelsen åpen programvare betegner programvare som oppfyller kravene beskrevet i avsnitt 3.2.

3.2 Hovedprinsippene til åpen programvare

Åpen programvare kan kjennetegnes gjennom den frihet hver bruker[9] har til å

- bruke programvaren på den måten han eller hun ønsker, på så mange maskiner som er ønskelig,og i enhver situasjon det er ønskelig

[6] Dette kan leses som tilgjengeligheten av Internett.
[7] The Open Source Initiatives definisjon av Open Source Software finnes på: http://www.opensource.org/osd.html
[8] Free Software Foundation definisjon av Free Software finnes på: http://www.fsf.org/philosophy/free-sw.html
[9] Når man i åpen programvaremiljøer snakker om brukere, tenker man på en kompetent programvareutvikler og ikke en vanlig ”pek-og-klikk” sluttbruker. Brukerperspektivet i miljøet er meget utviklerorientert.

- ha mulighet til å tilpasse programvaren slik at den tilfredsstiller hans eller hennes spesielle behov. Dette inkluderer forbedringer i programvaren, feilretting, utvidelser av programvaren og muligheten til å studere hvordan programmet virker
- redistribuere programvaren til andre brukere, som igjen kan tilpasse programvaren i henhold til sine behov

For å kunne oppfylle betingelsene ovenfor er det enda en betingelse som må tilfredsstilles:

- Brukeren av et program må ha tilgang til kildekoden til programmet.

Kildekoden til et program, som oftest skrevet i et høynivå programmeringsspråk som for eksempel Java, C eller C++, er en absolutt forutsetning for at en bruker skal kunne forstå hvordan programmet er byd opp og virker og deretter kunne tilpasse, utvide eller rette opp feil i programmet.

For å kunne opprettholde frihetene beskrevet i dette avsnittet, er det nødvendig å beskytte programvaren med lisensbetingelser og/eller copyright. Dette kan virke noe paradoksalt, men er programvaren uten lisensbetingelser, såkalt public domain, kan folk for eksempel ta egen copyright på programvaren, redistribuere selve programmet uten kildekoden eller bruke kildekoden som basis for proprietær programvare. Ulike åpen-programvare-lisenser blir beskrevet i avsnitt 3.4.

3.3 Forskjellen mellom åpen programvare og andre typer programvare

Åpen programvare er ikke det samme som gratis programvare. Gratis programvare eller "freeware" er betegnelsen på kompilerte[10] programmer som distribueres fritt. Det finnes mye gratis programvare, alt fra små hjelpeprogrammer til store kjente produkter som Microsofts Internet Explorer.

Kommersiell programvare som selges uten tilgang til kildekoden og muligheten til å endre og redistribuere denne, betegnes som proprietær programvare. Det er spesielt lisensbetingelsene til proprietær programvare som gjør at folk utvikler åpne alternativer. De fleste lisenser har begrensninger på antall maskiner programmet kan installeres på, muligheten til å endre noe i programmet og muligheten til videredistribuere programmet.

Lisensbetingelsene til proprietær programvare medfører at man kjøper en rett til å bruke programmet på visse vilkår, ikke selve programmet.

[10] Et kopilert program kan leses av maskinvaren som skal bruke programmet, og ikke av mennesker. Et program som er skrevet i et språk som mennesker forstår, blir kompilert til et språk som maskinvaren forstår. Når man kun distribuerer det kompilerte programmet, er det kun maskinvaren som kan lese og forstå programmet.

3.4 Kort om åpen-programvare-lisenser

Lisensbetingelser knyttet til programvare er et omfattende tema. Her vil vi gi eksempler på innholdet i åpen-programvare-lisenser og forklare hvorfor det er knyttet lisensbetingelser til åpen programvare.

Den første lisensen knyttet til åpen programvare ble utviklet av Free Software Foundation i forbindelse med deres GNU[11] programvare. En beskrivelse av prinsippene finnes på: http://www.gnu.org/copyleft/copyleft.html. Lisensen til GNU programvaren heter GNU General Public Licence og kalles kun for GPL[12]. Hele lisensen finnes som et vedlegg til rapporten. GPL er fundamentet for de fleste andre åpen-programvare-lisenser. The Opens Source Initiative har laget en beskrivelse av hva som skal til for å kalle en lisens en åpen programvarelisens [3].

To eksempler på åpen-programvare-lisenser blir presentert i vedlegg til denne rapporten. Det er GPL (Vedlegg III) og BSD (Vedlegg IV).

En av hovedgrunnene til at det er knyttet lisenser til åpen programvare, er for å beskytte programvaren slik at den fremdeles skal være åpen. Som nevnt tidligere kan åpen programvare bli gjort om til proprietær programvare hvis det ikke er knyttet lisensbetingelser til programvaren. Lisensbetingelser kan blant annet

- ivareta frihetene beskrevet i avsnitt 3.2
- ivareta betingelser som de opprinnelige rettighetshaverne ønsker (at navn på opprinnelige utvikler følger med, at lisensen distribueres med programvaren, etc)
- påse at rettighetene til de personene som har utviklet programvaren eller har copyright på navnet til programmet, blir ivaretatt
- pålegge at ny programvare basert på den åpne programvaren også forblir åpen programvare
- pålegge at all programvare som benytter den åpne programvaren også skal være åpen programvare

De ulike lisensene varierer mest når det gjelder de to siste kulepunktene. Mens GPL er meget streng når det gjelder at all programvare basert på den opprinnelige koden også skal være fritt tilgjengelig, tillater BSD-liknende lisenser at man kan lage proprietær programvare basert på den opprinnelige kildekoden.

Åpen-programvare-lisenser er ikke til hinder for at virksomheter selger programvaren og tjener penger på dette. De som selger åpen programvare, kan derimot ikke hindre andre i også å selge den samme programvaren. Gjennom lisensbetingelsene kan man ikke begrense den enkeltes rett til å kopiere eller endre programvaren, og man må også la kildekoden følge med de programmene

[11] GNU er en forkortelse av det sirkulære uttrykket Gnus Not Unix.
[12] Originalteksten finnes på: http://www.gnu.org/copyleft/gpl.html

man selger. GPL sier at endringer og forbedringer i koden også skal være åpen kildekode, og at alle kan bruke dette i egen virksomhet. Et godt eksempel på dette er installasjonsprogrammet RPM (RedHat Packet Manager) som er utviklet av Linuxdistributøren RedHat. Dette programmet er også en del av distribusjonen til Caldera og SuSE. Dermed blir forbedringer gjort av en distributør benyttet av de andre distributørene.

Det skal framgå tydelig på åpen-programvare-pakker som distribueres, at distribusjonen er omfattet av en åpen programvarelisens. Alle kan sette sammen pakker med åpen programvare og distribuere disse.

3.5 Eksempler på åpen programvare

Det finnes en mengde åpen programvare tilgjengelig. Mange av de mest benyttede programmene på Internett er åpen programvare. Programmene utvikles ofte i åpne prosjekter. Prosjekt betyr her en liten kjerne med utviklere som publiserer nye versjoner av programmet, og som koordinerer og bruker innsatsen til en hel rekke med frivillige bidragsytere for å videreutvikle programmet. Vi vil her kort beskrive noen av programmene.

3.5.1 Linux

Linux er det programmet som fikk folk utenfor åpen-programvare-kulturen til å bli oppmerksom på åpen programvare, og er det programmet som oftest trekkes fram når man skal forklare hva åpen programvare er.

Linux, eller mer korrekt GNU/Linux, er et UNIX-liknende operativsystem. Linux prøver å være i samsvar med POSIX-standarden for å være mest mulig kompatibelt med andre UNIX-systemer[13]. Et program utviklet for et POSIX-operativsystem, skal i teorien kunne benyttes på alle operativsystemer som følger POSIX-standarden.

Linux var opprinnelig designet for å kjøres på Intel-baserte PC-er, men finnes nå for andre maskinvareplattformer som Macintosh, Alpha[14] og Sun SPARC.

Linux er utviklet og distribueres under GPL. Dette betyr at Linux kan distribueres fritt så lenge kildekoden følger med. Alle endringer og modifikasjoner av Linux som ønskes distribuert, må også inneholde kildekode for endringene. På denne måten sørger GPL for at Linux forblir åpen, og at ingen kan lage en proprietær versjon av operativsystemet. Se for øvrig kapittel 4 for mer informasjon om Linux.

3.5.2 Apache

Apacheprosjektet er en frivillig felles programvareutviklingsinnsats med sikte på å etablere en robust, funksjonsrik og fritt tilgjengelig kildekode for en

[13] POSIX er utgitt som ”IEEE Standard 1003.1-1988 Portable Operating System Interfaces for Computer Enviroments og som ISO/IEC 9945.1: 1990.
[14] Dette var tidligere DEC Alpha, men er nå Compaq Alpha.

HTTP-tjener (i det følgende noe unøyaktig kalt en Webtjener), med egenskaper som er tilstrekkelige for næringslivsbruk.

Apachetjeneren er den mest brukt webtjeneren på Internett med ca 60 prosent markedsandel ifølge Netcrafts undersøkelser. Det finnes over 5 millioner Apache webtjenere på Internett i dag[15]. For en full og oppdatert oversikt, se http://www.netcraft.com/survey/.

Apache kan benyttes på en rekke plattformer og med mange operativsystemer. Blant de mest brukte operativsystemene for Apache webtjener finner vi: FreeBSD, Sun Solaris og Linux. Apache forsøker å være en referanseimplementasjon for HTTP-standarden for alle plattformer som den blir utviklet for.

Brukerne av Apache er mange. I Norge benytter blant annet følgende virksomheter Apache til sine eksterne webtjenester: Aetat, Datatilsynet, SSB, Statskonsult, Dagbladet, VG, Startsiden, Sol, Stocknet og Telenor[16].

3.5.3 Bind

BIND (Berkeley Internet Naming Deamon) er en implementasjon av DNS-protokollen (Domain Name System) og inneholder en fritt distribuerbar referanseimplementasjon av de viktigste komponentene i DNS, blant annet: en navnetjener (nameserver), et DNS bibliotek for å oversette mellom navn og IPadresser og verktøy for å verifisere at en DNS-tjener fungerer i henhold til standarden.

BIND er den dominerende navnetjeneren på Internett og er en forutsetning for at man skal kunne bruke navn, som for eksempel www.statskonsult.no, for å finne fram til IP-adressen til en maskin på Internett.

3.5.4 Perl

Perl er et programmeringsspråk som har spesielt gode tekstmanipulerings-egenskaper, og derfor er spesielt egnet til å lage interaktive og dynamiske websider. De fleste kommersielle websider som tilbyr dialog med brukerne og tilgang til databasetjenere, er utviklet ved bruk av Perl. Et konkret eksempel er Amazon.com.

Perl kan integreres i webtjenere, noe som fører til store forbedringer i prosesseringshastigheten. Kort sagt kan bruken av Perl gjøre kommunikasjonen med bakenforliggende systemer enklere og raskere.

[15] Dette er Netcrafts tall for oktober 2000.
[16] Opplysningene ble framskaffet gjennom bruk av Netcrafts tjenester i oktober 2000.

3.5.5 Sendmail

Sendmail er den mest brukte tjeneren for Internett e-post. Ca 75 prosent av alle e-posttjenere på Internett er ulike Sendmail-implementasjoner. Sendmail er fritt tilgjengelig og kan også redistribueres fritt.

Sendmail har i hovedsak tre typer av tjenester: e-post ruting (inkludert ulike former for autentiserings- og krypteringsmekanismer), e-post lagring (inkludert støtte for POP og IMAP) og muligheter for utvidelse av e-postprogrammet (Utvidelser som leveringskvitteringer og integrasjon mot en PKI).

Sendmail følger og utvikles i henhold til eksisterende e-poststandarder innenfor IETF. De siste versjonene av Sendmail har støtte for blant annet ESMTP [RFC 1869].

3.6 Utviklingsmetodikken for åpen programvare

Utviklingsmetodikken for åpen programvare er svært forskjellig fra utviklingen av proprietær programvare. Boken "The Cathedral & the Bazaar"[2] beskriver utviklingsmetodikken for, drivkreftene bak og kulturen innenfor åpen programvare. Det er viktig å vite hvordan åpen programvare blir til for å kunne forstå den konseptuelle forskjellen mellom åpen og proprietær programvare.

De fleste åpen-programvare-prosjektene starter med at en eller flere personer føler at en programvare mangler, eller at en eksisterende programvare er utilstrekkelig eller dårlig. De starter dermed å lage ny programvare, gjerne basert på eksisterende programvare der kildekoden er tilgjengelig. Når tilstrekkelige deler av programmet virker, ikke når det er ferdig, legger de det ut på en nyhetsgruppe eller liknende, slik at andre kan bruke programmet og eventuelt komme med kommentarer og forbedringer. Er det mange nok som har interesse av programmet og kommer med kommentarer, er et nytt åpen-programvare-prosjekt født.

Alle åpen-programvare-prosjekter har en eier. Eieren er den som mottar forslag til endringer i programmet, forslag til ny kode, rettelser, forbedringer og som gir ut nye versjoner av programmet. Dette er en kontinuerlig prosess, og i begynnelsen av et prosjekt kan det komme nye versjoner av et program så ofte som hver dag.

All kildekode blir publisert, og det er fritt for alle å bruke den publiserte kildekoden til eksisterende eller nye åpen-programvare-prosjekter.

Eierens oppgave blir å lage en oversiktlig og god struktur i programmet, administrere alle endringer og ny kode som blir foreslått, og ikke minst lage nye versjoner av programmet som har med de endringene og koden eieren mener er fornuftig. Dette betyr at ikke all kode kommer med, eller at alle forslag til ny funksjonalitet blir implementert. Alle programmer har en readme-fil som forteller om hvem som har bidratt med hva til programmet. Det er viktig å kreditere bidragsyterne.

Det er viktig for et prosjekt å få så mange brukere av programmet som mulig for derigjennom å få så mange bidragsytere som mulig til videreutviklingen av programmet.

3.7 Hvilken type programvare er mest egnet for utvikling som åpen programvare?

Hva kan utvikles som åpen programvare, og hva har blitt utviklet som åpen programvare? Dette er et viktig spørsmål, og svaret vil gi klare føringer på hva åpen programvare er egnet til.

Spekteret for programvare går fra infrastrukturprogramvare på den ene siden til sluttbrukerprogramvare på den andre siden. Infrastrukturprogramvare kan være alt fra implementasjoner av ulike tjenerprogramvare, operativsystemkjerner til DNS og TCP/IP. Sluttbrukerprogramvare er i denne sammenhengen programvare som den vanlige ikketekniske bruker benytter som tekstbehandlere, regneark, regnskapsprogrammer og ulike virksomhetsspesifikke systemer.

Tradisjonelt har utviklingen av åpen programvare skjedd på infrastruktursiden av programvarespekteret. Bind, Sendmail og Linux er alle gode eksempler. Andre programmer som Apache og Perl ligger litt mer mot midten, men kan ikke betegnes som ikketeknisk sluttbrukerprogramvare.

Det man kan se, er at åpen programvare ofte er programvare som andre programmer skal kjøres på toppen av, eller verktøy for å utvikle andre programmer. Ifølge Brian Behlendorf, hovedutvikler av Apache, er det flere grunner til at åpen programvare har blitt utviklet på infrastruktur og på tjenersiden av programvarespekteret. Noen av disse er:

- Sluttbrukerapplikasjoner er vanskelige å lage. Det er ikke bare fordi en programmerer må forholde seg til et stadig skiftende ikkestandardisert vindusutviklingsmiljø, men også fordi de fleste utviklere ikke er gode grafiske grensesnittdesignere. God sluttbrukerprogramvare krever gode grafiske brukergrensesnitt. Dette krever igjen testlaboratorier, brukerstudier, etc noe som åpen-programvare-miljøet som regel ikke har tilgang til.
- Kulturelt sett har åpen programvare vært fokusert på nettverksprogrammer og operativsystemer.
- Utviklingen av åpen programvare trives best der inkrementell utvikling er mulig. Historisk har dette vært enklere på tjenersiden enn på klientsiden, siden struktur og arkitektur på programvaren er bedre kjent her, og det er lettere å få et grunnleggende operativt program som man senere kan lage utvidelser av.
- Mye av den åpne programvaren ble skrevet av teknikere for å løse et problem som de hadde mens de utviklet annen programvare. Målgruppen for programvaren var andre teknikere.

For å utvikle et program trenger man god kjennskap til problemområdet, eller en god kravspesifikasjon. Tradisjonelt har ikke åpen-programvare-miljøet hatt kjennskap til behovene til ulike ikketekniske sluttbrukere og heller ikke hatt tilgang til gode kravspesifikasjoner. Måten åpen-programvare-prosjekter starter og programmer utvikles på er heller ikke så godt egnet for brukermedvirkning fra personer som ikke bidrar med kode. Men i en utviklingsgruppe for proprietær programvare er det derimot vanlig å ha med brukere.

Det etterfølgende sitatet er ment å illustrere nødvendigheten av å ha kunnskap om et område for å kunne utvikle gode programmer for det. Sitatet er av Larry Augustin, administrerende direktør (CEO) i VA Linux Systems[17]:

"Open-source developers understand UNIX. This is part of what made it possible to create a better UNIX-Linux. In order to create a better MS Office, open-source developers need to understand MS Office in as mutch detail as they understood UNIX. My fear is that the open-source developer community doesn't understand Office. It can't create what it doesn't understand. What we need is more developers using windows and Office."[5]

"De kan ikke lage noe de ikke forstår". Dette er det sentrale budskapet. Skal utviklere av åpen programvare for eksempel lage et regnskapssystem, må de kunne regnskap og alt regelverk og alle rutiner som er knyttet til dette, samt være gode programmerere.

3.8 Antatte fordeler med åpen programvare

Motivasjonen for å bruke og utvikle åpen programvare er forskjellige. Alt fra filosofiske og etiske til rent praktiske grunner er ofte representert. I dette avsnittet presenteres noen av de antatte fordelene ved bruk av åpen programvare.

Den første fordelen med åpen programvare er at den er gratis og fritt tilgjengelig. Denne fordelen er ikke unik for åpen programvare. Som det er beskrevet i avsnitt 3.3, finnes det mye proprietær programvare gratis tilgjengelig. Det som skiller åpen programvare fra annen programvare, og gir de antatte fordelene, er hvordan den enkelte kan utnytte hovedprinsippene som ligger bak utviklingen av åpen programvare. Her beskrives disse:

- *Tilgjengeligheten av kildekoden og retten til å tilpasse denne.* Tilgjengeligheten av kildekoden gjør det mulig å tilpasse og forbedre programmet for eget bruk. Det gjør det også mulig å lage versjoner for nye maskinvareplattformer og få detaljert kjennskap til hvordan programmet funksjonerer. Trenger man rettelser i programmet, kan man gjøre dette selv hvis man har kunnskapen. Lokale tilpasninger blir mulig, for eksempel å få programmet på sitt eget språk hvis leverandørene ikke lager en slik versjon.

[17] VA Linux systems selger maskiner med ferdig installerte Linux-systemer som er tilpasset kundens spesielle behov. De tilbyr også alle typer konsulenttjenester i forbindelse med Linux. Nettadresse http://www.valinux.com/

- *Mulighetene til å redistribuere rettelser og modifikasjoner*. Det er mulig å lage nye programmer basert på eksisterende kode for siden å distribuere dette programmet fritt. Anser man at det er feil og mangler i programmet, kan man rette disse og siden redistribuere den modifiserte versjonen. Viser det seg at den nye versjonen er bedre, vil denne som regel bli tatt inn i det opprinnelig programmet og det opprinnelige programmet blir bedre.
- *Friheten til å bruke programmet på den måten det passer den enkelte*. Mange gratis programmer har begrensninger på hvordan programmet kan benyttes. Et vanlig eksempel er at programmet ikke kan benyttes for kommersiell bruk. Slike restriksjoner finnes ikke under programvare som distribueres under GPL. Ønsker man å ta i bruk programmet på nye tekniske plattformer, i "embedded systems" eller annet, er dette helt greit.
- *Det finnes ingen ukjente "svarte bokser" i programmet*. I proprietær programvare kan deler av funksjonaliteten være ukjent. Det kan være virkninger i programmet som ikke er dokumentert for omverdenen som de som har laget programmet kan utnytte. Dette er ofte gjort som udokumenterte systemkall.
- *Utviklingen av programvaren er ikke avhengig av kun én virksomhet.* Hvis utvikleren av programmet ikke ønsker å lage nye versjoner, eller ikke ønsker å videreutvikle for din plattform, har du mulighetene til å få andre firmaer til å overta eller overta selv. I tilfellet Linux finnes det mange leverandører av det samme produktet som hver på sin måte videreutvikler det og tar det beste fra hva de andre gjør. Utviklingen og vedlikeholdet av programmet er ikke avhengig av ett firma, og man minsker dermed leverandøravhengigheten.

3.9 Antatte ulemper med åpen programvare

Her er noen antatte ulemper knyttet til åpen programvare:

- *Det finnes ingen garanti for at det vil skje noen utvikling av en spesiell programvare eller en videreutvikling av programvaren i ønsket retning.* Alle programvareprosjekter trenger menneskelige eller økonomiske ressurser for å komme i gang. Man vet aldri om produktet vil komme så langt at det kan benyttes, eller om det er nok interesse for produktet. Dette gjelder alle typer programvareutvikling, men usikkerheten er vesentlig større når det gjelder åpen-programvare-prosjekter.

  I alle programvareprosjekter er det en periode fra ideen blir utformet og arbeidet starter til det foreligger noe konkret som kan benyttes. Dette er den skjøreste fasen i et åpen-programvare-prosjekt. Særlig hvis et åpen-programvare-prosjekt blir startet uten sterk støtte fra en eller flere virksomheter, kan det være vanskelig å få koden opp på et slikt nivå at den tiltrekker seg andre programmerere som kan bruke og utvikle produktet videre. Er det ikke mulig å få nok støtte eller nok programmerere interessert, vil prosjektet bare dø eller sakte forsvinne.

- *Det er av og til vanskelig å vite om et spesielt åpen-programvare-prosjekt eksisterer og finne ut hvor langt det har kommet.* Det skjer liten

markedsføring av åpen programvare, så det er vanskelig å finne ut om det finnes åpen programvare som dekker dine behov.

- *Problemer knyttet til patentrettigheter og intellektuelle rettigheter.* Opphavsrettigheter, eller nærmere bestemt brudd på opphavsrettigheter, kan være et stort problem for åpen programvare. Det er vanskelig å vite om en bestemt algoritme eller måte å løse et problem på er patentert. Patenter på programvare er et problem for alle som utvikler programvare, men den rammer åpen programvare mer enn annen programvare av følgende grunner: Tilgjengeligheten av kildekode gjør det lettere å oppdage brudd på opphavsrettigheter. De fleste åpne-programvare-prosjekter har ikke finansielle krefter til å skaffe seg rettslig hjelp i tvister som gjelder brudd på opphavsrett.

4 Linux

Som beskrevet i avsnitt 3.5.1 er Linux, eller mer korrekt GNU/Linux, et UNIX-liknende operativsystem. Linux kommer i flere distribusjoner fra ulike leverandører og er også fritt tilgjengelig på Internett.

Dette kapitlet vil beskrive Linux i mer detalj, og fortelle om ulike distribusjoner av Linux og om programvare som er tilgjengelig for Linux. Kapitlet vil til dels være teknisk orientert og forutsetter noe teknisk kunnskap hos leseren.

4.1 Hva er en Linuxdistribusjon?

En Linuxdistribusjon er en sammensetning av mange ulike progravaremoduler, kalt pakker, som til sammen utgjør et komplett operativsystem. Det vi får når vi kjøper en distribusjon fra et av selskapene som distribuerer Linux, er i realiteten en rekke pakker som de har satt sammen for oss. RedHat Linux består av over 400 ulike pakker fra mange forskjellige utviklingsprosjekter.

En Linuxdistribusjon er mer enn bare et operativsystem, det er en samling av åpen-programvare-komponenter som gjør at brukerne kan utnytte operativsystemet bedre og mer bekvemt. Blant komponenter som følger med er grafiske brukergrensesnitt, kompilatorer, tekstbehandlere og ulik tjenerprogramvare. Alt dette er åpen programvare. I tillegg er det ofte med proprietær nytteprogramvare som kontorstøtte-pakker og spill.

Pakkingen av Linux sammen med de ofte medfølgende installasjonsprogrammene er en hovedgrunn til at man kjøper en distribusjon og ikke laster ned og setter det sammen selv.

4.2 Linux-kjernen

Den sentrale komponenten i Linux er kjernen. Skal man være presis, så er det kjernen som er Linux. Kjernen er det systemet som kontrollerer kommunikasjonen (eller grensesnittet) mellom andre programmer og maskinvaren.

Kjernen i et operativsystem sørger blant annet for filhåndtering, I/O, signalering, separering av prosesser, mottak av avbrudd-meldinger fra de ulike maskinvaredelene og kommunikasjonen mellom andre deler av operativsystemet og maskinvaren.

4.3 Designmål for Linux[18]

Linux ble utviklet for Intel 386 plattformen, og var i utgangspunktet ikke designet med portabilitet for øye. Linuxkjernen er ikke skrevet for å være portabel, men ble designet for å være effektivt.

Linuxkjernen ble til dels utviklet ved å se på fellestrekk ved eksisterende maskinvarearkitektur, og designe operativsystemet ut fra hva som var felles for disse arkitekturene. Kjernen er hva man betegner en monolittisk kjerne, og ikke en mikrokjerne, som var hovedtrenden da Linux ble utviklet.

Ved videreutviklingen av Linuxkjernen har Linus Torvalds lagt til grunn noen få prinsipper.

- Utgangspunktet er at man skal bruke sunne generelle designprinsipper istedenfor å designe for portabilitet. Bruken av god effektiv design og kode vil også føre til at systemet blir portabelt.
- Man skal så langt som mulig unngå programmerbare grensesnitt til kjernen. Grunnen til dette er at med en gang man får et grensesnitt inn mot kjernen, må dette opprettholdes i nye versjoner fordi folk begynner å programmere mot dette.
- Man skal være forsiktig med å tilføre nye egenskaper til systemet. Nye egenskaper må nøye vurderes før de tas inn i kjernen, siden kjernen påvirker hele resten av systemet.
- Kjernen må lages så modulær som mulig. Dette er av flere grunner. En grunn er at modulære systemer er lettere å tilpasse spesifikke maskinvarearkitekturer siden man ikke trenger å endre hele systemet. En annen grunn er at folk kan arbeide uavhengig av hverandre på forskjellige deler av kjernen.

4.4 Grafisk brukergrensesnitt

Linux er originalt uten et grafisk brukergrensesnitt, men det finnes flere grafiske brukergrensesnitt som kan brukes sammen med kjernen for å utgjøre et komplett moderne operativsystem. Linux har over 10 ulike grafiske brukergrensesnitt. Disse kalles vindushåndteringssystem (window managers).

De grafiske brukergrensesnittene kjører på toppen av XFree86, en fri utgave av X Windows System. X-systemet er infrastrukturen for alle Unix-liknende vindushåndteringssystemer. X-systemet er meget konfigurerbart, og man kan i stor grad selv bestemme hvordan brukergrensesnittet skal framstå. Det er mulig å få et X-system til å etterlikne både Windows 95/98 og Macintosh.

De to for tiden mest populære brukergrensesnittene for Linux er KDE (the K Desktop Enviroment) og GNOME (the GNU Network Object Model Enviroment). Alle Linuxdistribusjoner kommer med ett eller flere grafiske brukergrensesnitt.

[18] Teksten i avsnittet er en bearbeidet utdrag av et essay Linus Torvalds har i boken Open Sources

4.5 Kompilatorer

For å kunne tilpasse kildekoden og lage en egen versjon trenger man en kompilator som oversetter fra kildekode til maskinkode. Jo bedre kompilatoren er, desto lettere er det å lage tilpassende versjoner, eller versjoner for andre maskinvareplattformer. Linux er utviklet i programmeringsspråket C. Det følger med kompilatorer i alle distribusjoner av Linux. Den vanligste kompilatoren er gcc (GNU C Compiler).

4.6 Programvareutvikling

Linux er en utmerket plattform for programvareutvikling. Linuxdistribusjoner kommer som standard med en C og C++ kompilator og assembler. I tillegg finnes det utviklingsverktøy for nyere språk som Perl og Python. Grunnen til at det legges så stor vekt på programvare for programvareutvikling, er at muligheten til å tilpasse programvare er et av de store fortrinnene ved Linux. Uten gode programutviklingsverktøy ville tilgjengeligheten av kildekoden og muligheten til å endre koden hatt liten praktisk nytte.

4.7 Lisensbetingelsene for Linux

Linux er utviklet, og distribueres, under lisensen GNU General Public License (GPL). Se for øvrig avsnitt 3.4.

Linux er åpen programvare, men alle som ønsker det, kan sette sammen en distribusjon og selge denne. En forutsetning er selvfølgelig at lisensbetingelsene i GPL overholdes. Linux-distributører kan ikke begrense betingelsene i GPL eller videredistribusjon på noen måte. Skal du installere Linux på flere maskiner, er det derfor nok å kjøpe en pakke (CD) som kan benyttes for å installere Linux på alle maskinene. Det er ett unntak fra denne regelen. For å skape merverdi for sin egen distribusjon, legger noen leverandører med kommersiell programvare i pakken. Den kommersielle programvaren kan ha begrensninger på hvor mange maskiner de kan installeres på. Dette bør komme klart fram i selve pakken som distribueres.

4.8 Ulike distribusjoner

Vi har sett på fire ulike Linuxdistribusjoner, og vurdert ulike egenskaper ved distribusjonene. Distribusjonene er forskjellige både når det gjelder pris og medfølgende programvare. Vårt mål har ikke vært å kåre noen vinner, men å se på om de ulike distribusjonene bidrar med noe ekstra til Linux.

Her er en oppsummering av noen distribusjoners egenskaper:

Distribusjon/ Egenskaper	Caldera	Corel	RedHat	SuSE
Pris	kr 345	$ 89 + $ 63 frakt kr 1260	$ 80 + $ 38 frakt kr 977	kr 345
Type	Caldera Open Linux 2.3 Standard	Corel Linux OS Deluxe	RedHat Linux 6.1 Deluxe	SuSE Linux 6.4
Anskaffelse	Nett – via Akademika	Nett - direkte fra Corel	Nett – direkte fra RedHat	Nett – via Akademika
Tid for levering	2 dager	7 dager	5 dager	2 dager
Første gangs installasjon -	Partition Magic, Lizard	Corel Install Express	linuxconf	YAST 2
Installasjon av nye pakker	COAS	Corel package installer	RPM	RPM, SaX
Medfølgende tjener-programvare	Apache, Samba, Sendmail, DB2	Apache, Samba,	Apache, Samba, Sendmail, MySQL	Apache, Samba, Sendmail, MySQL
Medfølgende kontorstøtte-programmer	Star Office, Applixware, Wordperfect (prøve)	WordPerfect	Star Office	Star Office, Applixware, Wordperfect (prøve)
Drift og admin-istrasjon	COAS			
Brukerstøtte inkludert	90 dager e-post	30 dager e-post og tlf	90 dager web 30 dager tlf	60 dager e-post og tlf
Annen brukerstøtte tilgjengelig	Telefon mot en avgift			

5 Økonomien ved åpen programvare

Kapitlet vil både beskrive hvordan man kan tjene penger på åpen programvare, hvordan det offentlige kan finansiere utviklingen av åpen programvare og om bruk av åpen programvare kan redusere de totale IT-kostnadene.

5.1 Forretningsmodeller for åpen programvare

Et interessant spørsmål er hvordan det er mulig å tjene penger på åpen programvare. Programvaren er fritt tilgjengelig og kan lastes ned gratis. Dermed må man tjene penger på en annen måte enn å selge selve den åpne programvaren. Avsnittet vil beskrive ulike forretningsmodeller for å tjene penger på åpen programvare

5.1.1 Skape et merkenavn og distribuere en pakke av åpen programvare

Dette er den mest synlige måten å tjene penger på. Bedrifter som RedHat og SuSE setter sammen distribusjoner av Linux og selger disse pakkene sammen med begrenset brukerstøtte og trykte brukerveiledninger (bøker). Brukeren betaler ikke for selve programvaren, men for sammenstillingen av de ulike programmene, brukerstøtte og for pakkingen og distribusjonen. Kjøperen forholder seg til et merkenavn og kjøper et produkt fra dette firmaet istedenfor å kjøpe et liknende produkt uten merkenavnet.

5.1.2 Selge åpen programvare som man selv er hovedutvikler av

Som for avsnittet over er det et slags merkenavn som selv er produktet. Som et tillegg her er firmaet som selger den åpne programvaren, også hovedutvikler av programvaren. Dermed er det dette firmaet som har den beste kompetansen på produktet, og de har en stor fordel når de skal selge tjenester som brukerstøtte og spesialtilpasninger, siden de kjenner kildekoden best. En ulempe her er at man bruker mye ressurser på å utvikle selve programvaren, uten å få direkte tilbakebetalt disse kostnadene. Poenget er å ha bedre kunnskap enn konkurrentene og alltid være noen måneder foran når det gjelder nye finesser i koden. Cygnus, hovedutvikleren av gcc (Gnu C Compiler) kompilatoren, er et godt eksempel på en virksomhet som lever av å ha bedre kompetanse enn sine konkurrenter.

5.1.3 Proprietære produkter som øker verdien av den åpne programvaren

Mange virksomheter lager både åpen programvare og proprietær programvare som er basert på den åpne programvaren. Den åpne programvaren er beregnet på forbrukermarkedet, mens den proprietære programvaren er ment for forretningsmarkedet. Et eksempel på dette er Sendmail. Sendmail har en proprietær utgave der ny funksjonalitet kommer 4–8 måneder før den samme funksjonaliteten i den åpne utgaven.

5.1.4 Konsulenttjenester

Dette er en tradisjonell virksomhet. Gi bort produktet og selge tjenester som opplæring, drift og vedlikehold. Alle distributørene av Linux baserer seg på at de skal få en stor del av sine inntekter på salg av tjenester. Dette er også et av de viktigste argumentene som blir brukt for åpen programvare. IT-selskaper må fokusere på å tilby gode tjenester til brukerne når de ikke kan basere sin eksistens på inntekter fra lisenser, og tjenestene til brukerne vil dermed bli bedre[19].

5.1.5 Selge tilbehør

Dette kan virke trivielt, men salg av "tilbehør" er viktig for markedet rundt åpen programvare. Forlaget O'Reilly støtter utviklingen av åpen programvare ved å lønne sentrale utviklere av åpen programvare. De tjener inn dette på salg av bøker om Linux, Perl og annet som de utgir. Andre i denne kategorien er de som selger maskinvare med åpen programvare preinstallert, eller som selger kommersiell programvare for Linux. Som en kuriositet kan vi nevne at det er et stort marked for ulike former for maskoter knyttet til Linux.

5.2 Finansiering av åpen-programvare-utvikling

Dette avsnittet vil omhandle hvordan man kan finansiere utviklingen av åpen programvare. Avsnittet vil være et grunnlag for de forslagene rapporten framsetter på hvordan det offentlige kan støtte opp om utviklingen av åpen programvare, se kapittel 10. Avsnittet er i hovedsak basert på den EU-sponsede rapporten om åpen programvare[4].

5.2.1 Eksternt finansiert utvikling

Her blir utviklingen av åpen programvare betalt av en ekstern virksomhet. Det kan være flere grunner til at noen er villige til å finansiere utvikling av åpen programvare. Flere av bidragsyterne er virksomheter som lever av åpen programvare på en eller annen måte, se avsnitt 5.1.

En viktig bidragsyter er det offentlige, som gjennom midler til forskning og utvikling kan sponse utviklingen av ny programvare for spesielle anvendelsesområder. I Norge vil dette typisk være Norges Forskningsråd og Statens Nærings- og distriktsutviklingsfond.

En annen bidragsyter er selskaper som selger tilhørende produkter, og som vil tjene på utviklingen av en spesiell åpen programvare. På Linux finnes det en del proprietær programvare (spesielt kontorstøtteprogramvare) som har behov for drivere eller annen programvare for å virke korrekt på Linux. Disse betaler gjerne for utviklingen av denne åpne programvaren.

[19] Vi har til nå ikke sett dokumentasjon på at tjenestene til brukerne er blitt bedre eller dårligere fordi brukeren benytter åpen programvare

5.2.2 Internt finansiert utvikling

Virksomheter som selger åpen programvare eller tjenester direkte knyttet til åpen programvare, vil støtte utviklingen av åpen programvare for å øke markedet for denne typen produkter. Er de gode, vil et større marked bety større muligheter for å tjene penger. Alle de store distributørene av Linux bidrar på denne måten. Ved selv å betale for utviklingen av produktet håper man på å skape et større marked og derigjennom større muligheter for inntjening.

5.3 Åpen programvares betydning for de totale IT-kostnadene

Har åpen programvare noen vesentlig betydning for en virksomhets totale IT-kostnader? Dette spørsmålet er en av grunnene til at Statskonsult har vurdert åpen programvare. Det er umulig å svare generelt på dette, for alle virksomheter har ulike forutsetninger ved kjøp av nytt IT-utstyr. Det vi vil gi er en generell beskrivelse av hvordan man kan vurdere sine egne totale IT-kostander og fokusere på de områdene der det er forskjeller mellom åpen og proprietær programvare.

Vi har ikke vurdert nytteverdien eller egnetheten av IT-anskaffelsen for en virksomhet. Dett er noe alle virksomheter bør gjøre, men er umulig å gjøre på et generelt grunnlag. Det vi ser på her, er kostnadssiden i en kost/nyttevurdering.

Med totale IT-kostnader mener vi her kostnadene for anskaffelse og bruk av et bestemt IT-system i hele tiden IT-systemet er i bruk. Vi har delt ”livsløpet” inn i flere perioder som informasjonsinnhenting, anskaffelse, utrulling og igangsetting, opplæring, daglig bruk og vedlikehold og avhending og overgang til ny teknologi. Alle disse fasene vil bli omtalt nærmere.

- *Informasjonsinnhenting*. Denne fasen innebærer å skaffe seg informasjon om produktet man senere skal kjøpe. Her kan det være vanskeligere å finne informasjon om åpen programvare, siden distributørene som regel ikke har noen egen markedsavdeling. Informasjonen som finnes, kan være mer korrekt enn for proprietær programvare siden åpen-programvare-virksomheter i hovedsak ikke tjener pengene sine på salg av produktet. Dessuten er koden tilgjengelig for inspeksjon.

  Offentlig sektor kan gjennom sine innkjøpsordninger sørge for at informasjon om åpen programvare er like tilgjengelig som informasjon om annen type programvare.

- *Anskaffelse*. Anskaffelse i offentlig sektor må følge regelverket for offentlig anskaffelse. Dette regelverket skal i utgangspunktet stille alle potensielle leverandører likt. Vurderer man en anskaffelse kun ut fra pris, vil åpen programvare selvfølgelig komme best ut, siden denne har en meget lav initiell kostnad knyttet til seg. Det er verdt å merke seg at en del programvare kommer sammen med maskinvare eller er en oppgradering av

eksisterende programvare, så det er umulig å sette opp et regnestykke som er dekkende for alle virksomheter selv på dette isolerte området.

- *Utrulling og igangsetting*. Dette omfatter installasjon og tilrettelegging av systemet, slik at det er klart til bruk i hele virksomheten. Store virksomheter som selger proprietær programvare, er i dag bedre rustet for dette enn virksomheter som selger åpen programvare, fordi de tradisjonelt har hatt et større fokus på denne fasen. Skal store virksomheter benytte åpen programvare må selgerne av åpen programvare etablere et apparat som er i stand til å håndtere utrullingen i stor skala.

- *Opplæring*. Denne kostnaden bør være uavhengig av om det er et åpen-programvare-system eller ikke. Per i dag er det et større utvalg av opplæringstilbud for proprietær programvare enn det er for åpen programvare. Prisen er ikke forskjellig, men det er lettere å få et tilbud som passer den enkelte virksomhet med proprietær programvare. Det er også helt klart flere brukere som er kjent med eksisterende proprietær programvare, enn med åpen programvare, så det vil være kostnader forbundet med opplæring hvis man skal bytte system.

- *Daglig bruk og vedlikehold*. Dette er ofte den fasen som utgjør den største kostnaden knyttet til de totale IT-kostnadene. Det hevdes fra åpen-programvare-miljøet at åpen programvare vil gjøre vedlikehold billigere, men vi har ikke funnet noen dokumentasjon på dette så langt.

- *Avhending og overgang til et nytt system.* Nye IT-systemer blir ofte bygd for å erstatte eksisterende. Har man tilgang til kildekoden (ikke nødvendigvis åpen kildekode), vil ofte overgangen til et nytt system bli enklere, siden man har et bedre oversikt over hvordan det forrige systemet er implementert. Det som er viktig i denne fasen, er at systemet er mulig å dokumentere å at man kan ta med seg dataene fra det gamle systemet over til det nye.

Ved en vurdering av de ulike fasene i et IT-systems livsløp er det kun anskaffelsesfasen som klart skiller seg ut til fordel for åpen programvare. En stor kostnad forbundet med overgang til åpen programvare er opplæring av brukere i det nye systemet. For de andre fasene er det vanskelig å finne noen store forskjeller i antatte kostnader knyttet til bruken av åpen programvare.

Vår konklusjon på spørsmålet om det offentlige kan kutte kostnader ved en overgang til åpen programvare, er som følger: Det vil være kostnadsbesparende å benytte åpen programvare i virksomheter der de totale IT-kostnadene i hovedsak er knyttet til kjøp av lisenser, og kostnadene ved opplæring av brukerne på ”ny” programvare er relativt liten.

6 Eksempler på bruk av åpen programvare i forvaltningen

Utover bruken av Linux og Apache som for web-tjenester har vi ikke så mange eksempler på bevisst bruk av åpen programvare i forvaltningen. De vi har kommer fra utdanningssektoren.

6.1 Universitetet i Oslo

Universitetet i Oslo benytter åpen programvare på mange måter. Flere fakulteter har Linux tilgjengelig for sluttbrukerne. Mange har Linux som tjenermaskiner der de kjører til dels avanserte tekniske applikasjoner. Apache er utbredt som webtjener, og infrastrukturprogramvare som Sendmail og Bind er dominerende.

Det forannevnte er helt vanlig bruk av åpen programvare. En annen bruk av Linux som til nå ikke har vært så vanlig, men som er svært interessant, er å benytte Linux som en tynn klient. Dette betyr at Linux kun benyttes for å håndtere brukergrensesnittene fra programmer som kjøres på tjenermaskinene.

All brukerprogramvaren finnes på tjenermaskinen. Dette er både vanlig kontorstøtteprogramvare og mer faglig programvare. På tjenermaskinene kjører man Windows eller NT. Klienten benyttes kun for å kommunisere med tjenermaskinen og vise fram skjermbilder. Brukeren ser de vanlige Windowsskjermbildene, men det er et Linux operativsystem som benyttes for å vise fram skjermbildene. På denne måten kan man unngå å installere Windows på alle PC-ene som finnes ute hos sluttbrukerne.

Her kan man spare en del lisenskostnader ved å bytte ut Windows med Linux og samtidig beholde den gamle programvaren. Ulempen, eller kanskje fordelen, er at man går over til et miljø med tynne klienter der sluttbrukeren mister den direkte kontrollen over hva som skal finnes på egen maskin.

I miljøer der man ønsker å innføre tynne klienter til sine brukere og ikke la brukerne ha direkte tilgang til lokal programvare, kan det være fornuftig å bytte ut Windows med Linux på alle klientene. Bruk av Linux i miljøer der man ønsker å innføre tynne klienter, er økende i universitets- og høyskolemiljøer.

6.2 Høle barne- og ungdomsskole

Høle barne- og ungdomsskole har installert Linux på gamle PC-er som de har fått fra næringslivet. PC-ene kommer uten operativsystem, og dette må installeres. Ved å velge Linux som operativsystem har de redusert kostnadene til lisenser.

7 Kommersielle tjenester for åpen programvare i Norge

7.1 Kjøp av programvare

Kjøp av åpen programvare i Norge er enkelt. Vi bestilte programvaren fra Akademika og fikk den levert to dager senere på døren. Det fulgte med faktura som kunne behandles på vanlig måte. Det finnes flere butikker i Norge som selger åpen programvare via nettet, og flere butikker som selger Linuxdistribusjoner over disk.

Det er også mulig å bestille Linuxdisitribusjoner direkte fra leverandørene. Dette tar lenger tid, og med de ekstra kostnadene knyttet til frakt og tollbehandling er det billigere å kjøpe fra leverandører i Norge. Fordelen er at man kan få versjoner som ikke er tilgjengelig i Norge, og at man kan få de nyeste versjonene tidligere.

Når det gjelder kommersiell programvare for Linux, selges dette på lik linje med annen kommersiell programvare. Store aktører som IBM, SGI og Oracle har alle versjoner av sin programvare tilgjengelig på Linux. Disse virksomhetene har også et støtteapparat for å betjene de kundene som velger å kjøre deres programvare på Linux.

7.2 Kjøp av maskiner med ferdig installert programvare

Flere store og små leverandører selger maskiner med ferdig installert åpen programvare. Dette gjelder i hovedsak maskiner med Linux, der man i tillegg kan få tjenerprogramvare som Apache og Sendmail ferdig konfigurert på maskinene.

7.3 Konsulenttjenester

Konsulenttjenester tilbys både av rene konsulentselskaper og av de store kommersielle programvareleverandørene. Blant konsulentselskapene finnes det både de store internasjonale som dekker ”alt”, og små nisjeselskaper som kun fokuserer på Linux med tilhørende programvare.

7.4 Kurs og opplæring

Kurs og opplæring på Linux og annen åpen programvare tilbys av stadig flere. Kursene holdes gjerne av virksomheter som selger programvare for Linux/Unix, og av universiteter og høyskoler. Tilbudet av kurs og opplæring er ennå ikke så omfattende som for Windows-plattformen.

8 Bruk av åpen programvare som standardiserings- og utviklingstiltak

Standarder og standardisering er en viktig del av IT-utviklingen. Åpen programvare har en rolle å spille i det framtidige arbeidet innenfor standardisering. Noen arenaer innenfor IT-standardisering, som IETF, reflekterer kulturen til åpen programvare, mens andre områder ønsker å benytte åpen programvare som et virkemiddel for spredning og bedre interoperabilitet av standarder.

8.1 IETF i åpen programvaresammenheng

Måten IETF (Internet Engineering Task Force) er organisert og fungerer på er et godt eksempel på ”Basar”-utviklingsprinsippene framsatt av Eric Raymond [2].

IETF definerer åpne standarder for grunnleggende protokoller på Internett samt standarder for ulike fellesanvendelser som sikkerhet, transporttjenester og administrasjon. Alle IETF-standarder er fritt og gratis tilgjengelig på Internett.

For at en vedtatt IETF-standard skal fortsette å være en standard, må det innen et gitt tidsrom minst finnes minst to ulike implementasjoner av standarden som virker sammen. I denne sammenhengen er åpen programvare viktig for å få spredd implementasjoner av standarden og for å kunne teste ulike produkter mot hverandre.

IETF er åpen for alle, og du er ”medlem” gjennom å delta på diskusjonslister og møter. Alle opptrer som enkeltpersoner og ikke som deltakere for en virksomhet eller offentlig myndighet.

Både standardiseringsprosessen og standardene baseres på åpenhet.

8.2 Internett og åpen-programvare-referanseimplementasjoner

En referanseimplementasjon av en standard er en implementasjon som andre viser til, eller tester mot, når de skal lage interoperable produkter basert på denne standarden. En referanseimplementasjon bør implementere alle egenskapene til standarden og gjøre dette på en korrekt måte. Apache er et eksempel på en referanseimplementasjon av HTTP (standarden). Nettlesere må kunne kommunisere med Apache for at de kan hevde at de er en korrekt implementasjon av standarden.

Noe av suksessen til Internett er at produktene basert på Internett-standardene er interoperable. Vi får dermed et samvirkende hele. De fleste av de grunnleggende protokollene for Internett finnes som åpen-programvare-referanseimplementasjoner. Referanseimplementasjonene har ofte en tidlig

versjon og har en dominerende markedsandel, og de øvrige produktene må være interoperable med referanseimplementasjonen for å få innpass på markedet. Eksempler er TCP/IP, DNS, SMTP og HTTP.

8.3 Åpen programvare som katalysator for utbredelse av nye standarder

For å få utviklet produkter basert på standarder er det viktig at det finnes implementasjoner av standarden som man kan bygge videre på. Her er det viktig med åpen programvare. Finnes standarden implementert som en åpen-programvare-referanseimplementasjon kan alle benytte denne implementasjonen av standarden for å utvikle nye produkter. Finnes det kun proprietære implementasjoner av standarden, må folk som ønsker å utvikle nye produkter basert på standarden, først implementere sin egen versjon av standarden, eller kjøpe rettigheter til å bruke en proprietær utgave. Denne vil være et hinder for nyutvikling og kan sakke prosessen med å få tatt i bruk nye standarder.

I forbindelse med utviklingen av nye standarder for å støtte implementasjonen av EU-direktivet om elektroniske signaturer har EU-kommisjonen bedt om å få laget åpne referanseimplementasjoner av noen av de allerede vedtatte standardene fra ETSI. Dette for å lette produktutviklingen og forskningen på området.

I Norge kan Norges forskningsråd og Statens nærings- og distriktsutviklingsfond være mulige aktører for å fremme bruken av åpen programvare i utviklingsarbeid. Norges standardiseringsforbund kan være en aktør for å fremme åpen programvare innenfor arbeidet med IT-standardisering.

9 Vurdering av bruksområdene til åpen programvare

9.1 Vurderingskriterier for åpen programvare

Vi har skilt mellom åpen programvare som er ment for tjenermaskiner, og for infrastruktur og åpen programvare som primært er tenkt brukt av vanlige sluttbrukere når vi har valgt vurderingskriterier.

Vi har lagt vekt på følgende ting når det gjelder vurdering av åpen programvare for tjenermaskiner og infrastruktur:

- *Anskaffelse.* Programvaren skal være enkel å anskaffe.
- *Installasjon.* Det skal være lett å installere programvaren, både første gang og ved senere oppgraderinger.
- *Support.* Det skal finnes profesjonell support på programvaren.
- *Administrasjon og drift av systemet.*
  - Fleksibilitet, enkelhet og muligheter for tilpassing
  - Kompatibilitet med andre systemer
  - Markedsposisjon
  - Driftskostnader
- *Standarder.* Følger programmet gjeldende standarder på området?
- *Opplæring.* Det må finnes muligheter for opplæring i systemet.
- *Anvendbarhet for offentlig sektor.* Hvor anvendelig er programvaren for offentlig sektor?

Vi har lagt vekt på følgende ting når det gjelder vurdering av åpen programvare for sluttbrukere (operativsystem med programmer som tekstbehandler og e-post):

- *Anskaffelse.* Programvaren skal være rimelig og enkel å anskaffe.
- *Installasjon.* Det skal være lett å installere programvaren, både første gang og senere oppgraderinger. Nye programmer skal kunne installeres og fjernes enkelt.
- *Brukerstøtte og support.* Det må være mulig å få brukerstøtte for den installerte programvaren.
- *Opplæring.* Det må finnes muligheter for opplæring i bruk av de ulike programmene.
- *Brukervennlighet.* Programmene må være like ”brukervennlige” som tilsvarende kommersiell programvare. For vanlige kontorstøtteprogrammer vil Microsoft Office være et naturlig program å sammenlikne med.
- *Språk.* Programmene må være tilgjengelige på norsk.
- *Driftskostnader.* Kostnader for drift av programvaren over tid.
- *Anvendbarhet for offentlig sektor.* Hvor anvendelig er programvaren for offentlig sektor?

Valg av kriterier er gjort ut i fra hva vi anser som mest sentralt og dekker alle faser av anskaffelse og bruk av programvare. Ut fra kriteriene skal det være mulig å gjennomføre en analyse av de totale kostnadene for anskaffelse og bruk av programvaren[20].

9.2 Hvilke deler av forvaltningen er best egnet for bruk av åpen programvare?

Forvaltningen er svært variert. Ikke alle deler er like egnet for å ta i bruk åpen programvare. Etater og sektorer med store, tunge egenutviklede programmer eller der man er avhengig av egne saksbehandlingssystemer er ikke så velegnet for å ta i bruk åpen programvare fordi 1) det ikke finnes den typen åpen programvare, og 2) den typen programvare ikke er egnet til å utvikle som åpen programvare.

Områder der hovedbruken er en kombinasjon av tekstbehandling, e-post og Internett, kan være velegnet for bruk av åpen programvare. I denne kategorien finner vi blant annet mange departementer, noen direktorater og store deler av utdannelsessektoren. Alle disse vil kunne ta i bruk en eller annen form av åpen programvare. Utdanningssektoren har store variasjoner i den programvaren som blir benyttet, mens direktorater og departementer i stor grad benytter den samme typen programvare[21]. Evnen til å ta i bruk ny programvare virker dermed større i utdanningssektoren enn i departementene og direktoratene. I tillegg har utdanningssektoren som formål å lære opp sine brukere, noe som er en forutsetning for å ta i bruk nye typer av programvare. På høyskoler og universiteter finnes det også betydelig IT-kompetanse blant undervisningspersonalet. Sett under ett er utdanningssektoren den sektoren der det er mest fornuftig å starte for å ta i bruk åpen programvare i forvaltningen.

Vår vurdering er at åpen sluttbrukerprogramvare ikke per i dag kan erstatte de kommersielle alternativene når det gjelder funksjonalitet og integrasjonsmuligheter.

9.3 Hvilken type åpen programvare passer i forvaltningen?

Som med annen programvare er det store variasjoner i brukervennligheten og anvendeligheten til åpen programvare. Vi har i forbindelse med denne vurderingen sett på programvare for sluttbrukere og tjenerprogramvare. Programvare for sluttbrukere omfatter det grafiske brukermiljøet, tekstbehandling, regneark, nettleser og e-postklient. Tjenerprogramvare omfatter programvare for e-post, etablering av nettsider og vanlig filbehandling.

[20] Dette omtales gjerne som total cost of ownership (TCO) for IT-produkter.
[21] Dette kan leses ut av bakgrunnstallene til rapporten ”IT i staten 1999”.

9.3.1 Sluttbrukerprogramvare

Det finnes flere gode tekstbehandlere og regneark for Linux. Ulempen er at disse ikke er åpen programvare, og man dermed fremdeles har de samme ulempene som annen proprietær programvare har. De tekstbehandlerne og regnearkene som finnes som åpen programvare, mangler dessverre den helheten og de integrasjonene som de proprietære produktene har. Når det gjelder nettlesere og e-postklienter finnes mange av de samme produktene tilgjengelig både på Linux og Windows. Ikke alle av disse produktene finnes som åpen programvare, men det er ofte en lav eller ingen lisenskostnad knyttet til bruken av produktene.

Per i dag mangler tekstbehandlere, regneark og e-postklienter utviklet som åpen programvare den helheten og funksjonaliteten som finnes hos de kommersielle produktene. Tatt i betraktning at det for de fleste brukere vil medføre opplæring og omstilling å gå over til nye produkter, bør de nye produktene ha tilnærmet samme funksjonalitet som det produktet man skifter fra. Pris blir her mindre viktig for en sluttbruker. I situasjoner der man skal læres opp og man ikke har gjort seg avhengig av funksjonaliteten til de proprietære produktene, er åpen-programvare-alternativene velegnet. Vår vurdering er at åpen programvare beregnet på sluttbrukere vil egne seg best i utdanningssektoren og hos spesielt motiverte og interesserte brukere.

9.3.2 Tjenerprogramvare

Når det gjelder tjenerprogramvare er situasjonen annerledes. Programvare for drift og utvikling av nettsider er i stor utstrekning allerede åpen programvare. I tillegg finnes det flere eksempler på åpen tjenerprogramvare for e-post. På disse områdene er det helt klart et stort potensiale for åpen programvare i forvaltningen. Dessverre er det også på disse områdene man tilbyr rimelig eller gratis proprietær programvare, slik at overgangen til åpen programvare ikke medfører store økonomiske innsparinger. Bruk av åpen programvare kan være fornuftig for å hindre den ensrettingen av programvare som spesielt skjer innenfor departementer og direktorater.

På tjenerprogramvare er den åpne programvaren funksjonelt og administrativt minst like god som de proprietære alternativene. I noen utstrekning kan det mangle support på programvaren fra leverandørene. Dette er imidlertid i ferd med å endres, siden flere store leverandører tilbyr support på åpen tjenerprogramvare som Apache, Samba og sendmail.

Vår vurdering er at Linux er en velegnet tjenermaskin for offentlig forvaltning. Dette gjelder spesielt i kombinasjon med Apache som tjener for egne nettsider.

10 Konklusjoner, tiltak og anbefalinger

10.1 Konklusjoner

Det offentlige er ikke tjent med en for stor ensretting av den programvaren som benyttes. Hver enkelt virksomhet bør så langt som mulig stå fritt til å velge den programvaren som virksomheten finner mest hensiktsmessig for å løse sine oppgaver. Dette krever at virksomhetene har kunnskap om, og kjennskap til, ulike produkter og tjenester. *Åpen programvare er en billig måte folk kan gjøre seg kjent med ny programvare på*.

Åpen programvare er et konsept som forvaltningen bør være bevisst å utnytte. Tilgang til kildekoden betyr at flere kan få kjennskap til den grunnleggende teknologien, og dette kan gjøre det enklere å få spredt og utviklet ny teknologi. *Det er spesielt innenfor utdanning, forskning og utvikling at åpen programvare vil gjøre seg gjeldende*.

Åpen programvare er dominerende når det gjelder implementeringen av Internett-infrastrukturen. Både utviklingen og implementeringen av standarder for Internett er basert på ideene og tankene rundt åpen programvare. *Norske myndigheter bør være seg bevisst de mulighetene åpen programvare gir for en rask spredning av nye standarder i markedet.*

For å utvikle åpen programvare via nettet trenger de involverte en felles forståelse av og innsikt i det som skal utvikles. *Det er derfor velegnet å utvikle tjener- og infrastrukturprogramvare som åpen programvare*. Grunnen er at det her som oftest foreligger klare spesifikasjoner og standarder som man kan bruke når man skal lage produkter. Som regel har de som skal implementere en standard, også vært med på å definere denne. Sluttbruker-programmer er ikke så velegnet å utvikle som åpen programvare, siden det ikke finnes standarder for dette, og at det ikke er så lett å komme opp med en klar spesifikasjon av det som skal lages. Dette gjenspeiler seg også i hvilke programmer som finnes som åpen programvare.

Det er ikke alltid det er den beste teknologien som går seirende ut av en tvekamp. Dette er det viktig å ha med seg. Det er flere faktorer enn de rent tekniske som ligger til grunn når folk skal velge et operativsystem eller andre tekniske apparater. Andre viktige ting som markedsføring av produktet, design, pris, tilgjengelig tilbehør (for operativsystemer vil dette si hvilke programmer som finnes og hvilken maskinvare som støttes), hva du forbinder med produktet og ikke minst hvilke andre er det som bruker det.

Vår vurdering av Linux og åpen programvare blir dermed ikke en ren teknisk vurdering, men en helhetsvurdering der vi også tar hensyn til brukernes erfaringer og de investeringene som allerede er gjort på IT-området i offentlig sektor.

Det er ikke gitt at åpen programvare er billigere enn kommersiell programvare. *Etter våre vurderinger vil det være kostnadsbesparende å benytte åpen programvare i virksomheter der de totale IT-kostnadene i hovedsak er knyttet til kjøp av lisenser, og kostnadene ved opplæring av brukerne på "ny" programvare er relativ liten.*

Linux er i hovedsak et operativsystem som er ment benyttet på tjenermaskiner. Dette gjenspeiler også hvordan de fleste Linuxdistribusjoner blir markedsført og dokumentasjonen som følger med programmene[22].

Linux er en Unix-variant, og det er dermed naturlig at Linux først og fremst vil bli benyttet på de områdene der andre Unix operativsystemer tidligere har blitt brukt. Unix-systemer har i hovedsak blitt benyttet som tjenermaskiner, og vi ser at Linux også i hovedsak blir tatt i bruk som tjenermaskiner. De miljøene der sluttbrukerne har benyttet Unix, vil også være de første miljøene der sluttbrukerne tar i bruk Linux.

Det er spesielt innenfor undervisningssektoren at Linux kan spille en rolle både som tjenermaskiner og som operativsystem for sluttbrukere. Gjennom å oppmuntre til bruk av forskjellig programvare i opplæring kan brukerne få et bredere kunnskapsgrunnlag og derigjennom et bedre grunnlag når de senere skal velge programvare selv.

Linux er i dag velegnet som operativsystem for tjenermaskin, også for offentlig sektor. Vår vurdering er at åpen sluttbrukerprogramvare ikke per i dag kan erstatte de kommersielle alternativene når det gjelder funksjonalitet og integrasjonsmuligheter.

10.2 Mulige tiltak

Det finnes mange mulige tiltak for å få en større utbredelse av åpen programvare i offentlig forvaltning. Dette avsnittet omtaler noen av disse.

- Pålegge bruk av åpen programvare
- Støtte bruken av åpen programvare
- Distribuere og utvikle åpen programvare
- Støtte forskning og utvikling av åpen programvare

10.2.1 Pålegge bruk av åpen programvare

Et kraftig virkemiddel er å påby bruk av åpen programvare i gitte sammenhenger. Pålegg om bruk av en bestemt teknologi vil altid være risikabelt. Er det den rette teknologien man påbyr? Er virksomhetene som

[22] Følgende sitat er hentet fra den trykte dokumentasjonen som følger med Caldera OpenLinux 2.3: "*Today, Linux is first and foremost a server operating system. Although many applications are now appearing that allows Linux to be used as a primary workstation or desktop system, most users of Linux focus on the server capabilities of the operating system.*"

pålegges bruk, klare for å ta teknologien i bruk? Er mottakerne av pålegget motiverte for å ta teknologien i bruk?

I hovedsak bør en være tilbakeholden med å pålegge bruk av en bestemt teknologi, men heller framheve fordeler og ulemper med teknologien, slik at virksomhetene selv kan velge å ta teknologien i bruk.

10.2.2 Støtte bruken av åpen programvare

Med å støtte bruken av åpen programvare mener vi her å tilrettelegge omgivelsene slik at det blir enklest mulig å ta i bruk åpen programvare. Tiltakene kan spenne fra å forenkle anskaffelsesprosessen for åpen programvare gjennom opplæringstilbud for bruk av åpen programvare til hjelp til drifting av åpen programvare.

Ved å tilrettelegge for bruk av åpen programvare kan man stimulere til større bruk av denne typen programvare uten å tvinge noen til å bruke en type programvare de ikke ønsker.

10.2.3 Distribuere og utvikle åpen programvare

Et konkret tiltak for å støtte opp under bruken av åpen programvare er om staten selv distribuerer en egen versjon av Linux med annen tilhørende åpen programvare. Dette vil kunne gjøre det enklere for virksomheter å ta i bruk åpen programvare. På den annen side vil det kunne virke markedsvridende om det offentlige har sin egen Linux-distribusjon. Hva distribusjonen skulle inneholde vil også være et sentralt spørsmål. Et annet spørsmål er om en egen distribusjon vil være kostnadssvarende siden det allerede er gratis å videredistribuere de eksisterende distribusjonene av Linux.

En mellomløsning ville være at noen i det offentlige kopierte opp og distribuerte eksisterende Linuxdistribusjoner, slik at de som ønsket å prøve ut åpen programvare, kunne få denne på en enkel måte ved å henvende seg en offentlig etat.

Det offentlige bør også vurdere om de kan tilby programvare som de i dag har eierrettighetene til, som åpen programvare.

10.2.4 Støtte forskning og utvikling av åpen programvare

Ved å støtte utvikling av åpen programvare kan man bidra til å videreutvikle infrastrukturen samtidig som man gjør teknologien tilgjengelig for alle. Som nevnt tidligere er det en trend at implementasjonen av nye standarder skjer som åpen programvare. Setter man krav fra de som bidrar med penger til forskning og utvikling, at resultatet skal være tilgjengelig som åpen programvare, får man som en bieffekt at resultatet av forskningen enklere kan nyttiggjøres av andre. Vi anser at ved tildeling av forsknings- og utviklingsmidler bør det være et krav at programvare som utvikles, gjøres tilgjengelig som åpen programvare.

10.3 Anbefalinger

- Linux er et produkt som det offentlige bør støtte for å skape et press for videreutvikling av, og som et potensielt alternativ til Microsofts operativsystemer. Linux er i dag best egnet som et tjeneroperativsystem.
- Det offentlige bør støtte utviklingen av åpen programvare for å skape et alternativ til eksisterende programvare. Ny åpen programvare kan skape et press på videreutvikling av eksisterende programvare, og kan hindre en for stor ensretting av den programvaren som benyttes i offentlig sektor. Støtten kan komme gjennom forsknings- og utviklingstiltak.
- Ved tildeling av forsknings- og utviklingsmidler kan det være et krav at programvare som utvikles, gjøres tilgjengelig som åpen programvare.
- Det offentlige bør vurdere nærmere om de kan offentliggjøre kildekode som de har eierrettighetene til, som åpen programvare.
- Det offentlige bør oppmuntre til bruk av Linux og åpen programvare innenfor skole- og utdanningssystemet. Grunnen er at elever og studenter skal få kjennskap til flere ulike produkter, slik at de senere kan velge de produktene de selv ønsker på et bedre grunnlag.
- For å spare lisenskostnader kan brukte PC-er som gis til skoler, utstyres med åpen programvare.
- Infrastrukturen bør baseres på åpne standarder implementert som åpen programvare. Offentlig sektor bør sette krav til bruk av åpne standarder implementert som åpen programvare i den infrastrukturen den benytter. Et område der bruk av åpne standarder og løsninger basert på åpen programvare kan være fornuftig, er ved en eventuell etablering av en infrastruktur for distribusjon og håndtering av digitale sertifikater.

Vedlegg I Referanser

[1] Chris DiBona, Sam Ockman & Mark Stone (Editors): Open Sources – Voices from the Open Source Revolutin.
O'Reilly & Associates, 1999. ISBN 1-56592-582-3

[2] Eric S. Raymond: The Cathedral & the bazaar.
O'Reilly & Associates, 1999. ISBN 1-56592-724-9

[3] The Open Sourve Initiative. The open source definition, 1999.
Tilgjengelig på: http://www.opensource.org/osd.html

[4] Free Software / Open Source: Information Society Opportunities for Europe? Versjon 1.2 April 2000.
Working group on Libre software[23]
Tilgjengelig på: http://eu.conecta.it/paper.pdf

[5] Linux Journal. juni 2000 "The new beginning"

[6] NOU 2000:24 "Et sårbart samfunn"

[7] Open Source The unauthorized white papers. Donald K. Rosenberg
M&T books, 2000. ISBN 0-7645-4660-0

[23] Arbeidgruppen for Libre Software ble satt sammen på initiativ av EU-kommisjonens kommisjonær for området "Information Society".

Vedlegg II Forkortelser

BIND	Berkeley Internet Naming Deamon
BSD	Berkeley Software Distribution
DNS	Domain Name System
ESMTP	Extended Simple Mail Transfer Protocol
ETSI	European Telecommunications Standards Institute
GNOME	the GNU Network Object Model
GNU	Gnu's Not Unix
GPL	GNU General Public License
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IEC	International Electrotechnical Committee
IEEE	The Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IMAP	Internet Message Access Protocol
IP	Internet Protocol
KDE	the K Desktop Enviroment
OSI	Open Systems Interconnection
PKI	Public Key Infrastructure
POP	Post Office Protocol
POSIX	Portable Operating System Interface for unIX
RFC	Request For Comments
RPM	RedHat Package Manager
SMTP	Simple Mail Transfer Protocol
TCO	Total Cost of Ownership
TCP	Transmission Control Protocol

Vedlegg III GPL (GNU General Public Licence)

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee
your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library
General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or
use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to
certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that
they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the
software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is
modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect
on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent
licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General
Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under
copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter,
translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted,
and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program).
Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately
publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty;
and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or
work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a
whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary
way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a
warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program
itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered
independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when
you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose

permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the
distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above
provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source
distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial
distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source
code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a
special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components
(compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the
same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense
or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under
this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its
derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the
Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute
or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You
are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you
(whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot
distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the
Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through
you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole
is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole
purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous
contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he
or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places
the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among
countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to
the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the
option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a
version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For
software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be
guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED
BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE
PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY
SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER
PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING
ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM
(INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone
can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and
each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample;
alter the names:

Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more
useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Return to GNU's home page.

FSF & GNU inquiries & questions to gnu@gnu.org. Other ways to contact the FSF.

Comments on these web pages to webmasters@www.gnu.org, send other questions to gnu@gnu.org.

Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111, USA

Updated: 3 Jan 2000 rms

Vedlegg IV The BSD licence (Berkeley Software Distribution)

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the <ORGANIZATION> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

REFERANSER

Tittel:	Åpen programvare. Egnetheten til Linux og annen åpen programvare for statlig forvaltning
Forfatter(e):	Endre Grøtnes
Statskonsults rapportnummer:	
Prosjektnummer:	43 488
Prosjektnavn:	Åpen Porgramvare
Prosjektleder:	Endre Grøtnes
Oppdragsgiver(e):	Statskonsult
Resymé:	
Arbeidsområde:	☐ Styring og resultatorientering ☐ Omstilling og organisasjonsformer X Informasjonsteknologi ☐ Internasjonalisering ☐ Lederskapsutvikling
Emneord:	Linux, Open Source, Åpen programvare, Åpne standarder
Dato:	2. mars 2001
Sider:	50
Utgiver:	Statskonsult Direktoratet for forvaltningsutvikling Postboks 8115 Dep 0032 OSLO