Is open source software een economisch verantwoord alternatief?

Raphaël LHEUREUX

promotor :
Prof. Jeanne SCHREURS

Eindverhandeling voorgedragen tot het bekomen van de graad
Licentiaat in de toegepaste economische wetenschappen major
integrerend ondernemerschap

universiteit hasselt

Woord vooraf

Deze eindverhandeling vormt het sluitstuk van mijn opleiding tot Master in de Toegepaste Economische Wetenschappen aan de Universiteit Hasselt. De keuze van het onderwerp komt voort uit mijn sterke interesse in informatica en alles wat daarmee gerelateerd is.

Graag wil ik de verschillende personen bedanken die bijgedragen hebben tot de realisatie van deze eindverhandeling. Een speciaal dankwoord gaat naar mijn promotor Professor Jeanne Schreurs voor haar deskundige begeleiding. Daarenboven wil ik ook Michel van der Kleij, de IT-consultant van de onderneming uit de praktijkstudie, bedanken voor de rondleidingen, illustraties en professionele uitleg.

Tot slot een bijzonder woord van dank aan mijn familie, vrienden en vriendin voor hun steun doorheen mijn studies.

Samenvatting

Tegenwoordig proberen steeds meer bestaande ondernemingen te bezuinigen om een betere concurrentiële positie op de markt te bekomen. Ook voor ondernemingen in de opstart-fase wordt het steeds belangrijker om zo snel mogelijk 'break-even' te draaien of winst te maken. Dit laatste is eveneens nauw gerelateerd aan kostenbeperking. Uit de literatuur blijkt dat de informatica-afdeling heel vaak een doelwit is bij bezuinigingen. Hier worden immers relatief hoge kosten veroorzaakt, niet alleen door de aanschaf van hardware, maar ook door de software die gebruikt wordt door de informatici en het personeel van de onderneming.

Steeds vaker komen we de term 'Open Source Software' tegen of stuiten we op de afkorting 'OSS'. Dergelijke software is over het algemeen gratis en valt onder een licentie die stipuleert dat de broncode vrij moet blijven en dat deze vrij mag doorgegeven worden. Vrijheid van broncode houdt in dat iedereen deze code naar eigen wensen mag aanpassen. Alsmaar vaker wordt dergelijke software door de grote namen in de informatica-wereld opgehemeld. Grote onderneming als HP, SUN en IBM ondersteunen belangrijke OSS-projecten en Linux, het populairste OSS-besturingssysteem, ziet haar populariteit stijgen door de stabiliteit en functionaliteit die zij biedt.

Uitgaande van deze twee bedenkingen, besloot ik om te onderzoeken of Open Source Software een economisch verantwoord alternatief zou kunnen zijn voor de commerciële software die momenteel in de meeste ondernemingen gebruikt wordt.

Vooreerst heb ik OSS, door middel van een literatuurstudie en bevraging van bevoorrechte getuigen, geëvalueerd aan de hand van 9 criteria. Deze criteria zijn kwaliteit, veiligheid, flexibiliteit, ondersteuning, beschikbaarheid, gebruiksvriendelijkheid, leveranciersonafhankelijkheid, levensduur en kosten. Ondanks een aantal minpunten, bleek uit deze evaluatie dat OSS op verschillende vlakken beter scoorde dan haar commerciële variant met gesloten broncode (CSS of Closed Source Software). Daarenboven zijn kostenbesparingen effectief realiseerbaar door het (meestal gedeeltelijk) vervangen van CSS door OSS.

Een theoretische benadering wordt echter pas echt interessant, wanneer zij aan een praktijkgeval gekoppeld wordt. Daarom onderzocht ik Euroherbs BV, een Nederlandse onderneming die in 2000 een groot deel van haar commerciële software verving door OSS. Dit onderzoek gebeurde in samenwerking met Michel van der Kleij, de IT-consultant van de onderneming. De migratie bleek een succes te zijn. Tegenwoordig beschikt de onderneming over een volledig geïntegreerd en geautomatiseerd informatiesysteem waarvan de fundamenten zijn opgebouwd uit OSS. Dit zorgt ervoor dat flexibiliteit en portabiliteit in de hand gewerkt worden. Ook de veiligheid en continuïteit worden door OSS gegarandeerd. Daarenboven heeft de (gedeeltelijke) migratie naar OSS een heleboel kostenbesparingen met zich meegebracht. Dankzij dit alles is de directie van Euroherbs overtuigd dat OSS waarde heeft toegevoegd aan de onderneming. OSS is op 6 jaar tijd een vaste waarde geworden voor deze importeur en exporteur van Aziatische, geneeskrachtige kruiden. OSS kan dus wel degelijk een economisch verantwoord alternatief vormen voor de CSS die momenteel in vele ondernemingen wereldwijd enorme kosten veroorzaakt.

H1 Probleemstelling

1.1 Situatieschets

In een wereld, gekenmerkt door snelle vooruitgang en voortdurende verandering, neemt de competitie tussen de spelers op de verschillende markten sterk toe. Horngren e.a. (2003) stellen dat de belangrijkste troef in het concurreren met andere spelers op de markt erin bestaat om een beter inzicht te verwerven in de kostenstructuur van de onderneming. Op die manier kan meer doelgericht aan kostenbeperking gedaan worden en kunnen de 'juiste' kosten aan banden gelegd worden. Dit geldt uiteraard niet alleen voor bestaande ondernemingen. Wanneer een nieuwe onderneming opgericht wordt, is er nood aan risicokapitaal en dus aan een investeerder. Volgens Smith en Smith (2004) zal de investeringsbeslissing van deze laatste bepaald worden door de afweging van verwachte return en risico. Hoe hoger het risico, hoe meer return de investeerder zal eisen om in de nieuwe onderneming te stappen. Daarenboven is er een positieve correlatie tussen de omvang van de kosten in een bepaalde onderneming en het risico dat verbonden is aan het investeren in deze onderneming. Het beperken van de kosten is dus niet alleen interessant voor bestaande ondernemingen die competitiever willen worden, maar kan er ook voor zorgen dat ondernemers die op zoek zijn naar risicokapitaal voor een nieuwe onderneming, een interessanter business plan kunnen voorleggen en daardoor minder problemen zullen hebben om een investeerder aan de haak te slaan.

Door de snelle groei en toename aan belang van de informatietechnologie doorheen de voorbije decennia, lijkt het logisch dat de kosten die informatiesystemen in ondernemingen

met zich meebrengen een steeds groter deel van de bedrijfskosten uitmaken. Zowel de Vreede (2002) als Verhoef (2005) zijn van mening dat de toenemende concurrentie en een minder gustige economische situatie ervoor zorgen dat steeds meer ondernemingen willen bezuinigen. Zij hebben er dan ook begrip voor dat het algemeen management, waar zelden iemand met goede IT-kennis toe behoort, als snel haar oog laat vallen op de IT-afdeling wanneer er ‘gesnoeid’ moet worden. In een onderzoek voor Ematic, ontdekt Kivit (2004) dat 11% van de bevraagde IT-managers geconfronteerd worden met bezuinigingen. Het is immers zo dat niet alleen het aanschaffen, onderhouden en upgraden van hardware zware kosten met zich meebrengen. Ook de software die op deze computers gebruikt wordt heeft een prijskaartje. Software heeft ook nood aan onderhoud en moet geregeld van upgrades en bugfixes worden voorzien. Daarenboven is de aanschaffingskost van software wel éénmalig, maar moeten ondernemingen licenties betalen voor elke gebruiker van deze software. Deze licenties zijn (bijna altijd) beperkt in de tijd en moeten dus periodiek hernieuwd worden.

Tijdens het surfen op Internet en het doorbladeren van tijdschriften stootte ik doorheen de voorbije jaren steeds vaker op de term ‘Open Source Software’ of de afkorting OSS en op de communities die zich met de ontwikkeling, optimalisatie en distributie hiervan bezighouden. Toen dit fenomeen in 1991 oproer maakte door de verspreiding van Linus Torvalds’ Open Source besturingssysteem, namelijk Linux, was de belangstelling van de IT-wereld groot. Het was een kwalitatief hoogstaand besturingssyteem, dat zeer stabiel was en pronkte met vrijheid van broncode. De voornaamste reden voor deze enorme belangstelling was echter dat dit besturingssyteem, net als andere Open Source Software,

geen licentiekosten met zich meebracht. Om deze reden maakten grote namen als E-trade en Disney reeds bekend dat zij overstapten op Linux en Open Source Software (Theunissen, 2004). Voeg daarbij het feit dat vele 'groten' in de computerwereld zweren bij besturingssystemen gebaseerd op Linux en Open Source applicaties en de onvermijdelijke vraag rijst: "Waarom nog hopen geld versluizen naar de bankrekening van Bill Gates voor licenties van Microsoft Windows, Microsoft Office en andere commerciële softwarepaketten, als Open Source Software zo goed als gratis is en niet moet onderdoen op gebied van kwaliteit?'

1.2 Praktijkprobleem

Het praktijkprobleem bestaat er dus in dat vele bestaande ondernemingen willen bezuinigen om hun competitiviteit te doen toenemen en dat 'start-ups' het moeilijk hebben bij het aantrekken van investeerders door te hoge kostenratio's. Het informatiesysteem binnen een onderneming creëert vaak enorme kosten, niet alleen op gebied van hardware, maar ook wat software betreft. De hoge licentiekosten van de programmatuur die een onderneming gebruikt liggen mede aan de oorzaak hiervan. Er is wel degelijk een alternatief voor deze commerciële software die tegenwoordig wereldwijd verspreid is. Dat alternatief heet Open Source Software en brengt geen licentiekosten met zich mee. De afwezigheid van licentiekosten vormt echter geen garantie dat Open Source Software voor een bepaalde onderneming, globaal bekeken, goedkoper zal uitkomen dan haar commerciële variant. Er moet dus ook gekeken worden naar de (eventuele) kostenstijgingen die een overstap van commerciële naar Open Source Software met zich kan meebrengen. Zo moeten we rekening houden met mogelijke bijkomende kosten voor

opleiding van de gebruikers, vervangen van hardware, veranderen van de architectuur van het informatiesysteem en dergelijke, om te kunnen afleiden of een migratie in totaal een kostenbesparing, dan wel een kostenstijging zal veroorzaken.

1.3 Centrale onderzoeksvraag

Zoals reeds vermeld, brengt Open Source Software geen licentiekosten met zich mee en wordt deze software door vele belangrijke mensen in de IT-wereld opgehemeld. Met dit in gedachten ben ik tot de volgende onderzoeksvraag gekomen:

Is Open Source Software een economisch verantwoord alternatief voor commerciële software in ondernemingen?

Het is de bedoeling om een zo algemeen mogelijk onderzoek te voeren, dat voor alle soorten ondernemingen interessant om lezen is en een instrument kan zijn bij de afweging om al dan niet over te stappen op OSS.

1.4 Deelvragen

Om op de voorgaande centrale onderzoeksvraag te kunnen antwoorden, zal ik een aantal deelvragen onderzoeken die nauw gerelateerd zijn aan de onderzoeksvraag en die zullen helpen bij het vormen van een goed gefundeerd antwoord hierop.

Vooreerst zal ik een evaluatie maken van Open Source Software aan de hand van een aantal criteria. De bedoeling is om een vergelijking te maken met Closed Source Software, om zo een idee te krijgen van de vlakken waarop Open Source Software een verbetering met zich mee kan brengen.

Daarnaast heeft een migratie naar Open Source Software ook een effect op de uitgaven in verband met het informatiesysteem, en dus ook op de kostenstructuur van de onderneming in haar geheel. Deze effecten zullen onderzocht worden en de verbanden met de criteria uit de vorige deelvraag zullen gelegd worden.

1.5 Onderzoeksmethodologie

Voor dit onderzoek zal gebruik gemaakt worden van zowel primaire als secundaire gegevens. Om een evaluatie te maken van Open Source Software en de effecten van een migratie op de kostenstructuur te onderzoeken, werd een literatuurstudie gehouden. Als aanvulling hierop werden ook experts op het gebied van Open Source Software bevraagd. Daarna moet de praktijkstudie van een migratie een duidelijker beeld scheppen van hoe een dergelijke overstap naar OSS in de realiteit kan gebeuren. Op basis van de kennis vergaard uit de literatuurstudie, de verschillende beschouwingen van de bevoorrechte getuigen en de praktijkstudie, heb ik getracht mijn thesis zodanig op te bouwen, dat zij voor ondernemingen een instrument kan vormen bij het beoordelen van Open Source Software als een economisch verantwoord alternatief voor de commerciële software die zij gebruiken.

1.6 Overzicht van de hoofdstukken

Vooreerst biedt hoofdstuk 2 de lezer een kennismaking met Open Source Software. Hierin zal duidelijk worden welke de basiskenmerken zijn van dergelijke software en hoe de verschillende soorten licenties hiervoor zijn opgebouwd. Daarnaast zal ook de geschiedenis van Open Source Software uit de doeken gedaan worden. Hierbij worden de verbanden gelegd met de geboorte van het Unix-platform en later de ontwikkeling van het Linux-besturingssysteem door Linus Torvalds. Bedoeling van dit hoofdstuk is de lezer duidelijk te maken welke de ontstaansredenen waren van de 'Open Source Movement' en welke principes deze beweging, nu nog steeds, met man en macht verdedigt.

In hoofdstuk 3 wordt Open Source Software geëvalueerd aan de hand van 9 criteria. Deze zijn kwaliteit, veiligheid, flexibiliteit, ondersteuning, beschikbaarheid, gebruiksvriendelijkheid, leveranciersonafhankelijkheid, levensduur en kosten. Op basis van deze criteria zal duidelijk worden of Open Source Software een goed alternatief is voor de commerciële software die tegenwoordig in het leeuwendeel van de ondernemingen aanwezig is. Ook wordt verwezen naar een simulatie-spreadsheet, die handig kan zijn voor ondernemingen die het effect van een migratie op de kostenstructuur proberen te voorspellen.

Hoofdstuk 4 biedt een praktijkstudie, waarin een onderneming die een aantal jaren geleden gedeeltelijk overstapte op Open Source Software onderzocht wordt. Aan de hand van het theoretische kader dat in het voorgaande hoofdstuk gevormd is, zullen de effecten van deze migratie op het informatiesysteem van de onderneming en haar kostenstructuur onder de

loep genomen worden. Deze praktijkstudie gebeurde in samenspraak met de IT-consultant van de onderneming in kwestie en rondt af met een beknopte conclusie omtrent de bevindingen van deze consultant en die van de directie en het personeel. De bedoeling van deze praktijkstudie is om aan te tonen dat een migratie, niet alleen op papier, maar ook in de realiteit haalbaar is en op verschillende vlakken voordelig kan zijn voor de onderneming.

In hoofdstuk 5 zullen tenslotte algemene conclusies getrokken worden omtrent de onderzochte materie. Dit zal gebeuren aan de hand van de bevindingen uit het derde en vierde hoofdstuk. Daarenboven zullen in dit hoofdstuk ook een aantal tips meegegeven worden voor ondernemingen die een migratie overwegen. Ook deze laatste steunen uiteraard op de bevindingen uit het onderzoek.

H2 Open Source Software : een introductie

2.1 Wat is Open Source Software ?

Open Source Software is software waarvan de broncode ‘vrij’ is. Dit concept van vrijheid is driedelig: vrijheid van distributie, gebruik en wijziging. Open Source Software mag dus door iedereen gebruikt worden voor alle mogelijke toepassingen en zonder enige restrictie. Dit laatste impliceert dus dat Open Source Software ook kan gebruikt worden voor bijvoorbeeld genetisch onderzoek of voor het ontwikkelen van kernwapens. Daarenboven kunnen alle eindgebruikers de broncode inkijken en bijgevolg fouten in de code rapporteren of zelfs aanpassen. Open Source Software wordt dus met andere woorden gedefiniëerd door de licentie die ermee gepaard gaat (Evers, 2000). De uitgebreide definitie die in 1997 voor het eerst door Bruce Perens op papier werd gezet, bestaat uit 10 vuistregels die de kenmerken van Open Source Software beschrijven. Dit document is doorheen de jaren meerdere malen aangepast geweest. De huidige ‘Open Source Definition’ vindt U in bijlage 1.

Typerend voor de licentie van OSS is dat zij zich net van die rechten van intellectuele eigendom distantiëert, die door ontwikkelaars van commerciële software gebruikt worden om inkomsten te genereren. Belangrijkste rechten voor deze laatste zijn de geheimhouding van de broncode van de software die zij ontwikkelen en de licentiekosten op het gebruik ervan. Vandaar dat men commerciële software ook vaak Closed Source Software of CSS noemt. Er zijn verschillende licenties voor OSS, maar de twee belangrijkste zijn ongetwijfeld de BSD- en de GPL-licentie. Het verschil tussen beide bestaat erin dat de

BSD-licentie toelaat om de broncode te gebruiken voor commerciële (en dus closed source) software, zij het onder strikte voorwaarden, terwijl de GPL-licentie dit onder geen enkele voorwaarde toelaat. (Raymond, 1999)

Vaak denken mensen dat Open Source Software gebonden is aan technologie of dat dergelijke software enkel voor Linux bestaat. Er zijn echter ook tal van Open Source applicaties voor commerciële platformen zoals Microsoft Windows of Apple Mac OS. Uiteraard is het aanbod groter voor Linux-gebruikers, omdat dit besturingssysteem zelf al Open Source is. Open Source Software voor de verschillende platformen kan men makkelijk downloaden op het Internet. Doorheen de voorbije jaren heb ik een heleboel websites hiervoor bijgehouden. De meest interessante hiervan zijn opgenomen in bijlage 2.

2.2 De geschiedenis van Open Source Software

2.2.1 Vrije broncode: zo oud als software

In tegenstelling tot wat veel mensen denken is ‘Open Source’ allesbehalve een nieuw fenomeen. Open Source verwijst eigenlijk naar de manier waarop men software ontwikkelde vóór de jaren ’80 (Uytterhoeven, 1999). Al sinds de jaren zestig wijst dit begrip op de vrije beschikbaarheid van de broncode (Weber, 2004). In de jaren vijftig en zestig was het geen evidentie voor particulieren om computers te kopen (laat staan software te draaien) omdat deze in die tijd nog veel te duur waren. De meeste software werd ontworpen om op mainframe-computers te draaien waar dan eventueel thin clients voor de ‘gewone eindgebruiker’ op werden aangesloten. Grote ondernemingen of

universiteiten die dergelijke mainframes aankochten, kregen er de software bijgeleverd. Ook de broncode van deze software was inbegrepen in het pakket. Dit was niet alleen nodig voor ondernemingen die de software wilden aanpassen voor eigen gebruik, maar ook voor universiteiten en andere onderzoeksinstellingen waar het delen van kennis de grootste drijfveer vormt voor kennisontwikkeling (Wayner, 2000). Omdat leveranciers van hardware ook de software leverden en omdat deze software enkel compatibel was met de hardware waarbij ze geleverd werd, had de broncode geen enkele economische waarde (Lerner, 2000). Het was zowel voor de producent als voor de consument interessanter om de broncode vrij te houden. De consumenten konden zelf fouten opsporen en aanpassen, maar ook rapporteren aan de producent, waardoor de software doorheen de opeenvolgende (ontwikkelings)versies steeds meer naar de wensen van de consumenten evolueerde. Ook de producent had hier baat bij, en wel in die zin dat hij minder middelen moest inzetten om zijn software te optimaliseren (Weber, 2000).

2.2.2 Unix: de ommekeer

Stilaan werden computers populairder en begon menig liefhebber zich te storen aan de incompatibiliteit tussen software van de ene leverancier en hardware van een andere. Consumenten die een pakket hadden gekocht, bestaande uit hardware en bijhorende software, werden beperkt tot de gekochte combinatie. Andere hardware aanschaffen en daarop de software van het vorige systeem draaien was onmogelijk, net als het omgekeerde. Daarenboven was het onmogelijk om de software zodanig te manipuleren dat hij toch op een ander hardware-platform kon gebruikt worden. Men spreekt in dit laatste geval van inportabiliteit van software. Eind jaren zestig ontwikkelde BTL, een business

unit van AT&T, het revolutionaire besturingssysteem ‘UNIX’, geschreven in de nieuwe programmeertaal ‘C’ (die ook bij BTL ontwikkeld werd). Dit nieuwe besturingssysteem zou een oplossing voor het probleem van incompatibiliteit en inportabiliteit vormen. Zo geschiedde en het tijdperk van de platformonafhankelijkheid was aangebroken: software zou nu makkelijk overgezet kunnen worden naar de hardware van andere leveranciers. Een bepaald programma was nu met andere woorden niet meer gebonden aan een bepaald hardwareplatform (Moglen, 1999). UNIX was ook het eerste besturingssysteem dat ‘multi-tasking’ (het gelijktijdig uitvoeren van verschillende taken) en ‘multi-user’ (verschillende gebruikers op één besturingssysteem) ondersteunde en wordt nog steeds veel gebruikt als besturingssysteem voor servers, vooral dan de distributie van Solaris (Beekman, 2003).

Stilaan begonnen softwareontwikkelaars in te zien dat dankzij deze nieuwe portabiliteit, mensen zouden kunnen kiezen welke software ze wilden draaien, zelfs nadat ze een bepaald soort hardware hadden aangeschaft. De consument zou nu met andere woorden kunnen kiezen voor de beste software op de markt, omdat hij niet meer gebonden was aan de software die zijn hardwareleverancier meeleverde (McKusick, 1999). Midden jaren ’70 waren ondernemingen, zoals het toen net opgestarte ‘Micro-Soft’, bezig met het ontwikkelen van professionele software, waarvan de broncode geheim gehouden werd en waarbij inkomsten gegenereerd werden door middel van licenties. In bijlage 3 vindt U een brief waarin een gefrustreerde Bill Gates hobbyisten oproept om te stoppen met het delen van broncodes, omdat dit de waarde voor de ontwikkelaars vernietigde. Toen reeds waren er een heel aantal tegenstanders van deze commerciële of ‘proprietary’ software, zoals Gates die noemde. Het merendeel waren ‘hackers’ (oorspronkelijk verwees deze term naar

getalenteerde programmeurs en mensen die er van hielden om allerhande toestellen te gebruiken voor doeleinden waar ze niet voor bedoeld waren) die vonden dat de broncode van software vrij moest blijven. Levy publiceerde in 1984 het boek 'Hackers: Heroes of the Computer Revolution' waarin de principes van de 'hacker-ethiek' werden opgesomd:

- Toegang tot computers en alle andere dingen die je iets kunnen bijbrengen over hoe de wereld werkt moet vrij zijn voor iedereen.
- Alle informatie moet vrij zijn.
- Wantrouw autoriteit, promoot decentralisatie.
- Hackers moeten beoordeeld worden naar hun hacken, niet naar schijncriteria zoals graad, leeftijd, geslacht, ras of positie.
- Met een computer kan je kunst en schoonheid creëren
- Computers kunnen je leven verbeteren.

2.2.3 De Open Source beweging

In 1984 begon de Free Software Movement, opgericht door Richard Stallman, een voormalig onderzoeker van MIT, stilaan vorm te krijgen. Deze beweging verstond onder 'Free Software' hetzelfde als wat men nu Open Source Software noemt en werd opgericht als reactie op het geheimhouden van de broncode van software. Vele mensen lieten zich misleiden door de term en dachten dat 'free' stond voor kosteloos, terwijl dit eigenlijk verwees naar de vrijheid van de broncode. Een begrijpelijk misverstand, gezien dergelijke software ook vrij was van licentiekosten en kosteloos bemachtigd en doorgegeven kon

worden (Stallman, 1999). Weber (2004, p.5) vertaalde het als volgt: “Free as in speech, not free as in beer!”.

Toen Linus Torvalds eind jaren ’80 stilaan genoeg kreeg van de beperkingen in zijn besturingssysteem ‘Minix’ (gebaseerd op Unix), begon hij aan de ontwikkeling van zijn eigen besturingssyteem. In 1991 publiceerde hij de broncode ervan op het Internet en gaf het besturingssysteem de naam ‘Linux’ mee. Het vervangen van de laatste letter van zijn voornaam door een X verwijst naar de Unix-basis van het besturingssysteem. Linux werd een groot succes en ontketende de eigenlijke ‘Open Source Revolution’. (Torvalds, 2002)

In 1998 deed de term ‘Open Source’ voor het eerst haar intrede. Deze werd gebruikt tijdens de campagne van een aantal ex-programmeurs van Netscape. Zij hadden zich afgescheurd van hun voormalige werkgever en waren begonnen aan de ontwikkeling van een Open Source browser op basis van Netscape zelf. Deze browser werd Mozilla gedoopt en zorgde voor veel oproer in de computerwereld. Later dat jaar werd het Open Source Initiative of OSI opgericht. Deze organisatie stond erop een duidelijk verschil te maken tussen Free Software en Open Source Software. Volgens hen moest Open Source geassocieerd worden met collaboratieve ontwikkeling, terwijl dat bij Free Software helemaal niet noodzakelijk is. Schrijvers en communities schakelden stilaan over naar de afkortingen FOSS (Free and Open Source Software) en FLOSS (Free/Libre/Open Source Software) om te verwijzen naar de Open Source community die de idealen van beide filosofieën nastreeft: vrije beschikbaarheid van broncode en de mogelijkheid tot het kosteloos bekomen en doorgeven van de software. (Stallman, 1999)

H3 Evaluatie van OSS als alternatief voor CSS

3.1 Methode van evaluatie

Om Open Source Software te evalueren als alternatief voor haar commerciële variant, zal zij beoordeeld worden aan de hand van een aantal criteria. Voor de keuze van de criteria heb ik mij in hoofdzaak gebaseerd op wetenschappelijke werken in verband met Open Source Software en op de kennis die ik doorheen de voorbije jaren heb opgebouwd door met OSS bezig te zijn. Na het zorgvuldig afwegen van de verschillende mogelijkheden die ik uit de vergaarde kennis gefilterd had, kwam ik tot 9 criteria die gebruikt konden worden om Open Source Software op een degelijke en volledige manier te evalueren. Nadat ik OSS geëvalueerd had aan de hand van verschillende wetenschappelijke werken, heb ik ook twee specialisten ter zake geïnterviewd hieromtrent. De beknopte samenvattingen van deze interviews zijn terug te vinden in bijlage 4 en 5. Het betreft in de eerste plaats Michel van der Kleij, IT-consultant van een aantal ondernemingen, waaronder het Nederlandse Euroherbs BV, dat later nog in de praktijkstudie aan bod zal komen. Daarnaast werd ook Eric Smets, medeoprichter van FKS, bevraagd. FKS is gelegen in Alken en biedt aan andere ondernemingen de nodige ondersteuning bij het implementeren van OSS in hun informatiesysteem.

De criteria die gebruikt werden voor de evaluatie zijn kwaliteit, veiligheid, flexibiliteit, ondersteuning, beschikbaarheid, gebruiksvriendelijkheid, leveranciersonafhankelijkheid, levensduur/gebruiksduur en uiteraard kosten. Stabiliteit is niet opgenomen als apart criterium omdat dit in feite een deel vormt van het criterium kwaliteit. Het criterium

'kosten' zal achteraf apart besproken worden, omdat het centraal staat in deze thesis en een aantal van de andere criteria eerst onderzocht moeten worden om een gefundeerd beeld te vormen omtrent het effect van OSS op de kostenstructuur van de onderneming.

3.2 De criteria

3.2.1 Kwaliteit

Met de kwaliteit van software wordt bedoeld hoe stabiel, effectief en efficiënt de software is. Natuurlijk bestaat er net als bij commerciële software zowel slechte als goede OSS. Breedveld (1999) beweert echter dat OSS altijd van hogere kwaliteit is dan vergelijkbare commerciële software. Visser (2002) is ervan overtuigd dat OSS, wanneer men de totaliteit van de uitgebrachte software bekijkt, beter scoort op gebied van kwaliteit dan Closed Source Software. Godden (2000) vertelt hoe Linux in een vergelijkende test haar commerciële tegenhanger Windows NT overtrof. Gedurende een gans jaar crashte Linux slechts één keer omwille van een probleem met de harde schijf. Het duurde vier uur om dit probleem op te lossen, wat dus resulteert in een beschikbaarheid van het systeem van 99,95%. Windows NT daarentegen crashte 68 keer, waarvan 26 keer door hardware-problemen, 26 keer door problemen met het geheugen, 8 keer door fouten in het bestandsbeheer en 33 keer (!) door ongekende conflicten. Het oplossen van deze problemen kostte 65 uren, wat resulteert in een beschikbaarheid van het systeem van 99,26%. Linux werd in deze test dus niet bepaald met neuslengte het winnende besturingssysteem, maar verpulverde de (commerciële) vaste waarde van vele ondernemingen. Ook Smets verklaarde tijdens het interview dat Open Source

besturingssytemen, met name Linux en FreeBSD, naar zijn mening veel stabieler zijn dan de commerciële tegenhangers.

Maar waarom is de kwaliteit van OSS dan hoger dan die van CSS? Raymond (1999) onderscheidt twee modellen van software-ontwikkeling: het kathedraal-model en het bazaar-model. Volgens hem wordt CSS ontwikkeld volgens het kathedraal-model. Dit houdt in dat de verschillende onderdelen van een applicatie (modules), door verschillende programmeurs (of kleine groepjes van programmeurs) geschreven worden. Consumenten krijgen geen inzicht in de code en kunnen slechts kritiek geven op de software nadat die uitgebracht is. OSS daarentegen, wordt volgens Raymond ontwikkeld volgens het bazaar-model. Net als het grote aantal potentiële kopers die rondkuieren op een rommelmarkt zijn er bij de ontwikkeling van OSS grote aantallen gebruikers die voortdurend tussentijdse versies van de broncode van de software kunnen inkijken. Zij kunnen dan, nog voor het uitbrengen van de software, fouten in de code opsporen en deze aan de ontwikkelaars rapporteren. Zelfs indien ontwikkelaars van CSS de broncode van hun gedeelte zouden laten inkijken door hun collega's (inkijken door externen mag uiteraard niet bij CSS), is het aantal ogen die over de code waken veel groter voor OSS-projecten dan bij de commerciële tegenhangers. Vele handen maken licht werk, of wat Baarsma (2004) het 'multi-eye-effect' noemt. Raymond (1999, p.1) formuleert het als volgt: "Given enough eyeballs, all bugs are shallow". Raymond kwam tot dit besluit nadat hij bewust aan een OSS-project begon in de bazaar-stijl (het gaat hier om de applicatie 'Fetchmail') en tot de constatatie kwam dat het project een succes was in vergelijking met zijn voorgaande projecten in kathedraal-stijl. De gebruikers helpen de ontwikkelaars als het ware bij het

‘ontluizen’ van de broncode, waardoor de grens tussen gebruiker en ontwikkelaar zeer vaag wordt. Dit is de reden waarom veel van de huidige OSS-ontwikkelaars vroegere gebruikers zijn. Ook van der Kleij is van mening dat deze manier van werken de belangrijkste reden is voor de hogere kwaliteit van OSS. Hij verwijst naar de ‘bazaar-stijl’ door middel van de term ‘peer-to-peer’.

Zoals reeds werd vermeld is er ook binnen OSS een onderscheid tussen goede en minder goede software. Om de kwaliteit van bepaalde OSS te voorspellen, is het interessant om even een kijkje te nemen naar de community die zich bezighoudt met de ontwikkeling van dat specifieke stukje OSS. Golden (2005) bevestigt dat de community één van de belangrijkste aspecten van OSS is. Ook van der Kleij bevestigt dit tijdens het interview. Vermits de community instaat voor het testen en rapporteren van fouten in de software, kunnen we ervan uitgaan dat het aantal mensen in de community al veel vertelt over de software zelf. Hoe meer mensen de community dus telt, hoe hoger de kwaliteit van de software. Als de software niet goed was, zouden er immers niet zoveel mensen mee begaan zijn, vinden Duijnhouwer en Widdows (2003).

3.2.2 Veiligheid

Het concept van veiligheid verschilt enorm tussen OSS en CSS. Raymond (1996) noemt de manier waarop veiligheid wordt nagestreefd bij de ontwikkeling van CSS ‘security by obscurity’. Hiermee bedoelt hij dat de ontwikkelaars ervan uitgaan dat de software veiliger is omdat de broncode niet vrij beschikbaar is en dus niet ontleed kan worden door potentiële boosdoeners. Dit heeft als gevolg dat de ‘waterdichtheid’ van de code enkel

afhangt van de prestatie van de ontwikkelaar, terwijl ook deze niet volmaakt is en er dus al snel 'lekken' in de code sluipen. Raymond (1999) is daarom van mening dat het geheimhouden van de broncode niet leidt tot echte veiligheid, maar slechts tot een vals gevoel van veiligheid. Smets beweert dat de veiligheid van CSS daardoor in het algemeen lager ligt dan die van OSS. Zo haalt hij bijvoorbeeld aan dat de kopieerbeveiliging voor DVD's in een mum van tijd gekraakt was en dat tegenwoordig, ondanks technische en juridische inspanningen, nog steeds geen manier gevonden is om DVD's zodanig te vergrendelen dat hackers de beveiligingscode niet zouden kunnen kraken. Hier geldt immers hetzelfde principe: het aantal hackers dat DVD's wil kunnen kopiëren is veel groter dan het aantal ontwikkelaars dat aan de beveiligingscode werkt.

Bij OSS daarentegen is de broncode beschikbaar voor iedereen die dat maar wil. Zoals reeds aangehaald, zorgt dit ervoor dat veel meer mensen de code onder de loep nemen en de kans op het vinden van lekken enorm toeneemt. Ook Beale e.a. (2004) zijn er van overtuigd dan het opsporen van fouten in de code veel dynamischer gebeurt en dat oplossingen veel sneller beschikbaar zijn bij OSS, terwijl dit bij CSS vaak heel lang kan duren. In hun onderzoek bevestigen zij dat Red Hat, een ontwikkelaar van OSS, er gemiddeld 11 dagen over doet om een fout in de software op te lossen, terwijl Sun, ontwikkelaar van CSS, er maar liefst gemiddeld 89 dagen over doet. Knubben (2001) wijt dit aan het feit dat ontwikkelaars van commerciële software hun patches vaak gebundeld in 'Service Packs' uitgeven. Hierdoor wordt de oplossing pas beschikbaar wanneer zij samen met andere verbeteringen in één pakket wordt uitgegeven. Cowan (2003) zegt dat vrijheid van broncode ervoor zorgt dat niet alleen de ontwikkelaars en gebruikers (beschermers)

van de software meer invloed krijgen over de veiligheid, maar dat ook hackers met kwade bedoelingen (aanvallers) makkelijker op zoek kunnen gaan naar achterpoortjes en lekken.

Alhoewel OSS software er dus voor zorgt dat niet alleen mensen met goede bedoelingen meer inzicht krijgen in de broncode, levert de vrijheid van broncode toch een belangrijk voordeel. Wanneer een de broncode van CSS zwakke plekken heeft, worden deze zelden door de ontwikkelaars zelf gevonden. Vergeleken met het aantal ontwikkelaars dat aan een bepaalde applicatie werkt, zijn de hackers die er interesse in hebben overduidelijk in de meerderheid. De ontdekkers zijn hierdoor meestal hackers die door middel van 'trial & error' zwaktes van de software opsporen. Het uitbuiten van deze zwaktes door hackers is dan het eerste signaal naar ontwikkelaars om aan te geven dat er iets niet pluis is. Bij OSS echter, is de kans veel groter dat de eerste persoon die op een dergelijk lek in de code stoot een gebruiker, mede-ontwikkelaar of andere persoon met goede bedoelingen is. Het lek kan dan aan de ontwikkelaars gerapporteerd worden, nog voordat iemand er misbruik van heeft kunnen maken. (Payne, 2002)

Blankensteijn (2002) voorspelt echter een negatieve evolutie voor OSS op gebied van veiligheid. Hij is van mening dat de veiligheid van het commerciële besturingssysteem 'Microsoft Windows' lachwekkend is. De reden hiervoor is volgens hem dat er zoveel gebruikers zijn, dat dit besturingssysteem voor hackers met slechte bedoelingen het meest aantrekkelijk is. Linux-besturingssystemen daarentegen, worden door veel minder mensen gebruikt en kennen daarenboven een groot aantal varianten, waardoor het voor hackers veel minder interessant is om Linux-besturingssystemen te bestuderen en aan te vallen.

Blankensteijn merkt op dat Linux in de lift zit en dat het aantal gebruikers bijgevolg in enorme mate toeneemt. Hij is van mening dat dit doorheen de komende jaren gepaard zal gaan met een verminderde veiligheid van OSS.

3.2.3 Flexibiliteit

Volgens Baarsma (2004) is OSS veel flexibeler dan CSS. Zij meent dat, dankzij de vrijheid van broncode, de mogelijkheid bestaat voor de consument om gekochte software aan te passen aan de eigen wensen en behoeften. Smets haalt tijdens zijn interview aan dat een onderneming die een OSS-specialist in dienst heeft, niet meer moet wachten op een derde om haar problemen op te lossen, maar gewoon zelf kan overgaan tot de nodige aanpassingen. In een onderzoek voor Merit door Glott en Ghosh (2003) wordt aangegeven dat de consument steeds meer geïnteresseerd is in deze mogelijkheid. OSS kan deze wens van de consument dus vervullen.

Daarenboven verhoogt de openheid van de broncode ook de compatibiliteit en portabiliteit van CSS. Linux ondersteunt bijvoorbeeld 14 verschillende processorfamilies, terwijl Microsoft Windows alleen Intel- en AMD-processors ondersteunt en programma's ontwikkeld door Microsoft ook alleen op Microsoft Windows draaien (Knubben, 2001). Knubben vermeldt ook dat Linux de 3 bestandssytemen van Windows (FAT16, FAT32 en NTFS) ondersteunt, wat tot een bijna onbeperkte flexibiliteit leidt. Godden (2000) illustreert de hogere flexibiliteit van OSS door als voorbeeld te geven dat eenzelfde variant van Linux op zowel een high-end server, een lichte consumentenmachine, als een handheld kan worden geinstalleerd. Windows-gebruikers bijvoorbeeld, zouden respectievelijk

Microsoft Windows Server 2003, Microsoft Windows XP en Microsoft Windows Mobile moeten installeren op de genoemde soorten computers. De belangstelling van de consument voor dergelijke compatibiliteit en portabiliteit zou stijgen (Glott en Ghosh, 2003). OSS kan dus ook deze behoefte van de consument beter invullen dan CSS.

Volgens van der Kleij zorgt ook de afwezigheid van licentie- en aanschafkosten ervoor dat de flexibiliteit van OSS hoger ligt dan die van CSS. Dit laat ondernemingen namelijk toe om OSS gratis uit te proberen, alvorens over te gaan tot het implementeren ervan in het informatiesysteem.

3.2.4 Ondersteuning

In een recent werk maakt van den Berg (2005) een onderscheid tussen gebruiksondersteuning en probleemondersteuning. Gebruiksondersteuning houdt in dat er antwoord gegeven wordt op vragen in verband met het gebruik van de software. Handleidingen zijn hiervan een voorbeeld. Probleemondersteuning daarentegen, verwijst naar het oplossen van problemen met of in de software. Daarenboven geeft van den Berg ook aan dat er nog een belangrijk onderscheid moet gemaakt worden wat betreft ondersteuning: kosteloze ondersteuning (door de community) en betaalde ondersteuning door derden. De ondersteuning die een community biedt (zoals bijvoorbeeld online bugtrackers waar gebruikers fouten in de broncode kunnen rapporteren) zal in vele gevallen echter niet direct genoeg zijn voor grotere ondernemingen. Zij zullen meer gebaat zijn met de ondersteuning van derden die meer toewijding kunnen bieden aan die bepaalde ondernemingen. Dit zal uiteraard kosten met zich meebrengen. Voor dergelijke

ondersteuning onderscheidt van den Berg een aantal verschillende regelingen. Zo kan een onderneming een periodieke ondersteuningsovereenkomst afsluiten met een ondersteuningsfirma, waarbij zij een vast bedrag betaalt voor onbeperkte ondersteuning gedurende die periode, of kan zij per consultatie betalen, waarbij de kostprijs vaak gebaseerd is op het aantal werkuren. Sommige projectleiders van OSS voorzien zelf ook in dergelijke ondersteuning. Een voorbeeld hiervan vinden we terug bij MySQL, een zeer populair stukje OSS voor database servers (MySQL, 2005). Volgens van der Kleij is er ook voldoende OSS die ondersteund wordt door commerciële bedrijven. Zo bieden bekende namen als SUN, HP en IBM ook hun eigen OSS aan, die natuurlijk gepaard gaat met de professionele ondersteuning ervan.

Opmerkelijk is dat uit een onderzoek van Kivit (2004) voor Ematic blijkt dat het gebrek aan ondersteuning voor vele mensen het grootste nadeel van OSS is. Volgens de mensen die bevraagd werden is er minder ondersteuning beschikbaar voor OSS in de vorm van handleidingen en cursusmateriaal. Uit een onderzoek voor Merit door Glott en Ghosh (2003) blijkt daarenboven dat ondernemingen die CSS gebruiken, in grote mate de hulp van gespecialiseerde ondersteuningsfirma's inroepen (uiteraard tegen betaling). Zo zouden 24% van de ondernemingen zich hier vaak tot zeer vaak op beroepen, 72% van ondernemingen regelmatig en slechts 4% van de ondernemingen nooit. Het is dus duidelijk dat de behoefte aan externe consultants groot is. Volgens Knubben (2001) is dergelijke, betaalde en professionele ondersteuning voor OSS minder aanwezig dan voor CSS, maar neemt die voor eerstgenoemde stilaan toe, zij het dan vooral voor de meest succesvolle applicaties en besturingssystemen.

3.2.5 Beschikbaarheid

Volgens Knubben (2001) is OSS nog niet in alle segmenten goed vertegenwoordigd. Op het gebied van verticale (specifieke) software is er bijvoorbeeld minder OSS beschikbaar en een heel aantal veelbelovende projecten verkeren nog niet het maturiteits-stadium. Daarenboven merkt hij op dat OSS-ontwikkelaars een zo groot mogelijke compatibiliteit en portabiliteit nastreven, terwijl ontwikkelaars van CSS vaak net het omgekeerde proberen te bereiken. OSS zal heel vaak op CSS-systemen gebruikt kunnen worden, terwijl het omgekeerde zelden mogelijk is. Dit zorgt er dan ook vaak voor dat een heleboel mogelijke OSS wel beschikbaar is, maar niet gebruikt kan worden omwille van compatibiliteits- en/of portabiliteitsproblemen. Volgens Knubben (2001) wordt er ook op dit gebied vooruitgang geboekt. Zo bieden CSS-ontwikkelaars zoals Sun en Netscape tegenwoordig ook OSS-producten aan en werken anderen zoals Oracle en SAP aan de compatibiliteit van hun producten met OSS.

Smets merkt in zijn interview op dat voor verticale software, ook vaak geen CSS beschikbaar is die voldoet aan de vereisten. Nog vaker biedt CSS zoveel onnodige mogelijkheden aan dat de ondernemingen veel te veel zullen betalen, rekening houdend met het aantal functies die zij werkelijk zullen gebruiken. Daarenboven zorgt dit voor extra belasting van de hardware, waardoor de onderneming vaker en duurdere computers moet aankopen. Het voordeel van OSS is dat de broncode kan aangepast worden aan eigen noden. Ontbrekende functies kunnen toegevoegd worden, terwijl overbodige functies even snel uit de software verwijderd kunnen worden. Op deze manier is het dus mogelijk dat een bepaald stukje OSS enkel die functies bevat die de gebruiker echt nodig heeft. Hiervoor zal

dan veel minder zware hardware vereist zijn dan voor software die overladen is met onnodige functies.

3.2.6 Gebruiksvriendelijkheid

Alhoewel er hard gewerkt wordt aan de gebruiksvriendelijkheid van OSS, merken we bijvoorbeeld bij Linux-besturingssystemen dat de gebruiksvriendelijkheid toch nog steeds lager is dan die van bijvoorbeeld Microsoft Windows XP. Volgens Kenwood (2001) ligt de oorzaak hiervan in het feit dat Linux oorspronkelijk geschreven is voor programmeurs en niet voor doorsnee eindgebruikers. Schiff (2002) stelt dat hier nog twee andere oorzaken voor zouden kunnen zijn. Zo legt hij uit dat OSS-ontwikkelaars erkenning zoeken bij de gebruikers van hun software, die meestal veel meer kennis hebben omtrent software-ontwikkeling dan de doorsnee consument die een computer enkel gebruikt als tekstverwerker en voor toegang tot het Internet. Hij voegt hier nog aan toe dat de hoge gebruiksvriendelijkheid van commerciële software hoogstwaarschijnlijk een gevolg is van de keuze voor een product-differentiatie-strategie door CSS-ontwikkelaars.

Het eerste wat iemand zich zal afvragen, wanneer hij wil overstappen op OSS, is welk besturingssysteem en welke applicaties hij zal gaan gebruiken. Afhankelijk van wensen en behoeftes kan hij of zij op Internet een enorme hoeveelheid aan OSS terugvinden. Het aanbod is zo uitgebreid, dat het voor de ‘switcher’ een hele taak wordt om de verschillende opties te gaan vergelijken en af te wegen welke software het best voldoet aan zijn of haar noden. Om te beginnen zijn er een heleboel verschillende Linux-besturingssystemen.

Volgens Gagné (2004) zijn er meer dan 350 Linux-besturingssystemen beschikbaar op het Internet. Op Figuur 1 hieronder vindt U de meest populaire uit dit enorme aanbod.

Figuur 1: De tien grote Linux-distributies

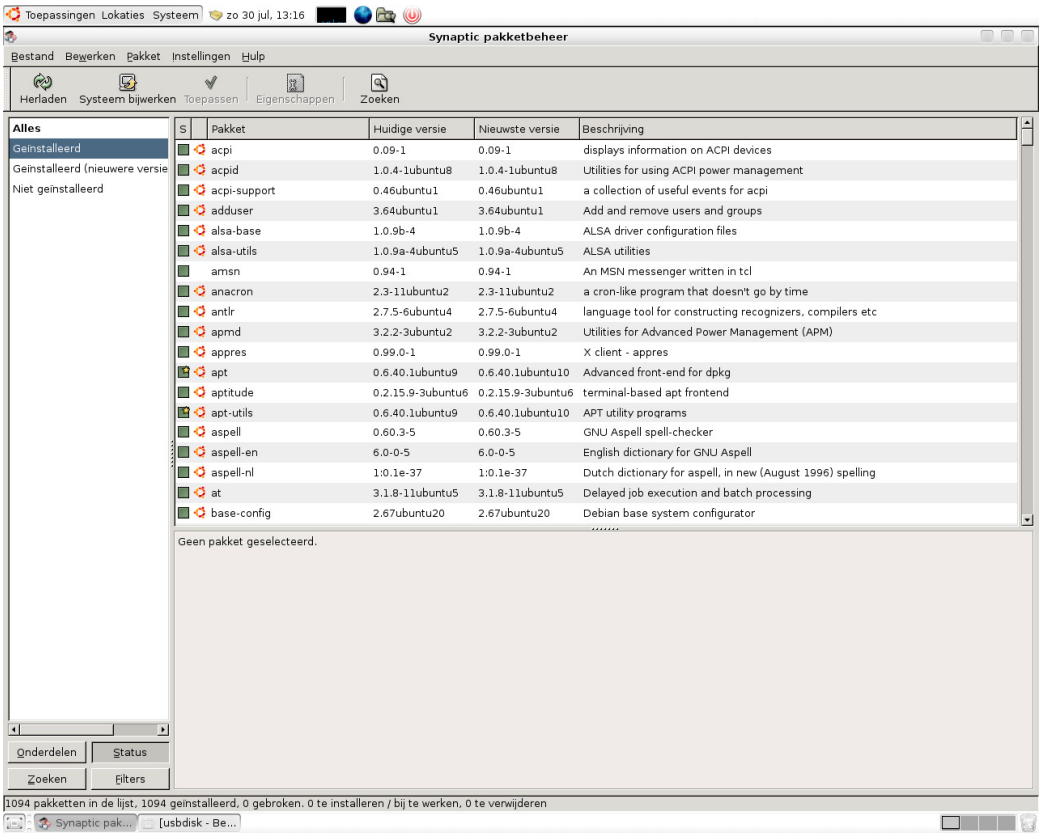
naam	eerste release	website
Debian	1993	www.debian.org
Slackware	1992	www.slackware.com
Fedora	1995	www.fedora.redhat.com
Mandriva	1998	www.mandriva.com
Suse	1996	www.suse.com
Ubuntu	2004	www.ubuntu.com
Knoppix	2003	www.knoppix.com
Gentoo	2002	www.gentoo.org
Mepis	2003	www.mepis.com
Xandros	2001	www.xandros.com

Bron: Linux Forums (2006)

Nadat de nieuwe gebruiker dan eindelijk een keuze gemaakt had tussen de verschillende besturingssystemen, moest hij vroeger ook nog eens alle applicaties gaan zoeken die hij nodig had en ervoor zorgen dat ze op zijn Linux-versie draaiden. Vaak werden applicaties alleen in de vorm van broncodes op het Internet geplaatst. De gebruiker moest deze dan eerst met een ‘compiler’ omzetten naar echte software die geschikt was voor zijn systeem. Het is duidelijk dat de gebruiksvriendelijkheid dus te wensen overliet. De Linux-ontwikkelaars onderkenden dit probleem en begonnen stilaan distributie-pakketten samen

te stellen. Deze pakketten bevatten een besturingssysteem, samen met een hele hoop interessante applicaties voor dat besturingssysteem. Aan de hand van een installatie-programma kan de gebruiker ook gemakkelijk zoeken naar nieuwe software voor zijn Linux-distributie. Een voorbeeld van een vrij nieuwe en spraakmakende Linux-distributie is Ubuntu. Dit is daarenboven de Linux-distributie die ik gebruik en waarmee dus deze thesis gemaakt is. Op de schermafdruk in Figuur 2 hieronder ziet U 'Synaptic', de applicatie voor pakketbeheer die meegeleverd wordt met de Ubuntu-distributie. Hierin moet de gebruiker gewoon op zoek gaan naar het soort applicatie dat hij nodig heeft en erop dubbelklikken om het te installeren. Een hele vooruitgang dus, vergeleken met het compilen van weleer.

Figuur 2: Synaptic, de applicatie voor pakketbeheer in Ubuntu



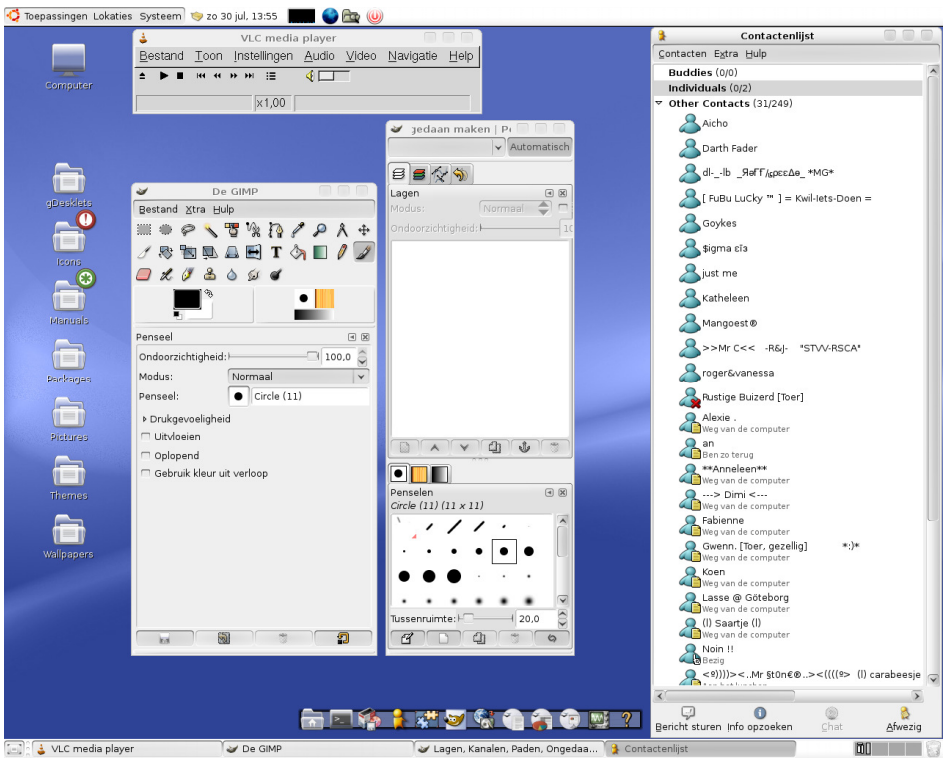
Volgens Baarsma (2004) is OSS op desktop-niveau, waar de grafische interface heel belangrijk is, minder gebruiksvriendelijk dan CSS. Op serverniveau is dit volgens haar minder belangrijk, omdat deze computers toch door professionelen worden gebruikt. Voor de doorsnee eindgebruiker is het daarentegen wel heel belangrijk om in een herkenbare omgeving terecht te komen. Smets is het hiermee eens, maar wijst ook op de keerzijde van de medaille: dankzij de hogere gebruiksvriendelijkheid, is niet alles via de GUI van Windows toegankelijk. Zo zal men bijvoorbeeld soms in de 'registers' van het besturingssyteem moeten duiken. Hiervoor moet ook een Windows-gebruiker teruggrijpen naar de 'command line' zoals die bij Linux-besturingssystemen vaak gebruikt wordt.

Zowel Baarsma (2004) als Knubben (2001) geven echter toe dat OSS de laatste jaren veel vooruitgang heeft geboekt. Baarsma stelt zelfs dat het verschil in gebruiksvriendelijkheid van de gebruikersinterface tussen OSS en CSS geen reden meer vormt om een overstap te beletten. Een aantal schermafdrukken vanop mijn huidige systeem illustreren dat de werkomgeving van een Linux-besturingssystemen zeer herkenbaar zou moeten zijn voor gebruikers van een Microsoft Windows-besturingssysteem met bijhorende applicaties.

Figuur 3 hieronder geeft duidelijk weer dat de venster-structuur die gebruikt wordt voor commerciële software, ook in OSS gebruikt wordt. Linux-besturingssystemen zijn niet meer zoals vroeger gekenmerkt door een zwart scherm met daarop ingewikkelde code in verschillende kleuren. Bovenaan bevindt zich de 'startbalk' zoals die ook in Windows terug te vinden is. VLC Media Player is een gratis alternatief voor Windows Media Player, GIMP is de OSS-variant van Adobe Photoshop en de contactenlijst uiterst rechts is die van

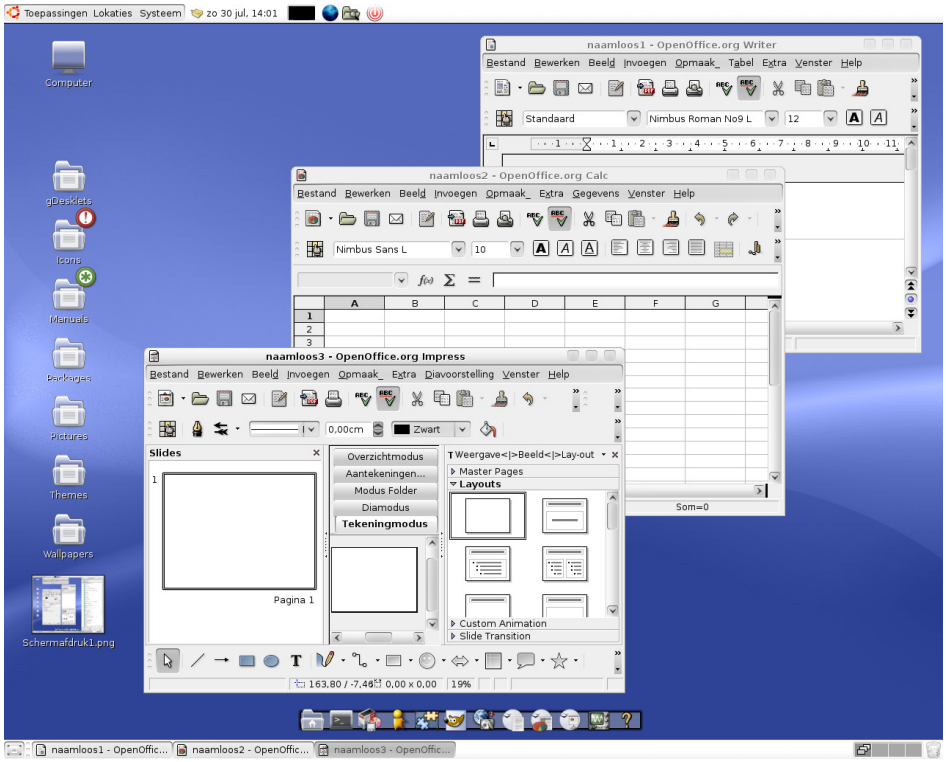
GAIM, de tegenhanger van MSN Messenger. Op een aantal geavanceerde mogelijkheden na, kan je met deze OSS-applicaties hetzelfde doen als wat je met de betalende versie ervan zou kunnen. Zo kan ik bijvoorbeeld wel chatten met mijn contactpersonen van MSN, maar kan ik geen video-gesprek houden met een webcam. Dit is echter een bewuste keuze, want als je even door de pakketten in Synaptic bladert, merk je als snel dat er een overvloed is aan applicaties voor instant messaging en dat sommigen ook de webcam-functie ondersteunen. Helemaal onderaan bevindt zich wat men noemt een ‘dock’. Hierin kan je snelkoppelingen plaatsen naar de applicaties die je het meest gebruikt. Deze dock is een applicatie voor gebruikers van (op Debian gebaseerde) Linux-besturingssystemen (zoals Ubuntu), waarvoor de inspiratie uit het Mac OSX besturingssysteem van Apple komt.

Figuur 3: GIMP, VLC en Gaim in Ubuntu



Figuur 4 toont de OSS-alternatieven voor de drie meest gebruikte applicaties van het Microsoft Office pakket. Deze applicaties zijn onderdelen van OpenOffice.org. Writer is de vervanger voor Word, terwijl het spreadsheet-programma Calc een goede vervanger is voor Excel. Impress is dan weer het alternatief voor wie voorheen presentaties maakte in Powerpoint. Ook voor Acces is er een alternatief, namelijk Base. Hierbovenop biedt OpenOffice nog een aantal applicaties die niet in de Microsoft Office suite zitten. Zo is Draw een handig programma om tekeningen, stroomdiagrammen en logo's te maken en te bewerken, terwijl Math een wiskundig programma is dat zijn weg weet met formules en vergelijkingen. OpenOffice.org vormt dus voor de doorsnee gebruiker op desktop-niveau een complete oplossing en is daarenboven volledig gratis beschikbaar. Dit pakket is niet alleen beschikbaar voor Linux-gebruikers, maar ook voor mensen die Windows of Mac OS draaien. Belangrijk is natuurlijk ook dat dit pakket dezelfde standaarden als Microsoft Office kan hanteren en dus volledig compatibel is. Zo kan men bestanden die in Microsoft Excel werden gemaakt gemakkelijk openen en aanpassen met Calc, terwijl ook bestanden die in Microsoft Word geschreven zijn, volledig bruikbaar blijven in Writer. OpenOffice ondersteunt daarenboven ook de standaarden van andere, minder bekende pakketten zoals Apple Works en Lotus Suite. We kunnen dus zeker spreken van 'open standaarden'.

Figuur 4: OpenOffice.org: Writer, Calc en Impress



3.2.7 Leveranciersonafhankelijkheid

Zodra een onderneming CSS aanschaft, ontstaat er in zekere mate leveranciersafhankelijkheid, waardoor de onderneming beperkt wordt in haar huidige en toekomstige opties en waarvan ook effecten op de softwaremarkt merkbaar zijn (Baarsma, 2004). De term die in de software-industrie voor dit effect wordt gebruikt is ‘vendor lock-in’ (Theunissen, 2004). In een onderzoek voor de Vlaamse Raad voor Wetenschapsbeleid ondervindt Monard (2004) dat de kosten veroorzaakt in Vlaanderen door dergelijke lock-in hoog zijn en dat er behoefte is aan alternatieven die deze lock-in elimineren.

Vendor lock-in kan beperkingen voor de onderneming met zich meebrengen op gebied van verschillende soorten keuzes. Zo zal de onderneming bijvoorbeeld verplicht zijn om te upgraden wanneer de huidige leverancier een nieuwe versie van de software heeft ontwikkeld en de ondersteuning van de oude versie beëindigt. Wanneer een onderneming daarentegen OSS gebruikt, kan zij zelf kiezen om al dan niet over te stappen op de nieuwe versie die beschikbaar gemaakt is (Baarsma, 2004).

Een andere vorm van lock-in situeert zich op niveau van de standaarden. OSS-ontwikkelaars wijken zelden van deze standaarden af. CSS-ontwikkelaars daarentegen hebben er baat bij om de consument te binden door eigen standaarden te gebruiken. Eens een onderneming dan gebonden is aan een bepaalde standaard, zal zij niet snel overstappen naar OSS, omdat dit door de verschillende standaarden, vaak hoge ‘switching-kosten’ met zich meebrengt. (Knubben, 2001)

Het voorgaande heeft natuurlijk ook zijn weerslag op de ondersteuning waarop de onderneming zich kan beroepen omtrent haar software. In het geval van CSS is de onderneming verplicht om ondersteuning bij de leverancier van die CSS te gaan zoeken, want enkel die leverancier heeft toegang tot de broncode. Ook wat betreft ondersteuning kunnen we dus spreken van vendor lock-in.

Een laatste gevolg van leveranciersafhankelijkheid, is het minder concurrentiëel worden van de softwaremarkt. Leveranciers en ontwikkelaars van CSS binden hun klanten en moeten zich daardoor steeds minder druk maken over de concurrentie. Dit heeft zowel

negatieve gevolgen op de kwaliteit van de software als op de prijszetting. De consument krijgt met andere woorden minder waar voor meer geld. Overstappen op OSS, zorgt er dus ook voor dat de marktmechanismen beter gaan functioneren en leidt tot hogere kwaliteit en lagere prijzen. (Mendys-Kamphorst, 2002)

3.2.8 Levensduur/gebruiksduur

Laurent (2004) ziet in OSS een ander groot voordeel. Volgens hem is de gebruiksduur van OSS in principe oneindig, terwijl die van CSS (door bijvoorbeeld vendor lock-in) beperkt is. Laurent is van mening dat dit één van de belangrijkste voordelen is die OSS aan profit-organisaties kan bieden.

OSS heeft een vrije broncode en kan dus doorheen de tijd aangepast worden aan de groeiende behoefte van de gebruiker (Raymond, 1999). Knubben (2001) besluit hieruit dat de levensduur van OSS door de gebruiker zelf bepaald wordt, terwijl CSS een relatief korte levensduur heeft. Daarenboven proberen ontwikkelaars van CSS gebruik te maken van ‘planned obsolence’ om hun inkomsten te verhogen (Katz en Shapiro, 1998). Hiermee wordt bedoeld dat de ontwikkelaars met opzet incompatibiliteiten veroorzaken met toekomstige hardware, zodat ondernemingen die nieuwe apparatuur aanschaffen, ook nieuwe investeringen moeten doen in software. Ook het opzettelijk ‘zwaarder’ maken van CSS door de ontwikkelaars, om klanten aan te zetten tot het kopen van duurdere hardware, is volgens van der Kleij bij OSS niet aan de orde. Dit gebeurt vaak door het integreren van een overvloed aan onnodige functies. Indien dit toch het geval zou zijn bij een bepaald

stukje OSS, kan gemakkelijk aan de broncode gesleuteld worden om dit probleem op te lossen.

3.3 Kosten

Zoals reeds vermeld brengt het gebruik van OSS geen licentiekosten met zich mee. Dit kan uiteraard leiden tot kostenbesparingen voor ondernemingen die commerciële software deels of volledig gaan vervangen door OSS. Licentiekosten zijn echter niet de enige kosten die met informatiesystemen gepaard gaan. Hahn (2003) zegt dat licentiekosten slechts een klein deel van de TCO (Total Cost of Ownership) van software uitmaken. De TCO van software bestaat uit nog een heleboel andere kostencomponenten. Volgens Breeveld (1999) is er echter geen éénduidige methode om deze TCO te berekenen. Of kostenbesparingen mogelijk zijn door het gebruik van OSS, zal dus afhangen van het effect van een migratie op de verschillende (mogelijke) componenten van de TCO. De meeste hiervan zijn gelinkt aan één of meerdere van de onderzochte criteria. De relevante kostencomponenten worden hieronder omschreven en besproken. Er zal niet dieper ingegaan worden op licentiekosten, omdat deze reeds besproken zijn.

3.3.1 TCO-componenten

3.3.1.a Aanschafkosten

Behalve de licentiekosten, die bij CSS enorm hoog liggen en bij CSS onbestaand zijn, onderscheiden we in de eerste plaats de aanschafkosten. Terwijl CSS vaak enorm duur is in aankoop, is OSS in principe gratis. De reden hiervoor is dat OSS door communities

beheerd wordt en dat de ontwikkelaars van de software door een OSS-licentie zodanig veel rechten aan de gebruiker toekennen, dat ze met de software vrijwel alles mogen doen. Een uitzondering op de afwezigheid van aanschafkosten doet zich voor bij distributiepakketten. De distributeurs bundelen dan een heleboel OSS in een pakket, dat tegen betaling wordt geleverd (Baarsma, 2004). De prijs van dergelijke bundels is echter verwaarloosbaar wanneer vergeleken met die van closed source equivalenten. Zo kost de desktopversie van SUSE Linux bijvoorbeeld slechts 50$ (Novell, 2006). Deze bundel bevat alle mogelijke OSS-applicaties die een doorsnee gebruiker zich maar kan inbeelden en nog veel meer. De applicaties gaan van kantoorapplicaties tot mail-programma's, van mp3-spelers tot handheld-synchronisatie, van spelletjes tot grafische applicaties, enzovoort. Microsoft Windows XP Pro daarentegen kost 299$ en biedt behalve een internetbrowser, mailprogramma en een paar kaartspelen, niet meer dan de basisfuncties van een besturingssysteem. Kantoorapplicaties bijvoorbeeld, dienen apart aangeschaft te worden. Zo kost Microsoft Office Professional, het pakket met kantoorapplicaties voor ondernemingen, 499$ (Microsoft, 2006). Het is dus duidelijk dat OSS ook op dit gebied enorm voordelig is.

3.3.1.b Hardwarekosten

Een tweede soort kosten zijn de hardwarekosten. Deze kosten worden vooral beïnvloed door de flexibiliteit van de software. Omdat OSS niet platformafhankelijk is en CSS wel, kunnen ook deze kosten gedrukt worden. De gebruiker is immers niet meer gebonden aan bepaalde leveranciers wat hardware betreft, en kan dus op de markt op zoek gaan naar de goedkoopste aanbieder. Daarenboven is het zo dat OSS minder hoge systeemvereisten

heeft dan CSS. De gebruiker zal dus niet zo'n hoge systeemspecificaties nodig hebben om de OSS vlot te draaien en kan daardoor dus goedkopere hardware gebruiken dan wanneer hij CSS zou gebruiken (Knubben, 2004). Een belangrijke reden hiervoor is dat bij OSS onnodige functies weggelaten kunnen worden, terwijl bij CSS een heleboel toeters en bellen ingebakken kunnen zitten die de gebruiker allerminst nodig heeft. Smets bevestigt in zijn interview dat door gebruik van OSS, dezelfde prestaties kunnen geleverd worden door goedkopere en/of oudere hardware.

3.3.1.c Zoekkosten

Daarnaast zijn er ook nog de zoekkosten (Knubben, 2001). Voor OSS wordt niet zoveel reclame gemaakt als voor CSS. Smets wijt dit onevenwicht aan de enorme marketing-budgetten waarover de grote namen in CSS-ontwikkeling beschikken. De bekendheid van OSS bij het brede publiek is daarom minder groot dan die van de meeste CSS. Het kan dus wat tijd kosten om de juiste OSS te vinden, terwijl het in sommige gevallen misschien zelfs mogelijk is dat er geen OSS-alternatief voorhanden is. Het is echter vanzelfsprekend dat deze zoekkosten tegenwoordig, vooral door het sterk geëvolueerde Internet, blijven afnemen en uiteindelijk verwaarloosbaar zullen worden.

3.3.1.d Operationele kosten

Professionelen dragen OSS hoog in het vaandel omwille van de hoge kwaliteit (Godden, 2000). Een rechtstreeks gevolg van een kwalitatief hoogstaande broncode is stabiliteit van de software. Zoals reeds eerder aangegeven, is deze in het algemeen veel hoger bij OSS

dan bij CSS. Volgens Knubben (2001) vermindert dit de kosten van onderhoud, terwijl Baarsma (2004) zegt dat de kosten veroorzaakt door down-time (de onbruikbaarheid van een computer door technische problemen) hierdoor ook sterk zullen afnemen. We kunnen dus besluiten dat de operationele kosten bij OSS lager liggen dan bij CSS.

Smets maakt tijdens zijn interview echter een belangrijke randopmerking. Operationele kosten bevatten een kosten-component die moeilijk te meten valt, namelijk de kost van inertie. Mensen willen vaak geen afstand doen van de software waar ze al jaren mee werken. Zo geeft van der Kleij als voorbeeld dat de directie van een onderneming die hij (met succes) hielp overstappen op OSS, weigerde om Microsoft Windows en Microsoft Office op hun laptops te laten vervangen door Linux en OpenOffice.org.

3.3.1.e Opleidingskosten

De opleidingskosten hangen volgens Baarsma (2004) sterk samen met de kwaliteit van de software en de flexibiliteit van de gebruikers (het personeel). Een voorbeeld uit een onderzoek (Knubben, 2004) waarbij systeembeheerders na een migratie naar OSS relatief snel en zonder het veroorzaken van noemenswaardige kosten, experts werden op het vlak van Linux, toont aan dat opleidingskosten van ICT-personeel niet altijd hoger liggen bij OSS. De systeembeheerders in kwestie waren echter gemotiveerde werkkrachten die zich vooral op het Internet stortten wanneer ze nood hadden aan informatie. Zijn leidden dus in principe zichzelf op, zonder betaalde tussenkomst van derden. Hierdoor blijft de kostenstijging beperkt tot de tijd die verloren gaat door het opzoekingswerk. Bij complexere OSS zal het ICT-personeel echter ook specifieke opleidingen moeten volgen,

wat uiteraard wel kosten met zich mee zal brengen. Natuurlijk is dit ook voor CSS zo. Op desktopniveau zou een migratie volgens Knubben (2001) geen echte kosten met zich meebrengen, omdat OSS op desktopniveau tegenwoordig voorzien is van een grafische gebruikersinterface die gebaseerd is op de meest populaire CSS-variant van de applicatie in kwestie. Baarsma (2004) daarentegen beweert dat sommige onderzoeken aantonen dat de opleidingskosten voor het gewone personeel bij OSS 150 tot 190% van de opleidingskosten bij CSS bedragen. Ze verwijst echter ook naar onderzoeken waaruit blijkt dat de opleidingskosten bij CSS en OSS niet substantieel verschillen en geeft zelfs aan dat de opleidingskosten bij OSS lager zouden kunnen liggen, als het personeel zelf de moeite doet om informatie op het Internet op te zoeken.

3.3.1.f Ondersteuningkosten

Wat ondersteuningskosten betreft, geeft Knubben (2001) aan dat ondersteuning voor de Linux-besturingssytemen en Apache (OSS voor webservers) van aanvaardbaar niveau is. Volgens hem zal OSS hier echter geen kostenvoordelen meebrengen, hoewel er ook weer op dit gebied veel via Internet kan opgezocht worden. Ook Baarsma (2004) is het hiermee eens. Zij is zelfs van mening dat ondersteuning voor OSS vaak meer kosten zal veroorzaken dan die van CSS. De reden hiervoor is volgens haar dat informatici die met OSS werken vaak hoger opgeleid zijn dan diegene die met CSS werken. Dit zou het gevolg zijn van de hogere specialisatie die nodig is voor het beheer van OSS. Volgens een onderzoek (Knubben, 2004) zal het aantal OSS-experten in de nabije toekomst sterk stijgen, waardoor hun lonen zullen dalen. Dit laatste zal natuurlijk een invloed hebben op de ondersteuningskosten die OSS met zich meebrengt. Uit datzelfde onderzoek blijkt ook

dat de kosten van technisch personeel voor ondersteuning tussen de 35 en de 60% van de totale kosten van het informatiesysteem van een onderneming uitmaken. Volgens Feller en Fitzgerald (2002) zouden deze kosten zelfs 70 tot 80% van de Total Cost of Ownership kunnen bedragen. Een TCO-component waar dus zeker rekening mee gehouden moet worden.

3.3.1.g Overstapkosten

Een laatste soort kosten die we kunnen onderscheiden zijn de overstapkosten of 'switching costs'. Hiermee wordt niet alleen de overstap van OSS naar CSS bedoeld, maar ook de overstap naar een nieuwe versie van een applicatie of naar een volledig andere applicatie. De eenmalige overstapkosten door een volledige migratie van een Microsoft-omgeving (zoals die in vele ondernemingen aanwezig is) naar een OSS-omgeving liggen relatief hoog. Deze kosten hebben vooral betrekking op de aanpassing van kennis, werkprocessen en beheer. De overstapkosten kunnen echter beperkt worden door een gedeeltelijke migratie, waarbij CSS alleen door OSS vervangen wordt, als deze laatste ook effectief verbeteringen en kostenbesparingen met zich meebrengt (Baarsma, 2004). De overstapkosten veroorzaakt door latere (dus na de migratie naar OSS) overstappen op andere of nieuwere OSS, liggen volgens Laan (2003) veel lager dan bij CSS, omdat ondernemingen bijvoorbeeld niet meer moeten wachten tot de licenties van de software verlopen zijn (want die zijn er niet bij OSS) en tot de andere investeringen in software zijn afgeschreven. Door de flexibiliteit van OSS kan deze ook aangepast worden aan veranderende behoeften van de onderneming. De levensduur van de software stijgt, wat de frequentie van het overstappen vermindert.

3.3.2 IDABC-spreadsheet & simulatie

Uit de analyse van de verschillende TCO-componenten blijkt dat er zowel kostenbesparingen als kostenstijgingen kunnen optreden wanneer een CSS-omgeving volledig omgezet zou worden naar een OSS-omgeving. Uiteraard is een initiële kostenstijging geen reden om te concluderen dat een overstap ook in de toekomst niet voordeliger zal zijn. Een begrip dat belangrijk kan zijn in deze problematiek is de payback-periode.

3.3.2.a IDABC-spreadsheet

IDABC is een programma van het Europees Parlement dat loopt van 2005 tot 2009 en dat onderzoek voert naar het gebruik van electronische overheidsdiensten. In het kader van dit programma werd een spreadsheet ontworpen die de kosten van informatiesystemen berekent, zowel onder CSS als onder OSS, en waarbij de kost en payback periode van een (volledige) migratie naar OSS afgeleid wordt. Dit gebeurt door variabelen in te vullen in de spreadsheet. Het gaat hier om basisgegevens zoals het aantal gebruikers, computers, netwerken, vestigingen, upgrades en dergelijke, die dan weer gespecifiëerd kunnen worden op andere werkbladen. Op deze werkbladen kunnen dan ook de gedetailleerde kostengegevens geraadpleegd worden. De spreadsheet en daarbij horende handleiding zijn beschikbaar op de website (IDABC, 2006).

Het is niet de bedoeling geweest van het IDABC om de spreadsheet te laten gelden als een allesomvattend TCO-model waarbij de verschillende componenten exacte bedragen voorstellen. De opzet van deze spreadsheet is daarentegen wel om beslissingsnemers in

ondernemingen een idee te geven van de belangrijkste kosten-determinanten van zowel CSS- als OSS-omgevingen en de belangrijkste verschillen tussen deze twee zichtbaar te maken (IDABC, 2006).

De spreadsheet is daarom in het kader van deze thesis misschien een interessant instrument voor beslissingsnemers in ondernemingen die een beter idee willen krijgen van de kosteneffecten die een migratie met zich meebrengt. Zij kunnen deze spreadsheet daarenboven ook gebruiken om een schatting te bekomen van de payback-periode voor een (volledige) migratie naar OSS binnen hun onderneming.

3.3.2.b Een voorbeeld

Om een idee te geven van de informatie die de spreadsheet kan leveren, is een voorbeeld opgenomen. Hieronder vindt U de tabellen uit het samenvattende werblad. Volgens de ontwerpers van deze spreadsheet zijn er 5 hoofdvariabelen die belangrijk zijn wanneer we het hebben over kosteneffecten. Het betreft het aantal interne computers, het aantal computers voor thuisgebruik (deze liggen buiten de firewall van de vestiging en hebben dus extra software nodig), het totaal aantal gebruikers, het aantal vestigingen met servers en het tijdstip van een eventuele upgrade in de 5 jaar waarover de spreadsheet loopt. De meer gedetailleerde variabelen zoals de gangbare kostprijs van een desktop-computer of server (die ingegeven kunnen worden op de andere werkbladen) worden behouden zoals ze door de IDABC-werkgroep in de spreadsheet zijn opgenomen. Deze gegevens zijn Europese gemiddelden en kunnen dus als algemene waarden gebruikt worden. Op de tabel

in Figuur 5 hieronder ziet U het standaard rekenvoorbeeld, zoals het ingegeven is in de spreadsheet.

Figuur 5: De 5 hoofdvariabelen

Number of Internal Desktops	**3.500**	Enter the total number of desktop machines connected to the Administration's network behind firewalls.
No of Remote /Home Desktops	**100**	Enter the total number of desktops machines which are connected via a WAN
Total Number of users	**4.000**	Enter the total number of users both internal and remote
No of sites	**10**	Enter the number of sites with server installations.
Year of Proprietary Upgrade	**3**	Enter the year in which the proprietary software is assumed to be upgraded.

Bron: IDABC (2006)

Het invoeren van deze gegevens resulteert op het samenvattende werkblad in 3 tabellen, namelijk de kosten voor een dergelijke onderneming indien zij CSS zou gebruiken, indien zij OSS zou gebruiken en tenslotte de kosten van een volledige migratie naar OSS en de eventuele payback-periode. Deze tabellen vindt U in Figuur 6, 7 en 8

Figuur 6: Kosten van commerciële software

	YEAR	1	2	3	4
Cost of Proprietary	Hardware	1.333.754	1.333.754	2.365.754	618.129
	Software	1.224.400	1.224.400	3.258.100	1.244.231
	People	4.298.582	4.298.582	4.844.255	4.254.712
	Total	6.856.735	6.856.735	10.468.108	6.117.071

Bron: IDABC (2006)

Figuur 7: Kosten van OSS

Cost of FOSS	Hardware	957.304	957.304	957.304	957.304
	Software	0	0	0	0
	People	1.340.893	1.340.893	1.340.893	1.340.893
	Total	2.298.197	2.298.197	2.298.197	2.298.197

Bron: IDABC (2006)

Figuur 8: Kosten van migratie en payback-periode

Cost of Migration	Hardware	109.250	14.250	14.250	14.250
	Software	2.480.000	0	0	0
	People	1.605.000	20.000	20.000	20.000
	Total	4.194.250	34.250	34.250	34.250
Payback Period	1 Year				

Bron: IDABC (2006)

Na een beetje experimenteren met de spreadsheet, werd mij al snel duidelijk dat veranderingen in de eerste 3 hoofdvariabelen (vooral het aantal gebruikers en het aantal interne computers), volgens deze spreadsheet, het meeste effect hebben op de payback-periode. Ik contacteerde het IDABC hieromtrent. De ontwikkelaars bevestigden dat het investeren in een overstap voor ondernemingen met meer computers en gebruikers sneller voor een positieve netto contante waarde zal zorgen, dan wanneer deze aantallen in de onderneming eerder beperkt zijn.

3.4 Conclusies

Hieronder vindt U de beknopte conclusies die getrokken kunnen worden na de evaluatie van OSS op basis van de 9 criteria.

- De kwaliteit van OSS variëert, net als bij CSS, sterk van product tot product. Terwijl OSS op gebied van efficiëntie en effectiviteit zeker niet moet onderdoen voor CSS, zal zij op gebied van stabiliteit in het algemeen zelfs beter scoren.
- Wat veiligheid betreft garandeert noch het bazaar-model bij OSS, noch het principe van security by obscurity bij CSS, de waterdichtheid van de broncode. Een voordeel dat OSS hier echter biedt, is dat er veel dynamischer kan ingespeeld worden op lekken in de broncode. In de toekomst moet echter wel rekening gehouden worden met het stijgende gebruik van OSS, dat ervoor zal zorgen dat ook deze software een populairder doelwit zal worden voor malafide hackers.

- De flexibiliteit van OSS is dankzij de vrijheid van de broncode veel hoger dan bij CSS. Op deze manier kunnen tekortkomingen in het informatiesysteem opgevangen worden en zijn compatibiliteit en portabiliteit gegarandeerd.
- Vooral professionele ondersteuning naar bedrijven toe is minder aanwezig voor OSS dan voor CSS. Het aantal aanbieders stijgt echter samen met de populariteit van OSS.
- OSS is minder beschikbaar dan CSS. Er is niet voor elk stukje CSS een goed OSS-alternatief. Omwille van de hoge flexibiliteit van OSS, is het echter mogelijk om hybride informatiesystemen op te bouwen, waarbij OSS en CSS mekaar aanvullen. Dankzij aanpassingen in de broncode van de gebruikte OSS kan zelfs ingespeeld worden op tekortkomingen van de gebruikte CSS.
- Op gebied van servers vormt de gebruiksvriendelijkheid van OSS geen probleem. Op desktop-niveau blijft OSS echter nog steeds een beetje achterop hinken. Een goede Internet-community en toenemende aandacht van de OSS-ontwikkelaars zorgen echter voor veel vooruitgang op dit vlak.
- Van leveranciersafhankelijkheid is bij OSS geen sprake. Bij het gebruik van OSS is er dus ook geen gevaar voor vendor lock-in. De onderneming blijft vrij in al haar keuzes en zorgt er door het gebruik van OSS voor dat de marktmechanismen in de software-markt beter werken.
- De levensduur van OSS is opmerkelijk langer dan die van CSS. Redenen hiervoor zijn de afwezigheid van verplichte upgrades, compatibiliteit en portabiliteit van de software en de mogelijkheid tot het aanpassen van de broncode aan de veranderende wensen en behoeften van de onderneming.

Op gebied van kosten kunnen we twee groepen onderscheiden. Vooreerst is er de groep waarbij gegarandeerde kostenbesparingen behaald kunnen worden. Daarnaast is er echter ook een groep kosten waarbij het uiteindelijke resultaat in elke situatie zal verschillen. Deze twee groepen worden hieronder kort besproken.

Kostenbesparingen worden door een overstap naar OSS in eerste instantie veroorzaakt op gebied van licentie- en aanschafkosten. Bij OSS zijn deze immers (mits uitzonderingen) afwezig. Hardwarekosten liggen ook opmerkelijk lager omdat OSS minder hoge systeemvereisten stelt dan CSS en dus goedkopere hardware kan aangekocht worden die daarenboven ook minder snel vervangen moet worden. Ook de operationele kosten liggen lager bij OSS, vooral omwille van hogere stabiliteit.

Zoekkosten daarentegen, zullen bij OSS hoger liggen dan bij CSS, als gevolg van de marketingcampagnes die de meest populaire CSS ondersteunen. Dankzij het Internet worden deze zoekkosten bij OSS stilaan verwaarloosbaar. De opleidingskosten kunnen ook een lichte stijging kennen bij een migratie naar OSS. Reden hiervoor is de hogere graad van specialisatie van het ICT-personeel. Wat de ondersteuningskosten betreft, kunnen we besluiten dat zij hoger kunnen liggen omdat er minder aanbieders zijn voor ondersteuning bij OSS dan bij CSS. Het aanbod neemt echter toe, wat resulteert in dalende ondersteuningskosten voor OSS. De overstapskosten voor een onderneming bij een migratie van CSS naar OSS zijn initiëel relatief hoog. Eens de overstap gemaakt is, zullen

andere overstappen (naar een ander hardware-platform, ander besturingssysteem, enzovoort) echter minder kosten veroorzaken dan in een CSS-omgeving het geval is.

H4 Praktijkstudie

4.1 Inleiding

Tot nu toe werd de migratie naar OSS op een theoretische manier benaderd. Alhoewel een theoretische basis zeer belangrijk is in een onderzoek als het deze, wordt het pas echt interessant als die theorie ook gekoppeld kan worden aan een geval uit de werkelijkheid. Daarom ben ik op zoek gegaan naar een onderneming die zich aan een (significante) overstap gewaagd heeft. Ik kwam terecht bij Michel van der Kleij, IT-consultant van een Nederlandse onderneming, die sinds een aantal jaren een groot gedeelte van de commerciële software die zij voordien gebruikte, vervangen heeft door OSS. Ze hebben er, naar eigen zeggen, nog geen seconde spijt van gehad.

4.2 Voorstelling van de onderneming

De onderneming die voor deze gevalstudie gekozen werd is Euroherbs B.V., gelegen in Westervoort, Nederland. Euroherbs is zowel importeur als exporteur van Chinese geneeskrachtige kruiden en aanverwante artikelen. Het klantenbestand is zeer uitgebreid en strekt zich uit over therapeuten en particulieren over bijna heel Europa. Nederland, België en Duitstalige landen zijn de belangrijkste partnerlanden. De leveranciers van Euroherbs bevinden zich in de Verenigde Staten, het Verenigd Koninkrijk, België en vooral het Verre Oosten.

4.3 Uitgangssituatie

Tot net voor de eeuwwisseling werkte Euroherbs met een Windows-omgeving en stand-alone pc's. Het besturingssysteem op deze stand-alone pc's was Windows 98. De belangrijkste software was grotendeels aangekocht of (in mindere mate) zelf gemaakt. Euroherbs gebruikte een boehoudpakket (Profile) met mogelijkheden voor loonadministratie, dat zij van een klein bedrijfje in Arnhem hadden gekocht. Dat bedrijfje stond ook in voor het onderhoud van het pakket. Er was ook een zelfgemaakte Dbase IV databank waarin productinformatie verzameld kon worden en die gebruikt werd voor het afdrukken van productlabels. Voor de verschillende kantoorwerkzaamheden werd gebruik gemaakt van Microsoft Office en Word Perfect. Het volledige informatiesysteem draaide dus op commerciële software. Van Open Source was geen sprake. Daarenboven was er zelfs geen netwerk aanwezig om de verschillende computers met elkaar te verbinden. Als je iets wilde doen waarvoor jouw computer niet was ingesteld, moest je dus gewoon op zoek gaan naar een computer in het bedrijf die dat wel kon. Ook het delen van bestanden was een bijna onmogelijke taak. Disketten waren in die tijd de enige interessante (en goedkope) media om bestanden mee te verzeulen. Deze manier van werken met computers stamde nog uit het 'tijdperk' dat de drie oprichters van de onderneming op de zolder boven de garage kruiden verkochten.

Het primaire bedrijfsproces is sinds toen eigenlijk niet in die mate veranderd dat er nood was aan een volledig nieuw informatiesysteem. De informatiestroom bij Euroherbs komt tegenwoordig nog steeds overeen met de gangbare manier van werken in vele bedrijven,

namelijk het ontvangen van informatie uit diverse bronnen en het verwerken van deze informatie, om dan uiteindelijk bestellingen te produceren en te versturen naar de klanten.

Op een gegeven moment werd echter duidelijk dat het informatiesysteem de bedrijfsprocessen niet langer efficiënt zou kunnen ondersteunen en dat het huidige informatiesysteem niet mee zou kunnen met de expansie van Euroherbs die stilaan op gang kwam. Bovendien waren de kosten van het informatiesysteem niet in verhouding met de toegevoegde waarde die het systeem creëerde. De conversie naar de Euro in de nabije toekomst en de vrees voor de gevolgen van de 'millenium-bug' samen met de wens om in de toekomst meer met Internet te gaan werken (bijvoorbeeld de mogelijkheid bieden aan klanten om online bestellingen te plaatsen) zorgden ervoor dat de knoop werd doorgehakt: het informatiesysteem van Euroherbs zou een ingrijpende verjongingskuur ondergaan! De directie had zich reeds geïnformeerd omtrent OSS en besloot dat het informatiesysteem in elk geval op Linux-fundamenten gebouwd zou worden.

4.4 Het nieuwe informatiesysteem

4.4.1 Hardware

De vernieuwing van het informatiesysteem ging gepaard met een verandering van de infrastructuur. Deze is tegenwoordig gebaseerd op een aantal 80x86 Linux servers en een 100Mbit netwerk. De meeste primaire en secundaire bedrijfsprocessen worden door deze servers ondersteund. De belangrijkste van deze servers is de applicatie-server. De andere

verrichten meer ondersteunende taken. Alle clients, waaronder ook nog enkele Windows-clients, zijn op dit netwerk aangesloten.

4.4.2 Commerciële software

4.4.2.a Linux

Op gebied van een aantal applicaties werd voor Linux gekozen. Het oude boekhoudpakket werd vervangen door Cash (Cash Software, Den Haag). Cash staat centraal in het huidige informatiesysteem van Euroherbs. Cash is eigenlijk veel meer dan alleen maar een boekhoudpakket. Naast financiële administratie, neem dit softwarepakket ook voorraad- en relatiebeheer voor haar rekening. We kunnen dus spreken van een soort ERP-software (Enterprise Resource Planning), ofwel functieoverschrijdende software die verschillende bedrijfsprocessen integreert en automatiseert (O'Brien, 2003). De keuze voor dit betalende pakket werd gemaakt, omdat er geen OSS op de markt was die dezelfde functionaliteit bood als Cash. Het gebruik van GNUcash bijvoorbeeld, was een optie, maar er zou dan zodanig veel aan de broncode gesleuteld moeten worden dat de keuze voor Cash voor de hand liggend was. Het is immers een pakket met veel mogelijkheden en is daarenboven betaalbaar. Daar tegenover staat natuurlijk wel dat het niet gaat om maatwerk, maar om een gestandardiseerd pakket. Op het eerste zicht waren er bij Cash dan ook een aantal tekortkomingen, maar volgens van der Kleij heeft de flexibiliteit van OSS in het algemeen, en Linux in het bijzonder, ervoor gezorgd dat een betaalbaar en flexibel administratief systeem op poten gezet kon worden.

Een andere Linux-applicatie waar even mee geëxperimenteerd werd is VMware. Deze virtualisatie-software zorgt ervoor dat je op één fysieke machine, twee verschillende besturingssystemen kan draaien. Op een computer met Linux als besturingssysteem, kan je dan VMware installeren om zo bijvoorbeeld Windows XP in een venster op je bureaublad te draaien. Op die manier zou er dus, als er nog nood zou zijn aan een computer met Windows erop, geen extra hardware aangekocht moeten worden. De prestaties van de hardware waren met VMware echter ondermaats. De directie besloot daarom een aantal goedkope computers te kopen om Windows te draaien waar dit noodzakelijk is.

4.4.2.b Windows

Een onderneming zoals Euroherbs ondervindt tijdens een migratie al snel dat er een heel aantal factoren zijn die het gebruik van Windows in de hand werken. Zo is de software die Nederlandse banken aanbieden om aan ‘online banking’ te doen enkel bruikbaar op een Windows-besturingssysteem. Ook overheidsinstelligen werken op dit gebied niet mee. Zo stelt het CBS (Centraal Bureau voor de Statistiek) enkel een Windows-versie ter beschikking van de software die verplicht (!) import- en exportgegevens door moet geven.

Omdat ook de clients die het Windows-besturingssysteem draaien moeten kunnen gebruik maken van Cash, is via het Internet ‘TigerTerm’ aangeschaft. TigerTerm zorgt voor de terminal-emulatie. Dit houdt in dat een computer, op afstand, als een terminal van een andere computer functioneert. De gebruiker ziet op zijn scherm dan hetzelfde als wat hij zou zien wanneer hij de Linux-client zou gebruiken om op Cash in te loggen.

Het pakket voor de loonadministratie is nog steeds het DOS-programma van weleer en draait zeer stabiel onder Windows 98. Omdat het momenteel nog voldoet aan de eisen die de directie heeft op gebied van loonadministratie, werd het niet vervangen en is men ook nog niet op zoek naar een andere applicatie.

Hiernaast worden nog een aantal alledaagse commerciële pakketten ingezet zoals anti-virus-software, drivers voor bepaalde printers, Microsoft Publisher voor de vormgeving van de website en Adobe Acrobat Pro om PDF-documenten te genereren.

4.4.3 Open Source Software

Om de belangrijkste OSS in één woord op te sommen, gebruikt van der Kleij het acroniem 'LAMP'. Dit staat voor Linux, Apache, MySQL en Perl. Deze combinatie vormt de schil rondom Cash en is dus van groot belang voor de onderneming.

Als besturingssysteem werd resoluut gekozen voor een Linux-distributie. Eerst werd geopteerd voor Redhat 9, een commerciëel ondersteunde distributie. Omdat Euroherbs geen grote onderneming is en het aantal gebruikers beperkt blijft tot maximaal 12, werd deze uiteindelijk vervangen door Fedora Core, de gratis variant van Redhat. Volgens van der Kleij werd voor dit besturingssysteem gekozen omdat het één van de meest complete distributies is en omdat Red Hat, een gerenomeerde naam in de OSS-wereld, achter dit product staat. Daarenboven beschikt Fedora Core over een sterke en uitgebreide community, wat resulteert in goede online ondersteuning. Dit zorgde reeds een aantal keren voor de ondersteuning die van der Kleij nodig had. Zo was er bij de migratie een

probleem met de afhandeling van de printer-poort. Om dit op te lossen heeft van der Kleij toen, met de hulp van de online community van Fedora Core, de kernel (basis van een besturingssyteem) zodanig aangepast dat dit probleem van de kaart was.

Apache is een Open Source web-server die ontwikkeld en onderhouden wordt door de Apache Software Foundation en die gebruikt kan worden op eender welk besturingssysteem. Sinds de eerste versie in 1995 het licht zag, is deze webserver uitgegroeid tot de meest populaire uit het aanbod (Kabir, 1998). In mei 2005 vertrouwde 70% van de websites wereldwijd op Apache en het gebruik ervan neemt nog steeds toe. Apache kan gebruikt worden in combinatie met verschillende programmeer- en scripttalen voor het gebruik van web-applicaties. Sinds de release van de 2.0 versie is de populariteit nog toegenomen, omdat Apache nu door Windows-gebruiker threaded (dus op de achtergrond) kan gebruikt worden. Dit komt de prestaties van de hardware ten goede. (Wheeler, 2005)

MySQL is een populair, relationeel databasemanagementsysteem dat zorgt voor de (op basis van tabellen) gestructureerde opslag van informatie. Hiermee wordt bedoeld dat ook de verbanden tussen de verschillende gegevens mee opgeslagen worden. Hierdoor kan de benodigde informatie door middel van queries (vraagformulieren) specifiek en zelfs in verwerkte vorm opgevraagd worden. (MySQL, 2005)

Perl (Practical Extraction and Reporting Language) is volgens van der Kleij het noodzakelijke bindmiddel tussen de verschillende applicaties. Het is een uitermate

flexibele scripting- en programmeertaal, die de mogelijkheden van het eerder vernoemde ‘C’ combineert met tal van andere mogelijkheden.

Ook voor kantoorwerkzaamheden wordt gebruikt gemaakt van OSS. Zo is OpenOffice, zowel op de Linux- als Windows-clients, de vervanger van Microsoft Office geworden. OpenOffice is volgens van der Kleij een zeer functioneel en gratis OSS-alternatief voor de commerciële versies die hun hoge aanschaf- en licentiekosten niet kunnen vertalen in betere kwaliteit. Voor het gebruik van het Intranet en Internet worden Netscape en Mozilla gebruikt.

Daarnaast wordt gebruik gemaakt van OpenSSH om op een veilige manier via de terminal van Linux-clients in te loggen op Cash en voor andere taken op het netwerk. SSH staat voor ‘Secure Shell’ en is dus in feite niet meer dan een beveiliging voor de terminal (communicatie-venster). Voor het sturen en ontvangen van mails wordt Fetchmail (zie Fetchmail-project van Raymond) gebruikt. Wat betreft het maken van backups, is Rsync ingeschakeld. Deze applicaties zijn volgens van der Kleij allemaal zeer stabiel en schieten op gebied van functionaliteit nergens tekort. Rest mij nog even op te merken dat de OSS die in deze paragraaf opgenoemd werd, makkelijk te downloaden is van het Internet. Ondernemingen kunnen deze dus zonder veel moeite uittesten. Hoe deze OSS ingezet wordt bij Euroherbs, wordt in het volgende gedeelte uit de doeken gedaan.

4.5 Werking van de mix

In dit gedeelte zal zo duidelijk mogelijk uitgelegd worden hoe en waarom deze mix van OSS en commerciële software zo goed haar werk doet.

4.5.1 Financiële administratie & integratie

Voor Euroherbs is gebruiksvriendelijkheid zeer belangrijk. Alle applicaties moeten makkelijk bedienbaar zijn voor de personeelsleden. Cash is bijvoorbeeld vanaf de Linux-clients bruikbaar gemaakt door de terminal, die op zijn beurt beveiligd is door SSH (meer bepaald OpenSSH, de Open Source versie). Op deze clients wordt dus gebruik gemaakt van de (ingebouwde) console van het Linux-besturingssyteem om in te loggen en is dus geen terminal-emulatie vereist. Hierdoor heeft Euroherbs een heleboel licentiekosten voor emulatiesoftware kunnen uitsparen. De personeelsleden voeren in de terminal hun login en paswoord in, waarna deze verbinding maakt met Cash op de applicatie-server. Op het scherm van de Linux-client wordt dan een venster met daarin de interface van Cash gestart. Op Figuur 9 onder deze paragraaf kunt U zien hoe die vensters eruit zien. Initieel vormde dit een probleem, omdat Cash logins en paswoorden op een niet-standaard manier afhandelt. Hiervoor werd in Perl een login-script geschreven, dat nu instaat voor de configuratie van de emulatie voor Cash.

Figuur 9: Cash

Bron: van der Kleij (2004)

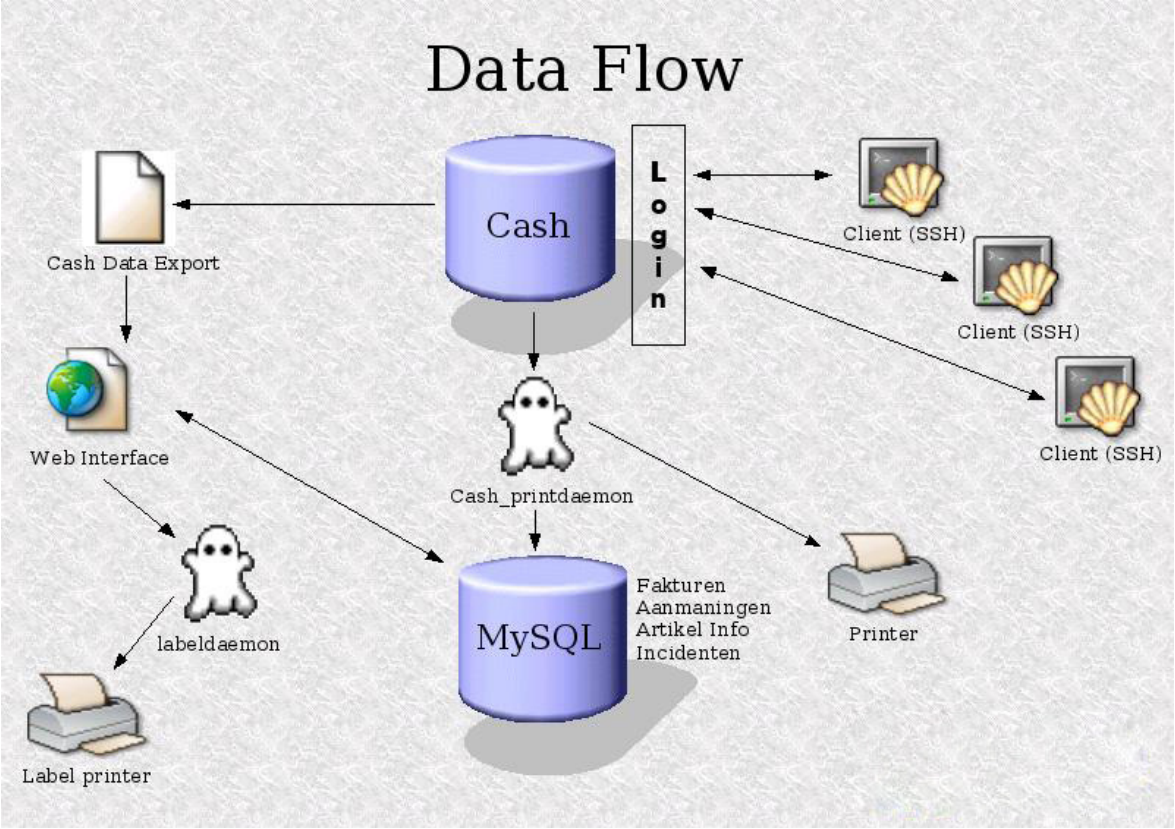
Een aantal andere taken, waaronder het afdrukken van product-labels, gebeurt via een web-applicatie op het Intranet. Deze web-applicatie draait op de Apache web-server. Voordat deze labels afgedrukt kunnen worden, moeten een aantal gegevens uit de database van Cash gehaald worden, zodat kan doorgegeven worden wat er op de labels moet staan. Ook hier ontstond een probleem. Cash is namelijk bedoeld om interactief gebruikt te worden en door het gesloten, interne formaat van de gegevens, zijn deze niet bruikbaar voor export naar een andere toepassing (in dit geval het script voor het printen van labels, namelijk labeldaemon). Al snel werd hier een mouw aan gepast. Cash bezit immers wel een functie waardoor data in eenvoudige tekstvorm kan worden geëxporteerd. De applicatie doet dit nu op regelmatige basis. Het resultaat van die export kan dan door verschillende applicaties

geconverteerd worden voor het gebruik ervan. Voor deze export moet echter wel een aparte Cash-licentie gebruikt worden, wat uiteraard licentiekosten met zich meebrengt. Deze zijn volgens van der Kleij echter te verwaarlozen, omdat ze niet opwegen tegen het voordeel dat deze techniek met zich meebrengt: perfecte integratie van de gegevens uit Cash in alle andere gebruikte applicaties.

Iets gelijkaardigs deed zich ook voor bij de verwerking van facturen en aanmaningen. Binnen Cash hebben deze twee soorten 'formulieren' een aantal onhandige eigenschappen (die vanuit boekhoudkundig perspectief kunnen verklaard worden, maar waar we niet op ingaan). Zo zijn de kopies van de facturen die worden gegenereerd niet altijd hetzelfde als het origineel en kunnen aanmaningen niet meer opnieuw ingekeken worden als dat reeds gebeurd is. De handmatige verwerking van de facturen en aanmanigen op papier kon een oplossing bieden, maar zou het personeel van Euroherbs een heleboel extra werk bezorgen. Om dit probleem op te lossen werd besloten om alle facturen en aanmaningen voortaan in een MySQL-database op te slaan. Alle printopdrachten uit Cash gebeuren nu via een Perl-script, namelijk Cash_printdaemon. Deze is uitgerust met de kennis om facturen en aanmaningen te herkennen en, naast afdrukken, ook op te slaan in de MySQL-database. Via de web-interface kunnen dan de facturen en aanmaningen door elk personeelslid opnieuw opgevraagd worden om af te drukken of gewoon te bekijken. Op Figuur 10 hieronder vindt U een schematische voorstelling van de data-flow voor financiële administratie binnen Euroherbs. Hierop is duidelijk zichtbaar hoe OSS (in dit specifieke geval LAMP) ervoor zorgt dat zowel commerciële software als OSS beter ingezet kan

worden in een specifieke omgeving. Dit schema werd ontwikkeld door Michel van der Kleij.

Figuur10: Data flow voor financiële administratie



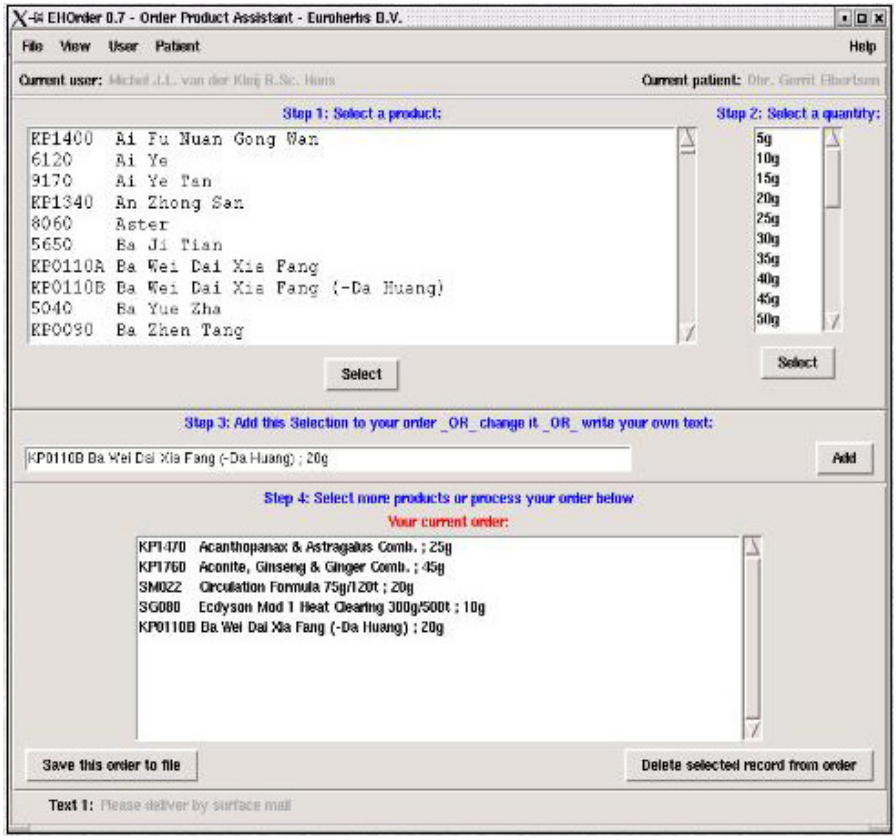
Bron: van der Kleij (2004)

4.5.2 Orderverwerking & portabiliteit

Bij het plaatsen van orders wordt, voor het meerendeel van de producten, gebruikt gemaakt van Chinese, Latijnse en Engelse namen. Om dit orderproces te ondersteunen en eenvoudiger te maken, zowel voor de klanten als voor Euroherbs zelf, werd een applicatie ontwikkeld in Perl/Tk. Tk is een extensie voor Perl die op een relatief eenvoudige manier

toelaat om een grafische interface aan een applicatie toe te voegen. De ontwikkeling van deze applicatie, namelijk EHOrder, duurde dankzij de kracht van de scripting-taal en de eenvoud van Tk slechts enkele dagen. Deze applicatie draait onveranderd onder zowel Linux als Windows. Omdat de applicatie in Perl is geschreven, kan deze ook heel makkelijk geport worden naar bijvoorbeeld een applicatie op het Intranet of op de website. Dankzij de gebruikte OSS kunnen we hier dus spreken van 100% portabiliteit, wat in het geval van toekomstige veranderingen aan hard- en software zal zorgen voor verhoogde compatibiliteit en lagere (tot zelfs geen) overstapkosten. Op Figuur 11 ziet U de gebruiksvriendelijke interface van de zelfgemaakte applicatie voor orderverwerking.

Figuur 11: EHOrder



Bron: van der Kleij (2004)

4.5.3 Conversie & tijdsefficiëntie

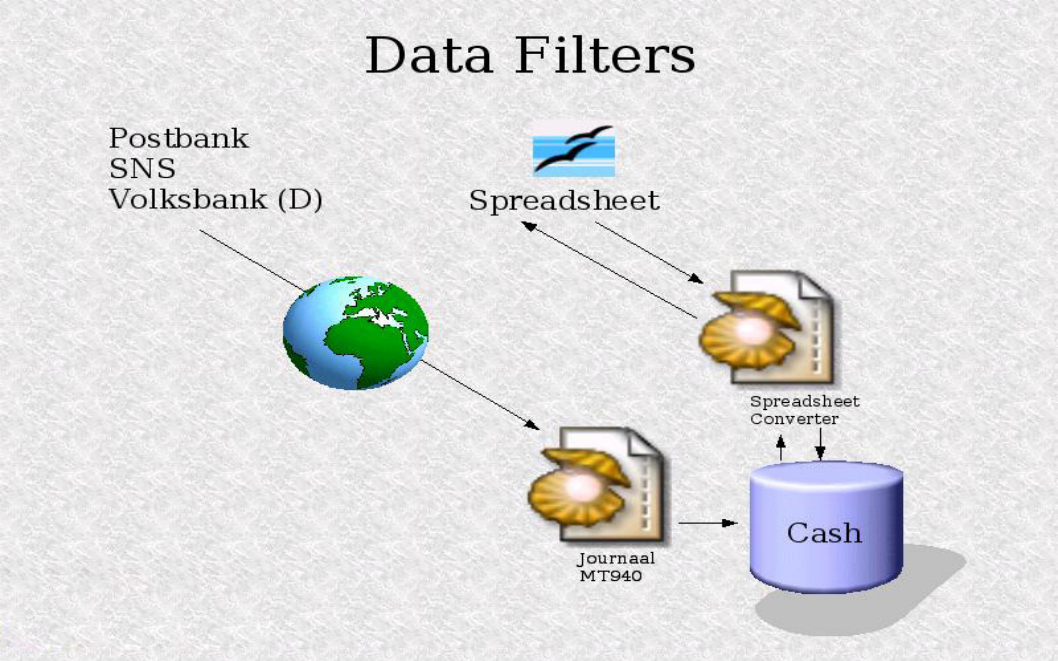
Zoals reeds vermeld is Cash bedoeld voor interactief gebruik. Bij Euroherbs stromen echter dagelijks grote hoeveelheden data de administratie binnen. Profile, de vorige applicatie voor financiële administratie, bevatte dus veel gegevens die overgezet moesten worden naar Cash. Men kon dit handmatig gedaan hebben, maar omdat dit een zeer tijdsintensieve bezigheid is (die een heleboel extra personeelskosten zou veroorzaken) besloot de directie om op zoek te gaan naar een methode van automatisering. Cash Software was bereid om deze taak op zich te nemen, maar na de goede ervaringen met OSS, werd besloten om ook bij de conversie weer beroep te gaan doen op haar flexibiliteit. Dit bleek een goede keuze te zijn, want er werd een tijdsefficiënt systeem voor conversie opgezet dat volledig op OSS steunde. De verschillende stappen van de conversie waren:

1. exporteren van de administratieve gegevens uit Profile
2. conversie naar het CSV formaat (comma-separated values) door middel van Perl, om de gegevens herkenbaar te maken voor spreadsheet-applicaties
3. importeren van de gegevens in OpenOffice Calc en verwerken van de spreadsheet
4. conversie van de gegevens naar het Cash-tekstformaat door middel van Perl
5. importeren van de gegevens in Cash

Dankzij deze tijdsefficiënte conversie werden een heleboel kosten vermeden. Daarenboven bleek deze methode (meerbepaald stappen 3 tot 5) ook toepasbaar te zijn voor andere wijzigingen van grote hoeveelheden data.

Ook bij het integreren van de bank-applicaties (van SNS, Postbank en de Duitse Volksbank) werd dankzij OSS heel wat tijd uitgespaard. Deze applicaties maken het mogelijk om via het Internet uittreksels en andere afschriften te downloaden. Deze afschriften kunnen daarna door een boekhoudpakket verwerkt worden. Het formaat van deze afschriften is afgestemd op de standaarden die gebruikt worden binnen Windows-omgevingen. Cash ondersteunt dit formaat echter niet. Het enige formaat dat in Cash geïmporteerd kan worden is ‘MT940’. Daarenboven is Cash nogal kieskeuring op de invoer van rekeningnummers. Zo moeten alle nummers aan de ’11-proef’ voldoen. Hierdoor zouden bijvoorbeeld Duitse rekeningen niet kunnen worden geautomatiseerd. Daarom werden een aantal data filters geïntegreerd, die door de Intranet-pagina aangestuurd worden. Deze filters formatteren verschillende bestandsformaten zodanig dat zij in Cash geïmporteerd kunnen worden. De besproken conversies zijn geïllustreerd in Figuur 12.

Figuur 12: Data Filters

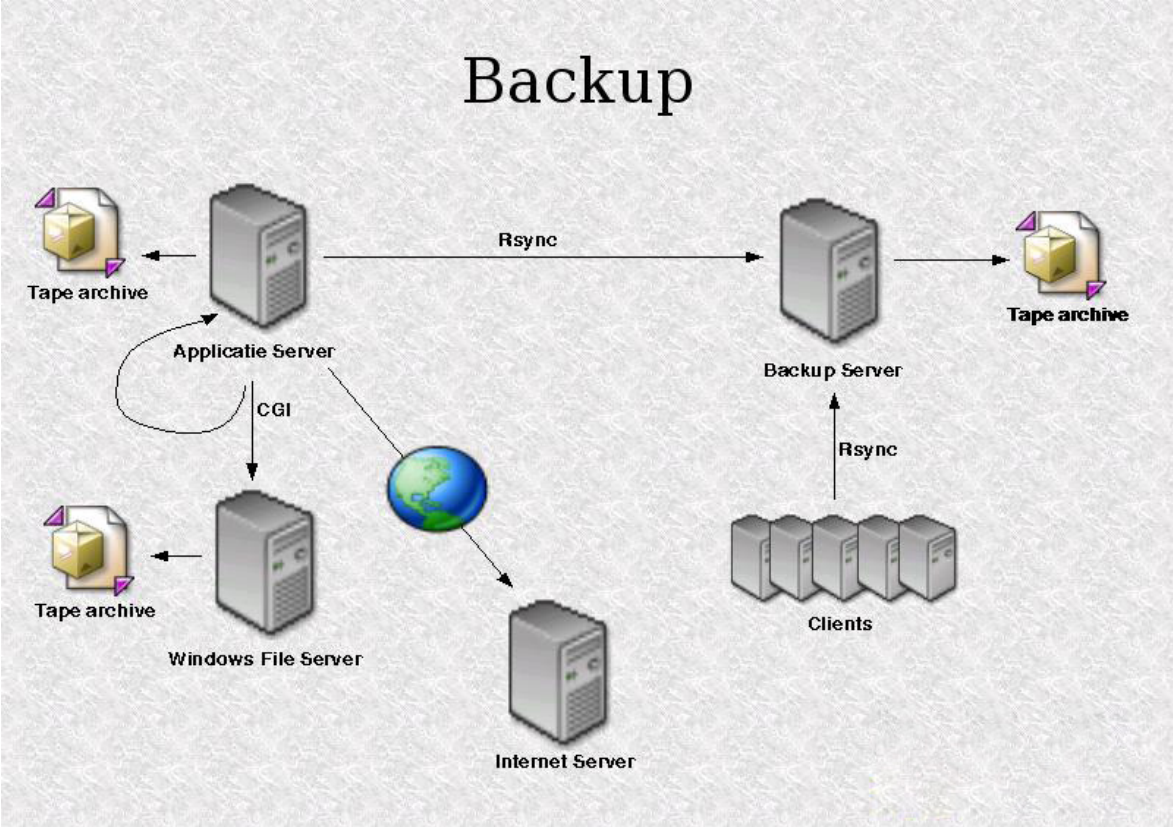


Bron: van der Kleij (2004)

4.5.4 Continuïteit & veiligheid

Om de continuïteit en de veiligheid van het informatiesysteem te garanderen, worden in de eerste plaats backups van data gemaakt. Omdat alle gebruikers inloggen op de applicatie-server waar Cash op draait en waarop de MySQL-database zich bevindt, is het zeer belangrijk dat er een backup gemaakt wordt van de informatie die via deze machine loopt. Er is hier immers sprake van wat men risico op ‘single point of failure’ noemt. Dit houdt in dat als deze applicatie-server zou crashen, het volledige informatiesysteem van Euroherbs in duigen ligt. Om dit te voorkomen, werden dus een aantal backups voorzien. Deze worden hieronder schematisch voorgesteld in Figuur 13.

Figuur 13: Backup



Bron: van der Kleij (2004)

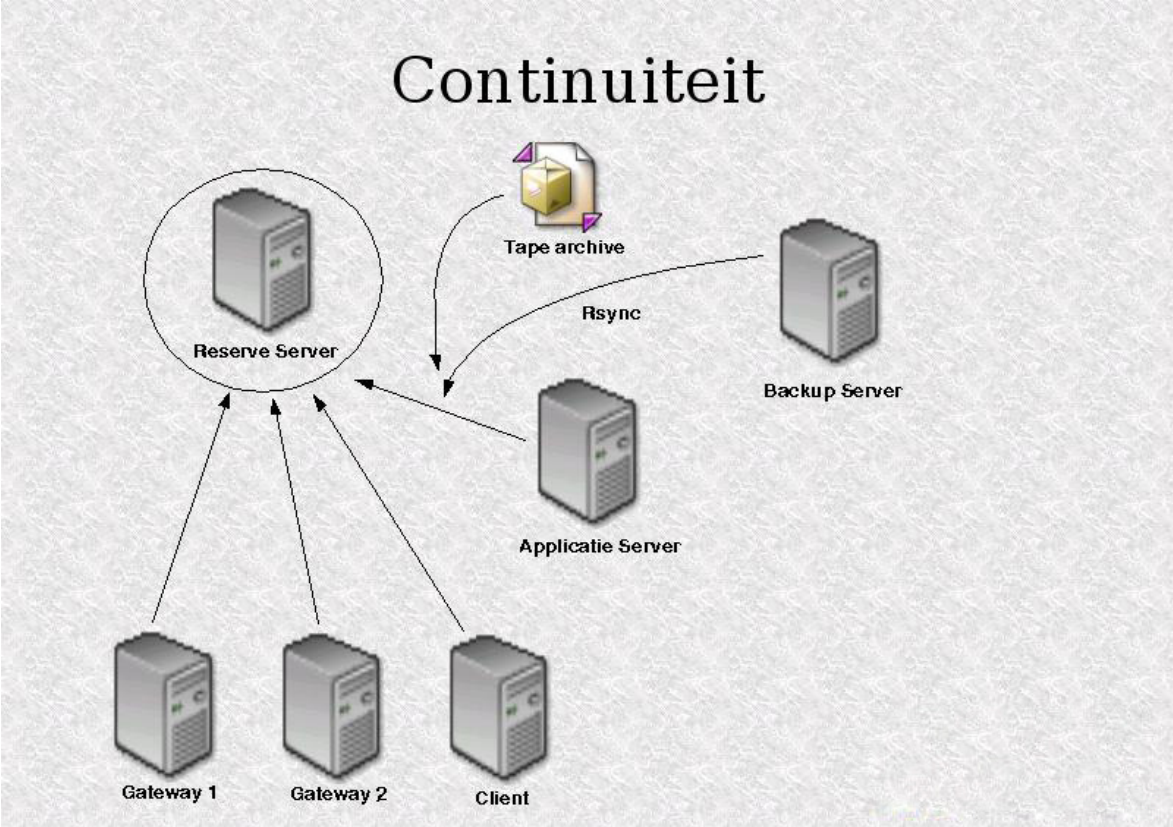
In de eerste plaats maakt de applicatie-server iedere nacht zelf een eigen backup door middel van een tape streamer. Daarnaast wordt een aparte backup-server gesynchroniseerd met de applicatie-server. Dit gebeurt door Rsync, een Open Source programma dat op de backup-server een exacte kopie creëert van de gegevens op de applicatie-server. Ook deze backup-server schrijft de data weg op een backup-tape. Op deze backup-server wordt daarenboven (ook weer via Rsync) elke dag automatisch een kopie gemaakt van de vitale mappen (mijn documenten, gedeelde mappen, enzovoort) op de Linux-clients. Voor de Windows-clients gebeurt dit ook, maar niet automatisch. De gebruikers moeten geregeld

het Open Source programma ‘Putty’ openen, om zo eenvoudig een backup te maken van de gegevens op hun client.

Daarnaast wordt ook geregeld een handmatige backup gemaakt van de gegevens op de applicatie-server, met name op een Windows file-server. De laatstgenoemde zet de gegevens op zijn beurt weer op tape. Deze server is echter al redelijk oud en voldoet niet meer aan de eisen die de onderneming stelt op gebied van functionaliteit. Om die reden werd deze handmatige backup reeds meer dan een half jaar niet uitgevoerd. Op Figuur 13 ziet U ook dat er een handmatige kopie gemaakt kan worden, die via het Internet op een Internet-server zou opgeslagen worden. Ook dit backup-systeem is momenteel niet meer in gebruik. Om deze laatste twee backup-servers te vervangen zal een Linux-werkstation geïnstalleerd worden dat twee keer per dag een synchronisatie zal uitvoeren op een (USB) harde schijf. Deze harde schijf zou dan dagelijks door een medewerker mee naar huis genomen worden.

Dankzij deze backups wordt de continuïteit van het informatiesysteem gegarandeerd. Voor een kleine onderneming, die dus betrekkelijk weinig middelen heeft, is elke verstoring van het bedrijfsproces een kost die de directie liever niet ziet opduiken in de boekhouding. Het voorkomen van dergelijke uitval is voor Euroherbs dus van groot belang. Wanneer uitval zich toch voordoet, is het voor de onderneming levensnoodzakelijk om deze verstoring zo kort mogelijk te laten duren. Op Figuur 14 ziet U hoe het informatiesysteem van Euroherbs hierop voorzien is.

Figuur 14: Continuïteit



Bron: van der Kleij (2004)

Bij Euroherbs wordt gebruik gemaakt van twee reserve-servers. Voor alle duidelijkheid is op de afbeelding slechts één reserve-server opgenomen. Deze reserve-servers staan opgesteld bij een medewerker thuis, op een redelijke afstand van de onderneming. Het zijn multi-boot machines die de identiteit kunnen aannemen van de twee firewalls (één hiervan is niet meer dan de ADSL-router) en een willekeurige client (om in te kunnen loggen op Cash). Deze servers worden geregeld handmatig gesynchroniseerd met zowel de tape-streamers als de backup- en applicatie-server. Wanneer een bepaalde server uitvalt, kan je in het multi-boot-menu op de reserve-server kiezen welke server hij zal moeten vervangen. De reserve-server wordt dan in het netwerk geplaatst en haalt zo recent mogelijke data uit

de tapes en andere servers. Deze reserve-server neemt dan de taak over van de server die hij vervangt. Op deze manier kan de verstoring van het bedrijfsproces door uitval van een server beperkt worden tot 10 à 30 minuten. De functionaliteit van het backup-systeem wordt twee keer per jaar uitvoerig getest en eventueel bijgewerkt.

4.6 Kostenbesparingen

Uiteraard heeft dit alles ook zijn weerslag op de kostenstructuur van de onderneming. Euroherbs heeft dankzij de introductie van OSS op een aantal vlakken kunnen besparen. Het resultaat is in Figuur 14 en 15 weergegeven.

Figuur 15: Kostenbesparingen op gebied van licenties

Besparing op	**Aantal**	**Kost/eenheid**	**Totaal**
Licenties OS	10	134	1340
Licenties firewall	2	66	132
Licenties Office	6	190	1140
Licenties anti-virus software	10	84	840
Licenties terminal-emulatie	10	69	690

Bron: van der Kleij (2004)

In de eerste tabel op Figuur 15 zijn de besparingen opgenomen die Euroherbs kon realiseren door geen licenties meer te moeten betalen. Deze tabel is gebaseerd op de toenmalige prijzen van de software die gebruikt werd voor de migratie. Deze besparingen worden bereikt over een periode van slechts één jaar. De totale besparingen die

gerealiseerd worden door de afwezigheid van licenties gedurende één jaar bedragen in totaal € 4142.

Figuur 16: Overige kostenbesparingen

Besparing op	Uren/eenheden	Waarde per eenheid	Totaal
Data-conversie	120	26	3120
Fout-correcties	24	50	1200
Bank-applicaties	24	70	1680
Onderhoud software	1	500	500
Vervanging hardware	10	400	4000
Ontwikkeling software	160	150	24000

Bron: van der Kleij (2004)

In de tweede tabel op Figuur 16 vinden we de besparingen in werkuren, evenals de besparing op gebied van vervanging van de hardware. Data-conversie kostte voordien meer dan 120 werkuren per jaar. Sinds de migratie gebeurt dit bijna volledig automatisch. Het gedeelte dat niet automatisch gebeurt is te verwaarlozen. Daarom werd in de tabel opgenomen dat hier 120 werkuren uitgespaard werden. Geautomatiseerde fout-correcties en bank-applicaties besparen de onderneming elk 24 werkuren per jaar, terwijl ook de jaarlijkse onderhoudsbeurt van het vroegere commerciële ERP-pakket nu wegvalt. Daarenboven heeft OSS ervoor gezorgd dat de onderneming geen nieuwe hardware heeft moeten aanschaffen, wat volgens van der Kleij wel nodig geweest zou zijn indien de migratie niet had plaatsgevonden. Ook de enorme kost die de ontwikkeling van

commerciële software met zich meebracht, is nu voorgoed verleden tijd. Deze laatste twee zijn éénmalige uitgaven die hier in het eerste jaar aangerekend worden.

We rekenen hier niet volgens het boekhoudkundige principe van kosten, maar met uitgaven, zoals dat volgens Mercken (2004) hoort bij een investeringsbeslissing. Het totaal aan besparingen uit deze tabel komt zo op € 34500. Opgeteld met de besparingen op gebied van licenties vormt dit dus een totale besparing van € 38642 tijdens het eerste jaar na de migratie. Voor de hierop volgende jaren zullen de besparingen bestaan uit de uitgaven die niet éénmalig zijn. Dat zijn alle genoemde uitgaven, behalve die voor vervanging van hardware en ontwikkeling van software. Uitgaande van een afschrijvingsduur van 5 jaar, zal er dus steeds een even lange cyclus ontstaan. In deze cyclus zal tijdens het eerste jaar telkens €38642 uitgespaard worden, terwijl in de 4 daaropvolgende jaren telkens een kostenreductie van € 10642 zal gerealiseerd worden. Abstractie makend van de indexatie van deze bedragen, kunnen we dus besluiten dat de migratie over elke periode van 5 jaar een totale kostenreductie van € 81210, of een gemiddelde jaarlijkste kostenreductie van € 16242 in de hand werkt.

4.7 Bevindingen van de gebruikers en de consultant

Doorheen de afgelopen zes jaar heeft Euroherbs een heleboel waardevolle ervaring opgedaan, zowel inzake OSS als inzake commerciële software. Na de volledige doorlichting van het informatiesysteem van Euroherbs, heb ik het samen met Michel van der Kleij gehad over de conclusies die hij kon trekken in verband met het nieuwe systeem

en OSS in het algemeen. Hierbij heeft hij ook rekening gehouden met de bevindingen van de directie en het personeel. Ik heb deze conclusies even op een rij gezet.

- De meest flexibele applicatie is er één met open standaarden, die bovendien generiek en eenvoudig in opzet en gebruik is.
- Hoe meer een commercieel pakket gebruik maakt van OSS-faciliteiten, hoe flexibeler het pakket is bij het inzetten voor verschillende taken.
- Belangrijke eigenschappen van software zijn mogelijkheid tot import, export, scripting en toegang tot de database.
- OSS laat toe om de tekortkomingen van commerciële producten op te vangen.
- Dankzij de mix van OSS en commerciële software wordt het voor de onderneming mogelijk om goedkopere software aan te schaffen en via OSS aan te passen aan de eigen noden en wensen.
- OSS is zeer stabiel en biedt dezelfde mogelijkheid inzake veiligheid en continuïteit als commerciële software.
- Overheden en banken werken, dankzij hun applicaties met gesloten bestandsformaten, nog steeds het gebruik van Windows in de hand.
- Dankzij OSS kan relatief goedkope software gekocht worden, die toch voldoende prestaties zal leveren.
- Er zijn voldoende leveranciers van OSS, waardoor ondernemingen niet gebonden zijn zodra ze bepaalde software gekocht hebben. Op die manier kan de continuïteit van de ondersteuning gegarandeerd worden.

- Als afsluiter merkte van der Kleij nog op dat OSS, dankzij het bovenstaande, nog een extra troef biedt voor ondernemingen: vrijheid van keuze! Hiermee bedoelt hij dat de ondernemingen niet alleen vrij zijn in de keuze van hardware, maar ook van andere software, leveranciers, ondersteuning, enzovoort.

Het is dus duidelijk dat het informatiesysteem van Euroherbs er, sinds de gedeeltelijke migratie naar OSS, op vooruit is gegaan. Een volledige overstap op OSS bleek ook hier niet verantwoord te zijn, maar uit de praktijkstudie blijkt dat het gedeeltelijk vervangen van commerciële software door OSS tot aanzienlijke kostenbesparingen kan leiden. Daarenboven heeft OSS ook de kwaliteit van het informatiesysteem op een hoger niveau gebracht. Bij Euroherbs is men het alleszins eens over één ding: “OSS is here to stay!”

H5 Algemene conclusie en aanbevelingen

In het eerste gedeelte van deze thesis werd onderzocht of OSS een goed alternatief vormt voor de CSS die momenteel in de meeste ondernemingen gedraaid wordt. Qua beschikbaarheid, gebruiksvriendelijkheid en (professionele, gerichte) ondersteuning blijkt OSS minder te presteren dan CSS. Er dient echter opgemerkt te worden dat er op deze vlakken sterke vooruitgang geboekt wordt. Ondanks het feit dat deze criteria (vooral ondersteuning) niet te verwaarlozen zijn, lijken deze criteria mij in het bedrijfsleven niet op te wegen tegen de voordelen die OSS op andere vlakken kan bieden. De hogere stabiliteit en flexibiliteit die OSS biedt, gekoppeld aan meer dynamische veiligheid, kunnen in ondernemingen een toegevoegde waarde voor het informatiesysteem vormen. Dankzij de leveranciersonafhankelijkheid, die onstaat door het gebruik van OSS, zullen ondernemingen die overstappen niet meer gebonden zijn en meer vrijheid kennen bij het maken van keuzes. Daarenboven is de levensduur van OSS in principe oneindig.

Wat kosten betreft is het moeilijk om een éénduidige conclusie te trekken. In principe is het steeds mogelijk om door middel van een gedeeltelijke migratie OSS kostenbesparingen te realiseren. Denk maar aan de eenvoudige vervanging van Microsoft Office door OpenOffice.org. Volledige migraties zullen daarentegen zelden interessant zijn, omwille van het beschikbaarheidsprobleem dat opduikt wanneer men op zoek gaat naar OSS. Ondernemingen kunnen natuurlijk de ontbrekende OSS laten ontwikkelen, maar dit brengt hoge kosten met zich mee. In dergelijk geval kan er een initiële kostenstijging ontstaan, maar volgens het IDABC zal de migratie op langere termijn altijd voordeliger uitkomen.

Zij merken ook op dat hoe meer gebruikers en computers er in een onderneming aanwezig zijn, hoe sneller de investering in een migratie naar OSS rendabel wordt. Vele ondernemingen willen echter dat de baten van een migratie meteen zichtbaar zijn. Indien het beschikbaarheidsprobleem opgelost wordt, zou dit ook bij volledige migraties de regel kunnen worden. Uiteraard staat de OSS-markt nog in haar kinderschoenen (Kivit, 2004). Naarmate de populariteit toeneemt, zullen er steeds meer aanbieders deze markt betreden en zal het beschikbaarheidsprobleem stilaan verdwijnen. In tussentijd kunnen ondernemingen gebruik maken van hybride informatiesystemen, niet alleen met het oog op kostenreductie, maar ook op het verhogen van de kwaliteit van het informatiesysteem op zich. OSS kan dus wel degelijk een economisch verantwoord alternatief vormen voor commerciële software, mits de onderneming overdacht te werk gaat bij de migratie.

Tot slot wil ik elke onderneming die een migratie naar OSS overweegt aanzetten om zoveel mogelijk informatie op te doen, alvorens over te gaan tot de implementatie. Een volledige overstap, als die al mogelijk is, zal niet altijd meteen voordeliger zijn. Het is daarom van het grootste belang dat beslissingsnemers in ondernemingen overdacht te werk gaan bij een migratie. Elke verandering brengt immers onverwachte gebeurtenissen mee. Ook de houding van de mensen binnen een onderneming is van belang. Inertie treedt vaker op dan we graag willen toegeven. Daarom wil ik alvast twee instrumenten meegeven die interessant kunnen zijn om een goede aanpak aan de dag te leggen.

In de eerste plaats raad ik de spreadsheet van IDABC aan om een beter inzicht te vekrijgen in de kostenstructuur van informatiesystemen, zowel voor OSS- als CSS-omgevingen. De

spreadsheet geeft ook een schatting van de kosten van een eventuele migratie en de payback-periode weer. Deze spreadsheet is, samen met een duidelijke (zij het Engelstalige) handleiding, beschikbaar op de website van IDABC (IDABC, 2005). Daarnaast is er nog het migratieplan van Socius, dat U terugvindt in bijlage 6. Dit migratieplan beschrijft 11 stappen die best overlopen worden bij een migratie en kan dus zeer handig zijn bij een overstap naar OSS.

Lijst van geraadpleegde werken

Baarsma, B. (2004) Kosten en baten van open standaarden en Open Source software in de Nederlandse publieke sector, Amsterdam, SEO

Beale, J., Meer, H., van der Walt, C., Deraison, R. (2004) Nessus Network Auditing: Jay Beale Open Source Security Series, Rockland, Syngress Publishing

Beekman, G. (2003), Computer Confluence, New Jersey, Prentice Hall

Blankensteijn, H. (2002) 'Groenlinks waarschuwt Microsoft', Overheid innovatief, N° 6, p.33

Breedveld, P., Koldewe, A., Scharis, R., Westerhof, R. (1999) De economische betekenis van Open Source Software in Nederland, International Data Corporation

Cowan, C. (2003) 'Software Security for Open-Source systems', Security & Privacy Magazine, Vol. 1, N°1

de Vreede, T. (2002) 'ICT-kosten groeien onstuitbaar', Automatisering Gids, week 48

Duijnhouwer, F. en Widdows, C. (2003) Capgemini Open Source Maturity Model, Brussel, Capgemini

Evers, S. (2000), An introduction to Open Source Software Development, Berlin, Technische Universität Berlin.

Feller, J. en Fitzgerald, B. (2002) Understanding Open Source Software Development, New York, Addison-Wesley

Gagné, M. (2004) 'Choosing a Linux Distribution', Tux Magazine (online) (geraadpleegd op 9 februari 2006). Beschikbaar op <URL: http://www.tuxmagazine.com/node/1000030>

Glott, R., Ghosh, R. (2003) Open standaarden en open source software in Nederland, Maastricht, Merit

Godden, F. (2000) 'How do Linux and Windows NT measure up in real life?', Gnet, januari

Golden, B. (2005) Succeeding with Open Source, New York, Addison-Wesley

Hahn, R.W. (2003) Government policy toward Open Source Software, Washington, Brookings Institution

Horngren, C.T., Datar, S.M. en Foster, G. (2003) Cost Accounting: A managerial emphasis, 11th edition, New Jersey, Pearson Education Inc.

IDABC (2006) 'IDABC – Documentation on Open Source Software' (online) (geraadpleegd op 3 juni 2006). Beschikbaar op <URL: http://europa.eu.int/idabc/en/document/2623>

Kabir, M.J. (1998) Apache Server Bible, Foster City, IDG Books Worldwide

Katz, M.L., Shapiro, C. (1998) Antitrust in software markets, Berkeley, University of California

Kenwood, C.A. (2001) A business case of Open Source Software, Massachusetts, Mitre

Kivit, J. (2004) Open Source in non-profit, 's-Hertogenbosch, Ematic

Knubben, B.S.J. (2001) Open Source Software, de bron geopend, Amsterdam, Universiteit van Amsterdam

Knubben, B.S.J (2004) Investeren in openheid, Den Haag, Stichting ICTU, Programma OSOSS

Laan, M. (2003) 'Amsterdam is duur Microsoft zat', Het Parool, 22 oktober

Laurent, A.M.S. (2004) Understanding Open Source Software and Free Software licensing, Cambridge, O'reilly

Lerner, J., Tirole, J. (2000), The Simple Economics of Open Source (Working Paper 7600), Cambridge, National Bureau of Economic Research

Levy, S. (1984), Hackers: Heroes of the computer revolution, New York, Doubleday

Linux Forums (2006) 'The ten major Linux distributions' (online) (geraadpleegd op 16 februari 2006). Beschikbaar op <URL: http://www.linuxforums.org/reviews/overview_of_the_ten_major_linux_ distributions.html>

McKusick, M. K., Raymond, E.S., Stallman, R. e.a. (1999), Open Sources: Voices from the Open Source Revolution, Cambridge, O'Reilly

Mercken, R. (2004) De investeringsbeslissing, Antwerpen, Apeldoorn

Mendys-Kamphorst, E. (2002) Open vs. closed: some consequences of the open source movement for software markets, Den Haag, Centraal Planbureau

Microsoft (2006) 'Microsoft Corporation' (online) (geraadpleegd op 4 mei 2006). Beschikbaar op <URL: http://www.microsoft.com>

Moglen, E. (1999), Anarchism Triumphant: Free Software and the death of copyright, Portland, University of Maine

Monard, E. (2004) Advies 86: Open Source Software, Brussel, Vlaamse Raad voor Wetenschapsbeleid

MySQL (2005) 'MySQL: The world's most popular open source database' (online) (geraadpleegd op 25 maart 2006). Beschikbaar op <URL: http://www.mysql.com/why-mysql>

Novell (2005) 'Novell: Worldwide' (online) (geraadpleegd op 4 mei 2006). Beschikbaar op <URL: http://www.novell.com>

O'Brien, J.A. (2003) Leerboek ICT-toepassingen, 11e druk, Schoonhoven, Academic Service

Payne, C. (2002) 'On the security of Open Source Software', Information Systems Journal, Vol. 12, N°1

Raymond, E.S. (1996) The Hacker's Dictionary, 3rd edition, Cambridge, MIT Press

Raymond, E.S. (1999), The Cathedral and the Bazaar, Cambridge, O'Reilly

Schiff, A. (2002) The Economics of Open Source Software, A Survey on the Early Literature, Auckland, University of Auckland

Smith, R.L. en Smith, J.K., Entrepreneurial Finance, 2nd edition, Indianapolis, Wiley Publishing

Theunissen, M. (2004) Factoren die de keuze voor Open Source Software bepalen, Amsterdam, Vrije Universiteit Amsterdam

Torvalds, L.B. (2002) Just for Fun: The Story of an Accidental Revolutionary, New York, Texere Publishing Ltd.

Uytterhoeven, G. (1999), 'Open Source Software (OSS) Ontwikkeling' (online) (geraadpleegd op 7 juni 2005). Beschikbaar op <URL: http://home.tvd.be/cr26864>

van den Berg, K. (2005) Finding Open Options, Tilburg, Universiteit van Tilburg

van der Kleij, M. (2004) Presentatie over de succesvolle mix van OSS en Commerciëel Linux in de praktijk, Westervoort, Tukcedo

Verhoef, E. (2005) 'De IT-tsunami', Automatisering Gids, week 2

Visser, J. (2002) Het Open Source Fenomeen, Diemen, Open Solution Providers

Wayner, P. (2000) Free for All: How Linux and the Free Software Movement Undercut the High-Tech Titans, New York, HarperBusiness

Weber, S. (2000) The Political Economy of Open Source Software, BRIE Working Paper 140 of the E-conomy Project, Berkeley, University of California

Weber, S. (2004) The Success of Open Source, Harvard, Harvard University Press

Wheeler, D.A (2005) 'Why Open Source Software / Free Software' (online) (geraadpleegd op 28 september 2005). Beschikbaar op <URL: http://www.dwheeler.com/oss_fs_why.html>

Lijst der figuren

Bijlagen

Bijlage 1: The Open Source Definition

Bijlage 2: Bronnen van Open Source Software op het Internet

Bijlage 3: Bill Gates' open letter to hobbyists

Bijlage 4: Interview: Michel van der Kleij

Bijlage 5: Interview: Eric Smets

Bijlage 6: Migratieplan van Socius

Bijlage 1: The Open Source Definition

(bron: http://www.opensource.org/docs/definition.php)

Introduction

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

Rationale: *By constraining the license to require free redistribution, we eliminate the temptation to throw away many long-term gains in order to make a few short-term sales dollars. If we didn't do this, there would be lots of pressure for cooperators to defect.*

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost–preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program.

Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

Rationale: *We require access to un-obfuscated source code because you can't evolve programs without modifying them. Since our purpose is to make evolution easy, we require that modification be made easy.*

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

Rationale: *The mere ability to read source isn't enough to support independent peer review and rapid evolutionary selection. For rapid evolution to happen, people need to be able to experiment with and redistribute modifications.*

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

Rationale: *Encouraging lots of improvement is a good thing, but users have a right to know who is responsible for the software they are using. Authors and maintainers have reciprocal right to know what they're being asked to support and protect their reputations.*

*Accordingly, an open-source license **must** guarantee that source be readily available, but **may** require that it be distributed as pristine base sources plus patches. In this way, "unofficial" changes can be made available but readily distinguished from the base source.*

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

***Rationale:** In order to get the maximum benefit from the process, the maximum diversity of persons and groups should be equally eligible to contribute to open sources. Therefore we forbid any open-source license from locking anybody out of the process.*

Some countries, including the United States, have export restrictions for certain types of software. An OSD-conformant license may warn licensees of applicable restrictions and remind them that they are obliged to obey the law; however, it may not incorporate such restrictions itself.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

***Rationale:** The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.*

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

Rationale: *This clause is intended to forbid closing up software by indirect means such as requiring a non-disclosure agreement.*

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

Rationale: *This clause forecloses yet another class of license traps.*

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

Rationale: *Distributors of open-source software have the right to make their own choices about their own software.*

Yes, the GPL is conformant with this requirement. Software linked with GPLed libraries only inherits the GPL if it forms a single work, not any software with which they are merely distributed.

10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

Rationale: *This provision is aimed specifically at licenses which require an explicit gesture of assent in order to establish a contract between licensor and licensee. Provisions mandating so-called "click-wrap" may conflict with important methods of software distribution such as FTP download, CD-ROM anthologies, and web mirroring; such provisions may also hinder code re-use. Conformant licenses must allow for the possibility that (a) redistribution of the software will take place over non-Web channels that do not support click-wrapping of the download, and that (b) the covered code (or re-used portions of covered code) may run in a non-GUI environment that cannot support popup dialogues.*

Bijlage 2: Bronnen van Open Source Software op het Internet

http://sourceforge.net/

http://www.ossl.nl/

http://freshmeat.net/

http://www.fsf.org/

http://www.opensource.org/

http://java-source.net/

http://www.tigris.org/

http://www.softpanorama.org/

http://osdir.com/

http://www.free-soft.org

http://www.w3.org/

http://csharp-source.net/

http://www.oss-watch.ac.uk/

http://www.opensourcemac.org/

http://www.theopencd.org/

http://www.openoffice.org/

http://www.jairlie.com/

http://www.opensourcecms.com/

http://www.blender.org/

http://www.dotproject.net/

http://www.mozilla.org/

Bijlage 3: Bill Gates' open letter to hobbyists

(bron: http://www.blinkenlights.com/classiccmp/gateswhine.html)

By William Henry Gates III

February 3, 1976

To me, the most critical thing in the hobby market right now is the lack of good software courses, books and software itself. Without good software and an owner who understands programming, a hobby computer is wasted. Will quality software be written for the hobby market?

Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. Though the initial work took only two months, the three of us have spent most of the last year documenting, improving and adding features to BASIC. Now we have 4K, 8K, EXTENDED, ROM and DISK BASIC. The value of the computer time we have used exceeds $40,000. The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however, 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent on Altair BASIC worth less than $2 an hour.

Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. We have written 6800 BASIC, and are writing 8080 APL and 6800 APL, but there is very little incentive to make this software available to hobbyists. Most directly, the thing you do is theft.

What about the guys who re-sell Altair BASIC, aren't they making money on hobby software? Yes, but those who have been reported to us may lose in the end. They are the ones who give hobbyists a bad name, and should be kicked out of any club meeting they show up at.

I would appreciate letters from any one who wants to pay up, or has a suggestion or comment. Just write to me at 1180 Alvarado SE, #114, Albuquerque, New Mexico, 87108. Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software.

Bill Gates

General Partner, Micro-Soft

Bijlage 4: Interview: Michel van der Kleij

Wat kan er over Open Source Software gezegd worden in verband met volgende criteria? Kunt U eventueel vergelijken met commerciële software?

Kwaliteit (stabiliteit, efficiëntie, effectiviteit)

In het algemeen dient gesteld te worden dat, zoals met zo vele onderwerpen, generalisatie erg gevaarlijk is. Ook binnen OSS doen zich kwaliteitsverschillen voor, maar algemeen gesteld bespreek ik hier de producten die als 'populairst' kunnen worden aangemerkt en welke vanwege deze populariteit een sterke community hebben die zich achter het betreffende product schaart. De ondersteuning van de community zal dan ook van doorslaggevend belang zijn.

De kwaliteit van OSS producten is vrijwel inherent goed vanwege de 'peer-to-peer' ontwikkelmethode waarbij verschillende ontwikkelaars snel feedback krijgen van vele testers/gebruikers en omdat de mogelijkheid tot het uitbrengen van patches en fixes zo snel tot stand komt. Door deze korte ontwikkelingscyclus komt men snel tot een goed werkend product. Het nadeel is dat veel producten weinig structuur qua ontwerp hebben, ik bedoel hiermee dat er vaak geen 'roadmap' is van te verwachten features. Deze verschijnen, dan wel

verdwijnen naar behoefte. Een nadeel van OSS is dat men voor het goed functioneren van met name hardware (denk aan drivers) vaak afhankelijk is van de medewerking van de leverancier, want anders is de ontwikkelaar veroordeeld tot reverse-engineering, hetgeen

vrijwel nooit eenzelfde oplossing oplevert als waneer met toegang heeft tot alle specificaties. Gelukkig wordt deze medewerking steeds meer toegezegd naarmate OSS een bekend fenomeen is geworden.

Veiligheid

Ik denk dat we hier een onderscheid moeten maken tussen beveiliging-georienteerde software zoals SSH, beveiligings-behoeftige software, zoals Postfix en Apache en overige software. De eerste twee categorieen zijn behoorlijk goed voor elkaar. Ook hier zijn de snelle
ontwikkelingscycli en het peer-to-peer element van de community verantwoordelijk voor. De enige categorie waar ik me zorgen over maak is de overige, want hier wordt vaak meer gedacht aan gebruikersvriendelijkheid die, zoals bekend, haaks staat op het beveiligingsniveau.

Wel is het zo dat ook de commerciele "overige software" categorieen dit probleem kennen. In het algemeen denk ik dat de (op beveiliging toegespitste) OSS wat beter is dan de commerciele vanwege de grote aantallen testers en de korte ontwikkelcyclus.

Flexibiliteit

Je zult merken dat veel van de onderwerpen die je hier noemt met elkaar te maken hebben. OSS' inherente flexibiliteit is een sleutelfactor in de strijd om de consument. Als bedrijf ben je met OSS flexibel vanwege de leveranciersonafhankelijkheid en de inzet van software die je nodig hebt en niet die waar je contractueel toe gebonden bent. Flexibiliteit

is ook de mogelijkheid om software uit te testen zonder tot aanschaf te hoeven overgaan. “Simply test until you find the best!”

Ondersteuning

Reeds genoemd is het belang van de ondersteunende community voor een bepaald product. Zelf ben je natuurlijk bekend met Ubuntu en als voorbeeld zie je daar een website, een forum en een lokale organisatie, toegevoegd aan de ondersteuning van een commercieel bedrijf. Over deze laatste, hoe "groter" de naam van zo'n bedrijf, bijv. geaccepteerde namen als Sun, IBM, HP, des te meer aantrekkingskracht OSS kan uitoefenen op gerenommeerde non-tech bedrijven als ING, Unilever, Shell etc.

Er zijn weinig bedrijven die de actieve ondersteuning van de community van een bekend product kunnen evenaren, zeker die van software-reuzen als Microsoft of Adobe niet. Voor een niche-product is deze ondersteuning echter veel minder aanwezig vanwege de benodigde, gespecialiseerde kennis die maar weinigen bezitten. Je ziet dan ook dat in deze hoek bedrijven juist achter de community zitten om deze op te bouwen waar mogelijk, mits ze natuurlijk interesse hebben in OSS.

Beschikbaarheid

Er zijn volgens mij twee soorten beschikbaarheid: vooreerst de beschikbaarheid van de software zelf, in termen van 99,998% uptime en daarnaast beschikbaarheid in de zin van hoe eenvoudig het is om software te vinden (de toegankelijkheid).

In het eerste geval zijn de gespecialiseerde producten zoals het Linux OS hier erg goed in. In mijn ervaring is het zo dat wanneer er een probleem optreedt met een systeem waar Linux op draait, dit vrijwel zeker een hardware oorzaak heeft. Windows daarentegen geeft soms zelf om onverklaarbare redenen de geest. Linux is in dit verband derhalve zonder meer te vergelijken met commerciele Unix-varianten als Solaris of AIX. De laatsten onderscheiden zich alleen nog door dure en op de eigen markt gerichte toepassingen om juist de beschikbaarheid van de combinatie hardware/OS verder omhoog te krijgen.

De toegankelijkheid van OSS daarentegen, is beduidend minder dan die van commerciele software. Commerciele software is weliswaar relatief duur, maar het ligt wel op elke plank van de willekeurige supermarkt en is door relatieve beginners al met (enig) succes te installeren. Het zoeken naar en installeren van OSS is, behalve voor een beperkt aantal populaire toepassingen zoals bijvoorbeeld OpenOffice.org, slechts weggelegd voor geinteresseerde en enigzins geoefende partijen. Ik doel hier met name de "configure/make/make install" software. Deze kan uitstekend werken maar is uiteraard minder toegankelijk voor het algemene publiek. Daarnaast is ook onderhoud ervan veel lastiger en wederom voor de expert/liefhebber.

Gebruiksvriendelijkheid

Traditioneel is dit een probleem voor desktop-gebruikers en ik zou willen voorstellen om dit onderwerp in twee delen te bespreken: voor desktop-gebruikers en op gebied van de server. Om met de server te beginnen denk ik dat de software hiervoor voldoende gebruiksvriendelijk is. Let wel dat ik het van enorm belang vind dat de beheerder weet wat

hij doet. OSS server- software als Apache, Postfix, maar ook Linux bieden de gebruiker op basis van hun Unix-genen voldoende mogelijkheid om de software te configureren en te gebruiken. In tegenstelling tot Windows administrators, welke simpelweg leren in de juiste volgorde een aantal velden in te vullen en knopjes in te drukken. Hierdoor gaat waardevolle kennis uiteindelijk verloren welke juist bij problemen essentieel blijkt te zijn!

De desktop is een iets ander verhaal. Terwijl ik denk dat het voor de geoefende en geinteresseerde gebruiker voldoende gebruiksvriendelijk is, vrees ik dat het algemene publiek er nog steeds moeite mee heeft. Lees het Ubuntu forum (maar uiteraard alle andere) er maar eens op na. Je vindt hier vragen die je op Windows-fora nooit zou tegenkomen. Hier geldt wel dat dit vaak komt door gebrekkige ondersteuning van de OSS community en van de leveranciers van hardware! Toch is het wel degelijk mogelijk om voor niet-veeleisende gebruikers een eenvoudig en vriendelijk systeem op te zetten, maar momenteel is de hulp van een forum of een ‘handige bekende’ nog te vaak gewenst.

Leveranciersonafhankelijkheid

Lijkt me bij een fenomeen als OSS bijna gegarandeerd. Toch gaat het hier niet alleen om de software zelf, maar ook om de ondersteuning ervan. Als bedrijf moet je er denk ik voor zorgen dat je nooit volledig afhankelijk wordt van wat of wie dan ook. Met OSS software is de vrije beschikbaarheid van een voldoend populair product langere tijd gegarandeerd. Het is wel uitkijken geblazen als de populariteit blijkt af te nemen omdat de aandacht van de community zich dan wellicht in ongewenste richting gaat begeven, maar dit geldt ook

voor commerciele aanbieders. Ondersteuning is een iets andere kwestie. Zorg er als bedrijf voor dat de medewerkers altijd ‘portable’ oplossingen gebruiken.

Levensduur/gebruiksduur

Het fraaie van OSS is dat je zelf de keus hebt, zoals Richard Stallman van GNU altijd zegt: “Free, not as in free beer, but as in free speech!” Je kunt ervoor kiezen om OpenOffice.org pas te upgraden als het je uitkomt en niet omdat je leverancier besluit met de ondersteuning op te houden om je daarmee te dwingen tot een nieuwe aankoop. Hierdoor is de gebruiksduur van OSS bijna per definitie langer dan die van commerciële. Omdat je ook zelf kunt kiezen
welke patches je aanbrengt, kun je zo erg lang doen met bepaalde software en toch ‘bijblijven’ op gebieden die je nodig vindt.

Voor grote bedrijven minder interessant (om bedrijfseconomische redenen van schaal) is het feit dat OSS ontwikkeld wordt door liefhebbers op basis van een modulaire structuur (men bouwt voort op de bijdragen van anderen) en veel langer op hardware kan blijven draaien. Ik draai bij diverse klanten met veel succes firewalls die gebaseerd zijn op een recente Linux versie maar lopen op een Pentium I van 200 Mhz! Uiteraard is dit voor kleinere bedrijven en particulieren wel degelijk een interessant verschijnsel.

Kosten (aanschafkosten, hardwarekosten, zoekkosten, operationele kosten, opleidingskosten zowel van ICT- als ‘gewoon’ personeel, ondersteuningskosten, overstapkosten)

Deze kosten vormen voor veel bedrijven dé drempel om de overstap te wagen. Euroherbs bevond zich in de gelukkige positie dat er nog geen echte infrastructuur (denk ook aan contractstructuren) was en dat een overgang werd geforceerd door de Euro en het ‘jaar 2000 probleem’. Men heeft van de nood een deugd gemaakt door meteen te kiezen voor een ommezwaai. Weinig bedrijven zullen echter in deze voordelige positie terechtkomen.

Wanneer er echter geen politieke belangen spelen, zijn het met name de flexibiliteit, leveranciersonafhankelijkheid, aanschaf- (simple commodity hardware), operationele en ondersteuningskosten waar de winst mee gemaakt wordt. De opleidings- en vervangingskosten zijn in deze som uiteraard de negatieve getallen. Voor vervangingskosten geldt echter wel dat ze niet noodzakelijkerwijs in één keer komen: een geleidelijke overgang is ook vanuit andere perspectieven te prefereren.

Bijlage 5: Interview: Eric Smets

Wat kan er over Open Source Software gezegd worden in verband met volgende criteria? Kunt U eventueel vergelijken met commerciële software?

Het is heel moeilijk om via een algemene regel na te gaan welke de verschillen zijn tussen twee specifieke onderdelen. De terugkoppeling in de antwoorden is een megeling van algemene antwoorden (wat je als gemiddelde kan nagaan) en heel specifieke antwoorden (toegepast op bepaalde software).

Kwaliteit (stabiliteit, efficiëntie, effectiviteit)

De kwaliteit van de Open Source Software is heel divers, door de eigen kennis en het beoordelen van de software op diverse merites kan men echter in de regel wel zeggen dat het naar stabiliteit, efficiente en effectiviteit zeker de vergelijking met de meest gangbare software kan doorstaan. Concrete voorbeelden hiervan zijn: Operating Systems
Linux en Freebsd zijn zeker zo stabiel als Windows om niet te zeggen stabieler (minder onderhevig aan virussen, ...). Door de correcte parametrering kan men veel efficienter werken met minder resourcen, deze laatste parameter-instelling is echter enkel weggelegd voor specialisten. Een linux OS is bijvoorbeeld sneller dan een Windows fileserver voor een zelfde hardware. Wat bijvoorbeeld databases betreft geraakt men in de regel met postgressql zeker zover als met oracle; hierbij zijn er ook clustering mogelijkheden. Deze laatste vergen wel meer instelwerk.

Veiligheid

Door het principe van meerdere mensen die open aan de code werken wordt het systeem direct veiliger. Hierbij wil ik opmerken dat het principe van ‘security by obscurity’ niet werkt. Al deze codes zijn zonder al te veel moeite te kraken. Voorbeelden zijn de DVD, DRM of modchips die gewoon gekraakt worden. Hierbij dient wel opgemerkt te worden dat er wel een zekere kennis nodig is van de zaken waarmee men werkt. Indien dit niet het geval is, worden er evengoed onveilige systemen geconfigureerd.

Flexibiliteit

De flexibiliteit voor de ontwikkelaars is groter omdat er hier niet gewacht hoeft te worden op een derde. Men kan zelf de problemen oplossen. Voor een klant van FKS hebben wij software ontworpen die gelijktijdig een 30 tal USB verbindingen diende te maken met PDA's. Dit is een oplossing die niet direct mogelijk is binnen de windows wereld. FKS heeft dit opgelost door te starten van een eerste versie van code die deze verbinding opzetten, hierna heeft FKS de code gedebugged/aangepast en op een bepaalde plaats ook hulp ingeroepen van de kernel developer voor het USB-stuk. Deze laatste bug heeft geresulteerd in een nieuwe versie van de kerneldriver voor USB, binnen de 48 uur. Eerlijkheidshalve dient hierbij opgemerkt te worden dat FKS goede debugrapportage en een eerste aanzet tot foutoplossing eveneens meegedeeld heeft. Deze snelheid van reactie kan enkel bereikt worden door grotere bedrijven die genoeg druk kunnen leggen op de gevestigde bedrijven in de closed source wereld.

Ondersteuning

De ondersteuning voor de leken lijken niet zo sterk gezaaid, waar men voor de meer traditionele IT support bijna op elke hoek van de straat een specialist vindt. Is dit in de open-source wereld minder makkelijk? Er zijn volgens mij wel genoeg bedijven die hierin mediator kunnen spelen of kunnen uitleggen op welke manier hulp kan ingekocht worden. Dit laatste is echter enkel voor de mensen die de regels kennen.

Beschikbaarheid

Voor diverse onderdelen kunnen er SLA afgesloten worden zoals voor de meer traditionele paketten. Er is echter niet steeds een goede open source oplossing voorhanden. Zoals er ook niet steeds een goede SAP oplossing voor handen is voor een specifiek onderdeel.

Gebruiksvriendelijkheid

Wat is gebruiksvriendelijkheid ? Indien gebruiksvriendelijk betekent dat men klikt in een GUI zonder te weten wat er gedaan moet worden, dan is de gebruiksvriendelijkheid minder als in de Windows-omgeving. De laatste jaren worden voor de meer alledaagse dingen ook GUI's gemaakt, die wel naar dezelfde commando line interface teruggrijpen. Om hier een heel eenvoudig voorbeeld van te geven: in Windows kan ook niet alles vanaf de GUI gedaan worden. Zo moet er bij sommige paketten ook in de registry gedoken worden, wat dan weer via command line dient te gebeuren.

Leveranciersonafhankelijkheid

Groter, er kan met de source naar een concurrent gegaan worden. In de open source wereld wordt er gestreden op basis van de diensten.

Levensduur/gebruiksduur

De levensduur wordt door de gebruiker bepaald niet door de aanbieder van de software.

Kosten (aanschafkosten, hardwarekosten, zoekkosten, operationele kosten, opleidingskosten zowel van ICT- als 'gewoon' personeel, ondersteuningskosten, overstapkosten)

Wat betreft de kosten kan men het volgende zeggen. Hardwarekost zijn lager voor een zelfde kwaliteit of performantie.Zoekkost zijn hoger omdat er voor de meer traditionele software-ontwikkelaars een groter marketing budget beschikbaar is. Het verschil hierbij is dat men denkt dat een bepaald product de goede oplossing is, terwijl dit niet noodzakelijk zo is.

Het gebruik van OSS veroorzaakt ook duidelijk minder operationele kosten: geen licentiebeheer, geen gedwonen upgrades, automatische upgrades, procedures reeds sinds lange tijd aanwezig. Opleidingskosten zijn in de regel hoger omdat er meer mensen zijn die Windows reeds kennen. Wat betreft de migratie van een Unix-omgeving is deze dan weerom gemakkelijk. Ook Linux-mensen kunnen makkelijk een Unix-systeem onderhouden. Voor de andere producten zijn dit standaard opleidingskosten voor paketten. Bijvoorbeeld een opleiding, mocht deze aangeboden worden, in openoffice.org is niet noodzakelijk duurder dan een opleiding voor Microsoft Office.

IT kosten naar personeel zijn voor een zelfde kwaliteit van personeel hetzelfde. Wat echter wel het geval is, is dat er in de OSS-wereld moeilijker mensen te vinden zijn.

De overstapskosten zijn analoog met de opleidingskosten. Wat echter wel het geval is, is dat er nogal eens kosten bijkomen omdat werknemers/medewerkers niet willen overstappen.

Bijlage 6: Migratieplan van Socius

(bron: Het overstappen naar Open Source Software binnen een organisatie moet goed voorbereid worden. Hieronder beschrijven we een mogelijk stappenplan.

1. Creëer een team dat over de nodige competenties beschikt én over de steun van de leiding.

2. Migratie biedt de kans om de basisarchitectuur zowel als de toepassingssoftware in huis in kaart te brengen en zo nodig te herzien. Het kan bijvoorbeeld een aanleiding zijn om data die her en der verspreid zijn en eventueel in verschillende bestandsformaten zijn opgeslagen, te centraliseren in één database.

3. Zorg ervoor dat de eigenheid van Open Source software duidelijk is: bij commerciële software kan je enkel support krijgen bij bedrijven die toegang hebben tot de broncode. Als de leverancier uit de markt verdwijnt zonder de code vrij te geven is er een probleem. Bij Open Source software is een hele 'Community' betrokken, zodat in het ergste geval nog steeds hulp te vinden is via mailing lists en on line fora.

4. Maak een volledige inventaris van de aanwezige besturingssystemen en softwaretoepassingen

5. Maak een inventaris van alle data en ga na onder welk formaat ze voor welke gebruikers resp. toepassingen beschikbaar moeten zijn.

6. Ga na op welke manier het systeem beveiligd is, wat de structuur is van de gebruikte paswoorden, wat eventuele nieuwe behoeften aan beveiliging kunnen optreden.

7. Maak een gedetailleerde berekening voor de kosten verbonden met de migratie. Maak een vergelijking tussen het onderhouden van de aanwezige omgeving enerzijds en de kosten resp. besparingen door migratie naar OSS anderzijds, over een redelijk tijdspad van meerdere jaren.

8. Raadpleeg de gebruikers. Motiveer de overstap naar OSS en geef 'early adopters' de kans om alvast te starten. Voorzie in een interne helpdesk via e-mail of intranet, waartoe de gebruikers zelf kunnen bijdragen.

9. Start met kleinschalige pilootprojecten. Deze leveren informatie op over de kostenstructuur en de reacties van de gebruikers en kunnen leiden tot validering of aanpassing van de architectuur.

10. Neem een beslissing over de snelheid van het migratieproces. De meest risicivolle optie is de 'Big Bang': iedereen schakelt op hetzelfde moment volledig over. Gefaseerde overgang in groepen is een betere optie, bijvoorbeeld te combineren met de vervanging van desktop PC's, die dan ineens met de nieuwe software uitgerust worden. Tenslotte is, in kleine organisaties, ook een gebruiker-per-gebruiker-scenario mogelijk. Voorzie dat oude en nieuwe systemen kunnen blijven samenwerken, zodat de dagelijkse activiteit niet onderbroken wordt.

11. Voer de migratie door voor de hele organisatie. Voorzie in permanente training voor gebruikers en technici. Volg de feedback van gebruikers op en los problemen die opduiken op.

Auteursrechterlijke overeenkomst

Opdat de Universiteit Hasselt uw eindverhandeling wereldwijd kan reproduceren, vertalen en distribueren is uw akkoord voor deze overeenkomst noodzakelijk. Gelieve de tijd te nemen om deze overeenkomst door te nemen en uw akkoord te verlenen.

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Is open source software een economisch verantwoord alternatief?
Richting: **Licentiaat in de toegepaste economische wetenschappen** Jaar: **2006**
in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Deze toekenning van het auteursrecht aan de Universiteit Hasselt houdt in dat ik/wij als auteur de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij kan reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

U bevestigt dat de eindverhandeling uw origineel werk is, en dat u het recht heeft om de rechten te verlenen die in deze overeenkomst worden beschreven. U verklaart tevens dat de eindverhandeling, naar uw weten, het auteursrecht van anderen niet overtreedt.

U verklaart tevens dat u voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen hebt verkregen zodat u deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal u als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze licentie

Ik ga akkoord,

Raphaël LHEUREUX

Datum: