Ik wil weten hoe Linux snel en goedkoop kan!

Als Linux gekozen wordt om de kostprijs te verlagen dan mag die winst niet verloren gaan door onvoorziene, extra ontwikkelkosten. Die vallen de eerste keer vaak tegen, vooral omdat er geen referentie is. Daarom stellen we de vraag of die manier om Linux in te zetten, de juiste keuze was. Dit aan de hand van een generiek stappenplan om de kosten laag te houden.

Inhoud

- Goed en snel starten spaart kosten!
- Vijf thema's en 10 stappen
  1. Wat kost meer?
     - Kleiner is duurder
     - PC technologie is de norm
  2. Wat is gratis?
     - Het web en het internet
     - Multimedia en VoIP
  3. Wanneer gaan we over?
     - Poorten versus nieuwbouw
     - De oplage is bepalend
  4. Waar beginnen we mee?
     - De hardware is onbelangrijk
     - Beginnen is altijd moeilijk
  5. Hoe moeten we kiezen?
     - Kies voor standaard en kies de defaults
     - Snel kiezen schaadt niet.

Goed en snel starten spaart kosten!

In *"Hoe kan gratis Linux toch te duur zijn?"* kunnen we lezen dat het belangrijk is om op de kosten te letten. Zowel op licentiekosten als op ontwikkelkosten. De licentiekosten van Linux zijn nihil, maar in praktijk vallen de ontwikkelkosten vaak tegen. Vooral als er geen realistische uitgangspunten gebruikt zijn.
Dat laatste is vaak te wijten aan gebrek aan ervaring en kennis van Linux, wat natuurlijk logisch is bij de eerste stappen op gebied van embedded Linux.

Vijf thema's en 10 stappen

Toch is het mogelijk om snel een globaal beeld te krijgen van of Linux mogelijk is, waar de kosten zitten en waar de kans op tegenvallers het grootst is. In dit artikel gebruiken we een handvol eenvoudige vragen, waarbij de antwoorden een goede eerste indruk geven of Linux mogelijk is. Hierbij kijken we er niet naar of Linux technisch mogelijk is, maar naar de economische haalbaarheid.

1. Wat kost meer?

Linux is gratis te downloaden en kan daarna aangepast worden naar behoefte. Dat is het concept van OpenSource. Maar elke aanpassing kost (ontwikkel)tijd en dus geld. Het is dus fijn als er weinig aanpassingen nodig zijn. Want behalve minder ontwikkeltijd, scheelt dat in het aantal fouten

en in de uitzoektijd. Bovendien kunnen we dan de belangrijkste stappen eerst zetten in plaats van te beginnen met de details. Daarover later meer.

Kleiner is duurder

Linux ondersteunt veel hardware en heeft heel veel features. Daardoor is Linux omvangrijk. De omvang van een Linux-PC wordt uitgedrukt in Gigabytes, niet in kilobytes! Veel hiervan is slechts nodig op sommige type hardware en is dus veelal overbodige ballast. Toch is het op 'normale computers' het goedkoopste om alles te installeren. Want optimaliseren is duur.
Traditionele embedded systemen worden typisch opgebouwd: je begint met niets en breidt dat stap voor stap uit totdat je de omvang hebt die gewenst is. Bij embedded Linux is dat anders. Een volledig Linux-systeem is gratis te downloaden. Daarna ga je dat downsizen: alles wat niet nodig is verwijderen. Je kunt het zo klein maken als je wilt, maar het wordt steeds lastiger, en dus duurder, om het kleiner te maken.

Natuurlijk zijn er veel zaken die snel en goedkoop te verwijderen zijn. Documentatie bijvoorbeeld. Maar bedenk dat er meerdere soorten documentatie zijn. 'Manual-pages' zijn makkelijk; gewoon die files deleten. Maar documentatie die ingebed is in code, is veel duurder om te verwijderen. Ook ongebruikte Linux-drivers zijn gemakkelijk te verwijderen. Maar bij andere onnodige code is dat veel moeilijker. Zo is 'swapping' niet nuttig in embedded systemen. Typisch staat het 'uit' in embedded Linux. Maar de code wordt niet verwijderd, dat is veel te duur!
Voordat we aan een embedded Linux systeem beginnen, moeten we ons afvragen hoeveel tijd we er voor overhebben om een klein systeem te maken. Hoe kleiner het systeem dat we willen, hoe duurder dat zal zijn om te ontwikkelen.

PC technologie is de norm

De meeste Linux-ontwikkelaars zijn PC gebruikers. Daarom wordt bijna alle PC-hardware ondersteund. Terwijl hardware die niet algemeen beschikbaar is, of waarvan de specificaties geheim zijn, vrijwel nooit ondersteund wordt.
Traditionele embedded systemen bevatten vaak veel specifieke hardware; dat kunnen FPGA's zijn, of componenten die niet in 'normale' computers te vinden zijn.
Wat de reden ook is, de kans om drivers te vinden voor componenten is het grootste als die componenten ook gebruikt worden in 'gewone' computers.
Dat geldt niet alleen voor gewone componenten, maar ook voor (communicatie)bussen. PCI en USB zijn heel gebruikelijk in de PC wereld. Maar ook in andere 'gewone' computers. Linux ondersteunt die out-of-the-box. Als er gewerkt wordt met typische embedded, of nog erger: eigen ontwikkelde bussen, dan zullen daar veelal drivers voor geschreven moeten worden. En dus zullen de kosten door extra ontwikkel- en testtijd oplopen.

2. Wat is gratis?

Projecten die Linux gebruiken, gebruiken bijna altijd ook andere OpenSource producten die net als Linux vaak gratis te gebruiken zijn. Dit gebeurt ook op 'normale' computers: de meeste software op een Linux-PC is eigenlijk geen Linux, maar afkomstig uit andere OpenSource projecten.

Het web en het internet

Linux is groot geworden door het internet. Er zijn dan ook heel veel netwerkapplicaties beschikbaar. Natuurlijk wordt TCP/IP ondersteund. Ook draadloze netwerken, ISDN en zelfs beveiligde netwerken zijn geen probleem.
Voor embedded systemen zijn er bovendien talloze webservers beschikbaar; deze kunnen gebruikt worden om het systeem te configureren.
Al deze software is gratis beschikbaar. Dat is een groot voordeel. Maar natuurlijk geldt dat alleen als er behoefte aan is. Zonder die behoefte, vervalt ook het Linux-voordeel. De keuze is aan u.

Multimedia en VoIP

Eén populair onderwerp in de OpenSource wereld mag special vermeld worden omdat sommige embedded systemen daar een voordeel mee kunnen behalen: Er is heel veel software beschikbaar om leuke dingen te doen met plaatjes, geluid en video. Zo is 'PODcasting' erg in opkomst: het op verzoek 'streamen' van audio, ofwel geluid versturen via het internet.
Dit klinkt wellicht erg ingewikkeld, maar omdat deze software als OpenSource beschikbaar is, is het goedkoop en eenvoudig om te realiseren met embedded Linux.
Ook voor VoIP ofwel 'bellen over het internet' is er veel software beschikbaar.
Veel moderne embedded systemen kunnen dergelijke moderne features gebruiken. In dat geval ligt er een kans met embedded Linux omdat veel software al beschikbaar is.

3. Wanneer gaan we over?

Ontwikkelen met embedded Linux gaat anders dan men in veel projecten gewend is. We hebben al een aantal voorbeelden gezien. Het moment van overstappen naar Linux heeft daarom veel invloed op het succes van de overstap. Snel en haastig overstappen zal niet werken. Maar wat zijn goede indicatoren?

Poorten versus nieuwbouw

Hier kunnen we kort over zijn. Overgaan op Linux is, als er verder niets veranderd, erg moeilijk. Bijvoorbeeld omdat er net te weinig geheugen is om Linux echt comfortabel te draaien. Trouwens, in hoeverre wordt de gebruikte hardware ondersteund? Ook kan het poorten van juist deze embedded toepassing erg lastig zijn.
Alleen maar overgaan op een ander OS is economisch vaak niet haalbaar. Technisch kan het wel; voor alle beschreven problemen zijn oplossingen. Maar het wordt dan duur, zeker voor de eerste keer.
Bij het ontwikkelen van een nieuw product, is de introductie van Linux het meest kansrijk. Dan kunnen hard- en software optimaal op elkaar afgestemd worden. En kunnen de ontwikkelinspanning geminimaliseerd worden. Ook bij een re-design zijn er goede kansen. Op dat soort momenten is er een mogelijkheid om voor net iets meer geheugen te kiezen. Of een chip te gebruiken die wel ondersteund wordt. Al die kleine dingen kunnen de ontwikkelkosten van Linux dramatisch beïnvloeden. En dus de overstap vergemakkelijken.
Natuurlijk, er zijn uitzonderingen. Als er al een industriële PC gebruikt wordt bijvoorbeeld. Of als het team veel Unix kennis heeft. Of omdat de bestaande software gewoon goed past op een Linux platform. Die voordelen maken het veel eenvoudiger om over te stappen naar Linux.

De oplage is bepalend

Linux heeft lage licentiekosten, maar wil graag krachtige hardware. Bovendien kan er vaak aanzienlijk bespaard worden op de ontwikkelkosten door een beetje extra geheugen te gebruiken. Of dit lonend is, is sterk afhankelijk van het aantal te fabriceren systemen.
Gezien de alsmaar lager wordende prijs van dat geheugen en de alsmaar groeiende vraag naar features blijkt het verstandig om het systeem te dimensioneren met 'te veel' geheugen. Dat maakt niet alleen de ontwikkeling goedkoper, maar komt later ook goed van pas. De praktijk leert namelijk dat anders de tweede (hardware) release alsnog met dat extra geheugen komt.
Voor de totale kosten, is het dan gunstig als er relatief weinig aantallen gemaakt worden. Linux is vaak goedkoop bij een relatief lage oplage.
Soms kunnen we dit inzetten als winnende strategie. Als de totale oplage slechts een vage schatting is, plan dan een beperkte eerste oplage met veel geheugen —de ontwikkelkosten zijn dan laag. En plan een tweede release met minder geheugen.
De initiële ontwikkelkosten zijn laag. Pas als meer zekerheid is over de totale oplage, wordt begonnen met het dure downsizen van Linux. Bovendien heeft dit een tweede voordeel: de time-to-market is veel korter worden.

4. Waar beginnen we mee?

Bij de ontwikkeling van embedded systemen staat de hardware traditioneel centraal. Terwijl Linux een platform met hardware abstractie biedt. Om de voordelen van embedded Linux optimaal tot hun recht te laten komen, is het belangrijk om even stil te staan bij dit beginsel.

De hardware is onbelangrijk

Linux gebruikt striktere grenzen, veel strikter dan traditionele embedded besturingsystemen. Er is een harde overgang tussen besturingsysteem en processen, tussen drivers en de rest, en tussen processen onderling. Die grenzen worden bovendien afgedwongen. Het voordeel is dat de '(eigen) toepassingen' –de software die de features implementeren– relatief onafhankelijk zijn van de gebruikte hardware. Zeker voor 'normale' computers is dat een voordeel.
Indien het niet mogelijk is om een grens te trekken tussen 'hardware specifiek' en 'hardware onafhankelijk', of als dat laatste deel niet het omvangrijkste is, dan is het moeilijk om embedded Linux te gebruiken. Gelukkig blijkt die grens in praktijk vaak gunstig uit te vallen; ook als dat in eerste instantie niet zo lijkt.
Na het trekken van die lijn, volgt een denkstap. Een stap die vreemd lijkt voor embedded programmeurs, maar erg gunstig kan zijn voor de ontwikkelkosten. Namelijk dat de hardware niet belangrijk is voor het merendeel van de software ontwikkeling!

Kunt u bedenken wat dat allemaal betekent?
In praktijk betekent dat, dat het belangrijkste ontwikkelwerk kan gebeuren zonder dat er hardware nodig is! Of eigenlijk met die hardware die beschikbaar is. Heel vaak betekent dat een oude PC, aangevuld met hardware die slechts lijkt op de te gebruiken hardware. Dat is pas goedkoop! Maar bovendien is het snel beschikbaar, wordt testen en debuggen veel eenvoudiger en dus zal het ontwikkelteam eerder aan de slag kunnen om veel sneller te werken.
Natuurlijk zal niet al het ontwikkelwerk gebeuren op zo'n geëmuleerd embedded systeem. Om drivers te schrijven, om te integreren en voor timing blijft echte hardware noodzakelijk. Maar Linux geeft de mogelijkheid te beginnen zonder dure, specifieke hardware. En eerder beginnen betekent sneller resultaat!

Beginnen is altijd moeilijk

Er is maar een manier om ervaring op te bouwen: gewoon beginnen. Hoe sneller hoe beter. Zo is overschakelen naar embedded Linux gemakkelijker als er een teamlid is dat Linux ervaring heeft. Al is het maar op de desktop. Maar ook 'embedded' en 'engineering' ervaring is nodig. Dat zijn aspecten die in veel 'normale' Linux distributies ontbreken. De meeste Linux CD's en de meeste boeken over Linux, richten zich op de typische gebruiker. Dus niet op de embedded ontwikkelaar. Begin daarom snel, maar begin wel met iets dat embedded Linux heet.
Voor de ontwikkelaars betekent dit: zo snel mogelijk wennen aan Linux, aan de typische begrippen en gewoontes van dit systeem. Zelfs een algemene, technische Unix cursus is nuttig. De diverse smaken Unix en Linux lijken voldoende op elkaar. Op elk Unix/Linux systeem is de gnu-compiler beschikbaar. Leer die eerst te gebruiken. Pas daarna is het nuttig om te leren hoe je daarmee kan 'cross-compileren' en wat 'canadian-cross' betekent.

Anderzijds is het nuttig om direct een geschikte Linux versie te kiezen. Daarbij zijn allerlei technische details minder van belang; veel belangrijker is de doelgroep. Zo is de gnu-compiler die bijgeleverd wordt bijna nooit geschikt, omdat die niet kan cross-compileren.
Maar zoek niet te lang naar zo'n systeem. In plaats van weken zoeken naar de beste Linux, is het beter om snel te beginnen om te leren wat 'een goede Linux' betekent.
Omdat u het toch wilt weten: als je met cross-compileren software kunt vertalen voor een ander type systeem, wat krijg je dan als je de compiler cross-compileerd? Als je dat goed doet: een native-compiler voor het target. Dat wordt 'canadian-cross' genoemd. En hoewel het erg handig is voor sommige embedded Linux projecten en heel makkelijk in gebruik, is het bouwen van zo'n canadian-cross zo complex dat maar heel weinig leveranciers het meeleveren.

5. Hoe moeten we kiezen?

Beginnen met Linux betekent keuzes maken. Veel keuzes maken. Zoals een (kernel)versie, een leverancier en de waarden van heel veel parameters. Voor beginnende Linux gebruikers is dat lastig. Vooral omdat onduidelijk is wat de consequenties zijn, zeker voor embedded systemen. Daarom een paar tips hoe snel en effectief te kiezen. De meeste keuzes zijn namelijk niet zo belangrijk.

Kies voor standaard en kies de defaults

Een van de belangrijkste voordelen van Linux, naast de licentiekosten, is het OpenSource principe van Linux. U heeft recht op de code en het recht om die te veranderen. Veel mensen hebben dat al gedaan. Dat is een van de redenen waarom u uit zoveel opties kunt kiezen. Weliswaar moet u kiezen, maar bijna elke keuze komt met een goede default. Kiezen voor die default-waarde is bijna altijd goed.
Zo zijn er ook heel veel andere standaard keuzes: Elk embedded Linux systeem gebruikt bijna de 'busybox', 'JFF2' is het standaard filesysteem voor embedded systeem en een 'oneven' kernel is niet bedoeld voor productiesystemen. Deze en andere keuzes zijn eenvoudig te maken door naar andere embedded systemen te kijken. Ook op internet is erg veel informatie te vinden.
Een belangrijke overweging voor elke keuze is 'wat doet de meerderheid'. Dat is niet alleen een makkelijke weg, het is ook een verstandige. Immers als u dezelfde opties kiest, gebruikt u ook de meest gebruikte code en dus de best geteste code. Alles wat standaard is, is veelvuldig gebruikt en getest. Niet alle code van Linux is even goed en er zijn zeker fouten te vinden. Door een verstandige strategie vindt een ander die fouten als eerste, en profiteert u van de oplossingen

Snel kiezen schaadt niet.

We hebben gezien dat Linux goedkoop is als we dicht bij de standaard Linux blijven. Dus zonder exotische hardware en met onbeperkt geheugengebruik. Dan zijn ook allerlei aantrekkelijke extra's als multimedia, networking of een laagdrempelige webinterface voor de configuratie heel goedkoop mogelijk. Daarnaast hebben we gezien dat sommige instapmomenten handiger zijn dan andere.
Vooral bij kleine projecten is het vaak belangrijk om snel te ontwikkelen. De oude wijsheid dat ontwikkeltijd niets kost als de stukprijs maar omlaag gaat, is dan niet geldig.
Natuurlijk, een embedded systeem zal niet louter bestaan uit PC-onderdelen. Noch kunnen we eindeloos besparen op ontwikkeltijd. We zullen ook naar de fabricagekosten moeten kijken. Want uiteindelijk zal Linux moeten gaan draaien op uw hardware en genoegen moeten nemen met beperkte resources.

Maar zouden we niet kunnen doen alsof? Althans voorlopig!
Veel potentiële besparingen worden in praktijk verspild. Verspild door eindeloos na te denken; over hoe en welke Linux te installeren en door uit te gaan van de technische mogelijkheden in plaats van de economische opties. Zo is het mogelijk om Linux-ervaring op te doen door meteen Linux-drivers te gaan schrijven, maar dat is zelden de efficiëntste manier. Zeker is, dat het veel doorlooptijd kost en dus de economische voordelen van Linux uitholt. Terwijl er nog wel een oude PC rondzwerft, die prima geschikt is voor (embedded) Linux. Probeer dat eens, dan weet u na een halve dag al wat Linux is en hoe groot of klein Linux is.

De kracht van Linux is de ontkoppeling tussen hard- en software. Die ongebruikte, oude PC is prima geschikt om een embedded platform te emuleren. Hoe ouder, hoe beter: een PC van 10 jaar oud lijkt sterk op het embedded systeem van vandaag. Gebruik dat als 'het embedded test systeem'. U kunt morgen al een programmeur vragen om de applicatie eens te compileren op dat embedded Linux systeem. Als u het niet doet, zult u nooit weten of het eind volgende week al deels werkt!

Snel ontwikkelen is de kracht van (embedded) Linux. Het is slechts zonde van de tijd om niet snel te beginnen. U kunt volgende week al weten of uw applicatie makkelijk om te zetten is naar Linux. Dan weet u ook hoeveel geheugen er te veel gebruikt wordt. Dus kunt u een realistische schatting maken van wat er bespaard moet worden. En hoeveel het oplevert. Hoe dat in praktijk heeft gewerkt, leest u in het artikel “Linux niet geschikt! Doel toch bereikt.”

Good Luck
ALbert Mietus