

L10.301

## Pretentie en doelstelling van het THE multiprogrammeringsproject.

Zoals bekend bestaat 90% van de wereldbevolking uit kruideniers. Omdat onder programmeurs het percentage ten minste zo hoog is, leid ik de overige sprekers in.

Het is mijn ervaring dat de overgrote meerderheid der programmeurs, systeembouwers en zelfs machinebouwers hun werk in eerste instantie beschouwen als een optimalisatieopgave, nl. de minimalisering van wat ze de "cost/performance ratio" noemen. Wij hebben geweigerd - of, zo ik wilt, ik heb geweigerd en mijn medewerkers zijn mij bijeen bijgevallen - om aan dit efficienystreven het primaat te geven en wij hebben ~~de constructie~~ het multiprogrammeringsproject in eerste instantie beschouwd als intellectuele uitdaging: hoe kan je zorgen dat in een dergelijk systeem de complexiteit je niet boven het loofd greift, hoe kan je zorgen dat je het ontwerp- en constructieproces blijft beheersen. Slechts binnen de spelruimte, die ons hierdoor gelaten werd hebben wij efficiencyoverwegingen ons duren en laten laten beïnvloeden.

Ik heb ervaren dat geschilderde relativering van de efficiency eis voor hen, die van efficiency een heilig ideaal gemaakt hebben, niets meer of minder dan heiligschennis is. Ik ben dan ook bejegend als de rebel, die zich niet aan de spelregels gehouden heeft. Wanneer na afloop blijkt, dat wij in weerwil van de verwachting waarvan we beschuldigd worden, een systeem geschapen hebben, waarvan de "performance figures" in dezelfde orde van grootte liggen, dan is de bood helemaal aan. Voor kruideniers is dit natuurlijk ook heel frustrerend en het ontlokt allerlei verzetsreacties tot bijna onverblende verdoelmaking toe.

Maar ik vraag u een pas terug te doen. Lange tijd hebben rekenautomaten alleen ~~ingebouwd~~ fixed point arithmetiek ingebouwd gehad, niet alleen omdat floating point arithmetiek moeilijker te bouwen was, maar ook omdat John von Neumann ~~daar~~ ervan overtuigd was, dat je floating point arithmetiek niet nodig had, je kon je probleem best schalen. Desalniettemin zijn software makers met verwachting van enige ~~efficiency~~ efficiency, ~~er~~ floating point arithmetiek gaan implementeren en nu zijn we blij, dat dit in de hardware zit. Je zou in een moment van optimisme kunnen denken, dat de mensheid van die ervaring geleerd zou hebben, maar niets daarvan. Een aantal jaren later bij de opkomst van de programmeertalen, werd er weer moord en brand geschreeuwd

nl. door die programmeurs die in een geïsoleerde exploitatie van een speciale machinecode hun hoogste goed zagen. En eindelijk zijn daarin blijven steken. En nu heb je de volgende generatie ruw- en brand-schreefprogramma's, nl. de programmeurs die de exploitatie van een specifieke configuratie als hun hoogste taak zien. Het wordt eentonig.

Maar het is duidelijk dat men de kunst van het programmeren, de kunst van het machinegebruik alleen dan verder kan helpen, wanneer men, zo daar goede gronden voor aan wezig zijn, bereid is de platgetreden paden te verlaten. En hierin ligt een van de redenen, waarom ik aan efficiency overweegingen, niet het prikkel wil geven; dat is nl. zo verlamdend. Ik wil ze wel in het oog houden, maar dat is heel iets anders. En er waren ditmaal goede gronden. Toen we er aan begonnen, zijn we nog voor gek verlaten - schriftelijk zelfs, maar ik zal de firma die dat gedaan heeft, niet noemen. U mag ~~want ikzelf~~ een keer raden - de ontwikkeling heeft aangetoond dat er in de rest van de wereld (laten we het voorzichtig uitdrukken) meer van zulke gekken te vinden waren.

Nu over het project zelf. Ik heb de EL X8, waarvoor we dit gedaan hebben, nooit beschouwd als ~~machine die is ontwikkeld voor~~ speciaal voor multiprogrammering ontworpen, ~~op zijn best~~ <sup>eer</sup> als inspirerende machine om het mee te proberen. Ik neem in dit verband speciaal de interrupt hardware, waarop je op logische gronden verteld kunt worden. ~~Ik wil hier graag met nadruk verklaren, dat wij het multiprogrammeringsproject~~ Ik wil hier graag met nadruk verklaren dat het antimeren van het multiprogrammeringsproject onverantwoord zou zijn geweest, als wij niet met zekerheid hadden kunnen aannemen dat we, in geval het niet zou lukken, de software van het MC achter de hand zouden hebben om hier het Rekencentrum te bedienen. Zonder dat vertrouwen hadden wij ons nooit kunnen permitteren dit stukje "Hogeschool programmering" weg te geven. Ik wil graag aan de vergeetbaarheid onthouden dat het tot begin 1967 geduurd heeft. Vandaar wij er vertrouwen in begonnen te krijgen, dat we een werkend "workable" systeem zouden produceren. En zelfs sindsdien heeft dat vertrouwen nog wel ons gewankeld. Je vergeet dit zo gauw, maar we zijn heel vaak heel bang geweest.

Wie een multiprogrammeringssysteem ontwerpt, loopt onmiddellijk tegen de tegenstrijdige problemen aan. Je kunt het je in de een mate maken en royaal met vaste tegenstellingen werken, je kunt het je moeilijk maken en zoveel mogelijk dynamisch uitwisselbaar houden. Wij zijn in de laatste richting vrij ver gegaan en hebben ons door dit bezogde raffinement allerlei ellende op de hals gehaald. Voor kenners neem ik de bankiersalgoritme als notoir voorbeeld; voorts neem ik de bereidheid van het systeem om de geheugenruimte, die normaal voor transport buffering ter beschikking staat, dynamisch in te krimpen als ~~de voor de programma verwerking benodigde geheugenruimte~~ <sup>meer</sup> ruimte voor de programma verwerking nodig blijft. ~~Men kan wel denken dat de~~ ~~aan stabiel van het systeem~~ Tilt men niet zo zwaar aan dit raffinement, dan kan men een goed stuk van het systeem beschouwen als "self inflicted pain". In antwoord hierop zien ik, zonder nu mijzelf als masochist te willen presenteren, willen zeggen "Daar ging het ons in zekere zin om." Wij zijn geen lustprogrammeurs in de zin, dat we het ons liever moeilijk dan makkelijk hebben gemaakt. Integendeel, elke ~~complicerbare~~ complicerende factor moest en aanvaardbaar doel dienen. Vooral bij werd toegepast en in het ontwerp stadium hebben we ~~er nog een heleboel afgepraat~~ <sup>dat</sup> ook ettelijke er weer uitgepraat. Het ging ons wel om ~~een~~ op gawe die alle mogelijkheden in zich borg om ondoenlijke moeilijk te worden <sup>dan</sup> te ontdekken hoe men vervolgen het werk zo makkelijk kan houden, dat men het tot een goed einde brengen kan.

Het ~~een~~ multiprogrammeringssysteem is met een dubbel doel op poten gezet. Ten eerste wilden we voor kleine programma's de turn-around time verkorten; ~~verkleinen van de turn-around time van een programma, dat is~~ ~~minimaal machinetijd vergt~~ ~~op~~ ~~dat~~ ~~was~~ ~~bekend~~ ~~dat~~ ~~by~~ ~~batch~~ ~~processing~~ ~~voor~~ ~~een~~ ~~programma~~ ~~het~~ ~~was~~ ~~bekend~~ ~~dat~~ ~~deze~~ ~~by~~ ~~batch~~ ~~-~~ ~~processing~~ ~~makkelijk~~ ~~in~~ ~~de~~ ~~uren~~ ~~kan~~ ~~lopen~~ ~~,~~ ~~ook~~ ~~als~~ ~~het~~ ~~programma~~ ~~zich~~ ~~zelf~~ ~~maar~~ ~~een~~ ~~minuut~~ ~~machinetijd~~ ~~nodig~~ ~~had~~ ~~en~~ ~~dit~~ ~~leek~~ ~~ons~~ ~~onaantrekkelijk~~ ~~en~~ ~~door~~

Het multi-programmeringssysteem is met een dubbel doel ontwikkeld. Ten eerste wilden we voor kleine programma's de turn-around time verkorten. Van batch-processing systemen was ons bekend, dat de turn-around time vaak in de uren ging lopen, ook al vergde het programma's maar enkele minuten rekentijd. Aangezien wij door de studenten bevalling vele kleine programma's verwachtten, ~~leek dit onaanvaardbaar~~ vonden we dit onaanvaardbaar. Een multi-programmeringssysteem, waarin een langdurig programma de machine niet blokkeert, schept een van de voorwaarden om de turn-around time voor kleinere ~~programma's~~ veel kleiner te laten zijn.

Ten tweede wilden we de mogelijkheid tot enkele gebruiksvormen scheppen, die zonder multi-programmering zouden afketsen op de ~~geheugen~~ <sup>inefficiënte</sup> benutting van het rekenorgaan. ~~Wij noemen twee aspecten. Ten eerste moeten~~  
~~We wilden de programmeur bevrijden van de kennis van de tweestrichigheid~~  
We wilden de programmeur ontlasten van de plicht rekening te houden met de tweestrichigheid van het geheugen: het systeem verzorgt automatisch de noodzakelijke transporten tussen langzaam en snel geheugen als het snelle geheugen te klein blijkt. Dit geschiedt ~~na~~ volgens "demand paging", d.w.z. informatie wordt pas uit het langzame geheugen overgehaald als ~~het nodig is~~ ~~het nodig is~~ ~~in het verwerkende proces~~ er aan toe is en de informatie ~~op dat moment~~ <sup>het dan volgende</sup> niet in de kern aanwezig is. Gedurende <sup>het dan volgende</sup> transport kan het verwerkende proces niet verder! Multi-programmering is een methode om gedurende die transporttijd nochtans employé voor het rekenorgaan te vinden.

~~Volgens welke van de machine kunnen gebreken in programma's,~~  
Vervolgens wilden we de machine kunnen gebruiken voor processen, die kwalitatief qua slechte op een fractie van ~~de machine~~ ~~de tijd~~ van het rekenorgaan bezlag leggen. Wij noemen in dit verband de extra bandreeler en bandpenseer, waarmee men "in een hoelge van de machine" ~~op een~~ ~~op een~~ banden kan copyeren, corrigeren etc. En de tweede teleprinter die voor conversationeel werk bestemd is, en <sup>experimenteel</sup> een dezer weken in het systeem zal worden opgenomen. De derde ~~toepassing~~ toepassing, waarmee we rekening ~~we~~ hebben willen houden was on-line koppeling met meet-apparatuur. Dit laatste is nog niet gerealiseerd, in de

enkele gevallen, dat er sprake van is geweest bleken de hiernaar  
 verbanden verplichtingen kwantitatief gezien prohibitief te zijn.

~~Wat mij niet hebben willen maken is een multi-access systeem dat  
 consoles alom bedient. Dit was een "twee jaar geleden" te  
 dertien en "maar ik houd er ernstig rekening mee, dat dit een  
 in zijn huidige vorm een model van voorbijgaande aard zal blijven.  
 Met een goedkoop en langzaam communicatiesysteem als een  
 teleprinter kan je een gebruiker inderdaad de illusie geven  
 dat hij op zijn werken bedient wordt, voornamelijk omdat dan  
 het "werken" een hydrovend proces is, dat de gebruiker alsmat  
 bezig houdt. Met krachtigere communicatiesystemen is het voorhands  
 onbetaalbaar. Als de multi-access faciliteiten - en dat gebeurt vaak -  
 aangegeven worden als de ideale "debugging aid", dan ga ik echter  
 stijgen, want dan wordt het <sup>systeem</sup> gebruikt voor de opsporing en de reparatie  
 van fouten tot het maken waarvan het systeem je eerst heeft uitgenodigd.  
 Zo het idee van "Begint eer ge bezint."~~

Wat mij niet hebben willen maken was een multi-access systeem  
 dat consoles alom bedient. Dit was een "twee jaren geleden" te  
 dertien en "maar ik houd er ernstig rekening mee dat dit in  
 zijn huidige vorm een model van voorbijgaande aard zal  
 blijven. Ik voor mij van de installatie van een oergetuigd systeem  
 ook wel durven aanbevelen voordat mij een verantwoorde  
 wijze van gebruik voor ogen stond. Maar zwaets hebben  
 we met dit systeem ten ongelukte dus niet geprobeerd.

Rest mij slechts te danken voor uw aanwysing  
 en uw belangstelling en het woord te geven aan  
 de volgende spreker.