Ik zal U van een papiertje toespreken, niet omdat ik dat liever doe, integendeel zelfs, maar opdat er na afloop geen verschil van mening hoeft te bestaan over wat ik gezegd heb.

Aan onze vriend Posthumus danken we het beeld, dat onderzoek en onderwijs voor de wetenschap zijn als schering en inslag van eenzelfde weefsel, waaraan onlangs de opmerking is toegevoegd dat als men ze scheidt, er ook slechts poetskatoen zal overblijven. Het is mij een voorrecht in heel kort bestek te mogen proberen U duidelijk te maken, wat dit voor de informatica inhoudt.

Het is wel bekend, dat electronische rekenmachines duur zijn, het is zelfs zo bekend , dat deze wetenschap dreigt ons kostenbeeld ernstig te vertekenen. De feitelijke uitvoering van de programma's mag door hun beslag op rekentijd dan duur zijn, de ontwikkelingskosten om deze programma's te maken is in het algemeen vele malen hoger. En het is onder andere om deze louter economische reden al, dat op de informatica de dure plicht rust te proberen om hierin een drastische verbetering te brengen, om programmeurs te kweken, die in hun beroepsuitoefening een orde van grootte effectiever zijn, dan wij tegenwoordig gewend zijn. Die grotere effectieviteit kan zich manifesteren in hun vermogen programma's te maken, die onder verstoking van minder rekentijd dezelfde resultaten bereiken, veel belangrijker is dat ze leren een orde van groote effectiever met hun eigen "brainpower" om te springen, want het is in hun manuren, dat de grote kosten gaan zitten.

Nu is een paar jaar geleden in een grote organisatie min of meer bij toeval aan het licht gekomen, dat de individuele effectieviteit van programmeurs, die door hun management nota bene als zo ongeveer gelijkwaardig beschouwd werden, bij nadere beschouwing onderling zo ongeveer een factor 20 uit elkaar bleek te liggen. Deze schokkende ontdekking was een sterke prikkel om te proberen achter de oorzaak van dit enorme effectiviteitsverschil te komen, dit met het uiteindelijke doel te komen tot een doceerbare methodologie waarmee deze enorme effectiviteitswinst kan worden binnengehaald.

Voor onze samenleving is het, voorzover die afhankelijk is van het succes waarmee automatische rekenmachines worden ingezet, van eminent belang, dat deze winst wordt binnengehaald, en hier is meer in het geding dan louter de prijs van programmaontwikkeling, uitgedrukt in manuren. In een moment van optimisme zou men een project, dat door omvang en complexiteit een groep van 10 man vele malen teboven gaat, als de nodige fondsen ter beschikking zijn kunnen proberen

te bemannen met zeg 200 man. Dit is geprobeerd maar de pogingen om de schaalvergroting in de programmeringsproblematiek op te vangen door er alsmaar meer mensen bij in te schakelen, zijn faliKant mislukt, nog helemaal los van het feit, dat deze aanpak een beetje onbetaalbaar begon te worden: in weerwil van de beste organisatorische maatregelen worden bij een dergelijke taakverdeling de onderlinge communicatiemoeilijkheden nl. onoverkoombaar. Bij de dan noodzakelijke informatieuitwisseling beginnen zich dezelfde congestieverschijnselen voor te doen als bij het autoverkeer in een grote stad. De moraal van deze, achteraf begrijpelijke, mislukkingen is, dat in plaats van de groepen tot onhanteerbare grootte te laten groeien, liever de effectiviteit van de individuele programmeurs genoemde orde van grootte opgekrikt zou moeten worden. Vandaar de levendige belangstelling voor een methodologie, die dat zou verwezenlijken, maar de brandende vraag is natuurlijk geweest of wij mochten hopen, dat een dergelijke methodologie ontwikkeld zou kunnen worden.

De aanvankelijke skepsis heeft gelukkig een aantal beoefenaren van de informatica er niet van weerhouden naar een dergelijke methodologie te zoeken en hun werk lijkt met meer succes bekroond te worden dan we aanvankelijk misschien mochten hopen. Het zoeken naar en beproeven van een dergelijke methodologie is bij lange na nog niet voltooid maar een aantal elementen beginnen zich met meer en meer overtuigingskracht af te tekenen. Voor de effectieviteitswinst bij het programmeerproces die dankzij een beter inzicht hierin nu haalbaar is, taxeer ik vergeleken bij hoe thans geprogrammeerd wordt een factor van tenminste 5, voor de toekomst acht ik een factor 30 of hoger geenszins onmogelijk. Op een gegeven moment laat het verschil in competentie zich natuurlijk niet meer in een factor uitdrukken, want dan krijg je te maken met het verschil tussen iets wel kunnen en iets niet kunnen, ongeacht hoeveel tijd je ervoor zou krijgen.

Het gaat hier om een uitgesproken Europese ontwikkeling, die in de Verenigde Staten uit de aard der zaak met belangstelling gevolgd wordt en sommigen heel wel aanspreekt, hoewel het niet een-twee-drie duidelijk is hoe men in het Amerikaanse werkklimaat de vruchten hiervan ten volle zou moeten plukken, een moeilijkheid die wel dezelfde oorzaak zal hebben als het feit dat in Amerika niet een analoge ontwikkeling zelfstandig, helemaal onafhankelijk van Europa, heeft plaatsgevonden. De voornaamste oorzaak van dit Amerikaanse achterblijven lijkt me de daar prevalerende neiging om aan kwantitatieve argumenten het primaat te geven, waardoor te vaak voortijdige kwantificering in de hand gewerkt wordt en vitale imponderabilia tussen wal

en schip komen. Vanuit technisch oogpunt bekeken lijken de toekomstmogelijkheden voor de Europese software-industrie dan ook heel gunstig, maar we zullen dan wel de moed moeten opbrengen om niet slaafs te volgen en niet vandaag uit Amerika te importeren, wat daar gister al als vergissing ontmaskerd is.

Eén ervaring van dit onderzoek wil ik niet onvermeld laten en die is, dat de invloed van een stuk gereedschap op de mens, die probeert dit stuk gereedschap te gebruiken, vele malen sterker is dan doorgaans wordt aangenomen. En waar het gereedschap in dit geval bestaat uit programmeertalen, moet je constateren dat wanneer aan een programmeertaal maar voldoende bezwaren kleven, zijn gebruikers voor het concipieren van hele klassen van mogelijke oplossingen mentaal volledig geblokkeerd worden; en in het gebruik is een dergelijke programmeertaal veel en veel te duur: na onevenredig veel denktijd komt de programmeur met een onbeholpen programma. Ik ben hier zó van onder de indruk, dat ik me ernstig afvraag of het moreel verantwoord is mensen af te richten [illegible] op het gebruik van in het bijzonder FORTRAN, COBOL en PL/1, omdat zo'n opleiding meer [illegible] lijkt op misleiding: wie dit doet bezondigt zich mijns inziens aan geestelijke misvorming, begaat onrecht jegens de slachtoffers en berokkent schade aan het vaderland. Je dwingt deze mensen tot net zo iets onbeholpens als met potlood en papier vermenigvuldigen en delen, maar dan in Romeinse cijfers.

Tenslotte -en ook deze laatste opmerking heeft met onderwijs te maken- de ontwikkeling van een methodologie is een steriele bezigheid, wanneer die methodologie niet overdraagbaar is en de enige overtuigende manier, die ik ken om deze overdraagbaarheid te testen is ... de overdracht te proberen! En hier zijn we terug bij de verwevenheid van onderzoek en onderwijs, waarmee ik begonnen ben. Het betekent dat de docent , opdat wat hij vertelt de moeite van het aanhoren waard blijft, de ondelegeerbare plicht tot onderzoek heeft, maar ook dat de student naast spons tevens wetsteen moet zijn. Mogen beide partijen tegen beide facetten van hun taak zijn opgewassen.