Titel: De software crisis, ontstaan en hardnekkigheid.

Spreker: prof.dr.Edsger W.Dijkstra
Burroughs Corporation/
Technische Hogeschool Eindhoven

Samenvatting:

Het ontstaan van de software crisis is een direct gevolg van de onderschatting van het revolutionaire karakter van de moderne computer, welks programmeeropgave ons confronteert met een intellectuele uitdaging zonder precedent. De ontwikkeling van een feilloos programma van behoorlijke omvang is een opgave waarvan de moeilijkheidsgraad aanvankelijk schromelijk is onderschat; het voortduren van de software crisis is grotendeels te wijten aan het feit dat deze moeilijkheidsgraad over het algemeen tot op de huidige dag ontkend wordt. Op de achtergrond van deze ontkenning zal nader worden ingegaan.

De software crisis, ontstaan en hardnekkigheid

Edsger W.Dijkstra
Burroughs Corporation/
Technische Hogeschool Eindhoven

Officieel bestaat de software crisis sinds october 1968; toen werd zijn bestaan namelijk openlijk toegegeven op de NATO Conferentie over "Software Engineering" die te Garmisch-Partenkirchen gehouden werd. Maar hij bestond natuurlijk al veel langer, want anders was die conferentie toen nooit georganiseerd. En in het jaar onzes Heren 1982 bestaat hij nog steeds, want anders was deze ASI leergang er niet aan gewijd. En dat de software crisis nog wel een tijdje zal voortduren is niet een gewaagde voorspelling. Deze voordracht zal hoofdzakelijk gewijd zijn aan zijn ontstaan en zijn hardnekkigheid.

Op de vraag "Hoe is de software crisis ontstaan?" is een heel makkelijk antwoord te geven. Waarom mislukken zoveel softwareprojecten? Heel eenvoudig: omdat programmeren blijkbaar iedere keer weer veel moeilijker is dan wij denken. Dit onontkoombare antwoord is evenwel te makkelijk, want het roept onmiddellijk de vraag op hoe de blijkbaar schromelijke onderschatting van de moeilijkheidsgraad van de programmeeropgave heeft kunnen ontstaan en blijkbaar zo'n taai leven kan hebben.

Voor de aanvankelijke onderschatting van de moeilijkheidsgraad van de programmeeropgave is een heel duidelijke verklaring; zij bestaat uit twee gedeelten. Ten eerste eiste in de beginjaren de constructie van betrouwbare hardware alle aandacht op. De electronica betrad het onontgonnen terrein van de digitale techniek, daarbij tegelijkertijd geconfronteerd met een schaalvergroting zonder precedent en ongehoorde betrouwbaarheidseisen.

Aan hardwareproblemen hadden we toen echt onze handen meer dan vol en de waarschuwing dat, als de machine eenmaal werkte, het gebruik niet triviaal zou zijn was het laatste, waar we behoefte aan hadden. Als de waarschuwing geuit werd --en dat werd hij wel-- , werd hij niet gehoord. Dit is maar al te begrijpelijk: het was immers niet de eerste zorg.

Ten tweede werd de waarschuwing slechts schuchter gegeven, niet alleen omdat er veel moed voor nodig is om een impopulaire boodschap van de daken te schreeuwen, maar ook omdat de feitelijke ervaring, die de waarschuwing zou moeten schragen, nog ontbrak. We wisten dat het eerst ontwikkelde programma --voor het genereren van een tabel van quadraten-- meteen had gewerkt; we wisten ook dat het tweede programma --voor het genereren van een tabel van priemgetallen-- dat tot ontnuchtering van zijn auteurs niet had gedaan. Maar met die ervaring ontmoedig je niet een hoopvolle mensheid. Het was pas hun tweede programma, die kerels hadden toch nog geen enkele ervaring, en, bovendien, wie is er nou in priemgetallen geinteresseerd? Het punt is, dat automatische rekenmachines toen inderdaad artefacten zonder enig precedent waren en dat de programmeeropgave inderdaad eisen aan ons intellect stelt, die er nog nooit aan waren gesteld. Zonder eerder aan een dergelijke discontinuiteit te zijn blootgesteld is het inderdaad heel moeilijk voorstelbaar dat iets, dat toch doenbaar moet zijn, onvoorstelbaar moeilijk uitpakt. We kunnen het naiveteit noemen, onschuld klinkt vriendelijker, maar over de aanvankelijke onderschatting hoeven we ons niet te verbazen.

Boeiender is de vraag, hoe het mogelijk is geweest, dat deze onderschatting zo'n taai leven leidt. Langzamerhand is er meer dan een kwart eeuw overweldigende experimentele ervaring dat programmeren op zijn zachtst gezegd een uiterst verraderlijke activiteit is, zo verraderlijk dat hij de inzet van ons

beste intellect rechtvaardigt. Maar de onderschatting duurt voort tot op de huidige dag. Deze ASI leergang is er een symptoom van: bij de voorbereidende sprekersbijeenkomst legde de secretaris, Mr. W.L. den Dekker, uit dat het traditioneel technisch/wetenschappelijke accent van dit soort leergangen dit maal "in verband met het onderwerp natuurlijk minder geprononceerd zou zijn". Een verklaring van het taaie leven van de onderschatting is ingewikkelder: een aantal oorspronkelijk onafhankelijke oorzaken blijkt elkaar wederzijds te versterken.

Het staat buiten kijf dat, toen het phenomeen computer zich kort na de tweede wereldoorlog aandiende, de internationale wiskundige wereld er niet rijp voor was. Traditioneel bestaat die wereld uit twee continenten, namelijk het wiskundig werelddeel dat zich bezighoudt met de problemen van het continuum en het deel dat zich op de discrete problematiek concentreert. Het zo eminent toepasbare werk van Newton, nu zo'n 300 jaar oud, is voor de continue wiskunde een enorme stimulans geweest en bij de stormachtige ontwikkeling van dat deel der wiskunde is de discrete wiskunde ontegenzeggelijk op het tweede plan geraakt. Maar de digitale rekenautomaat manipuleert per definitie nu eenmaal objecten in een discreet universum. Voeg hieraan toe, dat dit universum, hoewel groot, per definitie eindig is, en men kan zich misschien voorstellen dat het merendeel der wiskundigen van die generatie zich gemachtigd voelde het phenomeen computer te negeren: discreet en eindig moest het immers triviaal zijn. Men kan hun deze aanvankelijke beroepsblindheid bijna vergeven. Helaas moeten wij constateren, dat in de daaropvolgende 35 jaar deze beroepsblindheid nauwelijks is afgenomen en dat de wiskundige, die de intellectuele uitdaging van computing science serieus neemt, nog steeds een zeldzaamheid is en dat de gemiddelde wiskundige in bekrompenheid voor zijn niet-wiskundige medeburgers niet onderdoet. Overheerst bij wiskundigen eenmaal het gevoel dat computers niet de moeite waard zijn

om je mee bezig te houden, dan werkt dat vervolgens als een z.g. "self-fulfilling prophecy": als de intellectueel het best geequipeerden het vak links laten liggen, wordt het gebied bezet door tweede- en derderangs mensen en na enige tijd kan de echt knappe jongen zich nog moeilijker voorstellen dat daar een taak voor hem ligt. Ligt een stuk van de oorzaak van de software crisis bij het feit dat de wiskundige wereld door de bank genomen het gebied heeft veronachtzaamd, het taaie leven van de software crisis is mede het gevolg van het feit dat de wiskundige wereld door de inmiddels ontstane omstandigheden in deze veronachtzaming gesterkt wordt.

Naast deze culturele oorzaak is er ook een economische, die ging spelen op het moment dat Sperry-Rand als eerste rekenmachinefabrikant de UNIVAC als serieproduct op de markt bracht. Die markt moest gecreeerd worden, en die markt is gecreeerd door ondeskundigen, die over voldoende fondsen beschikten, gouden bergen te beloven. De gewenste ondeskundigheid was rijkelijk voorhanden in de top van grote bedrijven, die toen nog over geld beschikten; voor de gouden bergen zorgden de verkoopafdelingen der computerfabrikanten. Termen als "analytical engine", "mathematical machine" en "instrumentelle Mathematik" lieten er geen twijfel over bestaan waar in onze cultuur dit soort apparatuur was ontstaan en waarschijnlijk thuishoorde. Hoe verkoop je het dan aan een administrateur? Door met een groots opgezette campagne het wezenlijk wiskundige karakter van de hele gebruiksproblematiek onder tafel te praten, programmeren voor te stellen als iets dat iedereen in een drieweekse cursus kan leren, kortom door zo hardnekking te verklaren dat het goud der beloofde bergen massief is, dat genoeg mensen het geloven. In de jaren die daar op volgden is er voor het vak van programmeur volledig critiekloos geronseld. Hier in den lande was MULO ruimschoots genoeg, maar in andere landen was het geen haar beter. Het resultaat laat zich denken:

van de half-miljoen professionele programmeurs, die de wereld inmiddels telt is het merendeel van een incompetentie die elke beschrijving tart. Maar hier ligt wel een verklaring van het taaie leven van de software crisis: een incompetent arbeidsleger van een half miljoen ververs je niet een-twee-drie.

Heel pessimistische mensen betogen, dat de teerling is geworpen en dat we nooit van de horde incompetente programmeurs verlost zullen worden. Die sombere voorspelling is gebaseerd op de volgende, ongetwijfeld juiste observatie. Om met dit soort personeel er toch nog het minst slechte van te maken heb je veel van dat personeel nodig, dat vervolgens alleen maar werken kan in een organisatie zo strak, en ook zo infantiliserend, dat de mensen die je eigenlijk zou moeten hebben daar niet in passen. De observatie is ongetwijfeld juist; het is ook duidelijk dat het geobserveerde verschijnsel mede voor de taaiheid van het leven van de software crisis verantwoordelijk is. De pessimistische conclusie, dat de software crisis een eeuwig leven beschoren zal zijn lijkt mij evenwel een beetje haastig.

De sterkte van dit pessimisme verschilt van mens tot mens; over het algemeen is het in het bedrijfsleven sterker dan in het onderwijs. Het verschilt ook van land tot land; over het algemeen is het in de Verenigde Staten sterker dan in Europa. Dit laatste is niet verwonderlijk, aangezien de aftakeling van het onderwijs ginds nog verder is voortgeschreden dan hier. Het loont de moeite, daar en passant even de aandacht op te vestigen, omdat onze perceptie van wat de centrale problemen van computing science zijn de neiging heeft sterk beinvloed te worden door de Amerikaanse perceptie daarvan. Bij alle verhalen over de noden van "the casual user" en de intellectuele beperkingen van "the average programmer" moeten wij nimmer vergeten een duidelijk onderscheid te maken tussen enerzijds de intrinsieke

problemen van computing science en anderzijds de moeilijkheden veroorzaakt door het falend Amerikaans onderwijsbestel. Van een academicus vanzelfsprekend vergen dat hij zijn moedertaal in woord en geschrift voortreffelijk beheerst en effectief hanteert geldt hier inmiddels als een beetje overdreven, in de Verenigde Staten als absurd en irreeel. Alle Amerikaanse nadruk op "readability", "self-documentation" en soortgelijke, niet nader gespecificeerde deugden wordt daardoor wel begrijpelijk.

Waar het pessimisme overheerst, heeft dat een ongewenst neveneffect. Het leidt er toe, dat de software crisis gezien gaat worden als een ongeneeslijke ziekte, en gelijk wij allen weten is de ongeneeslijke ziekte de bij uitstek vruchtbare bodem voor kwakzalverij. En inderdaad: ik ken geen professie met een zo grote kwakzalverdichtheid als de onze! Dit is een aspect waar ik het merendeel van mijn academische collegae in de Onderafdeling der Wiskunde en Informatica van de Technische Hogeschool te Eindhoven geregeld aan moet herinneren. Het is een verschijnsel, dat zij in hun eigen professie niet meemaken en waarmee zij alleen geconfronteerd worden, wanneer een onderwijskundige hun pad kruist. Maar wie met pek omgaat, wordt er door besmet: je kunt van buitenstaanders nauwelijks verwachten dat zij kaf en koren scheiden, en de hoge kwakzalverdichtheid in onze professie maakt het voor de academische wereld eens zo moeilijk het vak serieus te nemen.

Ligt de schuld van de charlatannerie bij de kwakzalver of bij het publiek dat hij bedot? Wel, als dat publiek bedot wil worden, ligt de schuld dunkt me daar. In het bedrijfsleven is zulks dunkt me het geval. Met heel serieuze bedoelingen karakteriseerde ik enige manieren die bij het orde scheppen in wat anders onbeheerste chaos wordt veelbelovend leken en lanceerde ik termen als "structured programming", "layers of ab-

straction" en "separation of concerns". In de kortst mogelijke tijd werden niet de bedoelingen, maar wel de kreten gemeengoed en bevond ik me tegen wil en dank ingedeeld bij de goeroes. Als het zelfs de serieuze wetenschapsman zo vergaat, kunt U nagaan hoe moeilijk het is geen kwakzalver te worden.

Het bedrijfsleven is zo wanhopig, dat elke heilsboodschap er in gaat als koek. Aan heilsboodschappen is dan ook geen gebrek. Voor de wanhopige manager is het bijvoorbeeld een ontstellend geruststellende gedachte, dat de gebezigde programmeertaal de bron van al zijn ellende is. De stakker klampt zich vast aan de droom van de programmeertaal, waarin programmeren zo makkelijk is, dat het allemaal vanzelf goedkomt. De nieuwe programmeertalen als panacee gaan in de kwakzalverskraam over de toonbank als verse broodjes bij de bakker. In dit verband berucht is de IBM-advertentie in Datamation, 1968, waarin een stralende Susie Meyer --in kleuren!-- verklaart, dat de bekering tot PL/I het einde van al haar programmeerproblemen was. Hoe de arme Susie Meyer er een paar jaar later uitzag, vermeldt de historie helaas niet, maar het laat zich raden, want het wondermiddel heeft natuurlijk niet gewerkt. Wie de propagandaliteratuur voor Ada --de programmeertaal, die door het Amerikaanse ministerie van defensie wordt gepousseerd-- leest, moet constateren, dat de wereld in 14 jaar weinig is veranderd. (In Polen wordt Ada, terecht en onthullend, als "PL/II" aangeduid.)

Naast de potjes met programmeertalen hebben we de flesjes met nieuwe operating systems: op een regime van driemaal daags een druppel UNIX wordt U het eeuwige leven beloofd. En als dat niet werkt, dan is er nog het onfeilbare poeiertje van de "personal computer". Dat is helemaal heerlijk, want zonder dat U zich iets van al die andere patienten hoeft aan te trekken, kan dit poeiertje gemengd worden helemaal naar Uw persoonlijke behoeften.

Het laatste en gevaarlijkste luchtkasteel in deze windhandel is het "intelligent terminal" waartegen je "gewoon" kunt zeggen, wat je nodig hebt. Hier ligt de fraude er duimen dik boven op: de meeste mensen weten niet wat ze nodig hebben, en als ze het al zouden weten, kunnen ze het niet zeggen, noch "gewoon", noch "ongewoon". Desalniettemin schijnt de Japanse industrie te staan te trappelen om het te leveren, en wij zullen wel stom genoeg zijn om het te kopen. Ik denk, dat het daarmee net zo gaat als met homeopathische geneesmiddelen: je weet nooit of het helpt en, "Baat het niet, dan schaadt het niet.".

Deze laatste geruststellende overweging is overigens een ernstige misvatting. De z.g. "word processor" wordt aangeprezen met de overweging, dat het zo makkelijk is om iets in je tekst te verbeteren. Maar elke stroomlijning van het correctieproces is een staande invitatie voor de productie van dingen, die deze correctie nodighebben. Het is een volslagen illusie, dat de "word processor" de auteur helpt. Weinigen immers kunnen de verleiding weerstand bieden, maar iets te produceren, dat dan later wel gerepareerd of geperfectioneerd zal worden. Maar uit een rammelend concept is door bijvijlen nog nooit iets goeds ontstaan. Met een tekst die niet deugt, kun je maar een ding doen: naar de prullemand verwijzen. Als ik de Nederlandse overheid was, zou ik het gebruik van dit soort apparatuur niet verbieden; ik zou het zwaar belasten, net als alcohol en tabak.

De hoge kwakzalverdichtheid is mede verantwoordelijk voor de hardnekkigheid van de software crisis. Elke keer dat de patient bij een volgend lapmiddel geen baat heeft gevonden, wordt hij argwanender jegens de medische wetenschap. De onvervulde beloftes van volledige genezing maken hem doof voor wat de wetenschap verantwoord geven kan: suggesties ter verbetering van de toestand.

Laat mij evenwel niet zo in mineur eindigen. De toestand is inderdaad hopeloos als het wetenschappelijk onderwijs zich laat koejeneren door de eis van "maatschappelijke relevantie". Die leidt er namelijk toe dat de maatschappij krijgt waar hij om vraagt en dat, wanneer om lapmiddelen gevraagd wordt, er lapmiddelen worden geboden. Het "Mundus vult decipi; decipiatur ergo." is dan onverdund van toepassing. De toestand is evenwel niet hopeloos als het wetenschappelijk onderwijs niet voor zijn leidinggevende taak terugdeinst en de moed heeft de wereld te bieden, niet waar de wereld om vraagt, maar wat de wereld nodig heeft.

Over de hele wereld is de toestand aan de universiteiten tegenwoordig zo benard, dat op hen je hoop vestigen onverantwoord optimisme moet lijken. De wortels van onze universitaire traditie gaan echter zo diep, dat ik er van overtuigd ben dat die traditie een taaier leven beschoren zal zijn dan de software crisis van vandaag.

Nuenen, 4 juli 1982