Stichting NIOC en de NIOC kennisbank

Stichting NIOC (www.nioc.nl) stelt zich conform zijn statuten tot doel: het realiseren van congressen over informatica onderwijs en voorts al hetgeen met een en ander rechtstreeks of zijdelings verband houdt of daartoe bevorderlijk kan zijn, alles in de ruimste zin des woords.

De stichting NIOC neemt de archivering van de resultaten van de congressen voor zijn rekening. De website www.nioc.nl ontsluit onder "Eerdere congressen" de gearchiveerde websites van eerdere congressen. De vele afzonderlijke congresbijdragen zijn opgenomen in een kennisbank die via dezelfde website onder "NIOC kennisbank" ontsloten wordt.

Op dit moment bevat de NIOC kennisbank alle bijdragen, incl. die van het laatste congres (NIOC2023, gehouden op donderdag 30 maart 2023 jl. en georganiseerd door NHL Stenden Hogeschool). Bij elkaar bijna 1500 bijdragen!

We roepen je op, na het lezen van het document dat door jou is gedownload, de auteur(s) feedback te geven. Dit kan door je te registreren als gebruiker van de NIOC kennisbank. Na registratie krijg je bericht hoe in te loggen op de NIOC kennisbank.

Het eerstvolgende NIOC vindt plaats op donderdag 27 maart 2025 in Zwolle en wordt dan georganiseerd door Hogeschool Windesheim. Kijk op www.nioc2025.nl voor meer informatie.

Wil je op de hoogte blijven van de ontwikkeling rond Stichting NIOC en de NIOC kennisbank, schrijf je dan in op de nieuwsbrief via

www.nioc.nl/nioc-kennisbank/aanmelden nieuwsbrief

Reacties over de NIOC kennisbank en de inhoud daarvan kun je richten aan de beheerder:

R. Smedinga kennisbank@nioc.nl.

Vermeld bij reacties jouw naam en telefoonnummer voor nader contact.

Literate Programming in het onderwijs

E.W. van Ammers, M.R. Kramer
Vakgroep Informatica, Landbouwuniversiteit Wageningen

Inleiding

Het is gebruikelijk om gestructureerd programmeren te onderwijzen met behulp van stapsgewijze verfijning. Maar de werkstukken die studenten inleveren bestaan uit gecompleteerde programma's en de toegepaste verfijningen moeten daar als het ware uit worden ge-'parsed'. Het zou veel overzichtelijker zijn om omgekeerd te werken: documenteer de verfijningen en genereer daaruit de programmafiles. Knuth heeft deze benadering 'literate programming' genoemd.

Literate Programming

Literate programming heeft tot doel om aan een mens zo goed mogelijk uit te leggen wat een computer geacht wordt te doen. Daarbij dient het gedachtenpatroon van de oorspronkelijke ontwerper/programmeur als leidraad. Deze moet zijn successievelijke ontwerpbeslissingen documenteren in de vorm van discrete eenheden. In beginsel geeft elke ontwerpbeslissing aanleiding tot twee soorten tekst die conceptueel één geheel vormen: enerzijds een zekere hoeveelheid 'harde' code ten behoeve van de machine, anderzijds een (formele of informele) verantwoording ten behoeve van de menselijke lezer. Door die ontwerpstappen vast te leggen met behulp van een tekstverwerker of een formatter, kan men zeer toegankelijke documentatie verkrijgen die de gedachtengang van de programmeur goed weerspiegelt. Dergelijke documentatie is waardevol, zowel voor het beoordelen van de kwaliteit van een programma, als voor het onderhoud ervan.
De techniek van literate programming bestaat er nu eenvoudig uit dat de code-segmenten *automatisch* uit de documentatie worden gelicht en vervolgens weer in hun logische verband worden samengesmeed tot modulen. Op die manier correspondeert de code van de modulen dus gegarandeerd met de code die in de documentatie staat opgetekend. Deze eigenschap is door Van Wyk 'verisimilitude' gedoopt en zij is een krachtige stimulans om de documentatie van een programma consistent te houden met de actuele code. een 'literate program' *beschrijft* dus niet zozeer een programma, maar het is gewoon het programma in een speciale verschijningsvorm.

Ervaringen in het onderwijs

In Wageningen is ruime ervaring opgedaan met literate programming in practica en individuele projecten. Reeds bij het inleidende practicum worden verfijningsstappen geassocieerd met ontwerpbeslissingen. Die werkwijze wordt verderop in de studie uitgebreid tot literate programming.
De ervaringen van de studenten zowel als van hun begeleiders zijn overwegend positief. Studenten worden veel meer dan bij een conventionele benadering, gedwongen om rekening en verantwoording af te leggen van de door hen geproduceerde code. Deze extra mentale discipline wordt soms in eerste instantie als 'overhead' ervaren, maar dit is doorgaans van korte duur. Begeleiders kunnen aan de hand van de geproduceerde verfijningen de kwaliteit van een werkstuk aanzienlijk efficiënter beoordelen. Problematische stukken code zijn bijvoorbeeld gemakkelijk herkenbaar aan hun gekunstelde verfijningen.

Gereedschappen voor Literate Programming

Het genereren van de programmafile (module) vereist een speciaal gereedschap, een modulegenerator, dat in staat moet zijn om de codesegmenten te onderscheiden van 'gewone' teksten. We onderscheiden hier de gevallen dat de documentatie via een formatter (batch) wordt gemaakt, of dat daarvoor een tekstverwerker (interactief) wordt benut.

onderwijsinhoud informatica
onderwijsuitvoering curriculum
onderwijssector hoger onderwijs, scholing en training

Formatters
De inputfile van een formatter bevat, naast de eigenlijke te formatteren tekst, tevens commando's die de formatter aansturen. Door zo'n inputfile tevens te voorzien van commando's voor de modulegenerator, is het mogelijk de code-segmenten expliciet te markeren. Dergelijke systemen werken dus als volgt:
- prepareer de inputfiles (bijvoorbeeld met een editor)
- genereer de documentatie met behulp van de formatter uit de inputfiles
- extraheer de modulen met behulp van de generator uit de inputfiles.

Een bezwaar van deze methode is dat er eigenlijk met drie talen door elkaar heen wordt gewerkt, namelijk de programmeertaal, de commando's voor de formatter en de aansturing van de modulegenerator. Voorbeelden van batch-georiënteerde systemen zijn WEB en VAMP.

Tekstverwerkers
Een modernere benadering is om de documentatie te verzorgen met behulp van een tekstverwerker. De inputfiles voor de modulegenerator worden dan verkregen via ASCII-export (dit gaat prima met nagenoeg elke tekstverwerker). De generator moet vervolgens de code-segmenten in de geëxporteerde files kunnen herkennen en doet dat op basis van een goed omschreven programmeerstijl. De werkwijze is dan:
- prepareer de documentatie met de tekstverwerker
- exporteer de ASCII-files
- extraheer de modulen met behulp van de generator uit de ASCII-files.

Een modulegenerator van dit type wordt aan de Landbouwuniversiteit ontwikkeld. De eerste release is gepland in juni 1992.

Slot
Het merendeel der studenten en hun begeleiders ervaart de techniek van literate programming als een belangrijke vooruitgang. De interactieve benadering elimineert de belangrijkste bezwaren die bestaan bij het werken met een formatter.
Een secundair voordeel van literate programming is dat beschrijvend proza en programmacode met één en dezelfde tekstverwerker (editor) worden vervaardigd. Een aparte editor voor de programmafiles is dus niet nodig.