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.

Programmeeronderwijs aan de Technische Universiteit Delft

P.G. Kluit, W.J. Toetenel, F. Ververs
Faculteit Technische Wiskunde en Informatica, Technische Universiteit Delft

Inleiding

Het programmeeronderwijs binnen de opleiding tot Informatica-ingenieur aan de TU-Delft is in 1990 ingrijpend gewijzigd. Daarbij werd gestreefd naar een opzet die enerzijds motiverend en stimulerend zou zijn, anderzijds programmatuur-ontwikkeling vanaf het begin vanuit een abstract standpunt zou benaderen.

Probleemstelling

Tot 1989 werd het onderwerp programmaconstructie voor de informatica-studenten aan de TU-Delft op traditionele wijze behandeld in een drietal opeenvolgende colleges:

- Inleiding Informatica (Pascal inclusief procedures, recursie en pointers)
- Datastructuren en Algoritmen (min of meer CS2)
- Software Engineering.

Na afloop van deze colleges zouden begrippen als procedurele abstractie, data-abstractie, decompositie en information-hiding het geestelijk eigendom van de studenten moeten zijn. Dat bleek in de praktijk echter nauwelijks het geval. Op grond hiervan werd besloten (in het kader van een herziening van het gehele curriculum) het traject programmatuurontwikkeling vanuit deze doelstelling te herzien.

Analyse

Tijdens de analyse van het probleem kwamen een drietal aspecten naar voren:

- Zolang studenten niet over een taal beschikken om procedures en data abstract te beschrijven kunnen ze er ook moeilijk op abstract niveau over denken en redeneren.
- Wanneer in een later stadium een taal wordt geïntroduceerd om abstracties uit te drukken blijkt het voor de studenten erg moeilijk de abstracte begrippen los te maken van de reeds bekende concrete implementaties.
- Daarnaast hebben we de overtuiging dat vaardigheid in imperatief programmeren nog steeds in de ransel van een Delftse informatica ingenieur thuis hoort.

Een nieuwe aanpak

Na een tweetal min of meer experimentele jaren (1990-1992) begint de nieuwe cursus een vaste vorm te krijgen. De beide vakken Inleiding Programmeren en Datastructuren en Algoritmen zijn vervangen door de vakken Programmeren 1 en 2. In Programmeren 1 (twee kwartalen) wordt de specificatietaal SpeL geïntroduceerd, een subset van VDM-SL. (VDM-SL is de formele model-georiënteerde specificatietaal behorende bij VDM. Een ISO-standaard voor VDM-SL is in ontwikkeling). Om de praktische toepasbaarheid van de verschillende SpeL-constructies duidelijk te maken wordt vanaf het begin aandacht besteed aan modelleren. Voor eenvoudige SpeL-concepten (enkelvoudige typen, records, (recursieve) functies) worden vervolgens imperatieve implementaties geïntroduceerd, gebaseerd op Modula-2. Daarnaast wordt aandacht geschonken aan typisch imperatieve aspecten van Modula-2: assignment, herhaling en procedures. Het implementeren wordt geoefend in een practicum. In het derde en vierde kwartaal (Programmeren 2) wordt een en ander uitgebouwd: implementatie van samengestelde typen, algoritmiek.

Relatie met andere benaderingen

SpeL heeft een functionele taal als subset, zodat een vergelijking met programmeer-methoden gebaseerd op functionele talen (Scheme, ML, Miranda) voor de hand ligt. Als belangrijke verschilpunten noemen wij: SpeL is model-georiënteerd, kent noties als toestand en operaties op de toestand, en laat impliciete specificaties toe.

onderwijsinhoud informatica
onderwijsuitvoering curriculum, didactiek
onderwijssector hoger onderwijs

Slot

Ons inleidend programmeeronderwijs neemt de formele specificatietaal SpeL als uitgangspunt. Hierdoor vindt programmatuurontwikkeling vanaf het begin op systematische wijze plaats. Door het gebruik van SpeL zijn procedurele decompositie en abstractie zowel als data-decompositie en -abstractie vanzelfsprekende middelen bij programmatuurontwerp geworden. Dit geeft de student een gezonde basis voor Software Engineering. De gevolgde benadering doet echter vanaf de eerste dag een groot beroep op het abstractievermogen van de student.

onderwijsinhoud informatica
onderwijsuitvoering curriculum, didactiek
onderwijssector hoger onderwijs