

Een framework voor hermeneutische objectgeoriënteerde software-ontwikkeling: ontwerp en productie in één hand

Mehmet Aksit en Hans van Oosten

Dit artikel introduceert een innovatieve aanpak in systeemontwikkeling, die op dit moment wordt onderzocht en ontwikkeld aan de Universiteit Twente. Het is een objectgeoriënteerde aanpak die wat betreft structurering van de activiteiten zelf eveneens objectgeoriënteerd van aard is. Het is een aanpak die het mogelijk maakt verschillende technieken naar keuze te incorporeren in één framework. De aanpak komt tegemoet aan aantal bezwaren die 10 jaar geleden zijn geconstateerd voor de toenmalige ontwikkelmethoden (Rijsenbrij, 1993) en die ook nu nog gelden voor conventionele objectgeoriënteerde methoden. Hierbij kan b.v. gedacht worden aan het ontbreken van meta-informatie die de ontwerp-informatie kan correleren met het gerealiseerde product. Dit artikel beoogt de essentiële ideeën van deze aanpak toe te lichten, en bovendien aannemelijk te maken dat ontwerp- en operationele informatie van een softwaresysteem onlosmakelijk verbonden kan zijn met dat systeem, net zoals dat met andere industriële systemen zoals auto's het geval is.

1. Inleiding

Gedurende de laatste jaren is in de wereld van objectoriëntatie een duidelijke verschuiving waarneembaar van programmeren naar modelleren en ontwerpen. Zoals in de 70er jaren een grote verscheidenheid aan gestructureerde ontwikkelmethoden ontstond, zo is er nu een groot aantal objectgeoriënteerde (OO) methoden verschenen, waarvan een aantal voorbeelden zijn 'Object-Oriented Analysis and Design' (Coad & Yourdon, 1991a, 1991b), 'Object-Oriented Design' (Booch, 1990), 'Object Modeling Techniques' (Rumbaugh et al., 1991) en 'Responsibility-Driven Design' (Wirfs-Brock, 1991). In het algemeen maken deze methoden gebruik van het conventionele objectmodel en hanteren ze concepten zoals *klassen*, *objecten*, *overerving*, *berichtenstromen*, *toestandsdiagrammen*, *attributen* en *operaties*¹. Bovendien introduceren ze notatiewijzen, werkwijzen en tips, en een proces volgens welk de stappen doorlopen kunnen worden.

OO methoden beogen een consistente toepassing mogelijk te maken van de OO benadering in de totale levenscyclus. Inmiddels zijn met deze methoden de eerste ervaringen opgedaan waarvan verslag is gedaan op wetenschappelijke conferenties (OOPSLA, 1991, 1992). De vraag die met name gesteld wordt is of de OO aanpak in het algemeen en die volgens de OO methoden in het bijzonder de (hooggespannen) verwachtingen waarmaakt. In Nederland is o.a. aan de Universiteit Twente ervaring opgedaan met de toepassing van OO methoden bij een groot aantal pilotapplicaties in het TRESE project (Aksit & Bergmans, 1992). Het blijkt dat, zoals ook door anderen is ervaren, OO methoden een wezenlijke verbetering opleveren in productiviteit. In schril contrast hiermee staan echter andere bevindingen van het TRESE project. Men ontmoet onverwachte hindernissen als men volgens deze methoden software gaat ontwikkelen. Deels zijn deze problemen terug te voeren op de manier waarop OO modellen worden toegepast om software-systemen te bouwen, deels hebben ze te maken met het beperkte expressievermogen van de gebruikte OO modellen. De eerste categorie problemen noemen wij methodologische problemen, en de tweede modelleringsproblemen. In dit artikel gaan we alleen in op de methodologische problemen. De modelleringsproblemen zijn o.a. behandeld in de publicaties (Aksit et al., 1991, 1992, 1994) waarbij gebruik wordt gemaakt van het compositiefilter OO model. In het kader van deze artikelenreeks over systeemontwikkeling zullen ze in een volgend artikel apart onder de loep worden genomen.

Typische voorbeelden van methodologische problemen zijn de moeilijke uitbreidbaarheid van bestaande methoden, de semantische scheiding tussen ontwerp-informatie en de software zelf en het ontbreken van

¹ Ook methoden genoemd.

mogelijkheden om het ontwikkelproces te programmeren, problemen die overigens ook al kleefden aan gestructureerde methoden.

Ter overwinning van de methodologische problemen wordt in dit artikel een nieuw methodologisch raamwerk geïntroduceerd dat gebruikt wordt om zogenaamde *hermeneutische software* te bouwen. Een *hermeneutische methode* richt zich op de ontwikkeling van *hermeneutische softwaresystemen* welke de beslissingen van de ontwerpers, zoals die gedurende analyse-, ontwerp- en implementatiefasen worden genomen, integreren met het software-eindproduct. Het doel van dit artikel is om de basisconcepten aan te reiken die ten grondslag liggen aan hermeneutische software-ontwikkeling.

Het artikel is als volgt georganiseerd. Hoofdstuk 2 definieert een referentiekader voor OO software-ontwikkeling. Aan de hand van dit kader worden eisen beschreven voor OO methoden. In hoofdstuk 3 wordt enige achtergrondinformatie verschaft over de hermeneutische filosofie die ten grondslag ligt aan de hier beschreven manier van software-ontwikkeling, en over OO methoden en software-ontwikkelomgevingen. Hoofdstuk 4 behandelt de hermeneutische aanpak voor OO software-ontwikkeling, waarbij *leerdomeinen* worden geïntroduceerd als clusters van ontwikkelactiviteiten. Hoofdstuk 5 belicht vervolgens hoe leerdomeinen geïntegreerd worden met de applicatie software. In hoofdstuk 6 wordt de voorgestelde aanpak geëvalueerd in het licht van de eerder genoemde eisen voor OO methoden. Hoofdstuk 7 tenslotte bevat de conclusies.