07-07-2020

CWI introduceert nieuwe ontwerpmethodologie voor betrouwbare software

Het ontwikkelingsproces van software is in veel bedrijfs- en onderzoeksdomeinen uiterst belangrijk geworden. Om dit proces snel, efficiënt, betrouwbaar en geautomatiseerd te maken, is software stapsgewijs gebaseerd op zowel de beste vaardigheden van de ontwikkelaar als de feedback van de eindgebruiker. Onderzoeker Vlad-Nicolae Şerbănescu (CWI en LIACS) onderzocht hoe dit kan worden verbeterd door modellering en programmeertalen in dit proces te verenigen.

Op 10 juni heeft Şerbănescu zijn proefschrift 'Software Development by Abstract Behavioral Specification' verdedigd aan de Universiteit Leiden. Zijn onderzoek richt zich op het overbruggen van de kloof tussen modellering en programmeren om formele methoden en twee van de meest bekende en gebruikte talen voor softwareontwikkeling, de Java- en Scala-talen, te integreren. De resultaten zijn interessant voor software-ingenieurs, architecten en onderzoekers. Hij legt uit: "Zowel programmeertalen als modelleertalen hebben als doel om productontwikkeling te vergemakkelijken door het ontwerpen van correcte en betrouwbare applicaties. Er bestaat echter nog steeds een kloof tussen de twee domeinen, waarbij het proces van softwareontwikkeling vaak via twee afzonderlijke paden gaat met betrekking tot modellering en implementatie. Dit leidt mogelijk tot fouten en verdubbelt de ontwikkelingsinspanning. Mijn onderzoek laat toe dat softwareontwikkeling één continue stroom volgt van het begin tot het einde van de ontwikkelingscyclus".

Het werk in dit proefschrift is uitgevoerd als onderdeel van de Formal Methods Group bij Centrum Wiskunde & Informatica (CWI) in Amsterdam en Leiden Institute of Advanced Computer Science (LIACS) bij de Universiteit Leiden, onder leiding van prof. Frank de Boer en Dr Mohammad Mahdi Jaghoori. Dit onderzoek werd ondersteund door de Europese projecten FP7-610582 ENVISAGE (Engineering Virtualized Services) en FP7-612985 UPSCALE (From Inherent Concurrency to Massive Parallelism through Type-based Optimizations).