Space Flight Software is embedded software

Gerard Holzmann van NASA JPL: modelgestuurde verificatie is nu praktisch toepasbaar

Het 'Laboratory for Reliable Software' van het 'Jet Propulsion Laboratory' heeft als doel om op de lange termijn de betrouwbaarheid van JPL's missie-kritische software te verbeteren. LaRS wordt geleid door de Nederlander Gerard Holzmann, die eerder bij Bell labs werkte aan de Spin model checker. De groep doet onderzoek, formuleert coderingsrichtlijnen en produceert ook zelf software, onder meer voor het 'Mars Science Laboratory' dat eind 2009 wordt gelanceerd.

HANS VAN THIEL

De periodieke 'CWI Lectures in Mathematics and Computer Science' had dit jaar concurrency theorie als onderwerp, en een van de uitgenodigde sprekers was Gerard Holzmann. Na zijn promotie aan de TU Delft in 1979 werkte Holzmann, met een korte onderbreking, tot 2003 bij het beroemde Bell labs van AT&T en Lucent. In 2002 ontving hij de ACM System Software Award voor zijn werk aan de Spin verificatiesoftware.

Vanaf 2003 leidt Gerard Holzmann het 'Laboratory for Reliable Software' van het NASA 'Jet Propulsion Laboratory'. JPL is verantwoordelijk voor de onbemande ruimtemissies van de NASA en LaRS heeft tot doel om op de langere termijn de betrouwbaarheid van missie-kritieke software te verbeteren.

Ten tijde van ons geprek, in de bibliotheek van het Centrum voor Wiskunde en Informatica in Amsterdam, was de Phoenix capsule nog niet met succes op Mars geland. Afgezien van een globale softwareanalyse was de groep van Holzmann hier overigens niet direct bij betrokken, maar ontwikkelt wel zelf software voor het 'Mars Science Laboratory' dat eind 2009 wordt gelanceerd. Dat is vrij bijzonder, vertelt hij, want meestal duurt het een jaar of tien voordat een nieuw team ingeschakeld wordt bij de daadwerkelijke productie van software. "Als jullie het zo goed weten, maak dan zelf ook maar eens iets, zullen ze gedacht hebben," zegt Holzmann glimlachend.

LaRS houdt zich niet alleen bezig met onderzoek en opleiding, maar ontwerpt ook richtlijnen voor het schrijven van code. In de praktijk van JPL is dat uitsluitend C-code, met als uitzondering enige al langer bestaande Fortran-implementaties. Een samenvatting van deze voorschriften is onder de titel "The Power of Ten – Rules for Developing Safety Critical Code", verschenen in IEEE Computer (Juni 2006).

"Sommige missies, zoals de Cassinni-missie naar Saturnus, zijn tientallen jaren in voorbereiding voordat ze gelanceerd worden."

Betrouwbaarheid

Holzmann over de rol van zijn groep binnen JPL: "LaRS is in 2003 begonnen en we zijn nu met vier mensen. Elk jaar komt er één persoon bij. JPL heeft een lange geschiedenis in ruimteonderzoek, met name in orbiters en landers. In de jaren '60 ging alles nog mechanisch; de eerste sonde naar Mars had zo'n 50 regels assembly. Maar de hoeveelheid controlsoftware verdubbelt bij elke missie. De Phoenix heeft zo'n 400.000 regels C-code, inclusief commentaar, en het MSL (Mars Space laboratory) volgend jaar september krijgt er ongeveer 1,2 miljoen.

"Elke fout in de software kan de missie doen mislukken. Het gaat om bedragen van 500 miljoen tot 1 miljard dollar en ons doel is de kans op fouten te verkleinen door de software beter te ontwerpen, te testen en te verifiëren. We doen daarvoor gericht onderzoek naar wat er fout gaat, welk soort problemen er in de praktijk optreden.

"Space flight software is eigenlijk een schoolvoorbeeld van embedded software. Je hebt te maken met een gecontroleerde omgeving, een beperkt geheugen, een beperkte stack, noem maar op. De typisch gebruikte CPU, bijvoorbeeld, is de tegen straling geharde RAD750, een PowerPC van 200 MHz. Hoe verder de transistoren uit elkaar liggen, hoe minder gevoelig voor straling, en je hebt ook te maken met eisen aan het energieverbruik en -dissipatie. De CPU-snelheid is dus beperkt.

"De ontwerptijden zijn ook heel lang. Sommige missies, zoals de Cassinni-missie naar Saturnus, zijn tientallen jaren in voorbereiding voordat ze worden gelanceerd. LaRS is betrokken bij 10 van de 150 MSL softwaremodulen, maar we hebben invloed op hoe alle software binnen JPL wordt geschreven."

Holzmann haalt ter illustratie een memory stick uit zijn borstzak. "Het permanente geheugen in MSL is net zoiets als dit, al kost dat dan 200.000 in plaats van 20 dollar. Wij hebben het filesysteem geschreven en dat is volgens ons uitzonderlijk betrouwbaar. Het telt zo'n 6000 regels en wat er ook gebeurt, en wanneer dan ook, alle data

blijven bewaard. Aan die software module hebben we 2 jaar gewerkt."

Verificatie

De Spin Model Checker, waarvoor Holzmann in 2002 de ACM System Software Award ontving, is en wordt verder ontwikkeld. De tool is vrij beschikbaar, er zijn diverse boeken over verschenen, en Spin werkt nu niet alleen op C-broncode (via het ook vrijgegeven FeaVer) maar op gecompileerde C-code. Op een CWI-bijeenkomst hield Holzmann onlangs een presentatie over de parallelle implementatie van Spin op multicore systemen.

Holzmann over de wet van Moore: "Die zegt dat de transistordichtheid elke 18 maanden verdubbelt. Wat RAM betreft gaat dat nog steeds door, maar voor de CPU is dat in 2002, 2003 opgehouden. Vijf jaar geleden was de hoogste snelheid 2,5 GHz en nu is dat 3,5 GHz. Alle chipfabrikanten zijn bewust gestopt met het verhogen van de CPU-snelheid en in plaats daarvan overgestapt op multi-core. De snelste systemen hebben nu 8 kernen met 128 GB en over 2 jaar worden dat dus waarschijnlijk 16 kernen met 256 GB."

"Dat is een fundamentele verandering. Zes jaar geleden dachten we dat een tool als Spin door de wet van Moore vanzelf snel genoeg zou worden om standaard toe te kunnen passen bij elke compilatie. Dat is nog steeds het doel, maar dan moeten we de algoritmen parallel kunnen implementeren, toegespitst op meervoudige processoren. Dat lukt ook en we hebben nu bijvoorbeeld Swarm, een front-end voor Spin.

"Bij Bell Labs zat een aantal heel getalenteerde mensen en was er volledige vrijheid van onderzoek, maar je had niet veel te maken met het bedrijf zelf. Bij JPL werken we samen met de teams die de missies doen en is er veel meer invloed. Tools voor formele verificatie kunnen software veel betrouwbaarder maken en het is de bedoeling dat 'model checking' binnen 5 jaar bij elke ruimtemissie de standaard zal zijn. Ik heb mijn twijfels of dat lukt, maar bij LaRS zelf is natuurlijk iedereen hiermee vertrouwd. We verzorgen ook elk jaar colleges bij Caltech over model checking technieken, en mensen van JPL volgen die ook. (Caltech is het California Institute of Technology, de privé universiteit die met JPL is verbonden, red.)

Codering

Het streven is om de achterliggende techniek zoveel mogelijk te abstraheren in de tools. Van Spin zijn inmiddels, alle verschillende versies inbegrepen, 1 miljoen downloads geregistreerd, vertelt Holzmann. Maar verificatie en testen is niet alles; de programmatuur wordt ook betrouwbaarder met een bewuste programmeerstijl. Compileren met alle 'warnings' aan, bijvoorbeeld, en het beperken van de lengte van elke individuele functie tot 1 uitgeprinte pagina, vermijden van goto en recursie, declaratie van variabelen in het allerbinnenste blok, het vermijden van dynamische geheugentoewijzing en beperking van het gebruik van de preprocessor.

LaRS heeft voor JPL een verzameling heuristische regels opgesteld voor het schrijven van broncode, waarmee de kans op software fouten wordt verkleind (zie het IEEE artikel voor een samenvatting daarvan).

Twee regels die minder voor de hand liggen dan de al genoemde: de return waarde van elke functie moet worden gechecked door elke aanroepende functie, en het gebruik van pointers moet worden beperkt. Pointer dereferenties mogen niet voorkomen in macros of typedef declaraties, en pointers naar functies zijn helemaal verboden.

Holzmann op de vraag of het gebruik van pointers niet juist het wezen is van C: "Sommigen zeggen dat inderdaad, en je kunt een heleboel doen met pointers. Dat is precies het probleem en ze kunnen gemakkelijk verkeerd worden gebruikt, zelfs door ervaren programmeurs. Pointers kunnen ook problematisch zijn voor statische analyse tools.

"Een heel belangrijk hulpmiddel bij het schrijven van betrouwbare code zijn asserties. Volgens onze coderingsrichtlijnen moet elke functie er minstens twee hebben. Dat moeten Boolean tests zijn, vrij van neveneffecten en met een specifieke actie. Na testen en analyse door een tool kunnen zij worden uitgezet, als dat nodig is voor de performance".

"Het nut van deze richtlijnen wordt wel ingezien binnen JPL, maar het betrouwbaarder maken van de code vergt wel meer werk en dus tijd. Doordat ikzelf ook bij de productie betrokken ben draag ik dan eigenlijk twee petten. 'We kunnen het wel zo doen,' krijg je te horen, 'maar dan zijn we niet op tijd klaar voor de lancering!' Voor het Mars Science Laboratory duurt dat nog een jaar, maar de spanning begint bij alle betrokkenen nu al toe te nemen. Maar dat schijnt normaal te zijn, bij elke missie, zo is me verteld."■

Gerard Holzmann: "In de jaren '60 ging alles nog mechanisch; de eerste sonde naar Mars had zo'n 50 regels assembly. Maar de hoeveelheid controlsoftware verdubbelt bij elke missie. De Phoenix heeft zo'n 400.000 regels C-code, inclusief commentaar, en het MSL (Mars Space laboratory) volgend jaar september krijgt er ongeveer 1,2 miljoen." (foto's: CWI, Centrum voor Wiskunde en Informatica)