Interview Allan Kennedy, Kennedy Carter

Het model is

'Je kunt een executeerbare UML virtuele machine schrijven, net zoals je een Java virtuele machine kunt schrijven,' aldus CEO Allan Kennedy van het Engelse bedrijf Kennedy Carter. 'Vervolgens bouw, test, gebruik en onderhoud je het model, niet de code.' Kennedy Carter houdt zich al meer dan tien jaar bezig met software modellering en automatische code generatie.

HANS VAN THIEL

De 'Action Specification Language' van het bedrijf sluit aan bij de UML 1.5-specificatie[1] en KC is ook actief in de standaardisatie rond OMG[2] en MDA[3]. Code voor de F16 Modular Mission Computer is automatisch gegenereerd met tools van Kennedy Carter. PT Embedded Systems sprak met Allan Kennedy op de OMG Information Day die door Europees OMG vertegenwoordiger 'LogOn' op 18 Februari 2003 in Schiphol-Rijk werd georganiseerd.

Meneer Kennedy, kunt u om te beginnen, iets vertellen over de geschiedenis van Kennedy Carter?

'Al voor 1990 werkten Collin Carter en ik aan real-time modelleer methoden en we hebben al in 1988 een MDA-achtig proces gedefinieerd voor de 'Eurofighter Nose Radar', een stelsel regels om een functionele compositie te transformeren tot een OO ontwerp met een implementatie in Ada. Collin was Chief Software Engineer van Plessey Avionics tot 1988, toen we ons eigen bedrijf oprichtten. In 1994 hebben we onze 'Action Specification Language', waarmee modellen executeerbaar werden, vrijgegeven voor algemeen gebruik.

ASL bevat primitieven voor de creatie en verwijdering van instanties en associaties, navigatie van associaties, en het lezen en schrijven van attributen. De taal heeft sequentiele logica met vertakkingen en lussen en de datastructuren zijn verzamelingen, geordende verzamelingen en verzamelingen met mogelijk gelijke elementen[4].

De toestandsmachine is de fundamentele constructie voor het modelleren van software dynamica en ASL wordt gebruikt om de details te specificeren van elke toestandsactie. Daarmee is ASL een soort eenvoudige platform onafhankelijke programmeertaal die vertaald kan worden naar elke 3GL. Sinds de introductie hebben we met anderen samengewerkt aan standaardisatie en ASL is thans feitelijk versie 1 van de UML Action Semantics die is opgenomen in de UML 1.5 specificatie.'

'Het was voor Lockheed Martin een hele stap om automatische code generatie toe te passen in een luchtvaartsysteem'

Wat is het verband tussen UML en ASL? En wat is xUML?

'Eigenlijk was, vanuit onze optiek, de standaardisatie van grafisch modeleren in UML in 1997 een grote stap terug. Het introduceerde een enorme variëteit aan grafische notatie met een slecht gedefinieerde semantiek. Ons eigen werk was gebaseerd op de objectgeoriënteerde analysemethode van Shlaer en Mellor, toentertijd de meest precieze methode die er was. Maar een deelverzameling van UML, die wij xUML noemen, is goed genoeg gedefinieerd, met een operationele semantiek, om executeerbaar te zijn. Dus: xUML is UML min de vage gedeelten en plus actie semantiek. Met ASL kunnen we alle details nauwkeurig genoeg specificeren om een model executeerbaar te maken, maar zonder de platformonafhankelijkheid in gevaar te brengen.'

De 'Object Constraint Language' is ook een formele taal die geïntegreerd is binnen UML. Kun je OCL-beweringen, zoals pre- en postcondities, gebruiken samen met ASL?

'Nee, op dit ogenblik wordt ASL niet gebruikt als een 'constraint' taal maar het heeft een duidelijke overlap met OCL als een query taal. Ik ben er van overtuigd dat actietalen en OCL op enig moment in de toekomst zullen convergeren naar een gemeenschappelijke semantiek en syntaxis.'

Hebt U ook gekeken naar SDL[5] en/of MSC[6] als alternatieven voor UML? Die ITU[7] standaarden zouden preciezer zijn dan UML.

'Ik zou niet willen beweren dat die preciezer zijn en ze worden toch voornamelijk in de telecom industrie gebruikt. Jim Rumbaugh heeft wel eens gezegd: 'UML is een grote verzameling ingrediënten, en een goede kok weet welke ingrediënten hij moet gebruiken.'Wij vinden dat we alles wat we willen kunnen modeleren met xUML.'

De Model Driven Architecture die de OMG heeft geïntroduceerd en nu sterk aan het bevorderen is, past heel goed bij de Kennedy Carter benadering van software ontwikkeling.

'MDA is gebaseerd op het onderscheid tussen een 'Platform Independent Model' (PIM) en mogelijk meerdere, verschillende, 'Platform Specific Models'

het programma

(PSM) die van dat PIM kunnen worden afgeleid. Dat is een fundamentele scheiding van verantwoordelijkheden die teruggaat naar de vroegste dagen van gestructureerde methoden en die we altijd hebben aangemoedigd. Nu kan een PIM heel lang zijn waarde behouden, juist omdat het niet beïnvloed wordt door de invoering van snellere, goedkopere platformen in de loop der tijd. Als de PIM lang mee moet gaan is het essentieel om de correctheid te verifieren. We zouden het dus willen testen. Om het te testen, moeten we het executeren tegenover een test suite. En daarom zijn xUML en actie taal zo belangrijk voor MDA. Zonder die kunnen we er niet zeker van zijn dat we een PSM genereren vanuit een correct bronmodel.

Maar onze benadering gaat veel verder. Wij zien in dat met MDA platformonafhankelijke modellen eigenlijk componenten worden die te integreren zijn tot platformonafhankelijke systemen. Wij leren onze gebruikers om die model componenten er uit te halen die het meest herbruikbaar zijn. Juist die componenten, die wij domein modellen noemen, encapsuleren een enkel samenhangend onderwerp en publiceren een groep leverbare diensten waar andere domeinmodellen verbinding mee kunnen maken. Je zou dat, als je wilt, 'sockets' kunnen noemen.

Met xUML kun je alles modelleren omdat het maken van een precies UML model veel weg heeft van 'knowledge engineering'. Zo kun je zowel applicatie- als ontwerpkwesties modelleren in dezelfde taal. Zoals Steve Mellor altijd zei: 'Ontwerp is gewoon een ander onderwerp om te analyseren.' In de context van MDA: als je je kennis van goed ontwerp formaliseert in een precies UML model, ben je een PIM naar PSM vertaler aan het specificeren, met andere woorden, een model compiler.'

U noemt dit algemene principe ook wel 'software onafhankelijk ontwerp'?

'Modelleren is het formaliseren van kennis van deskundigen, en dat hoort onafhankelijk te zijn van enig software platform, net zoals het onafhankelijk hoort te zijn van enig hardware platform. Dat geldt voor alle aspecten van het software engineering proces: analyse, ontwerp, bouwen, testen, gebruik en onderhoud. En, net zoals je een Java programma draait op een Java virtuele machine, zo draai je een executeerbaar UML-model op een UML virtuele machine. Ontwikkelaars testen en debuggen het model en niet de code die daar uit voortkomt. Het levert ook een enorme kostenbesparing op dat je de code niet meer hoeft te onderhouden. Je onderhoudt alleen nog maar het model.'

Werkt U ook aan automatische correctheidsverificatie, zoals bijvoorbeeld Lucent in Bell Labs doet?

'Nee, dat is niet iets waar we ons mee bezighouden. Nog niet, in elk geval.'

> 'Modelleren is het formaliseren van kennis van deskundigen, en dat hoort onafhankelijk te zijn van enig software- en hardware platform'

Voor de F16 Modular Mission Computer is automatische codegeneratie toegepast.

'Dit was een project, dat begon in 1994, van Lockheed Martin om hardware en software in de F16 straaljager te vernieuwen. Een enkele computer moest drie andere computers vervangen, met als resultaat een dertigvoudige toename in doorvoersnelheid en een zestig procent vermindering in gewicht en in vermogensverbruik. Het was een zeer aanzienlijke verandering in het onderliggende platform en benadrukt hoe belangrijk platformonafhankelijk modelleren kan zijn bij de migratie naar krachtiger en doelmatiger platformen. Ongeveer twee jaar na het begin besloot Lockheed Martin om met behulp van onze iCCG[8] een automatische codegenerator te bouwen voor Ada '83. Mij is verteld dat nu tenminste vijftig procent van alle nieuwe mogelijkheden van het systeem die aan de MMC worden toegevoegd voor honderd procent automatisch worden gegenereerd uit executeerbaar UML. De kwaliteit en de productiviteit zijn sterk verbeterd, omdat de fouten introductie die gepaard gaat met handmatig coderen nu volledig is geëlimineerd, zo meldt het project. Het was uiteraard een hele beslissing voor Lockheed Martin om een luchtvaartsysteem, met een noodzakelijke veiligheidsgarantie, afhankelijk te maken van automatische generatie van code. Dit demonstreert de haalbaarheid van MDA en honderd procent codegeneratie voor kritische systemen.'

Hergebruik van legacy software is ook een belangrijk argument voor MDA.

'Het Lockheed Martin team genereert Ada uit xUML. Die code werkt samen met veel van de modelcomponenten die geïntegreerd zijn in de 'mission software' en die met de hand in Ada zijn geschreven. Met MDA tools gaat het integreren van legacy of andere componenten die met tools van derden zijn gebouwd heel direct.'

En wat is iUML?

'Dat is onze modelleer tool voor xUML. Je kunt er niet alleen platformonafhankelijke modellen mee ontwerpen, maar die ook meteen testen en debuggen met de geïntegreerde xUML simulator. De tool is beschikbaar voor Windows en Solaris, en voor Windows is een beperkte versie, iUMLite, gratis te downloaden van onze website. We hebben ook een tutorial geschreven met een benzinestation toepassing als voorbeeld.'

Ten slotte, gewoon uit nieuwsgierigheid, waar staat het voorvoegsel 'i' in de tool namen voor?

'Waar staat de 'i' in 'iMac' voor?' ■

Meer informatie

1. Website: www.kc.com
2. Enkele beschikbare white papers:
   - UML ASL Reference Guide
   - Configurable Code Generation in MDA using iCCG
   - Supporting Model Driven Architecture with eXecutable UML
3. Meer over Model Driven Architecture: www.omg.org/mda

Voetnoten

1) Unified Modeling Language
2) Object Management Group
3) Model Driven Architecture
4) Eng. 'bag'
5) Specification and Description Language
6) Message Sequence Charts
7) International Telecommunications Union
8) Configurable Code Generator