Bericht uit 

Om maar meteen met de deur in huis te vallen: ik heb last van het structuurvirus. Nu hoeft u niet meteen bloemen te sturen of bij me uit de buurt te blijven. Het virus is niet gevaarlijk, en ook niet besmettelijk, maar het is wel lastig. Ik zal u beschrijven wat eraan de hand is.

Kloppen en hakken of structuur?

Als ik een programma aan het ontwikkelen ben dan word ik in ernstige mate gehinderd door mijn diepliggende verlangen om de programmeertaal waar ik op dat moment mee werk 'recht te doen'. Dit heeft tot gevolg dat ik zal trachten om de taal esthetisch zo verantwoord mogelijk in te zetten teneinde het gestelde doel (een werkend programma) te bereiken. Bij programmeertalen waar geen dieper gelegen esthetisch ideaal achter ligt is dit vrij eenvoudig. Ik klop en hak er vrolijk op los, niet gehinderd door wroeging dat ik de taal onrecht aandoe. C en Perl zijn goede voorbeelden van talen waarbij ik mijn emoties de vrije loop kan laten gaan en zonder schuldgevoel de ene cryptische constructie na de andere uit mijn vingers kan laten vloeien. Echter, dit verandert als ik in Java aan het programmeren sla. Java kent dusdanige faciliteiten voor het implementeren van een mooie softwarearchitectuur en het goed structureren van een algoritme dat ik mij geroepen voel om het onderhavige programma volledig volgens de regels der Javakunst te bouwen. Dus niet zomaar wat bewerkingen loslaten op wat data die ik uit een file heb gelezen, maar eerst een mooie object-georiënteerde architectuur in de steigers zetten en dan met behulp van alle faciliteiten die in de Javataal en de Java-classlibraries zitten die architectuur implementeren. Voor ik het weet ben ik aan het mijmeren over de metadata-architectuur en wordt er een hele reeks 'design patterns' uit de kast gehaald. De invoerdata uit een file lezen kan natuurlijk niet, er dient eerst een abstract DataProducer-interface te worden geïmplementeerd, en dan een FileDataProducer-klasse om die data dan via die interface op te hoesten. Die klasse moet dan een kommagescheiden bestand inlezen. Maar dat is natuurlijk veel te pragmatisch, dus eerst maar eens nadenken over een metaclass die gestructureerde tekstbestanden kan inlezen, waarbij het concept 'kommagescheiden' via abstracte syntaxregels extern aan die klasse bekend kan worden gemaakt. Zomaar wat output naar de gebruiker schrijven kan natuurlijk ook niet, dus moet de user interface ook nog even geabstraheerd worden, en dat liefst op een manier die geen enkel type interface (tekst, grafisch, braille) uitsluit. Vervolgens moeten de klassen ook nog 'serializable' zijn en moeten de meeste klassen ook als Java Bean bruikbaar zijn. Rechtstreekse databasetoegang via JDBC is natuurlijk veel te eenvoudig. Daar moet op zijn minste een generieke 'object-relational'-laag tussen zitten die volledig spontaan objecten en hun relaties in databases kan opslaan en weer uit die databases kan instantiëren. Rechtstreekse communicatie tussen objecten is natuurlijk uit den boze, en zo wordt er alras een informatiebus in de JVM aangelegd waar objecten zich aan kunnen koppelen en waar berichten over heen en weer flitsen. U begrijpt, voor ik aan het oplossen van het originele probleem zou toekomen staat er een softwarehuis waarvoor een besturingssysteem zich niet zou hoeven te schamen. Echter, de tijd die het me zou kosten om het geheel op deze manier aan te pakken is vaak exorbitant ten opzichte van de eigenlijke probleemstelling. Dus moet het sneller, en pragmatischer, en dan krijg ik weer wroeging. En dan pak ik maar weer Perl waar ik zonder gewetensbezwaren '$n+=$_ while <STDIN>;' kan schrijven.

Jos Visser

is beroepstwijfelaar bij Open Solution Providers en kan worden bereikt via jos@osp.nl