Bericht uit Java

Over kleine en grote elegante oplossingen

Java is geboren vanuit een speurtocht naar elegante oplossingen voor platform onafhankelijkheid, geheugenbeheer, multiple inheritance, of foutafhandeling. Hoe die oplossingen in Java ingebed zijn, mag elegant heten. Elegantie is natuurlijk relatief en er zijn altijd mensen die een andere oplossing beter, mooier en misschien zelfs eleganter vinden. De meeste aspecten van Java zijn gewoon overgenomen vanuit andere talen en omgevingen. Elegante dingen verkopen zichzelf. Java verkoopt zichzelf.

In de kern van de zaak is Java vooral object georiënteerd. Ik ben natuurlijk geneigd om de elegante aspecten van Java toe te schrijven aan dat object georiënteerde. Kijk maar naar boeken en cursussen over object oriëntatie: grote kans dat daar de meest simpele, beheersbare, flexibele modellen worden gepresenteerd voor ogenschijnlijk taaie en onoplosbare problemen. Ga naar een Java-cursus en je leert hoe je met een minimale hoeveelheid code al hele leuke programmaatjes kunt maken. Lang leve de objecten. Maar daarna wordt het tijd om echt aan de slag te gaan.

De baas vindt het tijd dat we dat volgende systeem maar eens met Java moeten gaan doen: een groot team samengesteld, ontwikkelaars ingehuurd, definitiestudie, een stapel requirements, de planning tot het eind van het jaar is rond. Maar al na een half jaar, de eerste oplevering had al klaar moeten zijn, gaat het mis. De object georiënteerde modellen lijken helemaal niet zo meer zo inzichtelijk, beheersbaarheid wordt nu toch echt weer een probleem. We hebben keuzes gemaakt waar we niet gelukkig mee zijn maar we kunnen niet meer terug. Java lijkt ineens geen centimeter effectiever dan COBOL. Zoals bij het gros van alle IT projecten gaat het gewoon weer mis.

Wat ging er mis? De voorbeelden waarmee we Java en object oriëntatie geleerd hebben, lijken in geen enkel opzicht meer op wat we gemaakt hebben. Dat waren eigenlijk maar kleine simpele voorbeelden, geen real-life systemen. Vervolgens krijgt Java de schuld: "Java is niet geschikt voor grote projecten. Een systeem van duizenden Java classes kan natuurlijk niet beheersbaar zijn!" Maar het echte probleem ligt natuurlijk niet bij Java en ook niet bij object oriëntatie, maar bij de manier waarop we grote systemen aanpakken. De manier waarop we de complexiteit van het systeem in stukjes blijven hakken tot we denken dat we er mee aan de slag kunnen.

Wanneer ik in de gelukkige omstandigheid ben om een succesvol groot project mee te maken, blijkt telkens weer dat elegantie wel degelijk de clou is. Grote systemen kunnen wel degelijk flexibel en beheersbaar zijn. De uitdaging van deze succesvolle projecten is dat elegante oplossingen echt alleen maar in kleine teams bedacht kunnen worden. Met meer dan drie mensen worden maar zelden elegante oplossingen bedacht. Een team van drie mensen kan natuurlijk nooit een groot systeem maken. De uitdaging voor het maken van grote systemen is dan ook het organiseren van oplossingen die consequenties hebben voor het hele systeem, voordat we aandacht kunnen besteden aan de kleine aspecten van een systeem. Elk probleem wordt door een klein team benaderd. Als een oplossing niet elegant is, wordt het weggegooid. Individuele tabellen zijn nooit elegant, losse schermen zijn nooit elegant, losse stukken functionaliteit zijn nooit elegant.

Java is prima op te schalen naar grote systemen. Onze werkwijze is opgeschaald naar grote systemen. Dat hadden we al lang kunnen weten maar we werken met oogkleppen op. Geloof me, elegantie is de oplossing, hoe vaag dat ook mag klinken. Kijk maar naar succesvolle projecten om je heen. Waar elegantie is, is succes.

J. Meermans
is Java-kenner en te bereiken via
meermans@cibit.nl