SOGYO

* the art of IT

Whitepaper

Structure-First domain modeling

Agile development is populair in automatiseringsprojecten in het midden- en kleinbedrijf. Publicaties op dit gaan echter vaak in op situaties bij grotere bedrijven. Een ideaal agile team bestaat uit 5 tot 10 mensen. In onze praktijk komt het echter veel vaker voor om in teams van 2 tot 5 mensen te werken. Daarnaast zijn concrete processen en artifacts ofwel te vaag omschreven of zijn er simpelweg teveel om nog agile te blijven. Waar ligt nu de ideale mix van activiteiten en opleveringen in de praktijk?

De basis van onze methode is een modelgedreven aanpak. Deze aanpak heeft in beginsel veel weg van wat Eric Evans heeft omschreven als 'Domain Driven Design'. De kern van deze aanpak heeft tot doel een gezamenlijk vocabulaire te bouwen waardoor ontwikkelaars en designers kunnen praten met domein deskundigen. Deze domein deskundigen zijn klanten, eindgebruikers en andere belanghebbenden bij het eindproduct (uitgangspunt is dat er op zeker moment software wordt uitgeleverd aan deze belanghebbenden).

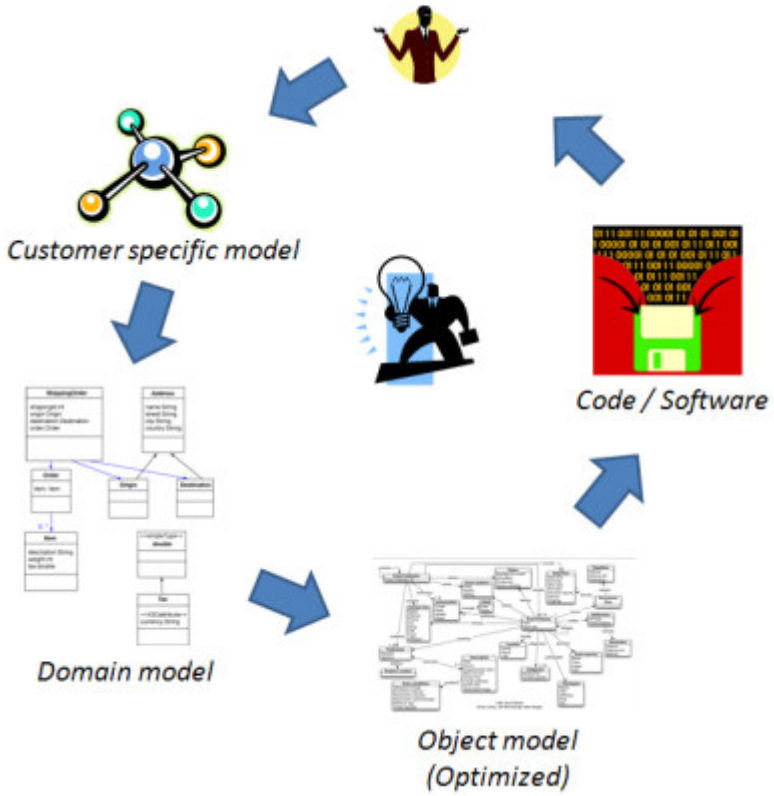


Figure 1

Onze ervaring leert echter dat modelleren in het begin veelal als lastig gevonden wordt; per situatie is er een klantspecifieke manier van omschrijven van de problematiek zijn. Deze klant-specifieke manier van communiceren brengen we onder de noemer 'Customer Specific Models' (CSM). Een transportbedrijf werkt met planborden, een document management organisatie met workflows – ieder specifiek domein kent zijn eigen manier van redeneren over de problematiek. Aan de software engineer is het dus in eerste instantie de taak om deze specifieke modellen te gaan vertalen naar een meer generiek model dat we aanduiden met 'Domain Model' (DM) en hier de belanghebbenden in mee te nemen. Vervolgens kunnen we dit vertalen in een 'Object Model' (OM) dat meer technisch van aard is; hierbij spelen bijvoorbeeld design patterns en meer infrastructurele zaken een rol.

© 1995-2008 Sogyo * The art of IT | t: 030 2202216 | f: 030 2205506 | info@sogyo.nl | www.sogyo.nl | www.software-innovators.nl

Een door ons veel gebruikte techniek om tot een DM te komen is een workshop waarbij verschillende belanghebbenden de materie *op hun eigen wijze* presenteren aan een aantal engineers. Dit kan bijvoorbeeld een software demo zijn of een middag meelopen met een aantal (eind) gebruikers van een bestaand systeem. Mocht er sprake zijn van nieuwbouw kan gedacht worden aan een meer vrije vorm als brainstorm sessies of een daadwerkelijke verkooppitch aan een aantal software engineers.

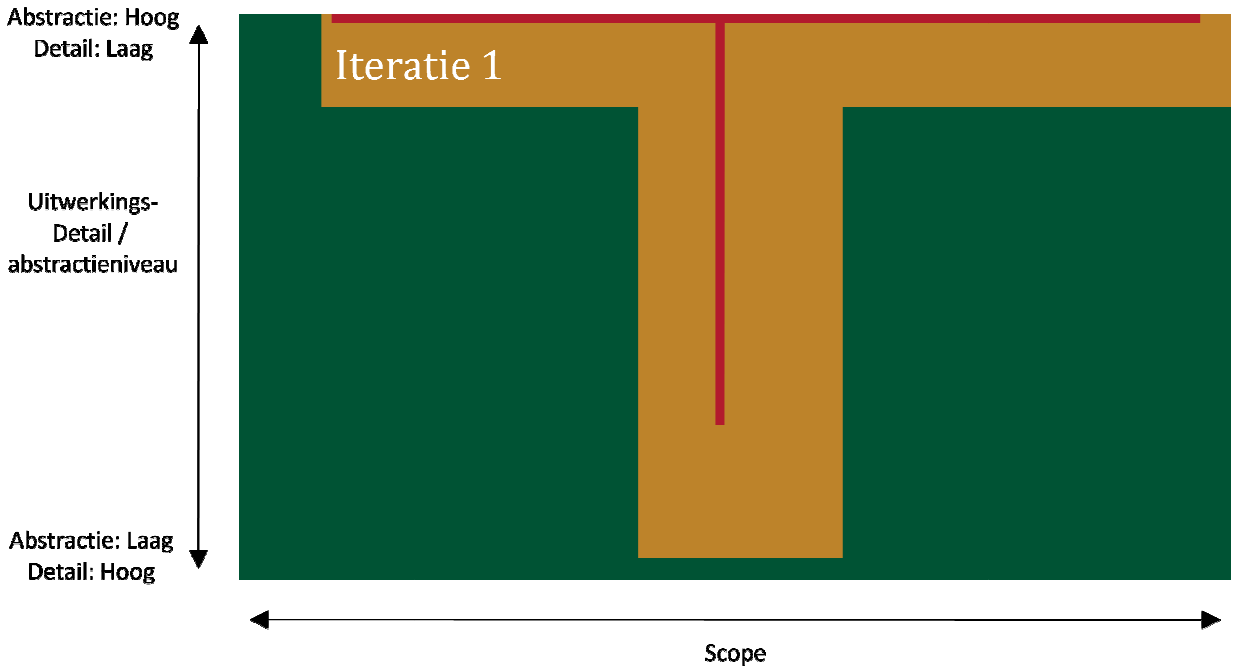
Tijdens dit soort workshops wordt door één persoon genotuleerd op post-its; de vermoedelijk relevante zelfstandige naamwoorden worden in enkelvoud genoteerd op een enkel velletje. Dit resulteert in tiental post-its voor een relatief afgekaderd bedrijfsdomein.
In de tweede helft van zo'n workshop worden de post-its min of meer at random van de stapel gepakt (hoewel het raadzaam is een grove sortering op relevantie te maken). Tijdens het pakken van een post-it plakt de workshopleider dit op een vel papier met termen die elkaar aanvullen of bijelkaar horen in hetzelfde gebied. Met de deelnemers wordt vervolgens beredeneerd wat de relatie is tussen deze termen. Zo kan het bijvoorbeeld zijn dat er een term 'klant' is en een term 'order'. De workshopleider zou de deelnemers kunnen vragen wat de relatie tussen deze twee termen is en een er associatielijn tussen trekken. Tip: veel software engineers zijn geneigd nu ook multipliciteiten aan te brengen. Het is raadzaam dit alleen te doen als je met ervaren deelnemers werkt.

Een voorbeeld van een uitkomst van een dergelijke workshop is hieronder weergegeven.

In feite kan je zeggen dat er al een eerste versie van een domeinmodel gecreëerd is na een dergelijke sessie. De relevante termen zijn voor een groot deel duidelijk en de eerste relaties zijn al gelegd.
Tip: maak een flip met belanghebbenden bij de software: actoren/gebruikers/stakeholders; deze hebben we op later moment nodig.

Nu kan een afbakening gemaakt worden van de relevante termen waarop ingezoomd wordt in de volgende activiteit. De vorm van opdeling die we daarbij aanhouden is iets dat wij het 'Iteratieve T-model' noemen – hieronder weergegeven.

© 1995-2008 Sogyo * The art of IT | t: 030 2202216 | f: 030 2205506 | info@sogyo.nl | www.sogyo.nl | www.software-innovators.nl



In dit model is de bulk van het te verrichten modelleer- en implementatiewerk als een rechthoek weergeven. Op de verticale as is de mate van detail (omgekeerd evenredig met de mate van abstractie) uitgezet, op de horizontale as de business-scope van de applicatie.

Op hoger abstractieniveau dienen we in een eerder stadium al veel breder bezig te zijn; vroege drafts van een conceptueel domeinmodel bestrijken vaak al zo'n 80% van de totale breedte van een systeem. We willen echter snel richting een detailniveau dat zo smal mogelijk is; alleen op deze manier is het mogelijk om korte iteraties te maken die daadwerkelijk iets opleveren.

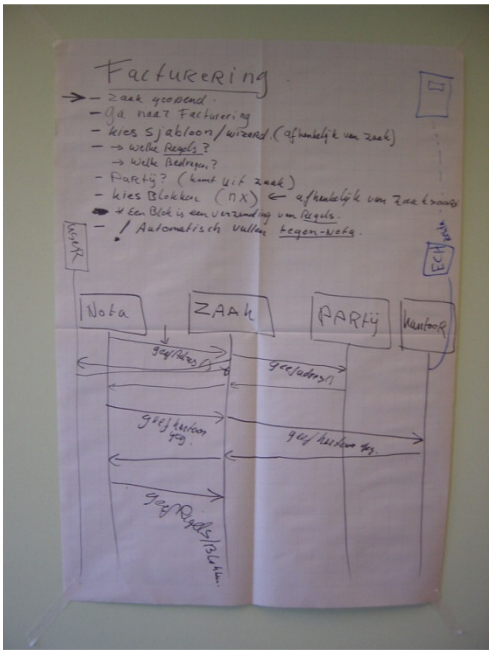
Het gele deel in bovenstaand model representeert de scope van de eerste iteratie. Om deze scope te bepalen kiezen we vaak een van de concepten die in het (draft) domeinmodel een rol spelen. Hiervoor kiezen we een proces dat we willen gaan automatiseren in de eerste iteratie. Gesteld dat dit een bestelmodule zou zijn zouden we kunnen inzoomen op de klasse 'Bestelling' of 'Order'. Een vraag die je kunt stellen aan domeindeskundigen is: hoe verloopt het bestelproces? Dit heeft echter vaak een nogal wollige uitleg tot gevolg die niet altijd even goed beklijft bij de verschillende aanwezigen. Een handigere vraag is: "Wat zijn de onderdelen die alle bestellingen (ongeacht een eventueel subtype) minimaal hebben?". Zaken als 'bestelnummer', 'verzendadres', 'afrekeningadres' etc. worden genoemd. Plak al deze onderdelen (voor zover ze nog niet in het model aanwezig waren) onder het 'bestelling' object in het model – op een andere kleur post-its.

De volgende stap is om met de deelnemers de post-its van de afwijkende kleur te gaan verdelen over het model. Wie of wat is verantwoordelijk voor het bijhouden van het verzendadres? Niet de bestelling. In vrijwel alle gevallen zullen we de post-its over meerdere objecten in het model verdeeld zien worden tijdens deze stap.

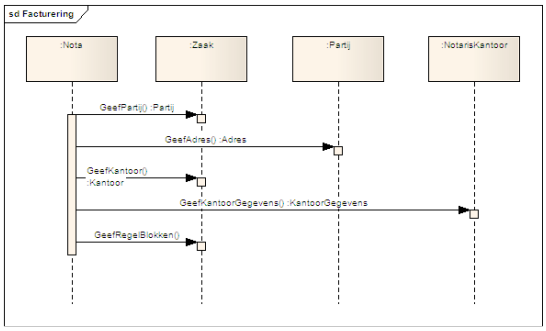
Ons doel is om nu het proces van een bestelling plaatsen te gaan uitwerken. We nemen hierbij als uitgangspunt het *doel* van dit proces; dat wil zeggen de bestelling is samengesteld. Uitgaande van een demand-chain modellering [Vens] gaan we het bestellingsobject vragen om zichzelf samen te stellen. Hiervoor maken we gebruik van het UML *sequence diagram*. Uitgangspunt is om voor ieder object met de afwijkende kleur post-its eronder in ons model een swimlane te maken. We starten met het bestellingsobject: BestelJezelf. Eventueel kan een betrokkene in de sequence opgenomen worden; het modelleringspatroon 'Ask the user' kan dan ook toegepast worden: introduceer een user object en geef

© 1995-2008 Sogyo * The art of IT | t: 030 2202216 | f: 030 2205506 | info@sogyo.nl | www.sogyo.nl | www.software-innovators.nl

deze een eigen swimlane. User input kan op deze manier meegenomen worden. Deze strategie is vooral toepasbaar als men moeite heeft om abstact over de materie na te denken.



Bovenstaande figuur is een voorbeeld van een sequence die in eerste instantie samen met deelnemers is samengesteld. Een ‘informele’ stappenlijst is eerst opgenomen, waarna we zijn gaan kijken welke objecten moeten gaan samenwerken om de sequence te volbrengen. Hieronder zien we het resultaat van de sequence uitgewerkt in UML. Hier zien we duidelijk dat een aantal objecten al eigen verantwoordelijkheden toebedeeld krijgt.



Teruggrijpend op figuur 1 kunnen we nu zeggen dat we de stap van CSM naar DM gezet hebben. De hierboven geschreven sequence kunnen we loslaten op een DM maar in feite maken we deze pas echt executeerbaar in een OM.
Nu we de interfaces voor deze functionaliteit min of meer duidelijk hebben kunnen we een scenario gaan opschrijven in een test. Deze testgedreven manier van werken komt voort uit de stroming ‘Test Driven Design’.

Deze manier van integratie van structureel- en interactiegerichte modellen noemen we ‘structure-first’. We leggen eerst een structuur neer en staven deze doormiddel van het loslaten van een interactiemodel.

© 1995-2008 Sogyo * The art of IT | t: 030 2202216 | f: 030 2205506 | info@sogyo.nl | www.sogyo.nl | www.software-innovators.nl