Boilerplates

Jeremy Dick, de laatste keynote speaker, van het Dream event begint zijn presentatie met een paar voorbeelden waarom natuurlijke taal niet het ideale medium is om requirements in te verwoorden. Requirements uitgedrukt in natuurlijke taal hebben vaak de onaangename eigenschap onvolledig, vaag, dubbelzinnig en moeilijk meetbaar te zijn. We hebben daarom regels bedacht om de kwaliteit van requirements vast te stellen. De meest gebruikelijke werkwijze is dat we eerst de requirements opstellen en dan achteraf toetsen of de requirements wel voldoen aan de gestelde kwaliteitscriteria.

door Reinoud de Leve

Jeremy Dick laat in zijn presentatie op het Dream Event zien dat we heel gemakkelijk deze werkwijze kunnen omdraaien. We hebben regels aan de hand waarvan we de kwaliteit van requirements kunnen beoordelen. “Waarom passen we die regels achteraf toe als we ze ook tijdens het schrijven van de requirements, of beter nog voordat we de requirements gaan opschrijven zouden kunnen toepassen?” Vraagt hij zich af.

> Een boilerplate bestaat uit de grammaticale structuur van een correct verwoord requirement van een bepaald type.

Het enige wat we daarvoor nodig hebben zijn volgens Jeremy Dick ‘Boilerplates’. Het begrip boilerplate stamt uit de tijd van de stoomlocomotief. Het is daarom niet toevallig dat uitgerekend een Engelsman dit begrip adopteert in de requirements engineering. Engeland is ten slotte het land waar de stoommachine het meest tot bloei kwam. In de tijd van de stoomlocomotief was een boilerplate de plaat staal waaruit het omhulsel van de boiler werd gemaakt. De staalplaat was vooraf op maat gebracht en voorzien van gaten op de plaatsen waar later kranen en buizen moesten worden gemonteerd.

Een boilerplate in termen van Jeremy Dick is een template op ‘statement-level’. Een boilerplate bestaat uit de grammaticale structuur van een correct verwoord requirement van een bepaald type. In de boilerplate zijn de specifieke aspecten van de requirement vervangen door een ‘placeholder’. Deze lijkt daarmee op de boilerplate uit het stoomtijdperk. Ook daar waren in de boilerplate gaten vrijgelaten om later in te vullen als men van de boilerplate een echte boiler maakte.
Het begrip ‘boilerplate’ heeft verwantschap met begrippen als ‘template’ en ‘pattern’. Het begrip ‘template’ wordt binnen requirements engineering meestal gebruikt voor documenten. Patterns refereren doorgaans ook naar iets breders dan een enkele zin. Een voorbeeld van een boilerplate is:

> The <user> shall be able to <capability> at a maximum rate of at least <quantity> times per <time unit>.

Met deze boilerplate kan bijvoorbeeld de volgende requirement worden verwoord: "The order entry clerk shall be able to raise an invoice at a maximum of at least 10 times per hour." De boilerplate biedt een handvat aan de persoon die de requirement moet opstellen. Hij moet alle placeholders in de structuur invullen of in ieder geval over de mogelijke invulling ervan nadenken. Als hij de placeholders in de boilerplate op de juiste wijze invult, heeft hij een requirement geformuleerd dat gegarandeerd voldoet aan een aantal kwaliteitscriteria. De requirement

is volledig, ondubbelzinnig, meetbaar enzovoort. Dit hangt natuurlijk wel af van de kwaliteit van de boilerplate.

Jeremy Dick geeft een voorbeeld. Bij het toepassen van boilerplates in het project Future Surface Combatant, een militair project met onder andere helikopters en een oorlogsschip, had men de requirement: “The FSC shall be able to launch helicopters within 3 minutes.” Het boilerplate om uit te drukken dat een systeem in staat moet zijn een specifieke actie uit te voeren binnen een bepaalde tijd ziet er als volgt uit:

> The <system> shall be able to <capability> within <time_quantity> <time_units> of <event>.

We zien nu als we de requirement met de boilerplate vergelijken dat er iets aan de requirement ontbreekt. De requirement is niet duidelijk over ten opzichte waarvan de tijd moet worden gemeten. Er moet aan de requirement nog een ‘event’ worden toegevoegd.

Boilerplates hebben niet alleen voordelen voor degenen die de requirements opstellen, maar ook voor degenen die de requirements moeten lezen. Zonder boilerplates zie je vaak dat verschillende specificeerders min of meer identieke requirements op een totaal verschillende manier verwoorden. Met boilerplates krijgen we een uniforme taal waarin iedereen dezelfde dingen op een zelfde wijze verwoordt. Je kunt die uniforme taal binnen een project, binnen een organisatie, of misschien zelfs wel over organisaties heen definiëren. Het voordeel daarbij is dat als het nodig is, je de boilerplate op één plaats kunt aanpassen en dat de aanpassing dan geldt voor alle requirements die met behulp van die boilerplate zijn geformuleerd.

Tenslotte kunnen boilerplates ondersteunen bij het opstellen van een consistent gemeenschappelijk vocabulaire. Een placeholder in een boilerplate moet je vervangen door een waarde van een specifiek type, bijvoorbeeld een toestand van het systeem. Je kunt dan afspreken uit welke mogelijke waarden men kan kiezen en wat de exacte betekenis van die waarden is. Zo bewerkstellig je dat niet alleen iedereen dezelfde vorm maar ook dezelfde begrippen gaat hanteren.

Het aantrekkelijke van het werken met boilerplates is dat het goed is in te passen in verschillende werkwijzen. Zo bieden requirement management tools de mogelijkheid om een repository van boilerplates toe te voegen. De gebruiker kan dan bij het invoeren van de requirement eerst de juiste boilerplate uit de repository kiezen en deze vervolgens completeren. Boilerplates zijn in verschillende projecten succesvol toegepast. British Airways heeft bijvoorbeeld de requirements voor haar systeem voor het afhandelen van bagage met boilerplates opgesteld. Maar ook als je omgeving nog niet expliciet met boilerplates werkt, helpt het je als je het concept zelf hanteert bij het opstellen van je requirements.