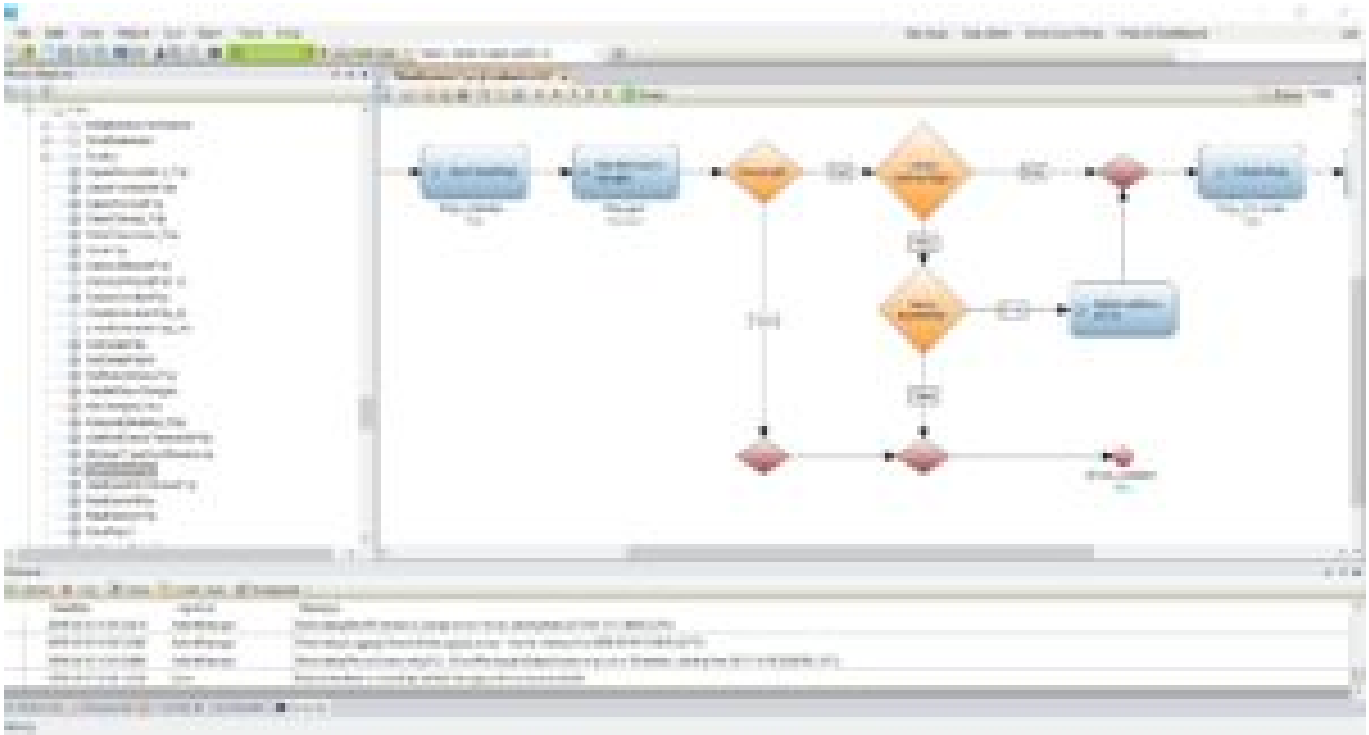
Zoals de term low-code al aangeeft, wordt er bij het ontwikkelen van een applicatie in een low-code platform zoals Mendix niet of beperkt ‘ouderwets geprogrammeerd’. Welke gevolgen heeft dit voor het testen van een applicatie? Betekent low-code ook dat er minder intensief getest hoeft te worden, dus low-test? Zijn de termen unittest, systeemtest en integratietest hiermee achterhaald? Is een acceptatietest voldoende?

Mijn ervaring met het testen van een Mendix applicatie (App) tot nu toe: dat hangt ervan af ….

Het idee achter low-code platformen is dat je op een visuele manier een applicatie opbouwt, waarbij je door het aan elkaar koppelen van diverse bouwstenen de gewenste functionaliteit realiseert. Uiteraard moet je eerst wel goed nadenken over wat je applicatie moet gaan doen en niet ‘zomaar beginnen’, maar je kunt al relatief snel een eerste resultaat opleveren en demonstreren aan de eindgebruiker. In een Agile omgeving werkt dit prima.

De functionele werking wordt opgebouwd via microflows waarmee een bepaalde actie wordt uitgevoerd. De bouwstenen van een microflow zijn afkomstig uit een bibliotheek. Door de visuele manier van bouwen is de werking van een microflow ook voor een tester redelijk goed te volgen.



In een eenvoudige App waarbij een beperkt aantal microflows een bepaalde systeemfunctie vervult, kun je al vrij snel volstaan met een Systeem/Acceptatie Test. Bij de ontwikkeling van een complexe App waarbij in een Agile setting wordt gewerkt met meerdere scrumteams wordt het testen ook een stuk complexer. De microflow structuur is complexer en microflows

worden hergebruikt op diverse plekken in de applicatie.

Veelal zal er hierdoor een (onbewuste) afhankelijkheid zijn tussen de user stories die in de diverse teams worden opgeleverd. De tester moet zich hiervan bewust zijn en ‘over de teamgrens heen’ testen. Dit kan tot een onverwachte extra testinspanning leiden die pas tijdens de sprint duidelijk wordt. Door in iedere sprint tijd in te ruimen voor automatisering van dit soort tests kan dit worden ondervangen.

In het Mendix project waarin ik werkzaam ben, ‘sloop’ de complexiteit en de afhankelijkheid tussen teams langzaam in de App. Dit was goed te merken bij het uitvoeren van de geautomatiseerde regressietest.

Low-test? In bepaalde situaties kun je je dat veroorloven, maar zodra een applicatie (landschap) complexer wordt, moet je een duidelijke testaanpak hebben voor de diverse testniveaus en testsoorten.

Douwe Wienke, testexpert bij Polteq

Douwe Wienke, testconsultant bij Polteq