Scrum# noodzaak bij integratie van software, elektronica en mechanica

Omdat Scrum niet goed kan omgaan met planningswijzigingen en verstoringen die tijdens een sprint optreden, moet het proces in dynamische tijden worden aangevuld. Scrum# voorziet hierin. Assembléon en Promexx hebben deze aanpak in praktijk gebracht bij de ontwikkeling van een compleet nieuwe pak-en-plaatsmachine. Met twee handen aan het stuur door de integratiefase van software, elektronica en mechanica.

Mark den Hollander Ron Piree

Ruim twee jaar geleden is Assembléon gestart met de ontwikkeling van de Iflex, een pak-en-plaatsmachine die de productiviteit van elektronicafabrikanten met een grote productmix meer dan dertig procent moet verhogen. Bij het project heeft het Veldhovense bedrijf besloten een open-innovatieaanpak te volgen met (voornamelijk regionale) partijen die de software, elektronica en mechanica voor hun rekening nemen. Partner voor de softwareontwikkeling is Promexx.

Assembléon stond voor de uitdaging om een compleet nieuwe machine te realiseren binnen korte tijd en met een beperkte organisatiegrootte. De korte time-to-market en de grote hoeveelheid geschat softwarewerk (circa tachtig manjaar) maakten het noodzakelijk om zo snel mogelijk te starten met de implementatie. Tel daarbij een beperkt budget, een nog niet duidelijk systeemconcept en een onvolledig eisenpakket en Scrum was een logische keus.

Chaotisch

John van Meel van Promexx en Giel van Doren van Assembléon hebben Scrum in 2008 geïntroduceerd binnen de Veldhovense organisatie. Voor de Iflex-ontwikkeling hebben we de werkwijze opgeschaald naar ongeveer veertig software-FTE's, verdeeld over vijf teams. Niet lang na de start van het project hebben we de aanpak uitgebreid, naar momenteel gemiddeld zo'n zeventig FTE's in tien (mono- en multidisciplinaire) scrumteams. De sprints van drie weken zijn gesynchroniseerd over alle teams. Het voordeel van één ritme is dat het project en al zijn stakeholders kortcyclisch denken, wat onder meer de afstemming vergemakkelijkt van intakes en *deliverables*.

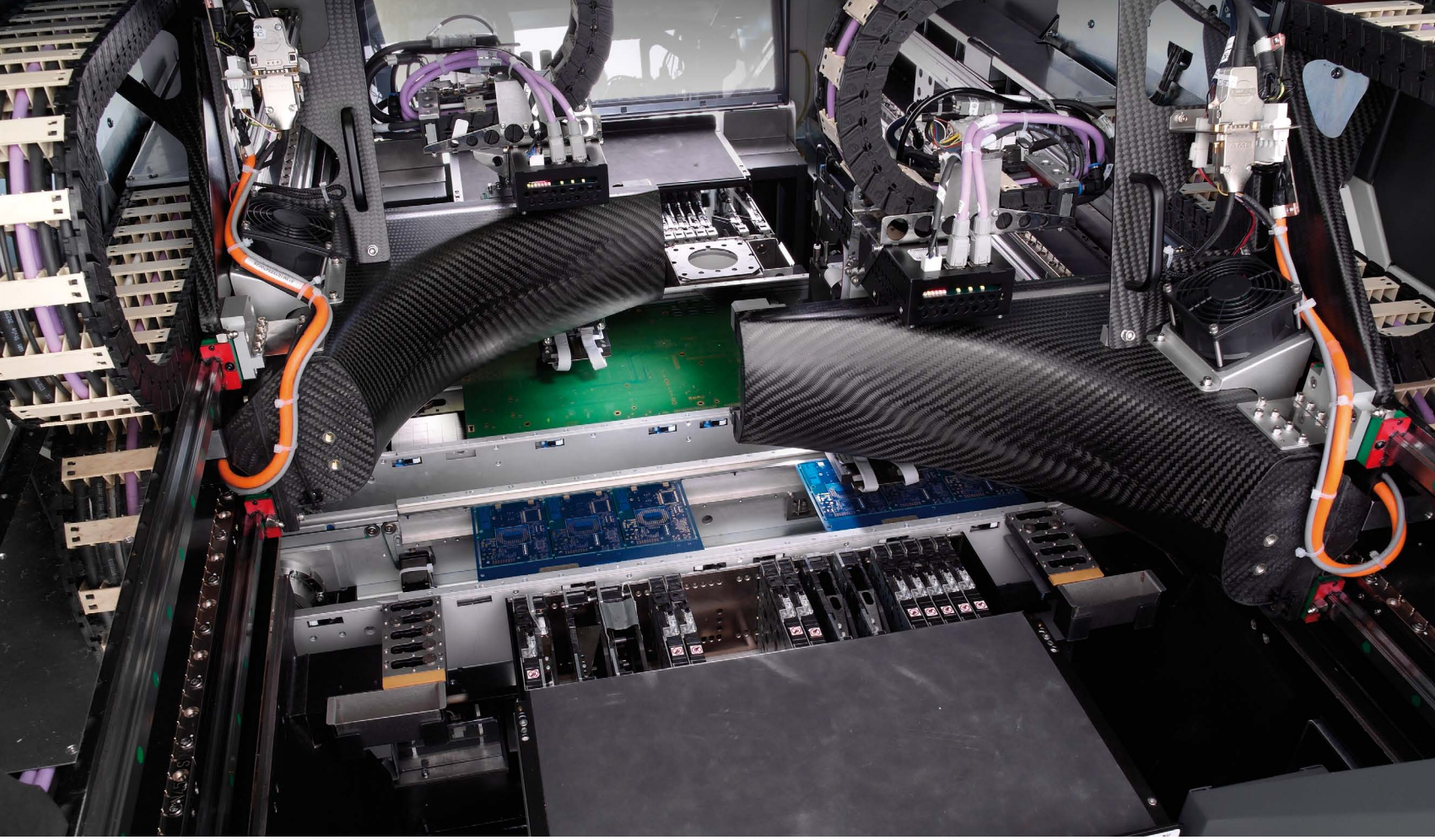
In het eerste projectjaar was de hardware voor de Iflex nog volop in ontwikkeling en niet beschikbaar als testplatform voor het softwareteam. Hierdoor konden we met dat team focussen op de definitie en de realisatie van de software, zonder al te veel last te hebben van verstoringen. In deze eerste fase konden we de Scrum-aanpak ten volle benutten. Voor integratietests hebben we zo veel mogelijk de bestaande generatie pak-en-plaatsmachines gebruikt als testplatform. Langzaam maar zeker zagen we echter steeds meer Iflex-subsystemen op de testvloer verschijnen, die we ook softwarematig met elkaar moesten integreren. Hiermee veranderde de focus van het softwareteam.

Met het beschikbaar komen van afgeronde story's en de toenemende mate van systeemintegratie begon ook het aantal softwaredefecten te groeien. Vooral bij het bereiken van een mijlpaal uit het machine-integratieplan vochten story's en defecten om de hoogste prioriteit. Daarnaast werden de softwarewerkzaamheden regelmatig verstoord door supportvragen vanuit de fabriek en de testruimte (over bijvoorbeeld inbedrijfstelling, integratietests en foutafhandeling). Zo kon de defectplanning, die we tot dan toe maakten bij aanvang van de sprint, op de eerste dag van de cyclus alweer achterhaald zijn. Ook eventuele urenstelposten voor defecten en supportvragen waren per definitie verkeerd gekozen.

Het resultaat was een chaotisch softwareontwikkelproces. Het was duidelijk dat we de grenzen van de Scrum-methodiek hadden bereikt. Deze situatie moesten we onder controle zien te krijgen. Of zoals Ken Schwaber, een van de grondleggers van Scrum, het treffend verwoordt: 'Operating at the edge of chaos (unpredictability and complexity) requires management controls to avoid falling into chaos.'

Echt stuur

Er moest een verandering komen in de werkwijze van het softwareteam, zodat we veel kortcyclischer op defecten konden reageren. Die aanpassing was op zich snel gevonden: Kanban. Door de werkvoorraad per software-engineer te beperken

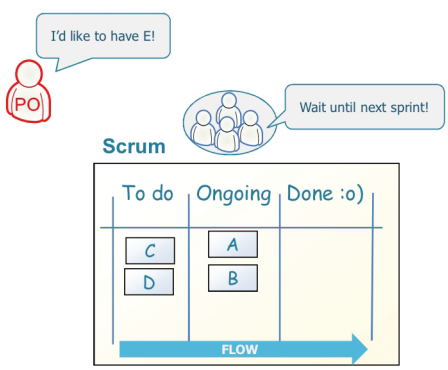


tot maximaal twee à vier defecten, kunnen we op elk moment de defecten die dan de hoogste prioriteit hebben *just in time* oppakken. Zo hebben we de eerste planningsproblematiek opgelost.

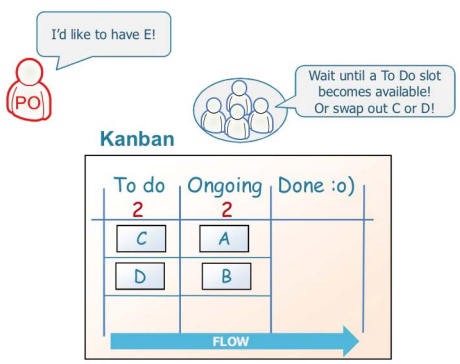
Vanwege de ad-hocmomenten waarop defecten aan het licht kwamen, ontstond al snel de behoefte om bijna automatisch te kunnen prioriteren zonder steeds eerst een CCB-overleg (Change Control Board) te hoeven houden. Daarom hebben we een mogelijkheid gezocht om een automatische prioriteitsbepaling uit te voeren op defecteigenschappen. Uiteindelijk hebben we gekozen voor de eigenschappen machinefunctie (MF, bijvoorbeeld *pick*, *place*, *transport* of *start-up*), *business value driver* (VD, bijvoorbeeld uptime, output of accuratesse) en *severity* (SV: *safety*, *critical*, *major*, *minor*). Per defect vullen we de waardes hiervoor in in de probleemtrackingtool Change Synergy. Door aan de machinefuncties en value drivers gewichtsfactoren toe te kennen, kunnen we de prioriteit eenvoudig berekenen met de formule (MF + VD) x SV.

De vraag was hoe accuraat deze automatische ranking eigenlijk is. Binnen Assembléon hanteerden de verschillende teams, proceseigenaren en CCB's allemaal hun eigen prioriteitenlijstjes. Deze hebben we naast de automatisch bepaalde lijst gelegd. In ongeveer tachtig procent van de gevallen bleek de match heel goed; in ieder geval de hoogste en laagste prioriteiten kwamen exact overeen. De tussenliggende verschillen hebben we nader bekeken, waarna we de defecteigenschappen waar nodig hebben bijgesteld. Gezond verstand en *sanity checks* blijven belangrijk bij het monitoren van de prioriteitsstelling, maar het komt slechts sporadisch voor dat een defect echt verkeerd is ingeschaald.

Hiermee hebben we een belangrijk resultaat bereikt: nieuw ingediende defecten kunnen we aan de hand van de drie genoemde eigenschappen automatisch prioriteren en bij het eerstvolgende Kanban-tijdslot toekennen aan een engineer. Voor de software-engineers die de defecten oppakken, verandert er helemaal niets in de manier van werken; vooral de administratieve verwerking en de verdeling van de defecten verloopt anders. Het eerst zo chaotische



Scrum ontmoedigt prioriteitswijzigingen tijdens een sprint (lead time bij Assembléon: maximaal drie weken).



Kanban gaat gemakkelijker om met prioriteitswijzigingen (lead time bij Assembléon: gemiddeld tien uur).

Illustratie: Henrik Kniberg, 'Kanban and Scrum – making the most of both' (2009)

To do	In progress	Completed without autotest	Completed with autotest
Most wanted		Requirements (onvoldoende/ onvolledig)/fout in softwaredesign	Requirements (onvoldoende/ onvolledig)/fout in softwaredesign
		Change request	Change request
Verificatie nachtbouw		Programmeerfout (statische check)	Programmeerfout (statische check)
		Programmeerfout (dynamische check)	Programmeerfout (dynamische check)
Kanban		Defectpreventie/ analyse verbeteringen/onderhoud	Defectpreventie/analyse verbeteringen/onderhoud
		Software-integratie (losse story's vormen niet de gewenste functionaliteit)	Software-integratie (losse story's vormen niet de gewenste functionaliteit)
		Hardware-software-integratie	Hardware-software-integratie
		Hardware niet oké/ externe softwarefout	Hardware niet oké/ externe softwarefout
		Problemen op lopende story's	Problemen op lopende story's
Kaizen		Duplicate/reject	

Het Kanban-bord van het brandweerteam, met swim lanes voor 'most wanted' (defecten die met de allerhoogste prioriteit worden afgehandeld), 'verificatie nachtbouw' (defecten uit de dagelijkse regressietests met hoge prioriteit), 'Kanban' (defecten die met de berekende prioriteit worden afgehandeld) en 'Kaizen' (procesverbeteringstaken). In de 'completed'-kolommen is de foutcategorie van een defect aan te geven.

softwareontwikkelproces is nu een beheerste stroom aan defecten, waarbij de focus op de hoogste prioriteiten procesmatig is geborgd. Door de gewichtsfactoren voor de machinefuncties en value drivers aan te passen, kunnen we die focus ook eenvoudig en razendsnel verleggen. 'Ik heb eindelijk een echt stuur in handen', aldus de teamleider die de defecten in- en verdeelt. Ook de rest van de organisatie ervaart deze manier van werken als zeer doeltreffend.

Brandweer

Kort na introductie van de werkwijze stagneerde het Kanban-proces echter omdat het uitdelen van de defecten plotseling veel trager verliep. Een nadere analyse duidde op keuzeproblemen bij de software-engineers die de defecten oppakten. Zij moesten daarnaast immers ook aan story's werken. De positieve kant van het verhaal was dat er nog nooit eerder zo snel en duidelijk een signaal was gekomen van stagnatie in het oplossend vermogen.

Door de constante afweging die de software-engineers moesten maken tussen story- en defectwerk vertoonde hun oplossnelheid te veel schommelingen. Het was duidelijk dat het Kanban-proces stevigere ondersteuning nodig had. Daarom hebben we het brandweerteam opgezet, bestaande uit zes software-engineers met vier fulltime taken. Ten eerste lossen ze softwaredefecten op. Ten tweede vangen ze supportvragen vanuit de fabriek, testruimte of testsites af en houden ze de functionele scrumteams in de luwte zodat die zich volledig kunnen focussen op storywerk. Ten derde analyseren ze de onderliggende oorzaak van de opgeloste defecten en initiëren ze preventieve acties. Ten vierde breiden ze de regressietestset uit zodat opgeloste defecten niet nogmaals ongemerkt worden geïntroduceerd. De leden van het brandweerteam komen vanuit de functionele teams en rouleren op sprintbasis. Zodoende blijft iedereen betrokken bij de eindgebruiker en vloeit de opgedane kennis weer terug in de functionele teams.

Met de start van het brandweerteam lag de oplossnelheid van de defecten al vrij snel op het gewenste niveau. Deze is ook nagenoeg constant gebleven. In de eerste sprint dat het brandweerteam operationeel was, hebben de leden zelfs nog ruim honderd uur aan support verleend aan de fabriek en de testruimte, maar dat heeft geen negatieve invloed gehad op de snelheid.

Assembléon gebruikt Scrum nu voor de realisatie van elke nieuwe functionaliteit. Hiermee kan het inspelen op de soms enorme dynamiek in de volgorde waarin functies voor klantspecifieke wensen moeten worden opgeleverd. Als toevoeging op de werkmethodiek hebben de introductie van Kanban, de semi-automatische prioriteitsbepaling en de invoering van het brandweerteam ervoor gezorgd dat de ontwikkelaars de instroom van defecten volledig procesmatig en volledig gefocust kunnen aanpakken. Met deze Scrum#-benadering kan het brandweerteam een nieuw defect van de hoogste prioriteit meestal nog dezelfde dag of uiterlijk de volgende werkdag oppakken. Daarnaast kunnen de functionele teams gefocust hun werk doen omdat de brandweer eventuele verstoringen afvangt.

Mark den Hollander is sinds 2007 vanuit Promexx werkzaam als senior software-engineer bij Assembléon en vanaf 2009 als teamleider van een Iflex-softwarescrumteam. Ron Piree is sinds 2009 vanuit Promexx werkzaam als projectleider bij Assembléon en vanaf 2011 als projectleider software voor de Iflex. Daarnaast verricht hij namens Promexx consultancywerkzaamheden bij andere partijen, onder meer op het gebied van Scrum en systeemarchitectuur.

Redactie Nieke Roos