Grip on software: understanding development progress of SCRUM sprints and backlogs

Helwerda, L.S.

Samenvatting

Grip op Software: Voortgang van ontwikkeling van SCRUM *sprints en backlogs beter begrijpen*

Complexiteit is alledaags in softwareontwikkelprocessen. Softwareontwikkelaars, kwaliteitsengineers en andere leidinggevenden ondervinden vaak uitdagingen tijdens de ontwikkeling, in het bijzonder het balanceren van inspanning die gemoeid gaat met allerlei taken behorende bij het implementeren en onderhouden van features in softwareproducten. Deze experts moeten omgaan met een digitaal ecosysteem dat altijd in verandering is, evenals met verschuivingen van prioriteit bij de wensen van de vertegenwoordigers van de gebruiker, zijnde de cliënt.

Om tijdens een ontwikkelfase gefocust te blijven op het implementeren van behoeftes, beschreven in user stories—niet vastgelegde plannen die lang geleden in de duur van het project zijn geformuleerd—kiezen softwareontwikkelteams er vaak voor om te werken volgens een Agile ontwikkelmethode zoals SCRUM. We willen de voorspelbaarheid van de sprintcyclus binnen SCRUM op de korte termijn en de planning van de product backlog op de lange termijn verbeteren en tegelijkertijd de flexibiliteit van het proces behouden.

De essentie van SCRUM zorgt ervoor dat herhaalde opeenvolgingen van bijeenkomsten en gebeurtenissen plaatsvinden. Dit maakt het mogelijk voor ons om gestructureerde informatie over de voortgang te extraheren uit verschillende systemen die de teams gebruiken als digitaal hulpmiddel. Na het vergaren van de data en het modelleren in een database kunnen we relevante metrieken en features beschrijven en selecteren voor een dataset, met als doel het verbeteren en evalueren van algoritmes uit machine learning.

We introduceren meerdere algoritmes, ten eerste voor het voorspellen van werk waarvoor een team een inzet zou kunnen afspreken om af te hebben voor het einde van een sprint in SCRUM, welke een paar weken duurt. Daarnaast onderzoeken we algoritmes voor het inschatten van de hoeveelheid werk op de backlog voor een langere tijd. De resultaten van deze modellen laten zien dat we kunnen aangeven of gekozen user stories—die story points krijgen als inschattingen voor de inspanning door experts—met redelijke betrouwbaarheid worden voltooid tijdens een sprint. We tonen ook aan dat het moeilijker is om te bepalen of bepaalde milestones in de toekomst haalbaar zijn, bijvoorbeeld door onvoorziene veranderingen in projectdoelen. Toch stellen deze algoritmes en datasets ons in staat om patronen uit de levenscyclus van een softwareontwikkelproject te benadrukken. Hiermee maken we nieuwe, interactieve informatievisualisaties beschikbaar die beslissingen ondersteunen.

Om de omvang van de dataset te vergroten en daarmee de algoritmes van verse, nieuwe data te voorzien, bouwen we een pipeline gericht op gegevensverzameling voor ons onderzoek, Grip op Software (GROS). De GROS-pipeline is ontworpen met het uitgangspunt van generaliseerbaarheid,

zodat het mogelijk is om data uit verschillende systemen te halen waar verschillende teams en organisaties mee werken. Bij twee Nederlandse overheidsdiensten voor IT—Stichting ICTU en Wigo4it—gebruiken we de componenten van de pipeline om datasets te verwerven en uit te breiden door regelmatig gedistribueerde agent-programma's data te laten verzamelen bij de organisatie. Een centrale opstelling verkrijgt eveneens bijgewerkte gegevens voor een gecombineerde dataset. We houden rekening met belangen rondom privacy en beveiliging door gevoelige informatie van projecten en persoonsgegevens te (pseudo-)anonimiseren.

Nadat nieuwe data is verzameld uit systemen rondom projectbeheer (Jira en Azure DevOps), versies van code (zoals Git, inclusief samenwerkingsfuncties uit GitHub en GitLab), kwaliteitscontrole (SonarQube en Quality-time), bouwplatforms, enzovoort, modelleren we de data op basis van overeenkomsten tussen systemen en leggen we nieuwe verbanden tussen deze systemen, zodat we opkomende metrieken kunnen afleiden uit overlappingen binnen data uit softwareontwikkelprocessen. Dit model leidt tot de inrichting van een MonetDB-database, waar grote aantallen registraties voor entiteiten en relaties uit het proces kunnen worden ingevoegd, bijgewerkt en massaal opgehaald met behulp van kolomgebaseerde opslag.

Het daadwerkelijke afleiden en selecteren van features voor de dataset vindt plaats met behulp van een nieuw opvraagsysteem met gecompileerde sjablonen. Deze sjablonen zijn bruikbaar om verschillende, op elkaar lijkende databronnen op te vragen, ongeacht of de data iets anders is gemodelleerd, om op die manier gevarieerde ecosystemen voor ontwikkeling bij de betrokken organisaties te ondersteunen. Gemeenschappelijke variabelen worden tussen de opvraagsjablonen gedeeld, wat het eenvoudiger maakt om bijvoorbeeld de inschattingen van inspanning op basis van story points te definiëren en tegelijkertijd ruis van de dagelijkse gang van zaken te verminderen.

We passen deze opvraagsjablonen toe op ons onderzoek om nieuwe fetures te maken, gebaseerd op adviezen van ontwikkelaars, SCRUM-coaches en kwaliteitsengineers, om tot een numerieke beschrijving voor verschillende metrieken, teameigenschappen en gebeurtenissen uit een sprint te komen. We splitsen onze dataset met voorbeelden van sprints op in sets voor trainen, testen en validatie, waarbij we rekening houden met het tijdsaspect van de data. Door alleen oudere sprints van een ontwikkelproject te gebruiken tijdens het trainen voorkomen we problemen waar de inspanning van een team in een latere sprint beïnvloed was door het resultaat van een eerdere sprint; deze sprint hoort dus niet voorbij te komen tijdens het testen en valideren.

Om de dataset en modellen te verfijnen, krijgen de afgeleide features ook scores die bepalen welke features het meest bijdragen aan het inschatten van een verwacht label, bijvoorbeeld een indicatie of stories klaar zijn aan het einde van een sprint. Daarnaast gebruiken we één van onze modellen om een representatieve deelverzameling van features te kiezen gebaseerd op een afstandsmaat, vergelijkbaar met clusteren.

Door de beperkingen in dimensies gebruiken we modellen voor classificatie en inschatting die gebruik maken van praktijken uit deep learning, met in het bijzonder structuren die uitlegbaar zijn aan belanghebbenden. We gebruiken een drielaags neural netwerk (DNN) voor classificatie—met een F1-score van 89.98%—en analogie-gebaseerde inspanningsschatting (ABE) voor het bepalen van het aantal story points dat het team zou kunnen afmaken tijdens een sprint, met inschattingen van het laatstgenoemde model die binnen een 25% marge voor 88.6% van de sprints in de validatieset van één organisatie zijn. De stabiliteit van het ABE-model is echter problematisch, omdat het niet overweg kan met gecombineerde data van meerdere organisaties, terwijl de classificatie van DNN statistisch gezien verbetert met grotere datasets.

Voor het, op langere termijn, voorspellen van backloggroottes, vergelijken we een lineaire regressie met Monte Carlo-simulaties op basis van verschillende scenario's, die meer data uit

eerdere voortgang van de backlog en andere veranderingen meenemen. We zien dat de Monte Carlo-methode in staat is om een juiste normaalverdeling van uitkomsten te genereren die het best lijkt op een uiteindelijke voltooiing van een project, met slechts een derde van het verloop van een project om de simulaties op te starten. Echter onderschatten alle modellen de backloggrootte, met als vermoedelijke reden veranderingen in projectdoelen.

De dataset en resultaten van de voorspellingen zijn de basis voor een centrale locatie van informatievisualisaties, die elk een vernieuwende blik bieden op patronen en situaties gevonden in ontwikkelprocessen. De interactieve visualisaties omvatten aanpasbare rapporten, panelen met gedetailleerde voorspellingen, tijdslijnen, netwerken als grafen, stroomschema's en kalenders met activiteitsniveaus. Deze ontwerpen dienen om verschillende aspecten van het gemodelleerde proces te belichten. Op basis van gesprekken met belanghebbenden maken we extra hulpgereedschappen beschikbaar voor de details die zij nodig hebben. Zo bieden wij intuïtieve toegang tot de resultaten, met focus op de belangrijkste onderdelen en opties om in te zoomen op fijnmazige informatie.

Daarnaast integreren we statusmetrieken van voorspellingen in kwaliteitsrapporten en bouwen we nieuwe grafieken voor backlogs die getoond kunnen worden binnen systemen voor projectbeheer, met links en referenties naar de visualisaties en brongegevens. Tests voor bruikbaarheid en toegankelijkheid tonen aan dat de opzet effectief is, met meerdere belanghebbenden die de visualisaties in hun werkwijze opnemen.

Over het geheel genomen hebben deze benaderingswijzen ervoor gezorgd dat we antwoorden en oplossingen kunnen geven voor onze onderzoeksvragen en doelen uit dit proefschrift, met de centrale focus op het begrijpen en verbeteren van het SCRUM-softwareontwikkelproces, met name de voorspelbaarheid van het leveren van stapsgewijze veranderingen tijdens sprints en bruikbare producten bij milestones op de langere termijn. Met behulp van het extraheren en analyseren van features die gebeurtenissen beschrijven maken wij een dataset en bouwen wij modellen gericht op verschillende voorspellingstaken. De pipeline voor gegevensverzameling en patroonherkenningsmethodes worden aangevuld met visualisaties in een gebruikersomgeving, wat een nieuwe terugkoppeling opstart met de belanghebbenden in het ontwikkelproject. Door de gegevensverzameling, database-opslag en analyse te versnellen, vergroten we de dataset regelmatig, wat leidt tot classificaties, inschattingen en informatievisualisaties die dagelijks—of vaker—worden vernieuwd. Alles bij elkaar genomen maakt dit de weg vrij voor een verbeterde SCRUM-softwareontwikkelmethode.