Studying the benefits of using UML on software maintenance : an evidence-based approach

Fernandez Saez, A.M.

SAMENVATTING

Het lijkt er op dat bedrijven geen gebruik maken van software modelleren omdat ze vrezen dat er up-font investeringen voor nodig zijn. Vanuit een economisch perspectief moet voor iedere investering een rechtvaardiging bestaan die bekijkt hoeveel een investering de organisatie oplevert. In de context van software projecten, moeten investeringen in modelleren gerechtvaardigd worden door voordelen, zoals verbeterde productiviteit en verbeterde product kwaliteit, welke waargenomen kunnen worden in latere stadia van software development en onderhoud aan software. Indien zulke voordelen afwezig of onvoldoende tastbaar zijn, dan wordt modelleren een activiteit zonder duidelijk toegevoegde waarde voor het systeem dat ontwikkeld wordt.

De grote vraag rondom software modelleren ligt feitelijk in de aanname dat kwantificeerbare voordelen oplevert aan software projecten. Het probleem is daarom hoe we kunnen onderzoeken en bewijzen of modelleren wel of geen voordelen oplevert voor software onwikkeling en onderhoud. Zo lang als deze vraag onbeantwoord blijft, zal het moeilijk zijn om het gebruik van modelleren te rechtvaardigen en ertoe te motiveren.

Om deze reden draagt dit proefschrift bij aan het gedeeltelijk beantwoorden van deze open vraag door haar onderzoek te concentreren op de voordelen van het gebruik van UML modelleren bij het onderhouden van software.

Onze aanpak voor het beantwoorden van de onderzoeksvragen in dit proefschrift is empirisch van aard. De gehanteerde onderzoeksmethoden zijn: survey, case study (op basis van interviews), en twee families van experimenten. We hanteerden mixed methods research als aanpak voor het combineren van kwantitatieve en kwalitatieve bevindingen in dezelfde onderzoeksrichting.

Onze experimenten werden uitgevoerd met studenten, terwijl de enquête en case study werden uitgevoerd met professionals, om realistiche meningen van beheerders uit de praktijk te verkrijgen over hoe UML-diagrammen (en documentatie in het algemeen) hen in staat stellen om hun onderhoudstaken uit te voeren.

In essentie is de bijdrage van dit proefschrift tweeledig. Ten eerste levert deze study solide empirisch bewijs over de potentiele voordelen van UML modelleren bij onderhoud van software. De bevindingen van deze study dragen hiermee bij aan de kennis op het gebied van software engineering. Op basis van ons evidence-based onderzoek presenteren we de volgende bevindingen:

- Het gebruik van een grafische notatie (in het byzonder UML) komt veelvuldig voor bij het process van begrijpen van een systeem dat onderhouden moet worden , en bij het ontwerpen van aanpassingen aan het te onderhouden systeem.
- De beslissing om UML te gebruiken hangt sterk samen met het opleidingsniveau en de ervaring van de onderhouds-ontwikkelaars, en hangt ook samen met de grootte van het onderhoudsteam en ook met de grootte van het te onderhouden systeem.

- UML wordt doorgaang gebruikt op een informele manier. Dit wordt ingegeven door het feit dat communicatie en het verkrijgen van een overzicht van het systeem de voornaamste doelen zijn. Een aanvullende verklaring is dat modellen begrijpelijk moeten zijn door alle gebruikers van de documentatie.
- Ontwikkelaars van software onderhoud gebruiken niet altijd de beschikbare documentatie (met of zonder diagrammen), en gebruiken enkel de source code en de daarin opgenomen toelichting. Dit is het gevolg van problemen bij het synchroon houden van de implementatie en de documentatie/modellen.
- Indien geen UML diagrammen beschikbaar zijn bij de initiele ontwikkeling van een software systeem, worden deze zelden gemaakt bij het onderhoud van dat systeem. Het toepassen van reverse engineering voor het verkrijgen van modellen is moeilijk, ondanks de beschikbaarheid van 'reverse engineering' tools, omdat de resulterende modellen handmatig opgeschoond moeten worden. Het gebruik van modellen die gemaakt zijn als onderdeel van het 'forward design' (FD) leiden in enige mate tot verbetering van het onderhoud van de source code. Enkel in het geval dat junior developers software onderhouden werd een hogere efficiency behaald met het gebruik van 'reverse engineered' (RE) diagrammen. Een mogelijke verklaring hiervoor is dat reverse engineered diagrammen in hoge mate corresponderen met de source code, zodat onervaren developers hier een voorkeur voor hebben. In de andere gevallen bieden forward designed class diagrammen een aantrekkelijkere balans tussen detail en relevante informatie dan reverse engineered class diagrammen.
- Het te prefereren niveau van detail (LoD) voor het beschrijven van diagrammen hangt af van de beoogde doellen van hun gebruik: er bestaat een lichte voorkeur voor het gebruik van hoog niveau van detail voor het begrijpen van een systeem, maar er is een voorkeur voor een laag niveau van detail voor het onderhouden van een systeem. Ondanks het feit dat ontwikkelaars stelden dat ze meer moeite hadden met het lezen van diagrammen met hoog niveau van detail, beschouwden de meerderheid van hen het waardevol dat enkele elementen van diagrammen in hoog niveau van detail worden weergegeven: attributen en operaties in class-diagrams en formele messages met parameters in sequence diagrammen.
- Enkele vermeende voordelen van het gebruik van UML tijdens onderhoud aan software zijn naar boven gekomen:
  - Bij het gebruik van UML is minder tijd benodigd voor het verkrijgen van een beter begrip van het te onderhouden systeem; het gebruik van UML leidt ook tot verbeterde opsporing van fouten in de software.
  - De reductie in de doorlooptijd van projecten wordt beschouwd as een voordeel van
  - het gebruik van UMLWanneer we kijken naar de voordelen bij de kwaliteit van het software product, dan meent de meerderheid van de software onderhoud-ontwikkelaars dat het gebruik van UML modelleren de kwaliteit van het uiteindelijke product verbeterd.
  - Wanneer we kijken naar de voordelen bij het process van het ontwikkelen van de software, dan melden software-onderhouds developers de volgende

voordelen van het gebruik van UML modellen: verbeterde communicatie, het voorkomen van de verdamping van kennis, het vereenvoudigen van de diagnose van problemen (in het byzonder rondom gedrags-modellen). UML modelleren wordt ook behulpzaam gevonden bij het verbeteren van het ontwerp van het systeem voordat aan de implementatie begonnen wordt (door het vergemakkellijken van peer-reviewing).

De tweede bijdrage van dit proefschrift is gerelateerd aan de beoefening van modelleren. Dit onderzoek geeft aanbevelingen voor 'best practices' bij UML modelleren. Deze aanbevelingen zijn gebaseerd op empirisch onderzoek bij zowel een industriele case studie, als ook op de meningen van experts. Dit onderzoek dient als gids and rechtvaardiging voor het voortzetten van het gebruik en het bijwerken van UML modellen, en voor het behalen van de voordelen van UML modelleren.

Ter afsluiting benadrukken we dat de voor- en nadelen van het gebruik van UML moeilijk te kwantificeren zijn omdat ze niet tastbaar zijn: geen enkel bedrijf houdt bij welke problemen ontstonden door miscommunicatie, door slechte ontwerp-keuzes, door achterstallige- of onbeschikbare documentatie. Een van de redenen dat dergelijke aspecten moeilijk te kwantificeren zijn, is dat modelleren van een van meerdere manieren is voor het bereiken van bepaalde doelstellingen. Om deze reden kunnen fouten niet teruggevoerd worden op een enkele oorzaak.

De verkregen bevindingen zijn een eerste aanzet voor het ontdekken van de contextuele factoren die van invloed zijn op het gebruik van UML bij het onderhoud van software in industriele projecten. Dit onderzoek heeft enkele vragen voor vervolgonderzoek opgelevert: een voorbeeld hiervan is de vraag naar het effect van verschillende UML notaties, waarbij formele en 'box-and-lines' vergeleken worden. Ook geeft dit onderzoek richtingen aan voor het ontwikkelen van tools die het gebruik van UML gedurende het software ontwikkeltraject beter ondersteunen.

De resultaten van dit onderzoek dragen bij aan de kennis over Software Engineering, en stellen software ontwikkelaars in de praktijk in staat om op basis van empirische bevindingen het gebruik van UML modellering te bevorderen en verdedigen. De aanbevelingen van dit proefschrift kunnen worden toegepast in de industriele praktijk van software onderhoud, en hiermee leiden tot verbetering van hun dagelijkse werk en verbetering van de kwaliteit van hun projecten.