Interactive scalable condensation of reverse engineered UML class diagrams for software comprehension

Osman, M.H.B.

Samenvatting

Softwareonderhoud is een van de meest kritieke activiteiten in de softwareontwikkelingscyclus (SDLC). Het bestaan van onderhoudsactiviteiten in software geeft aan dat de software nog steeds operationeel en relevant is. Onderhoud is noodzakelijk om te garanderen dat het systeem aan de veranderende verwachtingen van de gebruikers blijft voldoen. De primaire taak in softwareonderhoud is het verbeteren van fouten en het implementeren van veranderingsverzoeken.

Ongestructureerde code, onvoldoende domeinkennis en ontoereikende documentatie zijn veelvoorkomende problemen in softwareonderhoud. Deze problemen hebben een hoge impact op de prestatie van het onderhoud, omdat ze een significante invloed hebben op het begrijpen van de software. Softwarebegrip probeert kritieke informatie zoals de structuur van het systeem, het gedrag, en de interne en externe interacties van de modules uit te lichten. Onderzoek heeft aangetoond dat het begrijpen van de software de helft van de onderhoudstaak in beslag neemt. Softwaredocumentatie is een van de beste hulpmiddelen voor softwarebegrip. Echter, documentatie is vaak verouderd of in sommige gevallen niet beschikbaar.

Het achterhalen van softwarearchitectuur of softwarestructuur vanaf de implementatiecode is een gevestigd onderzoeksveld in softwareontwikkeling, genaamd Reverse Engineering. Tegenwoordig bieden diverse CASE-hulpmiddelen reverse-engineeringmogelijkheden die beloven de systeemarchitectuur terug te halen uit de broncode en deze te representeren in UML. Echter, het begrijpen van de resulterende UML-modellen is nogal ingewikkeld, omdat de gereverse-engineerde UML-diagrammen een enorme hoeveelheid informatie bevatten.

Gemotiveerd door het bovenstaande, richt dit onderzoek zich erop om een raamwerk te construeren dat het begrip van software ondersteunt door static analysis reverse engineering. Met enquêtes hebben we informatie van professionele softwareontwikkelaars verzameld over welke aspecten van een ontwerp zij indicatoren vinden voor het belang van informatie in softwareontwikkeling (wanneer deze gerepresenteerd worden met UML-klassendiagrammen).

De vondsten van het onderzoek leidden ons ertoe om objectgeoriënteerde ontwerpmetrieken en tekstmetrieken voor dit doel te gebruiken. Classificatiealgoritmes uit Machine Learning worden toegepast om een waarde te geven aan het belang van een klasse in het softwareontwerp, gebaseerd op bovenstaande metrieken. Deze machine-learningalgoritmes produceren een rangorde van klassen die de centrale informatie vormt voor het versimpelen van het gereverse-engineerde klassendiagram.

Uit de literatuur is het bekend dat verschillende belanghebbenden verschillende voorkeuren hebben voor het bestuderen van een systeem. In het bijzonder willen ze het systeem kunnen zien op meerdere niveaus van abstractie. Om dit te bewerkstelligen ondersteunt onze benadering het genereren van meerdere lagen op een schaalbare manier, dat de gebruiker in staat stelt om klassendiagrammen te bestuderen van hoog niveau van overzicht (met een klein aantal klassen) tot een gedetailleerder niveau (met een veel groter aantal klassen). Met het door ons geautomatiseerde raamwerk wordt het begrip van softwarestructuur van automatisch gereverse-engineerde klassendiagrammen praktischer. Onze oplossing draagt op deze manier bij aan het verbeteren van de efficiëntie in softwareonderhoud.