Onderzoeksvoorstel

Proof of concept: De update automatiseren van Angular versie 16 naar versie 20 in de applicaties van een financiële instelling.

Bachelorproef, 2025-2026

Wauters Sander

E-mail: sander.wauters@student.hogent.be

Samenvatting

Het overschakelen naar een nieuwere versie van een framework kan veel tijd in beslag nemen, vooranamelijk als de huidige applicatie meerdere versies achter loopt. Dit onderzoek heeft hierdoor als doel de update, van het web framework Angular van versie 16 naar versie 20, te automatiseren zonder de functionele of niet-functionele vereisten te schenden. Framework updates kunnen de performantie en veiligheid van de applicatie verbeteren, dus is het van belang deze uit te voeren. Deze updates zijn niet altijd even simpel om toe te passen. Functies worden aangepast of vervangen, naamgevingen veranderen, etc. In grotere codebases neemt het process van systematisch alle veranderingen aan te brengen veel tijd in beslag. Dit onderzoek zal nagaan op welke verschillende manieren deze aanpassingen automatisch kunnen toegepast worden. Het automatisatie process zal getest worden binnen een bedrijf dat beschikt over meerdere applicaties waarvan Angular van versie 16 naar versie 20 overgeschakel moet worden. De effectiviteit wordt beoordeeld aan de hand van het aantal veranderingen het process correct kan uitvoeren ten opzichte van het totaal aantal uit te voeren aanpassingen. Er wordt verwacht dat 75% van alle nodige veranderingen automatisch uitgevoerd kunnen worden. Verder verwacht dit onderzoek dat het automatisatie process uitgebreid kan worden voor gebruik bij de volgende versies van Angular mits de nodige aanpassingen.

Keuzerichting: Mobile & Enterprise development
Sleutelwoorden: Angular, Static code analysis

Inhoudsopgave

1. Inleiding

Het bedrijf Stater is een end-to-end dienstverlener voor zowel hypothecaire en consumentenkredieten, ze ondersteunen de kredietverstrekker voor de dienstverlening aan consumenten. Binnen het bedrijf zijn er verschillende applicatie dat gebruik maken van het Angular framework. Angular is een open-source front-end framework, gebaseerd op de TypeScript programmeertaal, dat gebruikt wordt voor de ontwikkeling van dynamische web applicaties. Momenteel is Angular versie 20 (v.20) de meest recente stabiele versie. Binnen Stater maken de applicaties gebruikt van Angular versie 16 (v.16), maar het bedrijf is van plan de applicaties te updaten naar de recentste versie, Angular v.20.

Software updates introduceren nieuwe functionaliteiten, kunnen de performantie van de applicatie verbeteren en verzekeren dat de software compatibel blijft met nieuwe software and hardware. Verder is het up-to-date houden van software cruciaal voor de cyberveiligheid te garanderen.

Vanwege het grote verschil in versies zal het updaten van alle applicaties veel tijd in beslag nemen. Hieruit komt de vraag: Is het mogelijk om een applicatie in Angular v.16 automatisch te updaten naar Angular v.20? Voor deze vraag te beantwoorden worden volgende deelvragen geformuleerd:

- Wat zijn de veranderingen tussen Angular v.16 en Angular v.20?
- Welke van deze veranderingen kunnen automatisch uitgevoerdt worden zonder de functionele en niet-functionele vereisten van de applicatie in drang te brengen?
- Wat zijn de manieren om code automatisch aan te passen?
- Welke manier(en) is meest geschikt voor toe te passen in deze context?

Gedurende dit onderzoek zal een applicatie ontwikkeld worden dat een software project in Angular v.16 automatisch update naar Angular v.20. In de rest van dit document wordt naar deze applicatie verwezen als de “updater”. De updater doorloopt de broncode van een applicatie en maakt een oplijsting van alle nodige aanpassingen en tracht de aanpassing zelf uit te voeren indien mogelijk. Voor de effectiviteit van de updater te meten, worden de uit te voeren updates opgedeeld in verschillende categoriën en wordt gemeten hoeveel van de nodige updates automatisch uitgevoerdt kunnen worden.

In de volgende sectie wordt een kort overzicht gegeven van de huidige stand van zaken binnen het probleem en oplossingsdomein. Hierna volgt een beschrijving van de methodologie waar de werking en evaluatie van de updater in meer detail beschreven worden. En tenslotte worden de verwachte resultaten besproken, waarin een inschatting wordt gegeven naar de bevindingen van het onderzoek.

2. Literatuurstudie

2.1. Angular versies

De **Angular update guide<empty citation>** geeft ons een compleet overzicht van alle aanpassingen die moeten gebeuren voor de applicatie tot versie 20 te updaten. Deze updates kunnen onderverdeeld worden in volgende categoriën:

- Packages updaten naar een nieuwe versie.
- Toevoegen van nieuwe configuraties.
- Vervangen van verouderde functionaliteiten.
- Hernoemen van functies en variabelen.
- Verwijderen van verouderde functies en variabelen.

2.2. Automatisatie process

De eenvoudigste manier om code in bulk aan te passen is het gebruik maken van regex gebaseerde find en replace tools. Dit is een snelle en makelijke manier om code aan te passen zonder dat er nood is aan speciale tooling. Regex is text gebaseerd en kan dus geen rekening houden met de semantiek van de programmeertaal. Als gevolg kunnen aanpassingen ongewenste gevolgen met zich meebrengen.

Een andere manier is abstract syntax tree (AST) gebaseerd refactoren. Een AST geeft informatie over de semantiek van de programmeertaal. Dit laat ons toe om een stuk code aan te passen enkel als het in een bepaalde scope zit. Voor het opstellen van een AST bestaan er verschillende open-source programmas. Een nadeel aan deze manier is dat de abstract syntax tree op de correcte manier uitgelezen moet worden om correcte aanpassingen te maken.

Nog een alternatief is om gebruik te maken van compiler tooling. Angular is gebaseerd op TypeScript en moet gecompileerd worden naar JavaScript. Compilers komen ingebouwd met de kennis van de verschillende scopes in een programmeertaal. Een nadeel aan deze aanpak is dat het uitbreiden van een compiler een complexe taak is.

Een ander optie is door Language Server Protocol (LSP) integratie. LSP’s komen met programmeertaal specifieke tools zoal: code completion, syntax highlighting, het markeren van warnings en errors, etc. In het begin van 2020 zijn LSP’s de “norm” geworden voor de implementatie van inteligente programmeertaal tools. Deze tools kunnen makelijk uitgebrijd worden door middle van script. Een nadeel is dat deze aanpak gelimiteerd is aan de functies van de LSP.

Tenslotte is het mogelijk om AI tools in te zetten voor deze aanpassingen te maken. Met de recente opkomst van AI tools dat specifiek gemaakt zijn voor programmeren is het mogelijk om deze taak uit te besteden aan deze tools. Het probleem met deze aanpak is dat AI tools niet 100% betrouwbaar zijn.

3. Methodologie

3.1. Literatuurstudie

Dit onderzoek start met een uitgebreide literatuurstudie naar de verschillen tussen Angular v.16 en Angular v.20. De nodige verandering worden opgelijst en onderverdeeld in verschillende categorieën. Deze lijst zal gebruikt worden voor de capaciteiten van de updater te bepalen.

Verder wordt onderzocht wat de verschillende manieren zijn om automatisch code te updaten. Eén of meerdere manieren worden verkozen om te implementeren op basis van volgende parameters:

- De complexiteit van de implementatie. Een voorkeur wordt gegeven aan het gebruik maken van bestaande tools over het ontwikkelen van nieuwe algorithme.
- De betrouwbaarheid van de output. Is het mogelijk om een correct configuratie een incorrecte output te krijgen?

Deze literatuurstudie neemt één tot twee weken in beslag en het resultaat dient als basis voor de volgende fase van het onderzoek.

3.2. Ontwikkeling

In deze fase van het onderzoek zal de updater ontwikkeld worden op basis van de voorafgaande literatuurstudie.

De updater heeft drie functies: het detecteren van aanpassingen in de broncode, het evalueren of deze aanpassingen automatisch kunnen uitgevoerd worden, en de aanpassingen uitvoeren indien mogelijk. Elke gedetecteerde aanpassing wordt geregistreerd, waaronder de locatie in de broncode, de aard van de wijziging, de categorie, en of de updater in staat is om de wijziging automatisch uit te voeren.

De updater wordt in een gecontroleerde omgeving getest om de stabiliteit te verzekeren. Voor het ontwikkelen van de applicatie wordt vier tot vijf weken voorzien.

3.3. Evaluatie

De updater wordt uitgevoerd op één van de applicatie binnen Stater. Vervolgens wordt het resultaat van de updater geëvalueerd aan de hand van het aantal uitgevoerde aanpassingen ten opzichte van het totaal aantal gedetecteerde aanpassingen.

4. Verwacht resultaat, conclusie

Op basis van de literatuurstudie en de gehanteerde methodologie verwacht dit onderzoek dat minstens 75% van alle nodige aanpassingen automatisch uitgevoerdt kunnen worden. Het automatisatie process zal naar verwachting de nodige tijd voor de applicaties te updaten verminderen. De hoeveelheid tijd dat in totaal bespaard zal worden zal afhangen van het aantal applicatie en de grote van de broncode dat geupdate moet worden. Het ontwikkelen van de updater heeft uiteraard ook tijd in beslag genomen. Deze oplossing zal wellicht meer tijd in beslag nemen als enkel één kleine applicatie geupdate moet worden.

Of dit de meest effectieve manier is voor deze casus op te lossen is open voor debat. Verder onderzoek kan uitgevoert worden naar de performantie, accuraatheid, en complexiteit van de verschillend implementaties besproken in de literatuurstudie.