

Proof of concept: De update automatiseren van Angular versie 16 naar versie 20 in de applicaties van een end-to-end kredietdienstverlener.

Wauters Sander
sander.wauters@student.hogent.be
Promotor: Irina Malfait
Co-promotor: Peter De Seranno (Stater)

Hogeschool Gent, Arbeidstraat 14, 9300 Aalst
Keuzerichting: Mobile & Enterprise development
Sleutelwoorden: Angular, TypeScript, Static code analysis, Automatisatie

Samenvatting

Applicatie ontwikkeld om de update van Angular-applicaties te ondersteunen. Getest door een Angular applicatie van v16 naar v20 te updaten. De evaluatie toont aan dat meer dan 25% van alle nodige aanpassingen automatiseerbaar is. Deze aanpak is bruikbaar voor het refactoren van verschillende TypeScript-applicaties. De oplossing is gebaseerd op de TypeScript-compiler voor een betrouwbare error detectie.

Introductie

Het Angular framework vereenvoudigt het ontwikkelingsproces voor het bouwen van dynamische webapplicaties. Zoals bij de meeste software ontvangt Angular regelmatig updates. Deze updates zijn noodzakelijk, omdat ze de cyberveiligheid verbeteren. Het toepassen van dergelijke updates is echter niet altijd vanzelfsprekend.

Angular verwijdert verouderde functionaliteiten uit het framework. Daardoor moet de broncode van een Angular-applicatie aangepast worden tijdens de update. Deze soort updates vinden plaats om de 6 maanden. Bij meerdere enterprise-applicaties kan de benodigde tijd voor dit updateproces snel oplopen. Het bedrijf Stater ervaart dit probleem. Stater is een end-to-end dienstverlener voor zowel hypothecaire als consumentenkredieten. Ze moeten al hun applicaties updaten van Angular v16 naar v20. De sprong van vier versies betekent dat er vermoedelijk veel aanpassingen in de broncode nodig zijn.

Methodologie

Deze studie begon met een literatuurstudie naar gekende methodes om code automatisch aan te passen. De beslissing werd genomen om een *updater* te maken op basis van de TypeScript Compiler API. Door de TypeScript-compiler programmatisch aan te spreken en te koppelen aan zoek- en vervangfuncties was het mogelijk om gericht code op te sporen en te veranderen.

Een collectie aan helperfuncties is ontwikkeld om de implementatie te vereenvoudigen. Door een programmatische instelling behalen we een hoge flexibiliteit met de updater. De updater zelf is een commandline-applicatie. Op deze manier kan het op meerdere projecten opgeroepen worden. Het volgende codefragment toont een voorbeeld van hoe we alle instanties van een functie gericht van naam kunnen veranderen doorheen meerdere applicaties.

```
1 const project = loadProject();
2 project.getSourceFiles().forEach((file) =>
3   findNodes(
4     file,
5     (node) =>
6       deepestInstanceOf(node, "async") &&
7       node.getKind() !== SyntaxKind.AsyncKeyword,
8     (node) => node.replaceWithText("waitForAsync"),
9   ),
10 );
11 await saveProject(project);
```

Onderzoeksvragen

In welke mate kan de automatisering van het updateproces van Angular v16 naar v20, bij meerdere applicaties, de onderhoudstijd voor de ontwikkelaars verlagen?

Deelvragen:

- Welke veranderingen moeten uitgevoerd worden om Angular van v16 naar v20 te updaten?
- Welke manieren bestaan er om code automatisch aan te passen zonder ongewenste veranderingen uit te voeren?
- Welke manier om code automatisch aan te passen is het meest geschikt om in deze casus toe te passen?
- Wat zijn statistisch gezien de meest voorkomende problemen bij het updaten van code?

Conclusies

De testresultaten tonen aan dat de updater een meerwaarde biedt in de ondersteuning van het updateproces. Het is mogelijk om een vierde van alle aanpassingen automatisch uit te voeren. De hoge flexibiliteit van onze aanpak maakt het mogelijk om de updater te herconfigureren voor toekomstige updates. Bovendien is de updater niet gelimiteerd aan het uitvoeren van Angular-updates. Dezelfde manier van werken kan toegepast worden om meer algemene refactoringen uit te voeren. Verder kan men de updater gebruiken op andere TypeScript-applicaties.

Categorie	Totaal		TypeScript		Testen		Syntax		Semantiek	
#Stappen	80	100,00%	50	100,00%	16	100,00%	24	100,00%	38	100,00%
Automatiseerbaar										
Volledig	22	27,50%	10	20,00%	3	18,75%	9	37,50%	1	2,63%
Deels	8	10,00%	8	16,00%	0	0,00%	5	20,83%	5	13,16%
Niet	50	62,50%	32	64,00%	13	81,25%	10	41,67%	32	84,21%
Detecteerbaar										
Volledig	24	30,00%	24	48,00%	6	37,50%	12	50,00%	13	34,21%
Deels	5	6,25%	5	10,00%	0	0,00%	2	8,33%	4	10,53%
Niet	51	63,75%	21	42,00%	10	62,50%	10	41,67%	21	55,26%

Resultaten van de updater om een Angular-applicatie te updaten van v16 naar v20.

Toekomstig onderzoek

Dit onderzoek focust op één van meerdere automatisatietechnieken. Een vergelijking van deze technieken voor toepassing in andere casussen kan best interessant zijn. Wanneer zou een integratie met een language server, of met AI, gepast zijn bijvoorbeeld?

Tijdens het schrijven van dit onderzoek is Angular v21 uitgekomen. Deze versie komt met nieuwe tools om AI beter te integreren in het ontwikkelingsproces. Voornamelijk beweren ze dat het AI toelaat om de nieuwste functionaliteiten te gebruiken. Deze tools zijn momenteel nog experimenteel, maar kunnen veelbelovend zijn.