Optimalisatie van objectgeneratie en geavanceerde codeanalyse voor visualisatie via TIA Portal Openness

Student: Vandenberghe Florian,

Promotoren: Lesage Wannes, Sweertvaegher Isabel

In samenwerking met: Actemium

Academiejaar 2023- 2024

I. Inleiding

A. *Actemium*

Actemium is een onderdeel van de Vinci Energies-groep en biedt wereldwijd advies, ontwerp en onderhoud voor industriële toepassingen.

B. *Actemium object manager*

Actemium Object Manager (AOM) is een tool ontwikkeld door Actemium voor het opzetten van PLC-projecten. In deze projecten worden objecten gebruikt om verschillende componenten, zoals motoren, ventielen en kleppen aan te sturen. Elk object bevat de programmeercode en bijbehorende I/O-instellingen van een component. AOM helpt automatiserings-ingenieurs bij het genereren van deze objecten en het toevoegen van de juiste I/O-instellingen. Door templates te configureren, kan AOM efficiënt code genereren voor alle objecten van een specifiek type. De objecten worden vervolgens handmatig gekopieerd vanuit de AOM naar het PLC-project.

C. *TIA Portal Openness*

TIA Portal Openness is een Application Programming Interface (API) die taken in TIA Portal automatiseert. Dit wordt mogelijk gemaakt door middel van Openness DLL-bestanden. Deze DLL-bestanden bieden een gestandaardiseerde manier om code te delen tussen verschillende toepassingen en stellen ontwikkelaars in staat om functionaliteiten van TIA Portal uit te voeren via andere software.

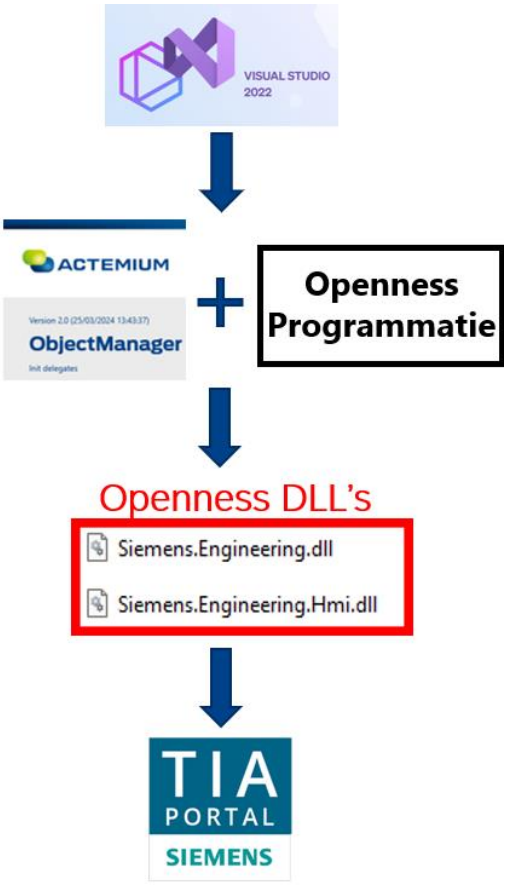
II. Doelstellingen

De masterproef heeft als doel de communicatie tussen AOM en TIA Portal te automatiseren. In een voorstudie worden verschillende tools onderzocht die de communicatie tussen AOM en TIA Portal efficiënt en gebruiksvriendelijk maakt. Hoewel de focus tijdens de voorstudie op TIA Portal Openness ligt, worden ook andere tools, zoals TIA Portal Add-Ins, OpennessScripter, Simatic Modular Application Creator, onderzocht. Het hoofddoel is om objecten vanuit AOM automatisch te genereren in een TIA Portal-project. Als hierna in AOM de objecten nog aangepast worden, moeten deze in TIA Portal ook geüpdatet worden. Het is van belang om te voorkomen dat objecten uit TIA Portal worden overschreven, aangezien deze mogelijk aanvullende aanpassingen van de programmeur bevatten die behouden moeten blijven. Daarnaast wordt het uitlezen van interlock-voorwaarden van objecten onderzocht om de operators te helpen bij het stellen van een betere diagnose tijdens een productieproces.

III. Resultaten

Hoewel alle tools handige functies bieden die relevant kunnen zijn voor de masterproef, steekt TIA Portal Openness er bovenuit. Omdat AOM ontwikkeld is in Visual Studio, kan Openness

rechtstreeks worden geïntegreerd, waardoor op een efficiënte manier een communicatiebrug ontstaat tussen AOM en TIA Portal (figuur 1).

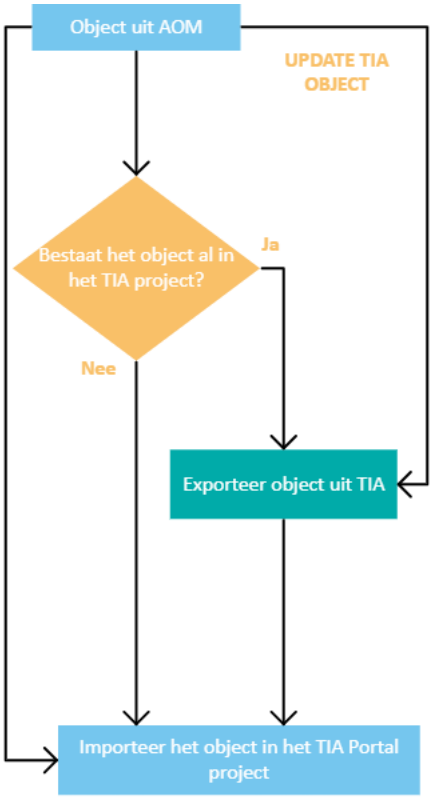


Figuur 1: Communicatie tussen AOM en TIA Portal door middel van TIA Portal Openness.

TIA Portal Openness maakt gebruik van het objectenmodel om acties op specifieke locaties binnen het TIA Portal project uit te voeren. Het is daarom essentieel om het objectenmodel grondig te bestuderen, omdat tijdens Openness-acties het model correct moet worden doorlopen om de gewenste actie te bereiken.

TIA Portal Openness biedt veel functies, waaronder het opstarten en laden van projecten, importeren en exporteren van bestanden met objectinformatie van de AOM. Het importeren van objecten gebeurt in twee stappen: eerst wordt het bestand toegevoegd als externe bron, waarna het als externe bron in het TIA Portal-project gegenereerd wordt. Deze versnellen het proces, maar zijn nog altijd afhankelijk van de verwerkingstijd van TIA Portal. Andere interessante functies omvatten exporteren van datablokken voor vergelijking van waarden op verschillende tijdstippen en codevergelijking voor offline/offline en offline/online situaties [1].

Bij het genereren van objecten worden verschillende stappen doorlopen (figuur 2). Eerst wordt gecontroleerd of het object al bestaat in TIA Portal. Indien niet, wordt het direct vanuit AOM geïmporteerd via een overeenkomstige Openness-actie. Indien wel, wordt het eerst geëxporteerd en vergeleken met het object uit AOM. De verschillen worden toegevoegd aan het TIA-object, waardoor het object wordt bijgewerkt zonder dat andere aanpassingen verloren gaan. Vervolgens wordt het bijgewerkte object geïmporteerd.

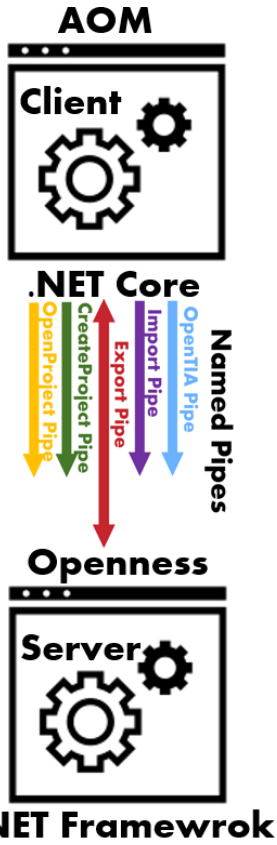


Figuur 2: Flowchart van objectgeneratie.

Het importeren en exporteren van objecten gebeurt via SCL-bestanden, waarbij de bijbehorende tagtabellen in XML-formaat zijn. Deze formaten worden bestudeerd en de nodige headers ontwikkeld, zodat TIA Portal de bestanden effectief kan inlezen tijdens het importeren. Bij het exporteren en bijwerken van de objecten wordt ook rekening gehouden met deze headers.

Tijdens het updaten wordt telkens het object uit TIA Portal naast het object uit AOM gelegd. Door middel van reguliere expressies [2] wordt een specifieke regel code uit het AOM-object gezocht in het TIA-object, bij een verschil wordt die regel code vervangen. Op deze manier wordt het object in TIA Portal bijgewerkt zonder volledig te worden overschreven. Verder worden deze reguliere expressies ook gebruikt om de interlockvoorwaarden te achterhalen. Voor een specifieke voorwaarde wordt naar de bijhorende interlockvoorwaarde gezocht. Op basis van de waarde wordt de SCADA aangepast zodat het de interlockvoorwaarde weergeeft.

In Visual Studio worden zowel .NET Framework als .NET Core ondersteund voor het bouwen van Windows-toepassingen, waarbij .NET Core een open-source en cross-platform is. De Actemium Object Manager (AOM) is ontwikkeld in Visual Studio met .NET Core, maar ondervindt problemen met het uitvoeren van Openness-acties omdat TIA Portal Openness alleen .NET Framework ondersteunt. Om dit op te lossen wordt bij het starten van de AOM een tweede applicatie op de achtergrond gestart in .NET Framework (Openness-applicatie), die via Named Pipes communiceert met de AOM om Openness-acties uit te voeren. Named Pipes is een inter-procescommunicatie (IPC) systeem waarbij de Openness applicatie als server dient en de AOM als client [3]. De Openness applicatie creëert een server voor elke Openness-actie, waarna de AOM-client verbinding kan maken met de specifieke serverpipe voor de gewenste Openness-actie. Nadat de verbinding is opgezet, stuurt de AOM het startsignaal en de benodigde gegevens naar de server, die vervolgens de Openness-actie uitvoert en feedback geeft aan de AOM (figuur 3).



Figuur 3: Communicatie tussen Openness en AOM door middel van Named Pipes.

IV. Besluit

Ondanks de complexiteit van TIA Portal Openness is het gelukt om automatisch objecten uit AOM te genereren in TIA Portal. Bij latere aanpassingen in AOM kunnen deze wijzigingen nu ook op een slimme manier worden toegepast op de objecten in TIA Portal, waardoor efficiënter en foutloos kan worden gewerkt. Daarbij worden de interlockvoorwaarden van de objecten gevonden en in de database geplaatst. Hoewel problemen zoals de compatibiliteit van TIA Openness ontstonden, is Named Pipes ontwikkeld om deze aan te pakken. Dit bracht weliswaar extra complexiteit met zich mee, maar het correcte gebruik ervan heeft tot succes geleid.

V. Referenties

[1] Siemens, „TIA Portal Openness,” 25 06 2021. [Online]. Available: https://support.industry.siemens.com/cs/document/109792902/tia-portal-openness-automation-of-engineering-workflows?dti=0&lc=en-BE.

[2] ByteHide, „From Zero To Hera Guide,” 21 05 2023. [Online]. Available: https://www.bytehide.com/blog/regex-csharp.

[3] M. Chinta, „Named pipes in .NET(C#),” 27 01 2024. [Online]. Available: https://medium.com/codenx/named-pipes-in-net-c-c0459e165371.