

Onderhoudbare detectie- en vergelijkingsketen voor in-house software in EPC en WSIT/CMDB

Projectplan

Quinten De Meyer

Student Bachelor in de Elektronica-ICT — Cloud & Cyber Security

Stagebedrijf: Pfizer Puurs — ICS Infrastructure

Stagecoach Thomas More: Cas Magnus

Stagementor Pfizer: Nikki Lissens

Inhoudsopgave

Verantwoording Actualisering Projectplan	4
1. Begrippenlijst	4
2. Context en probleemstelling	4
Bedrijfscontext	4
Probleemstelling	4
Reeds uitgevoerde voorbereiding	5
3. Centrale onderzoeksvraag.....	5
Deelvragen	5
4. Doelstelling.....	5
5. Diagram.....	6
6. Scope	6
Binnen de scope.....	6
Buiten de scope	6
7. Stakeholders, waardepropositie en business case.....	7
Stakeholders.....	7
Waardepropositie	7
Business case.....	7
<i>Aanpak en methodologie</i>	<i>7</i>
Software Development Life Cycle (SDLC)-fasen en tijdlijn.....	7
<i>FASE 1 – Analyse (Week 1–2).....</i>	<i>7</i>
<i>FASE 2 – Functioneel en technisch ontwerp (Week 3)</i>	<i>8</i>
<i>FASE 3 – Opbouw van bronstructuur en gegenereerde detectieprofielen (Week 4–5)</i>	<i>8</i>
<i>FASE 4 – Implementatie van endpointdetectie (Week 6–7).....</i>	<i>8</i>
<i>FASE 5 – Vergelijking met de CMDB (Week 8–9).....</i>	<i>9</i>
<i>FASE 6 – CMDB-updateproces (Week 10–11)</i>	<i>9</i>
<i>FASE 7 – Stabilisatie en optimalisatie (Week 12).....</i>	<i>10</i>
<i>FASE 8 – Oplevering (Week 13).....</i>	<i>10</i>
Test- en validatieaanpak	10
<i>Validatie van bronbestand</i>	<i>11</i>
<i>Validatie van generatie naar JSON-profielen.....</i>	<i>11</i>
<i>Testen van detectie op endpoints.....</i>	<i>11</i>
<i>Beoordeling van detectieresultaten</i>	<i>11</i>
Evaluatiekader.....	11
<i>Betrouwbaarheid van detectie.....</i>	<i>11</i>
<i>Onderhoudbaarheid van de bronstructuur.....</i>	<i>11</i>
<i>Overdraagbaarheid van de workflow.....</i>	<i>11</i>
<i>Evaluatie van de vergelijking met WSIT/CMDB</i>	<i>11</i>
8. Planningsoverzicht.....	12

9. Deliverables.....	12
Bedrijfsdeliverables.....	12
<i>Centrale Launcher-menu</i>	12
<i>YAML-bronbestand voor in-house software</i>	12
<i>Software Source Maintainer</i>	12
<i>EXE Reviewer</i>	12
<i>Python-script voor generatie van detectieprofielen</i>	13
<i>JSON-detectieprofielen</i>	13
<i>EPC-detectiebundle (Detection Engine)</i>	13
<i>Result Merger / Normalizer</i>	13
<i>WSIT Assets Export Script</i>	13
<i>Vergelijkingslogica (Comparison Engine)</i>	13
<i>Procedure en script voor CMDB-update (WSIT API Import)</i>	13
<i>Overdrachtsdocumentatie voor collega's</i>	13
Academische deliverables	14
10. Risico's en beperkingen	14
11. Succescriteria	14
12. Samenvattende slotbeschouwing	14
13. Verwijzingen	15

Verantwoording Actualisering Projectplan

Dit projectplan is na afloop van het stageproject geactualiseerd naar de feitelijke eindtoestand. Tijdens de realisatiefase bleek de workflow uitbreiding te behoeven om de beoogde onderhoudbaarheid en overdraagbaarheid te garanderen. Onvoorziene componenten, zoals de Software Source Maintainer, de EXE Reviewer, de Result Merger / Normalizer en de centrale Launcher, zijn gaandeweg toegevoegd. Dit document weerspiegelt de definitieve scope, deliverables en succescriteria van de opgeleverde EPC-CMDB Tool in overeenstemming met de feitelijke realisatie

1. Begrippenlijst

- **WSIT:** interne benaming voor *Workstation and Software Inventory Tool*, gebaseerd op Jira Service Management en Jira Assets.
- **CMDB:** *Configuration Management Database* binnen WSIT/Jira Assets, waarin software en endpoints als objecten worden beheerd in één objectschema.
- **EPC:** Endpoint Central, gebruikt voor endpointbeheer, scriptuitvoering op endpoints en aanvullende beheerfunctionaliteiten.
- **IID:** *Installation Instruction Document*; documentatie waarin installatie-informatie van in-house software wordt beschreven in de vorm van instructies.
- **YAML-bronbestand:** centraal bronbestand waarin in-house software gestructureerd wordt beschreven, bijvoorbeeld installatiepaden, bronmedia, versies, updatebronnen en prerequisites.
- **JSON-detectieprofielen:** gegenereerde JSON-artefacten op basis van het YAML-bronbestand, die door detectiescripts worden gebruikt voor endpointcontrole.
- **In-house software:** intern ontwikkelde of niet-standaardsoftware die niet betrouwbaar door de standaardinventarisatie van Endpoint Central wordt gedetecteerd.
- **Discrepantie:** verschil tussen de software-endpointrelaties in WSIT/CMDB en de feitelijke softwaretoestand op endpoints.
- **Endpointrealiteit:** de feitelijke toestand van software-installaties op een endpoint, vastgesteld door het EPC-detectiescript op basis van JSON-detectieprofielen. Dit begrip wordt in dit project gebruikt als tegenhanger van de administratieve softwareregistratie in WSIT/CMDB.
- **CD-shares:** gedeelde netwerklocaties waarop installatiebestanden van door Pfizer gebruikte software beschikbaar worden gesteld, doorgaans met een cd-nummer in het formaat 'CDxxxx'.
- **WPF:** Windows Presentation Foundation; een grafisch subsysteem van Microsoft voor het weergeven van gebruikersinterfaces in Windows-applicaties.

2. Context en probleemstelling

Bedrijfscontext

Vandaag is de Pfizer Global Supply site in Puurs-Sint-Amunds de grootste Europese productie- en verpakkingsfaciliteit van Pfizer en één van de grootste ter wereld. Hier worden jaarlijks meer dan 400 miljoen units inspuibare geneesmiddelen en vaccins geproduceerd onder strikte kwaliteits- en veiligheidsnormen om ervoor te zorgen dat elke patiënt een betrouwbaar en uitstekend vaccin of geneesmiddel krijgt (Pfizer Belgium, z.d.).

Probleemstelling

Tijdens mijn stage bij Pfizer binnen ICS Infrastructure werd vastgesteld dat de standaardsoftware-inventarisatie in Endpoint Central onvoldoende zicht biedt op in-house software. Voor dergelijke software ontbreken vaak

gestandaardiseerde detectie-indicatoren. Bovendien verschilt de installatieaanpak naargelang de software en de bijhorende installatie-instructies of het formele installatieplan.

Binnen dit project wordt er echter van uitgegaan dat deze instructies correct worden gevolgd. Het kernprobleem ligt daarom niet in een willekeurige of foutieve installatie-uitvoering, maar in het ontbreken van een onderhoudbare en reproduceerbare methode om de aanwezigheid van deze software op endpointniveau betrouwbaar te detecteren. Tegelijk wordt deze software vaak administratief geregistreerd in WSIT/CMDB. Daardoor kunnen verschillen ontstaan tussen de administratieve softwareregistratie en de feitelijke softwaretoestand op endpoints.

Reeds uitgevoerde voorbereiding

In een voorbereidende fase werd bestaande kennis over in-house software verzameld op basis van installatie-instructiedocumenten en installatiemedia. Daarbij werd nagegaan welke software binnen de afgesproken scope relevant is en welke kenmerken bruikbaar kunnen zijn voor detectie op endpointniveau.

Deze voorbereiding leverde een eerste overzicht op van softwarevarianten, installatiepaden, bronmedia, versies en andere relevante kenmerken. Die informatie vormt de basis voor de verdere uitwerking van het project.

De verdere projectuitwerking heeft tot doel deze verzamelde kennis te valideren, te structureren en om te zetten in een onderhoudbare detectie- en vergelijkingsketen.

3. Centrale onderzoeksvraag

Hoe kan een onderhoudbare en overdraagbare workflow worden uitgewerkt waarmee in-house software binnen de afgesproken scope betrouwbaar op endpoints kan worden gedetecteerd, vergeleken met de administratieve softwareregistratie in WSIT/CMDB en semiautomatisch kan worden opgevolgd?

De voorgestelde workflow wordt geëvalueerd aan de hand van het evaluatiekader dat in sectie 6.3 wordt beschreven.

Deelvragen

- Welke softwarekenmerken zijn nodig om interne software betrouwbaar te detecteren?
- Hoe kunnen deze kenmerken onderhoudbaar gemodelleerd worden in een centraal YAML-bronbestand?
- Hoe kunnen deze definities automatisch vertaald worden naar JSON-detectieprofielen voor runtimegebruik?
- Hoe kunnen endpointresultaten op een consistente manier vergeleken worden met WSIT/CMDB-gegevens?
- Welke workflow is haalbaar voor het opvolgen van discrepanties richting WSIT/CMDB?

4. Doelstelling

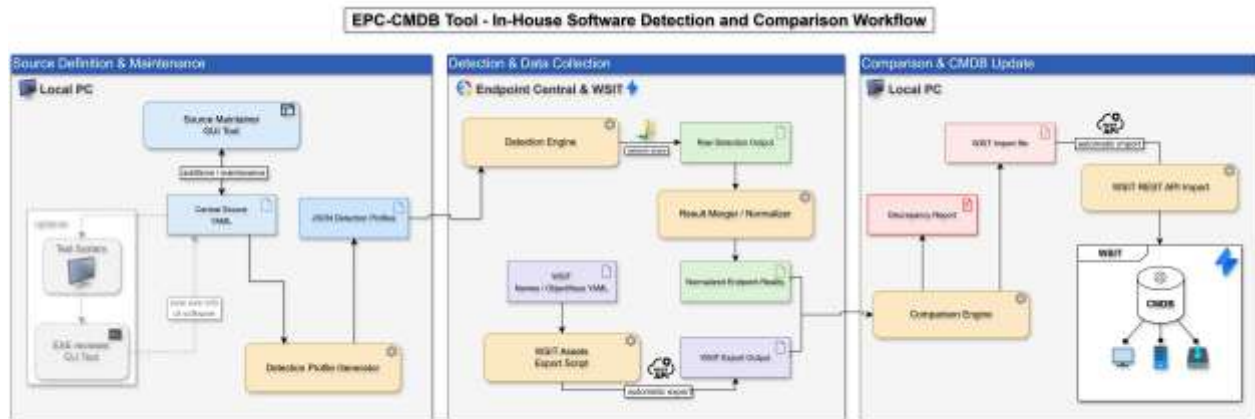
De doelstelling van dit project is het ontwikkelen van een onderhoudbare workflow voor het detecteren van in-house software op endpoints, het vergelijken van de resultaten met WSIT/CMDB en het semiautomatisch opvolgen van discrepanties binnen de afgesproken scope.

Deze doelstelling wordt concreet gerealiseerd via de volgende deelresultaten:

- een centraal en onderhoudbaar YAML-bronbestand waarin in-house software uniform wordt gemodelleerd;
- een Python-script dat deze definities automatisch omzet in JSON-detectieprofielen;
- een EPC-detectiescript dat op basis van deze JSON-profielen de feitelijke softwaretoestand op endpoints bepaalt;
- een vergelijkingsstap waarin endpointresultaten worden afgezet tegen de administratieve softwareregistratie in WSIT/CMDB;
- een semiautomatische procedure om relevante discrepanties op een beheerbare manier verder te verwerken;

- overdrachtsdocumentatie waarmee collega's het bronbestand in de toekomst kunnen uitbreiden zonder de volledige oplossing te herontwerpen.

5. Diagram



6. Scope

Binnen de scope

- in-house software binnen de afgesproken softwarelijst;
- endpoints binnen de afgesproken omgeving en testscope;
- detectie van installaties op basis van vooraf gemodelleerde softwaredefinities;
- vergelijking van EPC-resultaten met WSIT/CMDB-gegevens;
- een onderhoudbare bronstructuur via YAML en JSON-profielen.

Binnen deze scope wordt enkel software opgenomen waarvoor voldoende betrouwbare informatie beschikbaar is om detectiecriteria te definiëren op basis van bronmedia, installatie-informatie en IID's. Onder endpoints binnen de scope worden systemen verstaan waarop EPC-scriptuitvoering haalbaar is en waarop de oplossing functioneel getest en gevalideerd kan worden binnen de afgesproken projectcontext.

Buiten de scope

- universele detectie van alle software op alle endpoints;
- volledig automatische updates in WSIT/CMDB zonder validatie;
- beheer van software buiten de afgesproken scope.

7. Stakeholders, waardepropositie en business case

Stakeholders

Primaire stakeholders

- ICS Infrastructure team
- Beheerders van WSIT/CMDB
- Collega's die in-house softwaredefinities onderhouden

Secundaire stakeholders

- Teams die nood hebben aan correcte software-inventarisatie

Waardepropositie

De meerwaarde van dit project ligt in het verbeteren van de zichtbaarheid en betrouwbaarheid van gegevens over in-house software bij Pfizer Puurs. Door softwarekennis centraal te structureren en detectie te standaardiseren, vermindert de nood aan manuele interpretatie en worden discrepanties tussen endpoints en WSIT/CMDB werkelijk zichtbaar. Daarnaast wordt softwarekennis beter overdraagbaar voor toekomstig beheer.

Business case

De business case van dit project is het verminderen van manueel en foutgevoelig werk bij het beheren van gegevens over in-house software. Vandaag is kennis over in-house software verspreid over installatie-instructies, bronmedia, bestaande systeeminformatie, enkel WSIT en kennis van medewerkers. Daardoor kost het veel tijd om na te gaan welke software werkelijk op endpoints aanwezig is en of die toestand overeenkomt met de administratieve registratie in WSIT/CMDB. Door softwaredefinities te ontwerpen in één YAML-bronbestand en deze te vertalen naar JSON-detectieprofielen, wordt deze kennis herbruikbaar en onderhoudbaar. Het EPC-detectiescript maakt het mogelijk om de feitelijke softwaretoestand op een reproduceerbare manier vast te stellen, zonder dat voor elke controle opnieuw documentatie manueel moet worden nagelezen.

Aanpak en methodologie

Software Development Life Cycle (SDLC)-fasen en tijdlijn

FASE 1 – Analyse (Week 1–2)

Doel

In kaart brengen van het functionele en technische probleem, de gebruikte systemen en de scope van de oplossing.

Activiteiten

- Analyse van WSIT/CMDB, EPC en hun relevante gegevensstromen.
- Analyse van de beperkingen van standaardsoftwaredetectie in EPC voor in-house software.
- Inventarisatie van relevante in-house software binnen de afgesproken scope.
- Analyse van beschikbare installatiemedia en installatie-instructiedocumenten (IID's).
- Bepalen van de minimale en noodzakelijke softwarekenmerken voor detectie.
- Afbakening van endpointscope en softwarescope.

Deliverables

- Scopebeschrijving van software en endpoints
- Analyse van het probleem en de huidige werkwijze
- Eerste lijst met detectiecriteria per software
- Overzicht van relevante databronnen (WSIT, EPC, IID's, installatiemedia)

FASE 2 – Functioneel en technisch ontwerp (Week 3)

Doel

Uitwerken van de technische architectuur van de oplossing en het gegevensmodel voor in-house software.

Activiteiten

- Ontwerpen van de structuur van het YAML-bronbestand voor in-house software.
- Bepalen hoe softwareversies, bronmedia, updatebronnen en prerequisites worden gemodelleerd.
- Ontwerpen van het JSON-detectieprofiel dat door scripts gebruikt zal worden.
- Ontwerpen van de detectielogica op endpointniveau.
- Ontwerpen van de vergelijking tussen EPC-output en WSIT/CMDB-data.
- Ontwerpen van de werkwijze voor semiautomatische CMDB-updates.

Deliverables

- YAML-structuur en datamodel
- Prototype van JSON-detectieprofiel
- Eerste EPC-output en vergelijking

FASE 3 – Opbouw van bronstructuur en gegenereerde detectieprofielen (Week 4–5)

Doel

De manueel verzamelde softwarekennis omzetten in een onderhoudbaar YAML-bronbestand en deze automatisch vertalen naar JSON-profielen.

Activiteiten

- Structureren van softwarekennis in het YAML-bronbestand.
- Implementeren en testen van de generatie van YAML naar JSON-profielen met Python.
- Valideren van de gegenereerde profielen op volledigheid en consistentie.
- Toevoegen van extra detectiemethodes voor robuustere detectie.
- Opruimen van onduidelijke of vage definities waar mogelijk.

Deliverables

- Werkend YAML-bronbestand
- Werkend Python-generatiescript
- Gegenereerde JSON-detectieprofielen
- gevalideerde software detectie indicators

FASE 4 – Implementatie van endpointdetectie (Week 6–7)

Doel

Een werkend EPC-detectiescript implementeren dat op basis van de JSON-profielen die in-house software detecteert op de endpoints.

Activiteiten

- Implementeren van de detectielogica.
- Koppelen van het script aan de JSON-detectieprofielen.
- Voorzien van logging, foutafhandeling en outputnormalisatie.
- Testen op meerdere systemen en in meerdere softwarescenario's.
- Integreren van scriptuitvoering in EPC Custom Scripts.

Deliverables

- Werkend EPC-detectiescript
- Gestandaardiseerde scriptoutput
- rapport met resultaten op endpoints

FASE 5 – Vergelijking met de CMDB (Week 8–9)

Doel

Discrepanties identificeren tussen endpointrealiteit en de administratieve softwareregistratie in WSIT/CMDB.

Activiteiten

- Ophalen of exporteren van relevante softwaregegevens uit WSIT/CMDB.
- Normaliseren van EPC- en WSIT-output zodat vergelijking mogelijk wordt.
- Identificeren van ontbrekende, afwijkende of verouderde softwaregegevens via script.
- Analyseren van de belangrijkste discrepantiepatronen.

Deliverables

- Vergelijkingslogica of -script
- Discrepantierapport
- Analyse van belangrijkste verschillen

FASE 6 – CMDB-updateproces (Week 10–11)

Doel

Een werkbare en beheerbare manier uitwerken om discrepanties vanuit EPC-resultaten te verwerken richting WSIT/CMDB.

Activiteiten

- Evalueren van mogelijke updatepaden (import, CSV, API, ticketproces).
- Uitwerken van een semiautomatische update workflow.
- Testen van de gekozen aanpak op een beperkte set gevallen.
- Documenteren van randvoorwaarden en beperkingen.

Deliverables

- Procedure voor CMDB-update
- Eventueel ondersteunend update-script of exportformaat
- Testresultaten van updateflow

FASE 7 – Stabilisatie en optimalisatie (Week 12)

Doel

De oplossing robuust, onderhoudbaar en overdraagbaar maken. Verder optimaliseren voor snelheid en stabiliteit.

Activiteiten

- Optimaliseren van scripts op het gebied van performantie en betrouwbaarheid.
- Wegwerken van edgecases en inconsistenties in softwaredefinities
- Vereenvoudigen van de workflow voor toekomstige collega's.
- Verbeteren van documentatie en beheerinstructies.
- Eventueel verrijken van brondefinities via ondersteunende discovery.

Deliverables

- Geoptimaliseerde scripts
- Gestructureerd YAML-bronbestand en JSON-profielen
- Onderhoudsinstructies voor collega's

FASE 8 – Oplevering (Week 13)

Doel

De oplossing opleveren, documenteren en overdragen.

Activiteiten

- Finaliseren van de einddocumentatie.
- Voorbereiden van de eindpresentatie.
- Bundelen van alle scripts, profielen en bronbestanden.
- Opleveren van instructies en overdracht voor collega's.
- De effectiviteit van de oplossing aantonen.
- Toelichten van resterende beperkingen en toekomstige uitbreidingen.

Deliverables

- Eindpresentatie
- Definitieve scripts en bronbestanden
- Overdrachtsdocumentatie

Test- en validatieaanpak

Validatie van bronbestand

Vooraleer softwaredefinities gebruikt worden in detectieprofielen, wordt nagegaan of de informatie in het YAML-bronbestand voldoende eenduidig, volledig en intern consistent is. Hierbij wordt gecontroleerd of softwarekenmerken logisch aansluiten bij de beschikbare bronmedia, documentatie en gedefinieerde detectiekenmerken.

Validatie van generatie naar JSON-profielen

De automatisch gegenereerde JSON-detectieprofielen worden gecontroleerd op structurele correctheid, volledigheid en consistentie met het YAML-bronbestand. Daarbij wordt nagegaan of wijzigingen in de brondefinities correct worden overgenomen naar de detectieprofielen.

Testen van detectie op endpoints

Om te verifiëren dat het detectiescript in-house software binnen de afgesproken scope correct detecteert, wordt relevante software manueel geïnstalleerd op een testsysteem volgens de bijhorende IID of het formele installatieplan. Vervolgens wordt het detectiescript eerst lokaal op dit testsysteem uitgevoerd, voordat het via Endpoint Central op endpoints wordt ingezet. Aangezien binnen de projectcontext wordt uitgegaan van installaties die correct volgens IID of plan worden uitgevoerd, vormt deze testaanpak een representatieve validatie voor endpoints binnen scope.

Beoordeling van detectieresultaten

De detectie wordt als voldoende betrouwbaar beschouwd wanneer de oplossing reproduceerbare resultaten oplevert, geen evidente foutpositieven genereert, bewezen kan worden door manuele checks en discrepanties in een interpreteerbaar en uniform formaat zichtbaar maakt voor verdere opvolging. De beoordeling focust dus niet alleen op het correct detecteren van positief geteste software, maar ook op de consistentie en bruikbaarheid van de output binnen de verdere vergelijkings- en opvolgingsworkflow.

Evaluatiekader

Om te beoordelen in welke mate de voorgestelde workflow voldoet aan de centrale onderzoeksvraag, wordt een evaluatiekader gehanteerd dat focust op de betrouwbaarheid, onderhoudbaarheid en overdraagbaarheid van de oplossing. De evaluatie gebeurt aan de hand van een combinatie van technische validatie, functionele testen en kwalitatieve beoordeling binnen de projectcontext.

Betrouwbaarheid van detectie

De detectie van in-house software geldt als betrouwbaar wanneer het EPC-detectiescript op basis van de gegenereerde JSON-detectieprofielen reproduceerbare resultaten oplevert op meerdere endpoints binnen scope. Dat betekent dat identieke installatietoestanden bij herhaalde scriptuitvoering leiden tot consistente detectieresultaten, zonder evidente foutpositieven in negatieve testgevallen. De betrouwbaarheid wordt geëvalueerd door testinstallaties die correct volgens de relevante IID of het formele installatieplan zijn uitgevoerd, te vergelijken met de gegenereerde EPC-output. Indien de detectie onder deze voorwaarden correct werkt, mag worden verwacht dat dezelfde detectielogica ook correct functioneert op endpoints binnen scope waar dezelfde instructies correct werden gevolgd.

Onderhoudbaarheid van de bronstructuur

De oplossing wordt als onderhoudbaar beschouwd wanneer nieuwe software of wijzigingen aan bestaande softwaredefinities kunnen worden doorgevoerd door enkel het YAML-bronbestand aan te passen, zonder wijzigingen aan de detectiescripts zelf. De onderhoudbaarheid wordt geëvalueerd door tijdens het project meerdere iteraties van softwaredefinities door te voeren en te verifiëren dat deze correct doorstromen naar JSON-detectieprofielen en endpointdetectie.

Overdraagbaarheid van de workflow

De workflow wordt als overdraagbaar beschouwd wanneer collega's op basis van de documentatie het YAML-bronbestand kunnen aanpassen, de JSON-profielen opnieuw kunnen genereren en de detectie- en vergelijkingsketen kunnen uitvoeren zonder kennis van de interne script werking. Dit wordt beoordeeld op basis van de volledigheid en duidelijkheid van de overdrachtsdocumentatie en de mate waarin handmatige tussenstappen beperkt blijven.

Evaluatie van de vergelijking met WSIT/CMDB

De vergelijking tussen de endpointrealiteit en de administratieve softwareregistratie in WSIT/CMDB wordt als voldoende beschouwd wanneer discrepanties op een uniforme en interpreteerbare manier worden gerapporteerd, zodat verdere

opvolging mogelijk is. De evaluatie focust hierbij op de consistentie van de vergelijking en de bruikbaarheid van de output voor semiautomatische verwerking.

Op basis van deze evaluatiecriteria wordt in het besluit beoordeeld in welke mate de voorgestelde workflow een onderhoudbare en overdraagbare oplossing vormt voor het detecteren en opvolgen van in-house software binnen de afgesproken scope.

8. Planningsoverzicht

Week	Fasebeschrijving
1–2	Analyse van probleem, systemen, scope en databronnen
3	Functioneel en technisch ontwerp van de oplossing
4–5	Opbouw van het YAML-bronbestand en generatie van JSON-profielen
6–7	Implementatie van EPC-endpointdetectie
8–9	Vergelijking van EPC-resultaten met WSIT/CMDB
10–11	Uitwerking van het CMDB-updateproces
12	Stabilisatie, optimalisatie en onderhoudbaarheid
13	Oplevering, presentatie en overdracht

9. Deliverables

Bedrijfsdeliverables

Centrale Launcher-menu

Een overkoepelende, interactieve PowerShell-gebruikerslaag waarmee de volledige workflow consistent en opeenvolgend kan worden uitgevoerd en gecontroleerd.

YAML-bronbestand voor in-house software

Een onderhoudbaar bronbestand waarin in-house software gestructureerd wordt beschreven, inclusief installatiemedia, versies, updatebronnen, prerequisites en relevante detectie-informatie.

Software Source Maintainer

Een grafische WPF-desktopapplicatie (C#) waarmee beheerders het YAML-bronbestand foutloos en zonder syntaxkennis kunnen onderhouden.

EXE Reviewer

Een PowerShell-terminaltool uit te voeren op het endpoint zelf voor het scannen van bronpaden en het gericht classificeren van uitvoerbare bestanden ten behoeve van het bronbestand.

Python-script voor generatie van detectieprofielen

Een script dat het YAML-bronbestand omzet in een gestandaardiseerd JSON-profiel voor runtimegebruik.

JSON-detectieprofielen

De gegenereerde JSON-profielen die door detectiescripts op endpoints worden gebruikt.

EPC-detectiebundel (Detection Engine)

Een modulair, read-only Python-detectiescript inclusief geoptimaliseerde Operating System (OS)-specifieke runtimes en een PowerShell Upload Bundle Builder voor distributie via Endpoint Central.

Result Merger / Normalizer

Een transformatiecomponent die asynchrone endpointresultaten consolideert tot één uniform overzicht van de endpointrealiteit.

WSIT Assets Export Script

Een PowerShell-script dat de actuele administratieve software-endpointrelaties via REST API uit Jira Assets extraheert.

Vergelijkingslogica (Comparison Engine)

Een analytische module die de endpointrealiteit confronteert met de CMDB-export en een HTML-discrepantie-rapport inclusief CSV-doelstaat genereert.

Procedure en script voor CMDB-update (WSIT API Import)

Een REST API-importmodule die de berekende doelstaat (inclusief dry-run simulatie) semiautomatisch wegschrijft naar Jira Assets.

Overdrachtsdocumentatie voor collega's

Een gestructureerde Confluence-paginastructuur met gedetailleerde handleidingen en instructies per processtap.

Academische deliverables

- projectplan;
- technische documentatie van de volledige keten;
- handleiding voor het aanvullen en onderhouden van het YAML-bronbestand;
- documentatie van scriptuitvoering in testomgeving, lokaal en via EPC;
- eindpresentatie.

10. Risico's en beperkingen

- Niet alle in-house software is even goed of op eenzelfde manier gedocumenteerd in IID's of bronmedia;
- Voor oude versies van sommige software is geen IID beschikbaar;
- Niet alle software is op identieke wijze detecteerbaar, bijvoorbeeld door verschillen in versie, test-, validatie- of productieomgeving;
- API's, exports of updatekanalen kunnen beperkt zijn;
- De kwaliteit van de oplossing is afhankelijk van de kwaliteit van de softwaredefinities in het YAML-bronbestand.

11. Succescriteria

Het project wordt als succesvol beschouwd wanneer aan de volgende criteria is voldaan:

- Voor alle in-house software binnen scope bestaan onderhoudbare definities in het YAML-bronbestand, dat via een grafische interface (Software Source Maintainer) zonder syntaxfouten beheerd kan worden.
- Brondefinities kunnen zonder handmatige tussenstappen via de launcher worden omgezet in JSON-detectieprofielen.
- De core detectie-engine voert scans read-only uit binnen een geoptimaliseerde runtime van gemiddeld minder dan 20 seconden per endpoint, zonder foutpositieven te genereren.
- Asynchrone endpointdata wordt automatisch geconsolideerd en genormaliseerd tot een uniform brondocument.
- Discrepancies tussen de endpointrealiteit en de administratieve registratie in de CMDB worden overzichtelijk gerapporteerd in een HTML-impactrapport.
- Geïdentificeerde afwijkingen kunnen via een veilige simulatiemodus (*dry-run*) worden gevalideerd en via een REST API-importscript semiautomatisch worden doorgevoerd in Jira Assets.
- Toekomstige beheerders kunnen de volledige keten end-to-end aansturen via de centrale launcher en het bronbestand uitbreiden op basis van de Confluence-documentatie.

12. Samenvattende slotbeschouwing

Dit projectplan beschrijft een onderhoudbare werkwijze voor het detecteren van in-house software op endpoints en het vergelijken van deze resultaten met de administratieve registraties in WSIT/CMDB. Door softwaredefinities centraal te modelleren en detectie te standaardiseren, wil het project de zichtbaarheid van in-house software verbeteren en discrepanties beter opvolgbaar maken.

De verwachte meerwaarde ligt in een reproduceerbare en overdraagbare tool die het beheer van in-house software ondersteunt. Hiermee wil dit project bijdragen aan een betrouwbaarder en eenvoudiger in-house softwarebeheer binnen Pfizer.

13. Verwijzingen

Pfizer Belgium. (z.d.). *Productie-Puurs*. <https://www.pfizer.be/nl/over-ons/pfizer-in-belgie/productie-puurs>