# Structuur van het project.

Om dit project bruikbaar te maken voor de gebruiker zonder al te veel programmeer kennis is ervoor gekozen om de volgende structuur te hanteren:

# Lokalisatie

In deze folder zit een bestand genaamd: messages.xml

Hierin worden alle berichtgevingen van de fouten gedefinieerd. Het is hier mogelijk om verschillende talen te implementeren.

In dit bestand kan de tekst worden aangepast naar wens van de gebruiker.

### Regels voor het aanpassen van dit bestand

De id’s van de messages staan vast en als dezen worden aangepast kan de validatie stoppen met werken.

De attributen die worden toegevoegd, dit zijn de woorden tussen ‘{‘ en ‘}’, staan vast.

Deze attributen kan je wel:

* van plaats veranderen
* verwijderen

Deze attributen kan je niet:

* Van naam veranderen
* Nieuwe toevoegen met een nieuwe naam
* Kopiëren en meerdere keren gebruiken.

# Configuratie

In deze folder zit een bestand genaamd: configuration.xml

Hierin worden alle parameters gedefinieerd die gebruikt worden in de testen. Ze zijn opgedeeld per attribuut en hebben binnen een attribuut een naam die duidelijk maakt wat de functionaliteit van deze parameter is.

### Regels voor het aanpassen van dit bestand

De namen van de attributen & parameters, dit is tekst tussen ‘<’ en ‘>’, staan vast en kunnen niet worden aangepast.

Velden tussen parameters, dit is tekst tussen ‘<parameter>’ en ‘</parameter>’, kunnen worden naar behoeven worden aangepast. Let hierbij erop dat tekst ook tekst blijft en dat getallen ook getallen blijven.

# Validatie schema’s

De validatie schema folder is opgedeeld in 3 onderdelen

### POC schema

Dit is het bestand dat wordt gebruikt om de validatie uit te voeren. Hierin zitten alle bestanden gelinkt van de patronen, berichten, configuraties, functies en fases. Daarnaast worden in dit bestand al de namespaces goed gezet.

Als er bepaalde testen uitgezet moeten worden kan je in dit bestand deze uit commentariëren. Als er nieuwe testen gemaakt worden moeten deze in dit bestand worden toegevoegd met het relatieve bestands pad in een include en het id van de test in de active patterns binnen de corresponderende fase.

# Patterns

Dit is de folder waar alle tests worden gedefinieerd. Ze zijn onderverdeeld per attribuut om te zorgen dat het schaalbaar en overzichtelijk blijf bij uitbreiding van dit POC.

In elke folder zitten een aantal testen in de vorm van een .sch bestand. Deze zijn opgedeeld in 2 types testen:

* Structuur, deze testen zijn bedoeld om relatief simpele fouten af te vangen.
* Complex, deze testen zijn bedoeld om complexe situaties af te vangen, hierbij worden vaak aannames gedaan die ervan uitgaan dat de structuur testen geslaagd zijn.

### De structuur van een pattern (.sch bestand)

Het buitenste element is een pattern. Een pattern heeft:

* Een namespace, in ons geval gebruik wij de schematron namespace zodat niet overal de sch: prefix voor hoeft.
* Een Id. Dit id wordt gebruikt om zichzelf te identificeren bij het gebruik van fases.

Het 2e element is een rule met context. De context geeft aan welk element naar gezocht wordt om deze regel af te laten gaan. Aan dit element kan door middel van ‘[predikaat\_logica]’ een predikaat mee gegeven worden.

Bijvoordbeeld: als er 3 LSKabel elementen in een nlcs++ xml zitten, zal de inhoud van de rule 3 keer worden doorlopen met elke keer dat één LSkabel als basis element.

Hierna volgen een hoop let statements, hierin worden elementen en logica aan variabelen verbonden.

Belangrijke variabelen zijn: message & placeholders. Dezen worden gebruikt om de berichtgeving correct te maken.

Het laatste element is een assert, hierin wordt de test uitgevoerd en gekeken of er een verwacht resultaat is.

# XSL functies

Dit is de folder waar alle xsl-functies zijn gedefinieerd. Deze zijn er voornamelijk zodat er in de schema’s niet te veel herhalende blokken code staan.

### Geometry functies

In deze functies worden geometrische relaties verwerkt. Uitleg over de topologische relaties tussen objecten is in de bijlages toegevoegd als PowerPoint presentatie.

### Global functies

In deze functies worden object opgehaald vanuit de config.xml & worden de messages en placeholders geregeld.

### Helper functies

In deze functies worden veel gebruikte functies gedefinieerd. Dit zijn vooral functies om te casten van het ene naar het andere type of om code leesbaarder te houden in de schema’s.