WIPO Sequence Validator – Bedienungsanleitung

Version 2.2.0

Der Zweck dieses Dokuments ist es, den Ämtern für geistiges Eigentum beim Einsatz des WIPO Sequence Validator-Webdienstes zu helfen und sie bei der Konfiguration des Validators zu unterstützen.

Inhaltsverzeichnis

1. Einleitung

Der Hauptzweck des WIPO Sequence Validators (nachstehend "Validator" oder "das Tool" genannt) besteht darin, den Ämtern für geistiges Eigentum (IP-Ämtern) einen Webdienst zur Validierung von XML-Dateien im WIPO ST.26-Format zur Verfügung zu stellen, um sicherzustellen, dass diese mit dem WIPO-Standard ST.26 konform sind. Obwohl ein mit der WIPO Sequence Desktop-Anwendung erstelltes Sequenzprotokoll WIPO ST.26-konform sein wird, kann der Benutzer jedes Tool verwenden, das er für am geeignetsten hält.

Ziel dieses Dokuments ist es, den Aufbau, Einsatz, die Einstellungen und das Datei-Consumer-System des Tools zu erklären, das in den folgenden Abschnitten ausführlich beschrieben wird. Bei Fragen zur Fehlerbehebung wenden Sie sich bitte an das Wiki unter: https://www3.wipo.int/confluence/display/ST26software/Validator+Troubleshooting

1.1. Übersicht des Validator-Workflows

Das Tool bietet die folgenden vier Anwendungsfälle:

- Validierung einer WIPO ST.26-Datei;
- Abfragen des Status einer momentan ausgeführten Validierung;
- Aktualisierung der Konfigurationsdateien (nur für IP-Amt Admin); und
- Aufrufen eines Callback-Endpunkts mit dem Ergebnis des Validierungsprozesses, sobald der Prozess abgeschlossen ist.

Hinweis: Dieser Callback-Endpunkt[1] liegt außerhalb des Geltungsbereichs des Validators. Es obliegt den Ämtern, die diesen Dienst entwickeln und einrichten, den Endpunkt zu erstellen.

Das Tool besteht aus einer JAR-Datei, die als ein Webdienst ausgeführt werden kann, oder einer WAR-Datei, die auf einem Tomcat-Server bereitgestellt werden kann.

In beiden Fällen konsumiert das Tool zur Validierung des WIPO ST.26-Sequenzprotokolls Dateien aus einem lokalen Dateisystem, generiert einen Prüfbericht mit Validierungsergebnissen und liefert wahlweise die Ergebnisse des Validierungsprozesses, d.h. den Prüfbericht, durch den Aufruf eines Callback-Endpunkts.

Der Haupt-Workflow des Validators umfasst die folgenden Schritte:

- Das entsprechende IT-System des IP-Amts speichert eine WIPO ST.26 XML-Datei in einem "Inbox"-Standardordner oder dem in der Anforderung angegebenen Ordner.

- Das System des IP-Amts initiiert einen HTTP-Post, der die Validierung der Datei anfordert. Je nach Konfiguration kann das System des IP-Amts eine "vollständige" oder eine "Formalitäts"-Validierung der Datei anfordern. Beim "Formalitäts"-Validierungsprozess wird geprüft, ob es sich bei der ST.26-Datei um eine XML-Datei handelt und die Datei wird gegen die ST.26 DTD validiert. Der "vollständige" Validierungsprozess validiert die ST.26-Datei hinsichtlich den Geschäftsprüfregeln, die sich aus dem Inhalt von ST.26 ergeben, und führt außerdem den "Formalitäts"-Validierungsprozess durch.

  Hinweis: Es wird empfohlen, den "Formalitäts"-Validierungsprozess nur für das Online-Einreichungssystem zu verwenden, da dieser synchron durchgeführt werden kann, während

[1] Ein Callback-Endpunkt ist hier eine eindeutige Adresse, die durch eine URI identifiziert wird und an die Anfragen gesendet werden können

die "vollständige" Validierung für die Batchverarbeitung empfohlen wird, da sie wesentlich länger dauert.

- Sobald die Validierung abgeschlossen ist, wird in einer Antwort angegeben, ob die Datei die "Formalitäts"-Validierung bestanden hat oder nicht. Falls das IT-System des IP-Amts die "vollständige" Validierung ausgewählt hat, wird weiterhin angegeben, ob die Geschäftsprüfregelvalidierung korrekt gestartet wurde oder nicht.

- Führt der Validator eine "vollständige" Validierung durch, ruft er die XML-Datei aus dem "Inbox"-Ordner ab und startet die Geschäftsprüfregelvalidierung. Anschließend führt er Folgendes aus:

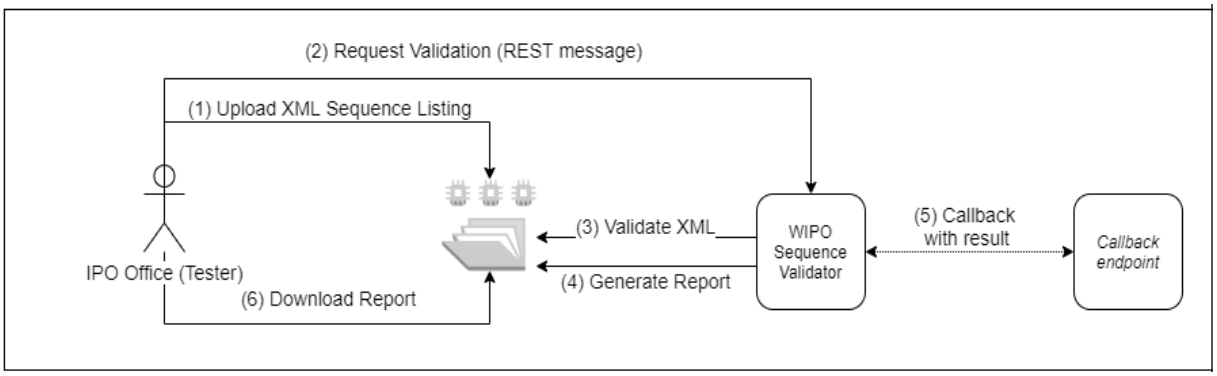
- Der Validator generiert eine XML-Berichtsdatei ("Verification Report" - Prüfbericht) in einem bestimmten "Output"-Ordner und bewegt die validierte WIPO ST.26 XML-Datei in einen "Outbox"-Ordner.

- Nachdem die Geschäftsprüfregelvalidierung abgeschlossen ist, wird der Callback-Endpunkt, falls konfiguriert, vom Validator aus aufgerufen und die Anforderung wird mit zusätzlichen Informationen in Bezug auf den Validierungsprozess aufgefüllt. Die Struktur der Anforderung und einige Beispieldaten finden Sie in Abschnitt 4 unten.

- Der Callback-Endpunkt sollte in der Antwort entweder einen leeren oder einen Erfolgscode zurückgeben (keine Fehler). [Hinweis: Dieser Schritt wird nur ausgeführt, wenn der externe Webdienst zur Verfügung gestellt und der Aufruf im Validator konfiguriert wurde.] Eine Konnektivität zwischen dem Validator und dem Callback-Endpunkt ist ebenfalls erforderlich. Wie bereits erwähnt, ist der externe Webdienst nicht Bestandteil des Validators und sollte von den Ämtern entsprechend dem weiter unten definierten Vertrag entwickelt und konfiguriert werden.

- Das System des IP-Amts kann den Prüfbericht aus dem "Report"-Ordner abrufen.

.



Hinweis: Der WIPO Sequence Validator entspricht dem WIPO-Standard für die Verarbeitung und Übermittlung von Daten über geistiges Eigentum unter Verwendung von Web APIs: WIPO ST.90.

1.2. Definition der Struktur des Validator-Dateisystems

Die Struktur des vom Validator verwendeten Dateisystems besteht aus fünf Ordnern:

- **"Inbox"-Ordner**: Dies ist der lokale Ordner, in dem WIPO ST.26-Dateien von einem IP-Amt zur Validierung bereitgestellt werden.

- **"Process"-Ordner**: Dies ist der lokale Ordner, den die in der "Inbox" befindlichen Dateien während der Verarbeitung vorübergehend durchlaufen. Dieser Ordner enthält zwei Unterordner:
    - **"Full validation"-Ordner (vollständige Validierung)**: Zur Speicherung der Dateien, die auf eine vollständige Validierung warten.
    - **"Formality validation"-Ordner (Formalitätsvalidierung)**: Zur Speicherung der Dateien, die auf eine Formalitätsvalidierung warten.
- **"Outbox"-Ordner**: Sobald die Validierung abgeschlossen ist, speichert die Anwendung die Source der WIPO ST.26-Datei in diesem lokalen Ordner.
- **"Report"-Ordner**: Dies ist der lokale Ordner, in dem die Ergebnisse der Validierung in einer Prüfberichtsdatei gespeichert werden.
- **"Params"-Ordner**: Dies ist der lokale Ordner, in dem sich eine JSON-Datei (.json) mit allen Validierungsparametern aus der Validierungsanforderung befindet, um Parameter für den asynchronen tiefen Validierungsprozess bereitzustellen.

Nachfolgend ein Beispiel für die Struktur des Dateisystems:

```
/temp/ST26
/temp/ST26/inbox
/temp/ST26/process/full
/temp/ST26/process/formality
/temp/ST26/outbox
/temp/ST26/reports
/temp/ST26/params
```

[WICHTIG: Standardmäßig sollte sich das Verzeichnis "/temp/ST26" in dem übergeordneten Verzeichnis befinden, in dem sich das Tool befindet. Wenn sich zum Beispiel die JAR- oder WAR-Datei in C:/dev befindet, sollte die Ordnerstruktur wie folgt erstellt werden: C:/temp/ST26/...]

2. Einsatz des WIPO Sequence Validators

Wie bereits erwähnt, wird der Validator als eines der beiden unten aufgeführten binären Formate bereitgestellt. Je nach Infrastruktur, in der das Amt den Validator einsetzen möchte, wird das IP-Amt ein binäres Format gegenüber dem anderen bevorzugt auswählen.

Die beiden vom Validator zur Verfügung gestellten binären Formate sind:

- **Binary SpringBoot JAR**: Dieses binäre Format ist eine ausführbare JAR-Datei. Dazu muss Java 8 installiert sein.
- **War Package Binary:** Dieses binäre Format ist für die Bereitstellung auf einem Servlet-Container vorgesehen. Ein mit Spring Boot 2 und Servlet Spec 3.1+ kompatibler Anwendungsserver ist erforderlich, wie z.B. Tomcat 8.5.

In den folgenden Abschnitten wird der Einsatz des Validators als eine Spring Boot-App oder als WAR innerhalb eines Java-Anwendungsservers beschrieben.

2.1. Starten des Validators als Spring Boot JAR

Das Spring Boot JAR enthält einen eingebetteten Server, der die Bereitstellung der Validator-API ermöglicht, ohne dass ein separater Server erforderlich ist. Dies vereinfacht die Konfiguration und den Einsatz auf der Infrastrukturebene erheblich.

Um den eingebetteten Server auszuführen, sollte der folgende Befehl ausgeführt werden.

Hinweis: Java 8 muss bereits auf dem Server installiert sein: Da Java die Verwendung von UTF-8 nicht garantiert, muss die Systemeigenschaft "file.encoding" auf "UTF-8" gesetzt werden. Dies kann durch Einfügen der folgenden Zeile erreicht werden:

```
java -D -jar wipo-sequence-validator.jar
```

Die Validator-API kann über eine Swagger-Benutzerschnittstelle aufgerufen werden:

http://[host-name]:8080/swagger-ui.html

Die Validator-API ist über die folgenden Endpunkte zugänglich:

http://[host-name]:8080/api/[version]/status

http://[host-name]:8080/api/[version]/validate

wobei das IP-Amt die folgenden Änderungen vornehmen muss:

- [host-name] muss durch den Hostnamen des Servers ersetzt werden; und
- [version] sollte durch die Version der Validator-API ersetzt werden (z.B. v1.0).

Standardmäßig wird der Server auf Port 8080 ausgeführt. Um den Port zu ändern, sollte die "--server.port"-Befehlszeilenoption wie hier gezeigt hinzugefügt werden:

```
java -D"file.encoding=UTF-8" -jar wipo-sequence-validator.jar –-server.port=<port-number>
```

Standardmäßig verwendet der Validator die Java Virtual Machine (JVM)-Standardspeichereinstellungen. Die maximale Heap-Größe beträgt standardmäßig ein Viertel des verfügbaren physischen Speichers.

Um die maximale Heap-Größe zu ändern, muss die Option "-Xmx" bei der Ausführung mit der folgenden Befehlszeile verwendet werden[2]:

```
java -D"file.encoding=UTF-8" -Xmx[size]-jar wipo-sequence-validator.jar
```

2.1.1. Einsatz des Validators als ausführbare Anwendung

Der Validator kann auch als vom Betriebssystem verwalteter Dienst installiert werden, um beispielsweise die Ausführung mit dem Start des Betriebssystems zu unterstützen.

Es ist möglich, eine Spring Boot JAR-Datei auf diese Weise für alle von WIPO Sequence unterstützten Plattformen zu konfigurieren: Windows, Linux und Mac OS.

In der folgenden Anleitung wird detailliert beschrieben, wie Sie einen Systemdienst erstellen, der die JAR-Datei für jedes Betriebssystem ausführt. Sie enthält auch Informationen zur Konfiguration der verschiedenen Optionen des Dienstes und der Ausführung der Anwendung:

https://docs.spring.io/spring-boot/docs/current/reference/html/deployment-install.html

[2] https://docs.oracle.com/javase/8/docs/technotes/tools/windows/java.html#BABHDABI

2.2. Einsatz als WAR-Webdienst

Bei der zweiten Art des bereitgestellten binären Formats kann das WAR-Paket in einem bestehenden Java-Anwendungsserver wie Apache Tomcat 8.5 bereitgestellt werden.

Hinweis: Ein mit Servlet 3.1 kompatibler Container ist erforderlich.

Die folgenden Anweisungen gelten für einen Tomcat-Anwendungsserver. Hier bezieht sich "$TOMCAT_ROOT" auf den Stammordner des Tomcat-Servers. Dieser Wert sollte durch den entsprechenden Wert für den Dateipfad ersetzt werden:

a) Server stoppen: $TOMCAT_ROOT\bin\catalina.bat stop

b) "war" nach $TOMCAT_ROOT\webapps\wipo-sequence-validator.war kopieren

c) Server starten: $TOMCAT_ROOT\bin\catalina.bat start

Hinweis: Da Java die Verwendung von UTF-8 nicht garantiert, muss die Systemeigenschaft "file.encoding" beim Start des Anwendungsservers auf "UTF-8" gesetzt werden. Dies kann durch Einfügen von -D"file.encoding=UTF-8"erreicht werden.

Auf die Validator-API kann, wie oben angegeben, über eine Swagger-Benutzeroberfläche zugegriffen werden:

http://host-name:8080/wipo-sequence-validator/swagger-ui.html

Die Validator-API ist über die folgenden Endpunkte zugänglich:

http://[host-name]:8080/wipo-sequence-validator/api/[version]/status

http://[host-name]:8080/wipo-sequence-validator/api/[version]/validate

wobei das IP-Amt die folgenden Änderungen vornehmen muss:

- [host-name] muss durch den Hostnamen des Servers ersetzt werden; und
- [Version] sollte durch die Version der API (z.B. v1.0) ersetzt werden.

Standardmäßig wird der Server auf Port 8080 ausgeführt. Um dies in einen anderen Port zu ändern, sollte die Tomcat-Konfigurationsdatei geändert werden, indem Sie den hier bereitgestellten Anweisungen folgen:

https://tomcat.apache.org/tomcat-8.5-doc/config/http.html#Common_Attributes

Standardmäßig verwendet der Validator die JVM-Standardspeichereinstellungen. Die maximale Heap-Größe beträgt standardmäßig ein Viertel des verfügbaren physischen Speichers.

Um die maximale Heap-Größe zu ändern, muss die Option "-Xmx" bei der Ausführung mit der Befehlszeile verwendet werden, wie oben in Abschnitt 2.1 angegeben.

3. Prüfbericht

Der vom Tool generierte Prüfbericht hat das XML-Format und das verwendete Template wird nachfolgend bereitgestellt:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VerificationReport productionDate="YYYY-MM-DD" sourceFileName="[ST.26 filename]">
   <VerificationMessageBag>
      <VerificationMessage>
         <Severity>[ERROR | WARN | XML_WARN |  XML_ERROR]</Severity>
         <DataElement>[ST.26 element]</DataElement>
         <DetectedSequence>[Sequence ID]</DetectedSequence>
         <DetectedValue>[value]</DetectedValue>
         <MessageKey>[Message key]</MessageKey>
         <ParameterBag>
                  <Parameter key="param key">Param value</Parameter>
         <ParameterBag>
         <LocalizedMessage> [Localized message] </LocalizedMessage>
      </VerificationMessage>

...

   </VerificationMessageBag>
</VerificationReport>
```

Ein Beispiel für einen solchen Prüfbericht finden Sie in Anhang I dieser Bedienungsanleitung mit den zulässigen Werten für diese Komponenten in Anhang III. Beachten Sie hinsichtlich des angegebenen Schweregrads die folgende Kategorisierung:

- ERROR - ein Fehler, der während der "vollständigen" Prüfung zurückgegeben wurde
- WARNING - eine Warnung, die während der "vollständigen" Prüfung zurückgegeben wurde
- XML_ERROR - ein Fehler, der während der "Formalitätsprüfung" zurückgegeben wurde
- XML_WARN - eine Warnung, die während der "Formalitätsprüfung" zurückgegeben wurde

Der Prüfbericht wird auch im HTML-Format unter Verwendung desselben Stylesheets erstellt, das in WIPO Sequence verwendet wird.

4. Callback-Endpunkt-Anfrage

Die vom Callback-Endpunkt an den Validator gestellte Anfrage sollte die folgenden Parameter enthalten, die die Dateispeicherorte und den Validierungsprozess näher beschreiben:

```
{
  "currentApplicationNumber": "string",
  "currentSEQLVersionNumber": "string",
  "parentApplicationNumber": "string",
  "parentSEQLVersionNumber": "string",
  "seqlInputLocation": "C:/temp/valid2Warning.xml",
  "verificationReportOutputPath": "C:/temp/report.xml",
  "nameFile": "valid2Warning.xml",
  "type": "full"
}
```

Das Feld "seqlInputLocation" der Validierungsanfrage sollte so festgelegt werden, dass der Pfad des zu validierenden XML-Sequenzprotokolls angegeben wird. Wenn das Amt dieses Feld leer lässt, versucht das Tool, die XML-Datei mit dem Dateinamen "nameFile" im standardmäßigen "Inbox"-Ordner zu validieren. Der Parameter "nameFile" gibt die zu validierende Sequenzprotokolldatei an.

Der "verificationReportOutputPath" innerhalb der Anfrage gibt den Dateispeicherort des vom Tool generierten Prüfberichts (.xml) an. Wenn der Benutzer das Feld leer lässt oder einen ungültigen Dateipfad eingibt, wird der Prüfbericht im standardmäßigen "Reports"-Ordner gespeichert.

4.1. Callback-Endpunkt-Anfrageformat

Ist die Eigenschaft "api.URL" konfiguriert, wird der Validator versuchen, die Ergebnisse der Validierung an einen Endpunkt mit der angegebenen URL zu senden.

Um mit dem Validator kommunizieren zu können, muss der Callback-Endpunkt mit dem folgenden Webservicevertrag (YAML) konform sein:

```yaml
openapi: 3.0.0
info:
  description: Callback for the WIPO Sequence Validator
  version: "1.0"
  title: WIPO Sequence Validator Callback
paths:
  /api/validator/callback:
    post:
      summary: Return the generated contract
      operationId: callbackUsingPOST
      requestBody:
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/ServiceRequest"
        description: request
        required: true
```

```
      responses:
        "200":
          description: OK
        "201":
          description: Created
        "401":
          description: UNAUTHORIZED
        "403":
          description: FORBIDDEN
        "404":
          description: ELEMENT NOT FOUND
        "500":
          description: INTERNAL ERROR SERVER
      deprecated: false
servers:
  - url: //localhost:8080/
components:
  schemas:
    Error:
      type: object
      required:
        - code
        - message
      properties:
        code:
          type: string
          example: INVALID_VALIDATION_TYPE
          description: error code
        message:
          type: string
          description: error message
        moreInfo:
          type: string
          description: extended info on the error
      title: Error
    MapEntry:
      type: object
      properties:
        key:
          type: string
          xml:
            name: key
            attribute: true
            wrapped: false
        value:
          type: string
```

```
    title: MapEntry
    xml:
      name: Parameter
      attribute: false
      wrapped: false
  ServiceRequest:
    type: object
    properties:
      currentApplicationNumber:
        type: string
      currentSEQLVersionNumber:
        type: string
      elapsedTime:
        type: integer
        format: int64
      endTime:
        type: string
      errorSummary:
        type: array
        items:
          $ref: "#/components/schemas/VerificationMessage"
      httpStatus:
        type: string
      parentApplicationNumber:
        type: string
      parentSEQLVersionNumber:
        type: string
      processID:
        type: string
      seqIDQuantity:
        type: integer
        format: int32
      seqInputQuantity:
        type: integer
        format: int32
      seqlType:
        type: string
      startTime:
        type: string
      totalErrorQuantity:
        type: integer
        format: int32
      totalWarningQuantity:
        type: integer
        format: int32
      verificationReportOutputPath:
```

```
        type: string
    title: ServiceRequest
  VerificationMessage:
    type: object
    properties:
      dataElement:
        type: string
        xml:
          name: DataElement
          attribute: false
          wrapped: false
      detectedSequence:
        type: string
        xml:
          name: DetectedSequence
          attribute: false
          wrapped: false
      index:
        type: integer
        format: int32
      key:
        type: string
        xml:
          name: MessageKey
          attribute: false
          wrapped: false
      locmessage:
        type: string
        xml:
          name: LocalizedMessage
          attribute: false
          wrapped: false
      params:
        type: object
        additionalProperties:
          type: string
      paramsForXML:
        type: array
        xml:
          name: ParameterBag
          attribute: false
          wrapped: true
        items:
          $ref: "#/components/schemas/MapEntry"
      reportValue:
        type: string
```

```
          xml:
            name: DetectedValue
            attribute: false
            wrapped: false
        sequenceIDNumber:
          type: string
        type:
          type: string
          xml:
            name: Severity
            attribute: false
            wrapped: false
      title: VerificationMessage
      xml:
        name: VerificationMessage
        attribute: false
        wrapped: false
```

Außerdem sollte die Anfrage ein JSON-Objekt mit dieser Struktur sein:

```
{
 "currentApplicationNumber": "string",
 "currentSEQLVersionNumber": "string",
 "elapsedTime": 0,
 "endTime": "string",
 "errorSummary": [
  {
   "dataElement": "string",
   "detectedSequence": "string",
   "index": 0,
   "key": "string",
   "locmessage": "string",
   "params": {
    "additionalProp1": "string",
    "additionalProp2": "string",
    "additionalProp3": "string"
   },
   "paramsForXML": [
    {
     "key": "string",
     "value": "string"
    }
   ],
   "reportValue": "string",
   "sequenceIDNumber": "string",
```

```
    "type": "string"
   }
  ],
  "httpStatus": "string",
  "parentApplicationNumber": "string",
  "parentSEQLVersionNumber": "string",
  "processID": "string",
  "seqIDQuantity": 0,
  "seqInputQuantity": 0,
  "seqlType": "string",
  "startTime": "string",
  "totalErrorQuantity": 0,
  "totalWarningQuantity": 0,
  "verificationReportOutputPath": "string"
}
```


Dies ist eine Beispiel-JSON-Instanz, die an den externen Endpunkt gesendet wird, der den Aufruf an den Validator durchgeführt hat:


```
{
   "processID": "1608194222169dvVE",
   "seqlType": "ST.26",
   "httpStatus": "SUCCESS",
   "currentApplicationNumber": "string",
   "currentSEQLVersionNumber": "string",
   "parentApplicationNumber": "string",
   "parentSEQLVersionNumber": "string",
   "verificationReportOutputPath": "C:/temp/report.xml",
   "startTime": "2020-12-17 09:36:54.000000",
   "endTime": "2020-12-17 09:37:26.000607",
   "elapsedTime": 32607,
   "totalWarningQuantity": 1,
   "totalErrorQuantity": 2,
   "seqInputQuantity": 3,
   "seqIDQuantity": 3,
   "errorSummary": [
      {
```

```
        "index": 0,
        "reportValue": "",
        "type": "WARNING",
        "params":com.wipo.st26.ipotool.models.ServiceRequest@5887858,
        "key": "X_EARLIEST_PRIO_APPLICATION_ID_MISSING",
        "locmessage": "Earliest priority application information is absent. It must be provided when a priority claim is made to an earlier application.",
        "detectedSequence": "",
        "dataElement": "PROPERTY_NAMES.EARLIEST_PRIORITY_APPLICATION"
    },
    {
        "index": 0,
        "reportValue": "-",
        "type": "ERROR",
        "params": {},
        "key": "INVENTION_TITLE_MISSING",
        "locmessage": "The invention title is missing. At least one invention title must be entered.",
        "detectedSequence": "",
        "dataElement": "PROPERTY_NAMES.INVENTION_TITLE_BAG"
    },
    {
        "index": 1,
        "reportValue": "-",
        "type": "ERROR",
        "params": {},
        "key": "INVENTION_TITLE_MISSING",
        "locmessage": "The invention title is missing. At least one invention title must be entered.",
        "detectedSequence": "",
        "dataElement": "PROPERTY_NAMES.INVENTION_TITLE_BAG"
    }
  ]
}
```

4.2. Prüfbericht

Wie in Abschnitt 3 erwähnt, befindet sich der generierte Prüfbericht nach der Validierung im "verificationReportOutputPath", der im vorliegenden Beispiel bei "C:/temp/report.xml" ist.

Der Inhalt dieses Berichts wird an den Callback-Endpunkt innerhalb des Feldes "errorSummary" des "ServiceRequest" gesendet. Ein Beispiel für dieses Feld findet sich in den oben in Abschnitt 4 angegebenen Beispielanfragen.

5. Konfiguration

5.1. Standardeinstellungen

Der Validator wird mit Hilfe einer Eigenschaftsdatei konfiguriert. Die Standarddatei "application.properties" hat die folgenden Werte:

```
############ WIPO Sequence Validator properties

## -- FOLDERS --

#Base path to be used by the rest of folders
app.basePath=/temp/st26/
#Folder to put the files to be processed
app.inboxPath=${app.basePath}inbox/

#Folder to store the ST26 files once validated
app.outboxPath=${app.basePath}outbox/
#Folder to store the validation reports
app.reportsPath=${app.basePath}reports/

#Folder to store the parameters
app.paramsPath=${app.basePath}params/

#Parent folder for full and formality folders
app.processPath=${app.basePath}process/

#Files in process for a full validation are stored in this folder
app.process.fullPath=${app.processPath}full/
#Files in process for a formality validation are stored in this folder
app.process.formalityPath=${app.processPath}formality/

alternativeResourceBasePath=${app.basePath}alt_resources

#Preferences
#To enable the rule VXQV49 set this value to true, default value is false.
app.preferences.optionalEnglishQualifierValue=false
# Please enter either: ERROR or WARNING to specifiy the type of the verification
message for the rule
 VXQV_49, default value is "WARNING".
app.preferences.optionalRuleType=WARNING

#locale used for the localized messages from the verification report
validator_locale=en

api.URL=
```

```
## -- Watcher

# These properties control the process looking for files in the folders to be
processed
# (see: https://docs.spring.io/spring-framework/docs/current/javadoc-
api/org/springframework/scheduling/concurrent/ThreadPoolTaskExecutor.html)

processing.delay=10000
processing.corePoolSize=5
#Max number of files being validated concurrently
processing.maxPoolSize=10
processing.queueCapacity=1000
processing.enabled=true


#### Logging (see https://logback.qos.ch/manual/configuration.html)

logging.level.root=info
logging.level.org.wipo=info
logging.pattern.console=%d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level %logger{36} -
%msg%n


# HEALTH ENDPOINT
management.endpoints.jmx.exposure.include=health
# Show details of health endpoint
management.endpoint.health.show-details=always
```

Um den Wert der hier bereitgestellten Parameter zu ändern, sollte eine alternative Datei "application.properties" verwendet werden. In der Dokumentation zu Spring Boot werden mehrere Optionen beschrieben: https://docs.spring.io/spring-boot/docs/2.0.6.RELEASE/reference/html/boot-features-external-config.html#boot-features-external-config-application-property-files

Die einfachste Option wäre die Bereitstellung einer neuen Datei "application.properties", nach der an den folgenden Stellen in der Reihenfolge der Operationen gesucht wird:

(a) Ordner "/config" im aktuellen Verzeichnis *[Hinweis: Wird der Validator als WAR-Datei in Tomcat bereitgestellt, befindet sich dieser Ordner im Ordner "lib", z.B. "/opt/apache-tomcat/lib/config"]*;

(b) Aktuelles Verzeichnis *[Hinweis: Wird der Validator als WAR-Datei in Tomcat bereitgestellt, befindet sich dieser Ordner im Ordner "lib", z.B. "/opt/apache-tomcat/lib/"];*

(c) $classpath oder config package; dann

(d) $classpath root.

Außerdem können Pfad und Name der Konfigurationsdatei angegeben werden, indem beim Starten des Tools der Parameter in der Befehlszeile festgelegt wird:

- Für den Einsatz mit JAR:

  java -jar -Dspring.config.location= <PATH_TO_FILE> wipo-sequence-validator.jar

- Für den Einsatz mit WAR auf Tomcat sollte der folgende Eintrag zu CATALINA_OPTS hinzugefügt werden:

  "export CATALINA_OPTS="-Dspring.config.location=<PATH_TO_FILE>"

Bei Verwendung von WAR kann auch die neue Datei "application.properties" in den Ordner "WEB-INF/classes“ der Webanwendung kopiert werden, oder die existierende Datei wird bearbeitet.

5.2. Configuring verification rule VXQV_49

```
app.preferences.optionalEnglishQualifierValue=false
app.preferences.optionalRuleType=WARNING
```

Der Wert "`optionalEnglishQualifierValue`" in der Datei "application.properties" kann auf 'true' gesetzt werden, wenn der Benutzer die Regel VXQV_49 aktivieren möchte, und der Schweregrad der Regel kann durch Aktualisieren des Wertes "`optionalRuleType`" auf 'ERROR' oder 'WARNING' festgelegt werden. Die Standardwerte für diese beiden Eigenschaften werden oben angezeigt.

5.3. Überprüfen Sie den Zustand ("health") des Endpunkts

```
# HEALTH ENDPOINT
management.endpoints.jmx.exposure.include=health
```

Der Validator-Dienst hat einen Endpunkt "/health" implementiert, der grundlegende Informationen über den Zustand ("health") der Anwendung liefert.

Um den Endpunkt "/health" zu untersuchen, lautet die URL: http://localhost:8080/wipo-sequence-validator /actuator/health. Der Endpunkt sollte Folgendes anzeigen:

- Der Zustand ist 'UP', solange die Anwendung in fehlerfreiem Zustand ist.
- Der Zustand 'DOWN' wird angezeigt, wenn die Anwendung aufgrund von Problemen wie der Verbindung mit der Datenbank oder fehlendem Festplattenspeicher usw. fehlerhaft wird.

Der Endpunkt "/health" zeigt nur einen einfachen 'UP'- oder 'DOWN'-Zustand an. Die folgende Eigenschaft in der Datei "application.properties" liefert die vollständigen Details, einschließlich des Status aller Zustandsindikatoren, der im Rahmen der Zustandsprüfung überprüft wurde.

```
# Show details of health endpoint
management.endpoint.health.show-details=always
```

Der Endpunkt "/health" enthält jetzt die Details des "DiskSpaceHealthIndicator“, der im Rahmen der Zustandsprüfung ausgeführt wird.

Der Endpunkt "/health" wird auf diese Weise als eine Reihe von Schlüssel/Wert-Paaren angezeigt und enthält weitere Details. Ein Beispiel ist unten dargestellt:

{"status":"UP","details":{"diskSpace":{"status":"UP","details":{"total":511123124224,"free":373225091072,"threshold":10485760}}}}

5.4. Lokalisierte Meldungen

Der Validator kann eine lokalisierte Meldung, z.B. im Prüfbericht, in jeder der zehn offiziellen PCT-Sprachen (Arabisch, Chinesisch, Deutsch, Englisch, Französisch, Japanisch, Koreanisch, Portugiesisch, Russisch, und Spanisch) bereitstellen.

Standardmäßig werden diese Meldungen in Englisch bereitgestellt. Um den Validator so zu konfigurieren, dass er diese Meldungen in anderen Sprachen bereitstellt, muss der Parameter "validator_locale" der Datei "application.properties" auf den entsprechenden Sprachcode eingestellt werden.

```
#Local used for the localized messages from the verification report
validator_locale=en
```

Hinweis: Der Validator sollte neu gestartet werden, damit die in der neuen Datei "application.properties" festgelegten Eigenschaften angewendet werden können.

5.5. Benutzerdefinierte Organismusnamen

Damit Ämter ihre eigenen, benutzerdefinierten Organismusnamen bereitstellen können, die nicht Teil der ursprünglichen vordefinierten Liste der Organismusnamen sind, kann eine Liste der benutzerdefinierten Organismen bereitgestellt werden, indem eine neue sogenannte Datei "custom_organism.json" im Ordner "alternativeResourceBasePath" erstellt wird. Diese Datei sollte die folgende Struktur aufweisen:

```
[
  {"value":"Custom Organism Sample"},
  {"value":"Custom Organism Sample 2"}
]
```

Hinweis: Im Gegensatz zur vordefinierten Liste der Organismusnamen sind alle Organismen in einer einzigen JSON-Datei enthalten, anstatt für jeden Buchstaben des Alphabets in eine JSON-Datei aufgeteilt zu werden.

5.6. Referenzieren von ST.26 DTD-Dateien

Standardmäßig referenziert der Validator die neueste Version von ST.26 DTD. Die aktuelle Version des WIPO Sequence Validators basiert auf Version 1.3 von WIPO ST.26 DTD[3].

[3] Valid as of April 18, 2022

Diese Kopie der neuesten Version von ST.26 DTD ist in der Validator-Bibliothek enthalten, die sich im Ordner "/src/main/resources" des Source-Codes befindet (dies ist der definierte Dateipfad, auf den innerhalb der JAR- oder WAR-Datei verwiesen wird). Sie wird in der Datei "catalog.xml" im selben Ordner referenziert, wie im Folgenden dargestellt wird:

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog">
  <public publicId="-//WIPO//DTD Sequence Listing 1.3//EN" uri="ST26SequenceListing_V1_3.dtd"/>
</catalog>
```

Anweisungen zum Einbeziehen einer neuen DTD finden Sie unten. Bei der Validierung wird die Version der DTD verwendet, die in der DOCTYPE-Deklaration der XML-Datei festgelegt ist. Zunächst wird die "publicId" verwendet, um den Speicherort der zu verwendenden DTD-Datei zu identifizieren. Ist die "publicId" nicht im Katalog enthalten, versucht das System, die DTD-Datei im Stammverzeichnis zu finden, in dem der Java-Prozess ausgeführt wird.

5.6.1. Angabe einer alternativen DTD-Version für die Validierung

Um WIPO ST.26-Dateien validieren zu können, die auf eine ältere Version der ST.26 DTD verweisen, muss diese ST.26 DTD-Datei dem Validator zur Verfügung gestellt werden, um eine entsprechende Validierung zu ermöglichen.

Um dies zu erreichen, gibt es zwei alternative Ansätze:

(a) Dekomprimieren Sie die JAR-Datei und fügen Sie einen Verweis auf die zusätzliche oder alternative ST.26 DTD-Datei in den Ordner "src/main/resources" ein. Ändern Sie die Datei "catalog.xml" und fügen Sie einen neuen Eintrag für die zusätzliche ST.26 DTD hinzu oder bearbeiten Sie den vorhandenen Eintrag.

For example:

```
<?xml version="1,0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog">
<public publicId="-//WIPO//DTD Sequence Listing 1.2//EN" uri="ST26SequenceListing_V1_2.dtd"/>
<public publicId="-//WIPO//DTD Sequence Listing 1.3//EN" uri="ST26SequenceListing_V1_3.dtd"/>
</catalog>
```

(b) Anstatt die JAR-Datei zu ändern, sollten Sie die folgenden Schritte befolgen:

(i) Kopieren Sie "catalog.xml" und alle DTDs in einen lokalen Ordner;
(ii) Ändern Sie "catalog.xml", um einen Verweis auf die zusätzliche ST.26 DTD einzubeziehen; und
(iii) Legen Sie diese Java-Systemeigenschaft beim Start fest: “xml.catalog.files=<path_to_catalog.xml>"

Hinweis: In Tomcat für Windows können Sie diese Umgebungsvariable hinzufügen:

```
set "JAVA_OPTS=%JAVA_OPTS%
 -Dxml.catalog.files=C:\\temp\\tomcat\\sharedclasspath\\catalog.xml"
```

[WICHTIG: Der Einbezug einer anderen Version der ST.26 DTD ermöglicht die "Formalitäts"-Validierung der XML-Datei gegen die referenzierte ST.26 DTD, aber die "vollständige" Validierung erfordert wahrscheinlich eine Änderung des Source-Codes, um die Prüfregeln zu implementieren. Daher wird empfohlen, mehrere DTDs nur bei der Durchführung einer "Formalitäts"-Validierung zu verwenden].

6. Validator REST API

In diesem Abschnitt werden die Anwendungsfälle der Validator-API spezifiziert. Es gibt drei Dienste oder Anwendungsfälle:

- Validieren einer Datei im "Inbox"-Ordner;
- Validieren einer Datei als Teil der Anforderung; und
- Abfragen des Status einer Validierung.

Die API-Spezifikation für diesen (OAS 3.0 Complete API [YAML File]) Dienst ist vollständig in Anhang II zu finden.

6.1. Validierung der WIPO ST.26-Datei

Anforderungs-Mapping	/api/v1/validate
Methode	POST
Konsumiert	application/json
Generiert	application/json
Operation	Anfordern der Validierung einer bestehenden WIPO ST.26-Datei im "Inbox"-Ordner. Liefert eine eindeutige „verificationId“ zum Abrufen des Status der Validierungsanforderung.

Anfrage	{
	"currentApplicationNumber": "string",
	"currentSEQLVersionNumber": "string",
	"parentApplicationNumber": "string",
	"parentSEQLVersionNumber": "string",
	"seqlInputLocation": "string"(Location of Input.xml file),
	"verificationReportOutputPath": "string" (Destination of report.xml),
	"nameFile": "file.xml",
	"type”: "string" (Possible values: full | formality),
	}

Antworten	• '202': “Accepted". Die WIPO ST.26-Datei hat die Formalitätsvalidierung bestanden und und ihre Prüfung hat begonnen. Dieser Code enthält auch eine Antwortmeldung, die einen eindeutigen Code zum Abrufen des Prüfberichts ("verificationID") angibt. Die WIPO ST.26-Datei wird zur Bearbeitung in den "Process"-Ordner bewegt.

	• ‘400’: "Bad request". Die REST-Anfrage war nicht wohlgeformt oder die WIPO ST.26-Datei hat die XML-Validierung nicht bestanden. Dieser Code wird durch eine Antwortmeldung unter genauer Angabe des Fehlers ergänzt. • '404': "Not found". Die WIPO ST.26-Datei wurde nicht im "Inbox"-Ordner gefunden. • '500': "Internal Server Error". Es ist ein interner Fehler aufgetreten. Dieser Code wird durch eine Antwortmeldung unter genauer Angabe des Fehlers ergänzt.
Vorbedingung	Die WIPO ST.26-Datei muss in dem vom Benutzer angegebenen "Inbox"-Ordner bereitgestellt werden.
Nachbedingung	Die WIPO ST.26-Datei wird zu dem in "verificationReportOutputPath" angegebenen Ort oder in den Ordner "Outbox" zu folgendem Ort bewegt: “/[outbox]/[verificationID]/[file.xml]” Der Prüfbericht wird erstellt unter: “/[reports]/ [verificationID]/report_[file.xml]”

Entsprechende OAS 3.0 Definition [in YAML-Spezifikation]

```
 /api/v1.0/validate:
post:
  tags:
    - validation-controller
  summary: 'Request the validation of an existing ST26 file in the inbox folder.
Returns a unique
  verificationID for retrieving the status of the validation request'
  operationId: validationFileUsingPOST
  consumes:
    - application/json
  produces:
    - application/json
  parameters:
    -
      in: body
      name: request
      description: 'ST26 File name for validation'
      required: false
      schema:
        $ref: '#/definitions/ValidationRequest'
  responses:
    '200':
      description: OK
      schema:
```

```
        $ref: '#/definitions/ValidationResponse'
    '201':
      description: Created
    '202':
      description: 'Accepted. The ST26 file passed the formal validation and their
verification has
       started. This code will be complemented with a response message indicating a
unique code for
      retrieving the verification report (verificationID). ST26 file is moved to
the ⍰process⍰ folder
       for processing'
      schema:
        $ref: '#/definitions/ValidationResponse'
    '400':
      description: 'Bad request. The REST request was not well formed or the ST26
file did not pass
      the XML validation. This code will be complemented with a response message
indicating the detail
      of the error'
    '401':
      description: Unauthorized
    '403':
      description: Forbidden
    '404':
      description: 'File not found. The ST26 file was not found in the Inbox
Folder'
    '500':
      description: 'Server Error. An internal error happened. This code will be
complemented with a
     response message indicating the detail of the error'
  deprecated: false
```

6.2. Status der Validierung anfragen

Anforderungs-Mapping	/api/v1/status
Methode	POST
Konsumiert	application/json
Generiert	application/json
Operation	Anfordern des Validierungsstatus für eine bestimmte WIPO ST.26-Datei.

Anfrage	{
	verificationID: {type: string}
	}

Antworten	• '200': "Success". Dieser Code enthält auch eine Antwortmeldung, die eine eindeutige ID mit dem Status des Prüfprozesses angibt, ausgewählt aus: ○ "RUNNING": die Datei wird gerade bearbeitet. ○ "FINISHED-VALID": die Datei hat die Formalitätsvalidierung bestanden und das Ergebnis der Validierung ist im "Reports"-Ordner verfügbar. ○ "FINISHED-INVALID": der Prozess ist abgeschlossen, aber die Datei hat die Formalitätsvalidierung nicht bestanden. Das Ergebnis der Validierung ist im "Reports"-Ordner verfügbar. ○ "NOT_FOUND": die "verificationID" wurde nicht gefunden. ○ "VERIFICATION_ID_ERROR": die "verificationID" wurde nicht in die Anfrage einbezogen. • ‘400’: "Bad request". Die REST-Anfrage war nicht wohlgeformt. • '500': "Server Error". Ein interner Fehler ist aufgetreten. Dieser Code enthält auch eine Antwortmeldung unter genauer Angabe des Fehlers.
Nachbedingung	Der Validator liefert den Status der Validierung.
Annahmen	-

Entsprechende OAS 3.0 Definition [in YAML-Spezifikation]

```
paths:
  /api/v1.0/status:
    post:
      tags:
        - validation-controller
      summary: 'Request the validation status for a specific ST26 File'
      operationId: getStatusUsingPOST
      consumes:
        - application/json
      produces:
        - application/json
      parameters:
        -
          in: body
          name: request
          description: 'ST26 File name Object for validation status'
          required: false
          schema:
            $ref: '#/definitions/ValidationStatusRequest'
      responses:
        '200':
          description: 'This code will be complemented with a response message
indicating the Status
          of the verification process: RUNNING (the file is being processed)
FINISHED-VALID
          (the file passed the formality validation and the result of the
validation is available
          in the reports folder) FINISHED-INVALID
          (the file passed the formality validation and the result of the
validation is available
          in the reports folder)'
          schema:
            $ref: '#/definitions/ValidationStatusResponse'
        '201':
          description: Created
        '400':
          description: 'Bad request. The REST request was not well formed'
        '401':
          description: Unauthorized
        '403':
          description: Forbidden
        '404':
          description: 'Not Found'
        '500':
```

```
          description: 'Server Error. An internal error happened. This code will be
complemented with
          a response message indicating the detail of the error'
      deprecated: false
```

[Anhang I folgt]

Anhang I: Beispiel eines Prüfberichts

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VerificationReport productionDate="2020-12-17" sourceFileName="valid2Warning.xml">
    <VerificationMessageBag>
        <VerificationMessage>
            <Severity>WARNING</Severity>
            <DataElement>PROPERTY_NAMES.EARLIEST_PRIORITY_APPLICATION</DataElement>
            <DetectedSequence></DetectedSequence>
            <DetectedValue></DetectedValue>
            <MessageKey>X_EARLIEST_PRIO_APPLICATION_ID_MISSING</MessageKey>
            <ParameterBag/>
            <LocalizedMessage>Earliest priority application information is absent.
It must be provided
             when a priority claim is made to an earlier
application.</LocalizedMessage>
        </VerificationMessage>
        <VerificationMessage>
            <Severity>ERROR</Severity>
            <DataElement>PROPERTY_NAMES.INVENTION_TITLE_BAG</DataElement>
            <DetectedSequence></DetectedSequence>
            <DetectedValue>-</DetectedValue>
            <MessageKey>INVENTION_TITLE_MISSING</MessageKey>
            <ParameterBag/>
            <LocalizedMessage>The invention title is missing. At least one
invention title must
             be entered.</LocalizedMessage>
        </VerificationMessage>
        <VerificationMessage>
            <Severity>ERROR</Severity>
            <DataElement>PROPERTY_NAMES.INVENTION_TITLE_BAG</DataElement>
            <DetectedSequence></DetectedSequence>
            <DetectedValue>-</DetectedValue>
            <MessageKey>INVENTION_TITLE_MISSING</MessageKey>
            <ParameterBag/>
            <LocalizedMessage>The invention title is missing. At least one
invention title must
            be entered.</LocalizedMessage>
        </VerificationMessage>
    </VerificationMessageBag>
</VerificationReport>
```

[Anhang II folgt]

Anhang II: Vollständige API-Spezifikation (YAML)

```yaml
swagger: '2.0'
info:
  description: 'API for the WIPO Sequence Validator'
  version: '1.0'
  title: 'WIPO Sequence Validator API'
host: 'localhost:8080'
basePath: /
tags:
  -
    name: validation-controller
    description: 'Validation Controller'
paths:
  /api/v1.0/status:
    post:
      tags:
        - validation-controller
      summary: 'Request the validation status for a specific ST26 File'
      operationId: getStatusUsingPOST
      consumes:
        - application/json
      produces:
        - application/json
      parameters:
        -
          in: body
          name: request
          description: 'ST26 File name Object for validation status'
          required: false
          schema:
            $ref: '#/definitions/ValidationStatusRequest'
      responses:
        '200':
          description: 'This code will be complemented with a response message
indicating the Status
          of the verification process: RUNNING (the file is being processed)
FINISHED-VALID
          (the file passed the formality validation and the result of the
validation is available
          in the reports folder) FINISHED-INVALID
          (the file passed the formality validation and the result of the
validation is available
           in the reports folder)'
          schema:
            $ref: '#/definitions/ValidationStatusResponse'
```

```
        '201':
          description: Created
        '400':
          description: 'Bad request. The REST request was not well formed'
        '401':
          description: Unauthorized
        '403':
          description: Forbidden
        '404':
          description: 'Not Found'
        '500':
          description: 'Server Error. An internal error happened. This code will be
complemented with
          a response message indicating the detail of the error'
      deprecated: false
/api/v1.0/validate:
post:
  tags:
    - validation-controller
  summary: 'Request the validation of an existing ST26 file in the inbox folder.
Returns a unique
           verificationID for retrieving the status of the validation request'
  operationId: validationFileUsingPOST
  consumes:
    - application/json
  produces:
    - application/json
  parameters:
    -
      in: body
      name: request
      description: 'ST26 File name for validation'
      required: false
      schema:
        $ref: '#/definitions/ValidationRequest'
  responses:
    '200':
      description: OK
      schema:
        $ref: '#/definitions/ValidationResponse'
    '201':
      description: Created
    '202':
      description: 'Accepted. The ST26 file passed the formal validation and their
verification has
```

```
      started. This code will be complemented with a response message indicating a
unique code for
      retrieving the verification report (verificationID). ST26 file is moved to
the ◆process◆
      folder for processing'
      schema:
        $ref: '#/definitions/ValidationResponse'
    '400':
      description: 'Bad request. The REST request was not well formed or the ST26
file did not pass
      the XML validation. This code will be complemented with a response message
indicating
      the detail of the error'
    '401':
      description: Unauthorized
    '403':
      description: Forbidden
    '404':
      description: 'File not found. The ST26 file was not found in the Inbox
Folder'
    '500':
      description: 'Server Error. An internal error happened. This code will be
complemented with a
      response message indicating the detail of the error'
  deprecated: false
definitions:
  ValidationRequest:
    type: object
    required:
      - nameFile
      - type
      - verificationReportOutputPath
    properties:
      nameFile:
        type: string
        example: file.xml
        description: 'File Name Validation'
      type:
        type: string
        example: 'full or formality'
        description: 'Type of validation'
      currentApplicationNumber:
        type: string
        example: 1.3
        description: 'The application number associated with the sequence listing'
      currentSEQLVersionNumber:
```

```
        type: string
        example: 1.2
        description: 'the version number of this sequence listing (internally
assigned by an Office)'
      parentApplicationNumber":
        type: string
        example: WIPO-1234
        description: 'Any associated parent application'
      parentSEQLVersionNumber:
        type: string
        example: 1.1
        description: 'The version number of the parent’s sequence listing'

      seqlInputLocation:
        type: string
        example: /st26/inbox/file.xml
        description: 'Contains the path of the input xml file to be validated'
      verificationReportOutputPath:
        type: string
        example: /st26/outbox/file.xml
        description: 'Will contain the destination path of the report.xml generated
by the tool.'

    title: ValidationRequest
    description: 'Class representing a response Validation status File by the
application.'
  ValidationResponse:
    type: object
    required:
      - verificationID
    properties:
      errorMsg:
        type: string
      verificationID:
        type: string
        example: 1552208288697FNc2
        description: verificationID
    title: ValidationResponse
    description: 'Class representing a response Validation ST26 File by the
application.'
  ValidationStatusRequest:
    type: object
    required:
      - verificationID
    properties:
      verificationID:
```

```
        type: string
        example: 1552208288697FNc2
        description: verificationID
    title: ValidationStatusRequest
    description: 'Request of the validation status of an ST26 File'
  ValidationStatusResponse:
    type: object
    required:
      - status
    properties:
      status:
        type: string
        example: 'RUNNING - FINISHED_VALID - FINISHED_INVALID'
        description: 'Valdation Status File'
      reportPath:
        type: string
        example: /st26/reports/1552208288697FNc2/report_file.xml
        description: ReportFilePath
    title: ValidationStatusResponse
    description: 'Response with the validation status for a specific
verificationID.'
```

[Anhang III folgt]

Anhang III: Eigenschaftsnamen (JSON)


```
{
    "dtdVersion": "PROPERTY_NAMES.DTD_VERSION",
    "fileName": "PROPERTY_NAMES.FILE_NAME",
    "softwareName": "PROPERTY_NAMES.SW_NAME",
    "softwareVersion": "PROPERTY_NAMES.SW_VERSION",
    "productionDate": "PROPERTY_NAMES.PRODUCTION_DATE",
    "project": "PROPERTY_NAMES.PROJECT",
    "nonEnglishFreeTextLanguageCode":
"PROPERTY_NAMES.NON_ENGLISH_FREE_TEXT_LANGUAGE_CODE",
    "originalFreeTextLanguageCode":
"PROPERTY_NAMES.ORIGINAL_FREE_TEXT_LANGUAGE_CODE",
    "name": "PROPERTY_NAMES.PROJECT_NAME",
    "applicationIdentification": "PROPERTY_NAMES.APPLICANT_IDENTIFICATION",
    "applicationIdentification.filingDate":
"PROPERTY_NAMES.APPLICANT_IDENTIFICATION",
    "applicantFileReference": "PROPERTY_NAMES.APPLICANT_FILE_REFERENCE",
    "priorityInformationBag": "PROPERTY_NAMES.PRIORITY_INFORMATION_BAG",
    "earliestPriorityApplicationIdentification":
"PROPERTY_NAMES.EARLIEST_PRIORITY_APPLICATION",
    "earliestPriorityApplicationIdentification.filingDate":
"PROPERTY_NAMES.EARLIEST_PRIORITY_APPLICATION",
    "applicantName": "PROPERTY_NAMES.APPLICANT",
    "applicantList": "PROPERTY_NAMES.APPLICANT",
    "applicantName.name": "PROPERTY_NAMES.APPLICANT_NAME",
    "applicantName.languageCode": "PROPERTY_NAMES.APPLICANT_LANGUAGE_CODE",
    "applicantName.nameLatin": "PROPERTY_NAMES.APPLICANT_NAME_LATIN",
    "inventorName": "PROPERTY_NAMES.INVENTOR",
    "inventorList": "PROPERTY_NAMES.INVENTOR",
    "inventorName.name": "PROPERTY_NAMES.INVENTOR_NAME",
    "inventorName.languageCode": "PROPERTY_NAMES.INVENTOR_LANGUAGE_CODE",
    "inventorName.nameLatin": "PROPERTY_NAMES.INVENTOR_NAME_LATIN",
    "inventionTitleBag": "PROPERTY_NAMES.INVENTION_TITLE_BAG",
    "sequenceTotalQuantity": "PROPERTY_NAMES.SEQUENCE_TOTAL_QUANTITY",
    "sequenceDataBag": "PROPERTY_NAMES.SEQUENCE_DATA_BAG",
    "sequenceIDNumber": "PROPERTY_NAMES.SEQUENCE_ID_NUMBER",
    "length": "PROPERTY_NAMES.SEQUENCE_LENGTH",
    "INSDSeqMoltype": "PROPERTY_NAMES.SEQ_MOL_TYPE",
    "INSDQualifierMolType": "PROPERTY_NAMES.QUAL_MOL_TYPE",
    "organism": "PROPERTY_NAMES.ORGANISM",
    "division": "PROPERTY_NAMES.DIVISION",
    "INSDSeq_other-seqids": "PROPERTY_NAMES.INSDSEQ_OTHER-SEQIDS",
    "featureTable": "PROPERTY_NAMES.FEATURE_TABLE",
    "featureKey": "PROPERTY_NAMES.FEATURE_KEY",
    "featureLocation": "PROPERTY_NAMES.FEATURE_LOCATION",
```

```
    "featureQuals": "PROPERTY_NAMES.FEATURE_QUALS",
    "qualifierId": "PROPERTY_NAMES.QUALIFIER_ID",
    "qualifierName": "PROPERTY_NAMES.QUALIFIER_NAME",
    "qualifierValue": "PROPERTY_NAMES.QUALIFIER_VALUE",
    "qualifierTranslatedValue": "PROPERTY_NAMES.QUALIFIER_TRANSLATED_VALUE",
    "INSDSeqSequence": "PROPERTY_NAMES.SEQ_SEQUENCE"
}
```

Beachten Sie, dass dies ein Teil des Source-Codes ist und bei Bedarf zusammen mit der Implementierung aktualisiert wird.

[Ende des Dokuments]