WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

WIPO Sequence Validator web service

Webinar training

What we are covering today

- Purpose of the Validator service
- Comparison to the desktop tool
- What is a web service?
- Introduction to Validator endpoints (main functions)
- Operation and Installation of the web service
- Request/Response parameters
- Configuration of the web service

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

WIPO Sequence Validator: purpose

- **Aim**: ensuring filed sequence listings comply with WIPO ST.26
- **Implementation**: web service produced for IP Offices to validate filed sequence listing
- **IP Offices:** incorporate into their own IT environments
- **Applicants**: receive the same report indicating any errors/warnings

WIPO Sequence Validator vs WIPO Sequence Desktop

Validator service	Desktop Tool
Designed for Offices	Designed for applicants
Web service	Desktop application
No interface	Interface
Integrated into IT environment	Run by user
Two levels of validation	Only full validation
More than one DTD	Only latest DTD

What is a web service?

- Service operating between two electronic devices communicating over the Internet or Intranet
- Uses a standardized XML/JSON request/response format
- Does not need to be implemented in a particular programming language (language agnostic)
- Modular, dynamic and distributed
- A type of API that operates using SOAP or REST as means of communication
- A web service contract defines the “what”/”how” and “where”

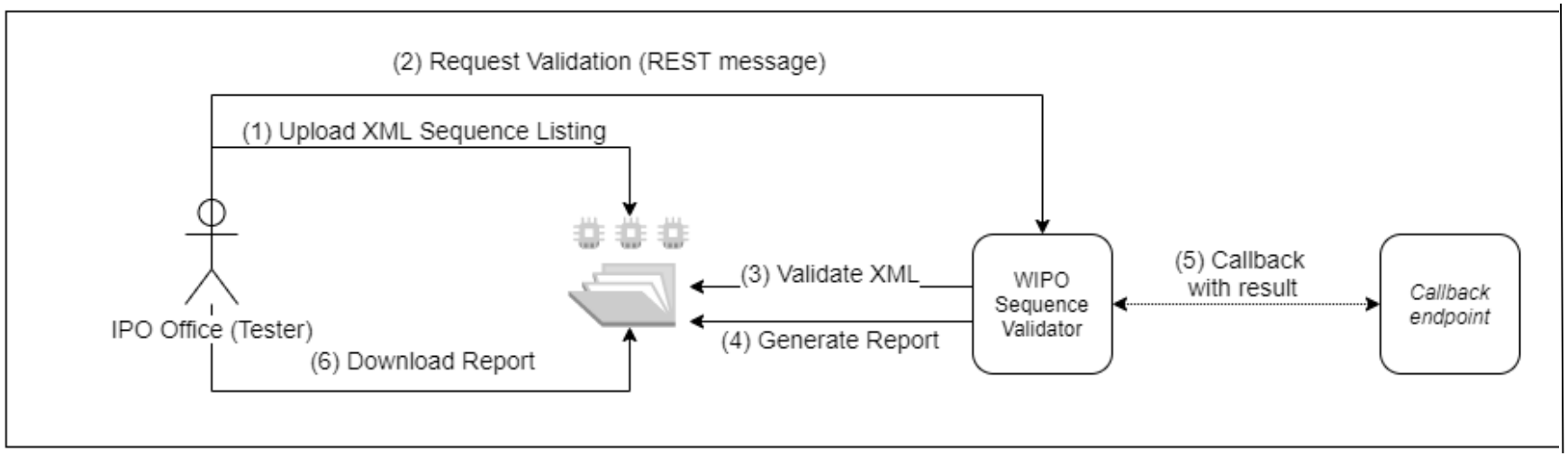
OAS API Specification

- Specification for RESTful APIs
- REST stands for REpresentational State Transfer – architecture which uses a subset of HTTP
- With this specification, both humans and computers can understand the capabilities of the API without accessing the source code
- Programming language agnostic

WIPO Sequence Validator: basics (1)

- Functions to:
  - Validate an ST.26 sequence listing
  - Provide status of validation
  - Callback to endpoint with messages from verification report
- Compliant with WIPO ST.90: https://www.wipo.int/export/sites/www/standards/en/pdf/03-90-01.pdf
- Web API specification: OAS 2.0
- Was not designed to be exposed externally

WIPO Sequence Validator: basics (2)



Installation of the Validator

- Requires Java 8 installed (Java 11 supported in the future)
- Folder structure must be established in parent location of where web service is located
- Basic default configuration can be tailored by providing own application.properties file
- Full set of instructions available in the WIPO Sequence Validator Operations Manual: https://www.wipo.int/export/sites/www/standards/en/sequence/wipo_sequence_validator_operations_manual.pdf

Default Folder structure

temp/st26

- temp/st26/inbox
- temp/st26/outbox
- temp/st26/params
- temp/st26/process
- temp/st26/reports

Deployment of the web service

- There are two types of services provided: WAR and JAR
  - JAR – built in server
  - WAR – Offices will require a Tomcat server, or any application server compatible with Spring Boot 2 and Servlet Spec 3.1+ installed

- Swagger UI can be used to provide basic parameters/perform testing at: http://[host-name]:8080/swagger-ui.html

- **Offices must provide their callback endpoint as this does not form part of the tool**

Deployment of Service as a JAR

- **java –D”file.encoding-UTF-8” –jar wipo-sequence-validator.jar**

- Default port is 8080 but the server port can be changed using:

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

Deployment of Service as a WAR

- For Tomcat 8.5:
- Stop server: “$TOMCAT_ROOT\bin\catalina.bat stop”
- Copy WAR to “$TOMCAT_ROOT\webapps\wipo-sequence-validator.war”
- Start server: “$TOMCAT_ROOT\bin\catalina.bat start”

- “$TOMCAT_ROOT” refers to the root folder of the Tomcat application server

- On initiation of application server, must set file encoding to UTF-8

- Default port is again 8080. To change, the “`port`” attribute in the Tomcat configuration file must be set. Reference: https://tomcat.apache.org/tomcat-8.5-doc/config/http.html#Common_Attributes

Validate function

- **Description:** Request the validation of an existing ST.26 file in the inbox folder. Returns a unique verificationID for retrieving the status of the validation request upon request
- **Endpoint (JAR):** http://[host-name]:8080/api/[version]/validate
- **Endpoint (WAR):** http://[host-name]:8080/wipo-sequence-validator/api/[version]/validate
- **Produces**: application/json
- **Input**: nameFile and type of validation
- **Output**: verificationID and verification report

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

“Full” versus “Formality” validation

- There are two types of validation that the Validator can perform: *full* and *formality*
  - **Formality**: Is the XML file valid and complies with the ST.26 DTD (severity: XML_WARN | XML_ERROR)
  - **Full**: formality validation AND compliance verification rules derived from WIPO ST.26 (severity: ERROR | WARN)

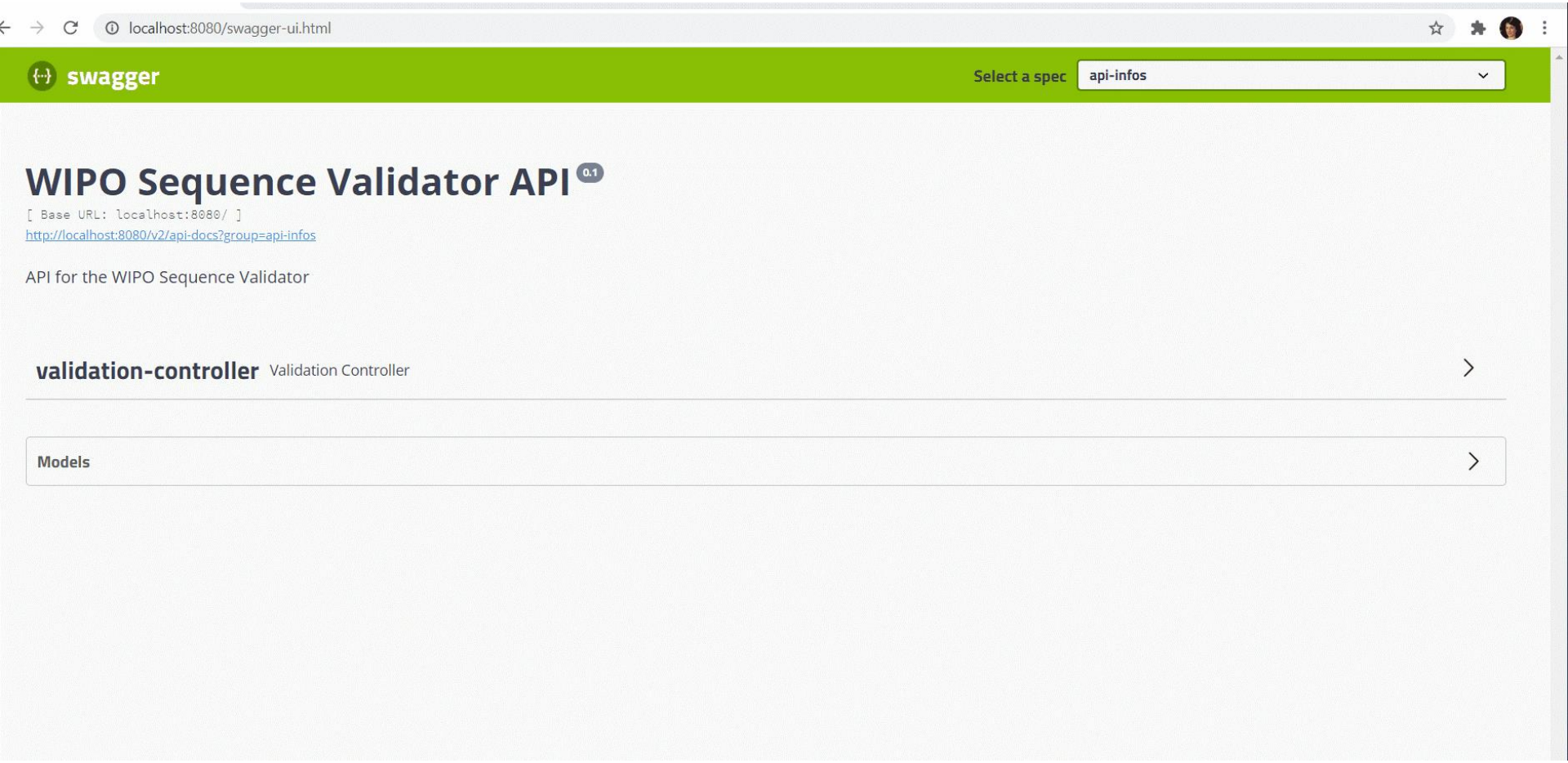
- Formality validation can be provided synchronously to users at filing. The full validation should be performed asynchronously

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

After validation

- Sequence listing file is moved to the local “Outbox” folder
- Verification report is generated in local “Reports” folder
- Temporary folders in local “Process” folder are deleted
- There is a verification ID which identifies the particular verification process and the folder which is created is labelled with this ID
- Optional callback to endpoint to provide response which includes verification report (if configured by IP offices)

Swagger UI (demo)



[hostname]:8080/swagger-ui.html

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

(Check) Status function

- **Description:** Request the validation status for a specific ST.26 File
- **Endpoint (JAR)**: http://[host-name]:8080/api/[version]/status
- **Endpoint (WAR):** http://[host-name]:8080/wipo-sequence-validator/api/[version]/status
- **Produces**: application/json
- **Input**: verificationID
- **Output***:* status (RUNNING/FINISHED_VALID/FINISHED_INVALID/NOT_FOUND/VERIFATION_ID_ERROR)

Callback Endpoint

- Callback endpoint: a separate API that receives the information from your web service
- Validation can also be performed by the callback endpoint making a call to the web service
- Will be performed asynchronously
- Must comply with the web service contract provided in the operations manual
- Request and response formats must also be provided in the standard format defined

Request format

- Validate function

```
{
  "currentApplicationNumber": "string",
  "currentSEQLVersionNumber": "string",
  "parentApplicationNumber": "string",
  "parentSEQLVersionNumber": "string",
  "seqlInputLocation": "string",
  "verificationReportOutputPath": "string",
  "nameFile": "file.xml",
  "type": "full or formality"
}
```

- Status function

```
{
  "verificationID": "1552208288697FNc2"
}
```

Response provided (1)

- **processID**: already provided by WIPO Sequence Validator
- **seqlType**: a fixed value: 'ST.26'
- **httpStatus**: HTTP Status code from service: 'Success' / 'Failure' along with the HTTP Error Code and corresponding description
- **applicationNumber**: for the patent application the sequence listing is a part of e.g., PCTUS1234567 or 23456789.
- **currentSEQLVersionNumber**: the version number of this sequence listing (internally assigned by an Office e.g., 1.1.
- **parentApplicationNumber**: any associated parent application e.g., 12345678.
- **parentSEQLVersionNumber**: the amendment number of the parent's sequence listing (internally assigned by an Office e.g., 3.2.
- **verificationReportOutputPath**: provides the complete path for the report
- **startTime**: the time the service started validation e.g., 2019-07-29 15:59:37.784
- **endTime**: the time the service concluded the validation e.g., 2019-07-29 15:59:37.859
- **elapsedTime**: provides the total time required for validation of the sequence listing e.g., 0 hr(s) 0 min(s) 0 sec(s) 75 ms

…

Response provided (2)

- **totalWarningQuantity**: sum of all the warnings from the verification report e.g., 2.
- **totalErrorQuantity**: sum of all the errors from the verification report e.g., 0.
- **seqInputQuantity**: total number of sequences from general information section in project input by the applicant e.g., 10.
- **seqIDQuantity**: verification step to check the number of sequences in the sequence listing against the number input by the applicant e.g., 10.
- **errorSummary** a summary of the errors reported in the verification report, with the following details:

[

{errorIndexID (this is the sequential numbering of each error code reported, e.g., 1, 2, 3, etc) : , errorCode: (numerical or alphaNumeric identification of the error, e.g., 1(####) for errors, or 2(####) for warnings) , description (error code-description pair) :, sequenceNumber (SEQ ID), xPath: (It can be used to navigate through elements and attributes in an XML document e.g., /ST26SequenceListing/SequenceData[1]/INSDSeq/INSDSeq_sequence) , characterRange: (e.g., 10-20 starting from character position 10 to position 20 in the element INSDSeq_seqence)

} , { (same information reported for the next error) }, { .. .. }

Verification report

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<verificationReport productionDate="YYYY-MM-DD" sourceFileName="[ST.26 filename]">
    <verificationMessages>
        <message>
            <severity>[ERROR | WARN | XML_WARN |  XML_ERROR]</severity>
            <dataElement>[ST.26 element]</dataElement>
            <detectedSequence>[Sequence ID]</detectedSequence>
            <detectedValue>[value]</detectedValue>
            <messageKey>[Message key]</messageKey>
            <params>
                <param key="param key">Param value</param>
            <params>
            <localizedMessage> [Localized message] <localizedMessage>
        </message>

...

    </verificationMessages>
</verificationReport>
```

Verification report: Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VerificationReport productionDate="2021-05-10" sourceFileName="EPOShort.xml">
    <VerificationMessageBag>
        <VerificationMessage>
            <Severity>WARNING</Severity>
            <DataElement>PROPERTY_NAMES.FEATURE_QUALS</DataElement>
            <DetectedSequence>2</DetectedSequence>
            <DetectedValue>2'-O-methyladenosine</DetectedValue>
            <MessageKey>X_FEATURE_QUALS_QUAL_NOTE_VAL_OTHER</MessageKey>
            <ParameterBag/>
            <LocalizedMessage>Verify that the 'note' qualifier describes the 'OTHER' value for the 'mod_base' qualifier.</LocalizedMessage>
        </VerificationMessage>
        <VerificationMessage>
            <Severity>WARNING</Severity>
            <DataElement>PROPERTY_NAMES.FEATURE_QUALS</DataElement>
            <DetectedSequence>6</DetectedSequence>
            <DetectedValue>2'-O-methyladenosine</DetectedValue>
            <MessageKey>X_FEATURE_QUALS_QUAL_NOTE_VAL_OTHER</MessageKey>
            <ParameterBag/>
            <LocalizedMessage>Verify that the 'note' qualifier describes the 'OTHER' value for the 'mod_base' qualifier.</LocalizedMessage>
        </VerificationMessage>
        <VerificationMessage>
            <Severity>ERROR</Severity>
            <DataElement>PROPERTY_NAMES.APPLICANT</DataElement>
            <DetectedSequence></DetectedSequence>
            <DetectedValue>{"name":"DAIICHI SANKYO COMPANY, LTD.","languageCode":""}</DetectedValue>
            <MessageKey>LANGUAGE_CODE_MISSING</MessageKey>
            <ParameterBag/>
            <LocalizedMessage>The language code has not been entered.</LocalizedMessage>
        </VerificationMessage>
    </VerificationMessageBag>
</VerificationReport>
```

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

Configuration of parameters- default

- Base path to be used by the rest of folders=**../temp/st26/**
- Folder to put the files to be processed=Base Path+/**inbox/**
- Folder to store the ST26 files once validated=Base Path+/**outbox/**
- Folder to store the validation reports=Base Path+/**reports/**
- Parent folder for full and formality folders=Base Path+/**process/**
- Folder to store the parameters=Base Path+/**params/**
- AlternativeResourceBasePath=Base Path+/**alt_resources**
- Localization used for the localized messages in the verification report = **en**
- URL of the callback endpoint that will be used for informing of the results of the validation=**http://callbackservice/api/endpoint**
- **Also logging etc.**

Alternative Configuration (1)

- Configuration is possible through provision of a new application.properties file

- If an alternative application.properties file is provided the tool will search for an alternative in the following order:

- A “/config” folder within the current directory
- The current directory
- A $classpath or config package; then
- The $classpath root OR
- At the location specified using parameter on command line

- **Note: The Validator will need to be restarted before the new configuration is applied**

Alternative Configuration (2)

- In particular, the two following two customizations can be made:

- Message language for verification report – set the "`validator_locale`" parameter of the application.properties file must be set to the appropriate language code e.g., "`validator_locale=es`" to change the language to Spanish.

- Custom organism names: provide a single JSON file of organism names which do not form part of the packaged list as:

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

- The new custom_organism.json file should be located at location indicated by alternativeResourceBasePath.

Alternative configuration: DTD (1)

- By default, the Validator service will validate against the latest DTD : provided under "`/src/main/resources`" (currently version 1.3)

- It is possible to also validate against an older DTD, using two different methods:
  - First:
  - Uncompressing the JAR file and include a reference to the additional or alternative ST.26 DTD file in the "`src/main/resources`" folder;
  - Modifying the "catalog.xml" file by adding a new entry for the additional ST.26 DTD or editing the existing one.

Alternative configuration: DTD (2)

- Second, instead of modifying the JAR file:
  - Copy “catalog.xml” and all the DTDs into a local folder;
  - Modify “catalog.xml” to include a reference to the additional ST.26 DTD; and
  - Set this Java system property on launch: “xml.catalog.files=<path_to_catalog.xml>”

- **Note: this will only work for the formality check as there may be updates necessary to the business verification rules derived from the body of the Standard**

Q&A session

standards@wipo.int

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

References

- WIPO Sequence home page (Validator binaries and operations manual in English):

https://www.wipo.int/standards/en/sequence

- WIPO Standard ST.26:

https://www.wipo.int/export/sites/www/standards/en/pdf/03-26-01.pdf

- Implementation of WIPO ST.26 FAQ:

https://www.wipo.int/standards/en/sequence/faq.html

- OAS 2.0 specification:

https://swagger.io/specification/v2/