Daten
Competence
Center e. V.

Functional interface description

Planning review for kitchen furniture

Based on the API documentation dated February 18th, .2020
(Version 1.2) - DCC, design review task force, derived from
the API documentation for design validation created by
Frank Nanninga (RMTSoft GmbH & Co. KG)

Content

Introduction

The interface description explains the kitchen furniture planning review. Based on the electronic form of EDI / EDIGRAPH data.

Concepts and Process

The relevant data transmitted by the sending system via REST and the receiving system confirms the receipt. After the validation has been carried out, the processed result will send back to the calling system via REST. Alternatively, the result can also be retrieved from the sending system via HTTP-GET.

With regard to data protection, attention is drawn to compliance with the GDPR. We recommended to use HTTPS and TLS in version 1.2 or higher for data transmission.

Format Guidelines

The planning data must be stored in industry-standard file format (in one ZIP compressed archive with EDI / EDIGRAPH file per planning supplier).

The EDI file format is based on EANCOM 97A. The EANCOM file within this ZIP file is always named "ORDER.EDI". The address data should be anonymized for data protection reasons. In order to be able to carry out a statistical evaluation, it would be desirable to make the commission number anonymous. This could be done using a hash algorithm. This hash value should be unique for each commission. In the JSON structure there is a field "commissionHash.

The validation result is answered in the form of a JSON structure, whereby the structure is "lower case sensitive" and therefore correct upper- and lower-case letters must be observed.

All time stamps in the JSON must be in ISO 8601 format (UTC time) in order to be prepared for working with suppliers and customers outside the German time zone.

```
YYYY-MM-DDTHH:MM:SS.ffffZ
```

Authentication

Authentication is via OAuth2 (Client Credential Flow). The OAuth2 token is transmitted as a bearer token via the HTTP header in the Authorization field.

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
```

Error Messages

Each HTTP request received from a server is answered with an HTTP status code.
In case of a potential issue, the server will respond with one of the following HTTP status codes.

Status Code	Description	Meaning
400	BadRequest	Server cannot process the request
401	Unauthorized	Invalid authentication
412	Precondition Failed	Invalid unique ID
422	Unprocessable Entity	Corrupt Data
500	InternalServerError	Unexpected server error
502	BadGateway	Invalid response from another server
503	ServiceUnavailable	Service not available

The sending system displays the error messages in the relevant language.
Optionally, the server responds with a JSON message as additional information to classify the potential issue.

```
{
  "error" : "CorrelationId not found"
}
```

Troubleshooting Test

To simulate negative responses, add the key “OCS-Mock-Response” to the request header in your API call. This field contains a JSON structure with the following content.

```
{
  "mock_application_code": "INTERNAL_SERVER_ERROR"
}
```

Available error codes.

Status Code	Value	Description
200	NO_RESULT_PROVIDED	No result available
400	BAD_REQUEST	Server could not process the request
401	NOT_AUTHORIZED	Request cannot be completed without valid authentication
412	PRECONDITION_FAILED	Invalid unique ID
422	UNPROCESSABLE_ENTITY	Corrupt Data
500	INTERNAL_SERVER_ERROR	Unexpected server error
502	BAD_GATEWAY	Server could not fulfil its function as a gateway or proxy
503	SERVICE_UNAVAILABLE	Server is temporarily unavailable

REQUEST - Send Planning Review

To execute the planning review API, you have to put the planning data as content in a JSON structure. Send that JSON structure with an HTTP POST request to the planning review API.

The JSON structure contains the following mandatory fields (optional "buyerQualifier", "replyTo")
The replyTo field is the specification for an endpoint to receive the result.
For example, a separate endpoint could be specified for development.
The endpoint for the productive environment would then be stored in the web service.

API Endpoint: https://{url}/ordervalidation

Fieldname	Description	Type
requestedBy	Name (ID) of the planning program	string
requestedByVersion	Version of the planning program	string
replyTo	***(optional)*** https://{reply-server}/url/endpoint	string
supplier	Manufacturer's IDM number according to IDM tables	string
buyer	Customer number of the dealer at the manufacturer	string
buyerQualifier	***(optional)*** Identifier for the customer number	string
commissionHash	Hash value of the anonymized commission	string
language	ISO 3166 language code according to IDM tables	string
mimetype	Media type see RFC2045 (Chapter 5)	string
content	ZIP File (see above Format Guidelines), base64 encoded	string

```
{
   "requestedBy"  :  "YOUR-APP-NAME",
   "requestedByVersion" : "v1.0",
   "replyTo" : "https://reply-server/url/endpoint",
   "supplier"  :  "2222",
   "buyer"  :  "3333",
   "buyerQualifier"  :  "GLN",
   "commissionHash" : "2D3YKwf1AWc3 …",
   "language"  :  "DE",
   "mimetype"  :  "application/zip",
   "content"  :  "UEsDBBQAA ... (Order.zip content) ..."
}
```

RESPONSE - Send Planning Review

If the request is successful and contains all valid data, then the receiving system sends back a JSON structure containing a "correlationId". This "correlationId" is a unique ID that can be used to match the result using this ID.

Fieldname	Description	Type
messageId	Unique message identifier	string
correlationId	Unique GUID for the entire operation	string
issued	Time stamp of the response in ISO8601 format	string
status	Status of the verification: - received (Receive data)	string

```
{
  "messageId" : "ID:PCSPC13-28421-1350452145851-11:1:1:1:1",
  "correlationId" : "79bcafb8-e1e7-4b42-91d5-9bd42e9e6eec",
  "issued" : "2022-02-15 10:10:00",
  "status" : "received"
}
```

Provide result via callback

An end point is required in the calling system in order to be able to receive the validation result. The authentication on the recipient system can be done via OAuth2.

Retrieve result (Polling)

The retrieval is done via an HTTP GET in combination with the specified ID, so no additional JSON data is required for the query.

API Endpoint: https://{url}/ordervalidationresult

Structure of the URL:

```
https:// << any-url >>/ordervalidationresult?correlationId=79bcafb8
```

This method would be important for standalone solutions, so the callback method would be preferable.

Response after verification completed

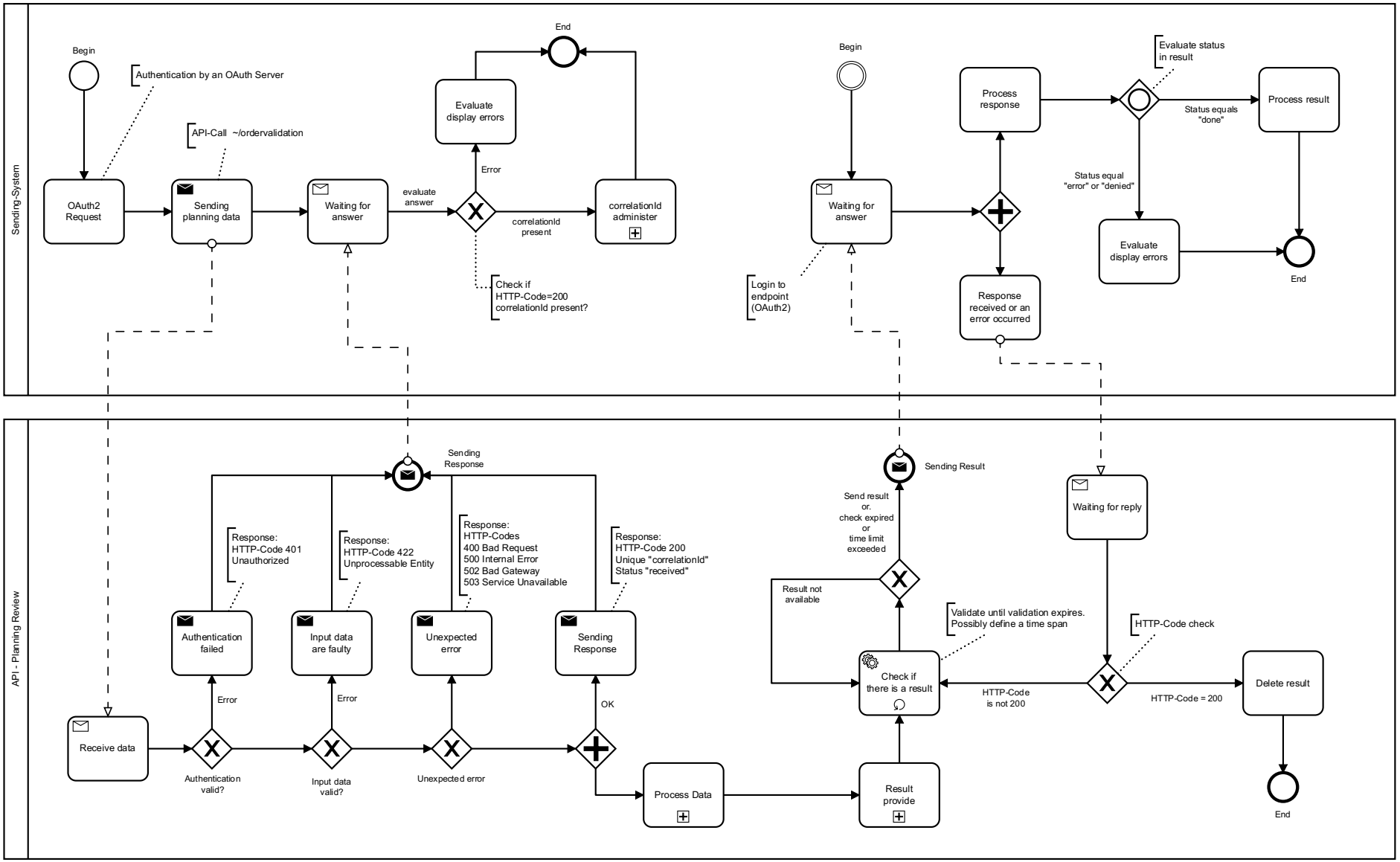
After the verification has been completed, the response contains the following data structure.

Fieldname	Description	Type
messageId	Unique message identifier	string
correlationId	Unique GUID for the entire operation	string
issued	Time stamp of the response in ISO8601 format	string
status	Status of the verification: - done (Verification completed successfully) - error (Verification error) - denied (Verification was rejected)	string
supplierName	Manufacturer name	
logo	Manufacturer logo as svg, base64 encoded	string
serviceLine	Manufacturer information, e.g.: a service hotline	string
resultMessages	List of results (Count 0-n)	array
number	Unique number of the validation rule	number
name	Name of the validation rule	string
levelCode	Level of the validation rule - 1 = info - 2 = warning - ~~3 = validate~~ (reserve) - 4 = error	number
message	Short text of the message	string
description	Long text of the message	string
lineItems	List of affected articles (0-n)	array
ediReferenceNumber	EDI reference number of the item	string
ediPlanningNumber	EDI planning system position number	string
type	Item description	string
variants	List of variants (see below)	array
media	List of additional information (Count 0-n)	array
description	Description	string
mimetype	Media type e.g., image/png	string
content	Binary content, base64 encoded	string
externals	List of external links (Count 0-n)	array
description	Description of specified URL	string
source	URL of the external link	string
variants	List of variants (Count 0-n)	array
headerNumber	Variants head number	number
variantType	Variant type	number
variantValue	Value of the variant	string

Example of the Result


```
{
  "messageId"  :  "ID:PCSPC13-28423-1540452147871-13:1:1:1:1",
  "correlationId"  :  "79bcafb8-e1e7-4b42-91d5-9bd42e9e6eec",
  "issued"  :  "2022-02-15 10:10:23",
  "status"  :  "done",
  "logo" : "image/svg+xml;base64,PHN2Zy . . .",
  "serviceLine" : "Service Hotline (+49)-12-(34)",
  "resultMessages" : [{
     "number" : "8",
     "name" : "Refrigerator door not centered at handle position 6",
     "levelCode" : 1,
     "message" : "The appliance door is supplied with handle position 2",
     "description" : "Possibly unassembled handle! Handle not in the middle!",
     "lineItems" : [{
         "ediReferenceNumber" : "9",
         "ediPlanningNumber" : "9.0",
         "type" : "GR201122Z",
         "variants" : []
     }],
     "media" : [{
         "description" : "Text for the graphic",
         "mimetype" : "image/png",
         "content" : "iVBORw0KGgoAA . . ."
     }],
     "externals" : [{
         "description" : "Description of specified URL",
         "source" : "https://sample.com/info/movie.mp4"
     }],
     "variants" : [{
         "headNumber" : 1,
         "variantType" : 100,
         "variantValue" : "6005",
     }]
  }]
}
```

Flowchart (Callback)



Flowchart (Polling)