USS Connect

Interface description ordering for partner by Unitedprint

1 Introduction

This document describes the SOAP-interface of Unitedprint (UP) controlling API-Calls (Application Programming Interface).

The interface creates the possibility to create order at UP, getting status information from UP and to define new product configurations.

This documentation contains all information that is necessary to use the interface for UP-partners.

A valid UP customer account, customer number and password is required as well as a certificate provided by UP for the use of the interface. This creates the prerequisites for secure access to the data provided by UP. The customer account must be enabled by UP for interface operations.

The certificate is used for communication with the product catalogue as well as for the order placement.

Described in detail:

- Creation of the certificates
- Data management of the product catalogue
- Creating order
- Transmission of status values for the partner order

The documentation and some code examples are created in PHP on a windows system.

2 Customer account

A prerequisite for use of the interface is a UP account.
If you do not already have a valid customer account, please register with one of the UP-portals in your country.

3 Certificates

Three independent certificates can be used for communication to use the interface.

- Sample (Open test system) – Provided in downloadarea
- Dev (customer account test system) – Provided from UP-IT
- Productive system (customer account live system) – Provided from UP-IT

The certificates provided by UP-IT are packed files (archives) in the format **.RAR*.

The following tools are recommended for further processing:

- WinZip / WinRar
- OpenSSL

As installation path for OpenSSL we use the following directory:

C:\OpenSSL-Win32

The archive provided by UP-IT contains the following files:

- *accountname_**dev**_date.crt*
- *accountname_**dev**_encrypted.key*
- *accountname _**prod**_date.crt*
- *accountname _**prod**_encrypted.key*

These files are stored in the following directory:

„C:\Work\Cert“

To decrypt the files your UP customer number is necessary.

- Dev-System

Call *„C:\OpenSSL-Win32\bin\openssl.exe“*

rsa -in *„C:\Work\cert\accountname_**dev**_encrypted.key“* -out *„C:\Work\cert\accountname_**dev**_decrypted.key“*

- Productive system

Call *„C:\OpenSSL-Win32\bin\openssl.exe“*

rsa -in *„C:\Work\cert\accountname_**prod**_encrypted.key“* -out *„C:\Work\cert\accountname_**prod**_decrypted.key“*

Now the final certificate files have to be created.

- Sample

„Sample.crt“ is already finished and needs no further processing.

- Dev-System

Content oft he file *„accountname_**dev**_20161004.crt“* including

-----BEGIN CERTIFICATE-----

and

-----END CERTIFICATE-----

is copied into a new text file with a text editor.

The text block from

-----BEGIN RSA PRIVATE KEY-----

to

-----END RSA PRIVATE KEY-----

from the file *„accountname_**dev**_decrypted.key“* will be attached after the certificate block. Afterwards the file is saved e.g. as *„accountname_**dev**.cert“.*

- Productive system

The area of the file *„accountname_**prod**_20161004.crt“* from

-----BEGIN CERTIFICATE-----

to

-----END CERTIFICATE-----

is copied into a new text file with a text editor.

From the file *„accountname_**prod**_decrypted.key“* the text block from

-----BEGIN RSA PRIVATE KEY-----

to

-----END RSA PRIVATE KEY-----

will be attached after the certificate-block. Afterwards the file is saved e.g. as *„accountname_**prod**.cert“*.

4 Construction of the SOAP connection

4.1 Setting up the SOAP client:

For all sample scripts, the password, customer number and portal name must be changed to the values sent by UP.

```
$args -> accountInformation -> password = 'pwd';
$args -> accountInformation -> customerNumber = 123456789;
```

```
$args -> accountInformation -> portalName = 'portal';
$args -> accountInformation -> languageCode = 'de';
$args -> accountInformation -> countryCode = 'DE';
```

4.2 Call the SOAP service

```
$options = array(
'soap_version' => SOAP_1_1,
'exceptions' => true,
'trace' => 1,
'cache_wsdl' => WSDL_CACHE_NONE,
'local_cert' => 'C:/xampp/htdocs/js/Certs/sample.cert',
'charset' => 'utf8',
'stream_context' => stream_context_create(['ssl' => ['verify_peer' => false, 'verify_peer_name' => false,
'allow_self_signed' => true]])
);
```

The variable *„stream_context“* has to be set only in the sample or Dev system, in the live environment it has to be omitted.

The variable *"local_cert"* specifies the certificate used for the sample, dev or productive system.

```
$soap_service = "http://interface.onts.unitedprint.dev/wsdl2.1";
```

In the variable the web service is specified.

- Sample and Dev: *“http://interface.onts.unitedprint.dev/wsdl2.1"*
- Productive: *"http://interface.unitedprint.com/wsdl2.1“*

5 Product catalogue

In the product catalogue all products that can be ordered are maintained.

To order a product by UP, this must first be created via the catalogue interface. For this all properties of the product are defined and stored under an article number. This item number can then be used in the order processing.

5.1 Creation of a new product

http://interface.unitedprint.com/ws2/catalogue.php

The individual product properties are selected. A unique combination of all available options is stored as a *„product“*. This *„product“* will be used later in the ordering process in the field:

orderPositions[$pos] -> orderInformation -> customerProductId

The fields *„order quantity“* and *„shipping type“* which are stored by the product are overwritten with a later purchase.

5.2 Product properties by creating a new product

Product properties using the example product „Flyer“

- product
- page
- format
- format direction
- paper
- paper description
- color
- finishing
- finishing description
- processing
- creasing
- perforation
- amount
- shipping

The selection of the properties must be from „top“ to „bottom“ in order to be able to exclude combinations which cannot be ordered. The overlying properties have an influence on which options can be selected in the groups below.

5.3 Display product catalogue

http://interface.unitedprint.com/ws2/catalogueList.php

Here the products saved by the partner of the catalogue are listed with their group assignment

6 Order creation

6.1 General

With creating an order, a product previously created in the product catalogue is sent to the interface together with the order data, order quantity and shipping method.

Order quantity and shipping type are overwritten in the following fields:

orderPositions[$pos] -> orderInformation -> orderQuantity

orderPositions[$pos] -> orderInformation -> shippingType

6.2 Script

„http://interface.unitedprint.com/ws2/order.php“

In the example the parameter *„validate“* can be set

```
$args -> validate = ($_GET{'validate'} == 'on' ? 1 : 0);
$erg = $client -> createOrder($args);
```

This parameter can be used to check an order. With *„validate = 1“* no order is created, only a check of the order is done, without saving it at UP.

If the check is successful, the expected delivery date, net and gross price as well as the currency are returned.

With *„validate = 0”*

```
$args -> validate = 0
$erg = $client -> createOrder($args);
```

The order is written in the UP system and the order number is returned.

The return values of the order unit are explained in the next chapter.

Required fields for the order creation are:

- environment
  - portalName
  - countryCode
  - languageCode
  - partyGroupName
- accountInformation
  - customerNumber
  - password
- orderInformation
  - customerProductId
  - orderQuantity
  - shippingType
  - orderName
- dataSourceInformation
  - dataSourceHost
  - dataSourceUser

- dataSourcePassword
- dataSourcePort
- dataTransmission

- messageLevel
  - all, medium, silent

6.3 PDF-Download FTP

The download of the product PDF usually takes place via a FTP server.

```
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourceUser = 'username';
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourcePassword = 'passw0rd';
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourcePort = 21;
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourceHost = 'ftp://testftp/filename.pdf';
$args -> orderPositions[$pos] -> dataSourceInformation -> dataTransmission = 'download';
```

Here the access data for the FTP server (username, password, port) and the name of the PDF to be transferred to UP are specified. The field *„dataTransmission" must contain the value "download".*

6.4 PDF-Download URL

The PDF download can also be carried out via the URL of a web server, access to the PDF must be possible and the variables

„dataSourceUser"

and

„dataSourcePassword"

must not be empty.

```
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourceUser = 'xxx';
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourcePassword = 'yyy';
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourcePort = 21;
$args -> orderPositions[$pos] -> dataSourceInformation -> dataSourceHost = 'ftp://testftp/filename.pdf';
$args -> orderPositions[$pos] -> dataSourceInformation -> dataTransmission = 'download';
```

6.5 Order_simple.php

Very simple example in which 250 flyers with delivery address and PDF-upload are ordered.

```
$args -> orderPositions[$pos] -> orderInformation -> shippingType = "Express";
```

Possible values for the shipping type are:

- economy
- standard
- express
- priority

These shipments may be restricted depending on the product.

$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> salutation = "Herr";

Possible values for the salutation are country-independent:

- Herr (Mr)
- Frau (Mrs)

6.6 Order_senderaddress.php

In this example the senders address of an order is changed.

The possible address fields are filled with sample data.

```
$pos = 0;
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> company   = "Abs_Firma";
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> salutation = "Herr";
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> firstName  = "Abs_Vorname";
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> lastName = "Abs_Nachname";
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> street = "Abs_Strasse";
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> zipcode  = "90429";
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> city  = "Abs_Ort";
$args -> orderPositions[$pos] -> shippingInformation -> senderAddress -> country   = "DE";
```

6.7 order_differentinvoiceaddress.php

In this example a different invoice address is created in the order.
The possible address fields are filled with sample data.

```
$args -> paymentInformation > invoiceAddress -> company = "Firma";
$args -> paymentInformation > invoiceAddress -> salutation = "Herr";
$args -> paymentInformation > invoiceAddress -> firstName = "Vorname";
$args -> paymentInformation > invoiceAddress -> lastName = "Nachname";
$args -> paymentInformation > invoiceAddress -> street = "Strasse";
$args -> paymentInformation > invoiceAddress -> zipcode = "90429";
$args -> paymentInformation > invoiceAddress -> city = "Ort";
$args -> paymentInformation > invoiceAddress -> country = "DE";
```

6.8 order_multiplepositions.php

This example creates an order with multiple purchase order items.
The possible fields are filled with sample data.

```
$pos = 0;
```

```
$args -> orderPositions[$pos] -> orderInformation -> customerProductId = "default_flyer";
$args -> orderPositions[$pos] -> orderInformation -> orderQuantity = 250;
$args -> orderPositions[$pos] -> orderInformation -> shippingType = "Economy";
$args -> orderPositions[$pos] -> orderInformation -> orderName = "MeinAuftrag";
$args -> orderPositions[$pos] -> orderInformation -> customerComment = "Kundenkommentar";
$args -> orderPositions[$pos] -> orderInformation -> customerCopy = 0;
```

In the first position a flyer with an order quantity of 250 and delivery type economy is created.

```
$pos = 1;
$args -> orderPositions[$pos] -> orderInformation -> customerProductId = "default_aufkleber";
$args -> orderPositions[$pos] -> orderInformation -> orderQuantity = 1000;
$args -> orderPositions[$pos] -> orderInformation -> shippingType = "Express";
$args -> orderPositions[$pos] -> orderInformation -> orderName = "MeinAuftrag";
$args -> orderPositions[$pos] -> orderInformation -> customerComment = "Kundenkommentar ";
$args -> orderPositions[$pos] -> orderInformation -> customerCopy = 0;
```

In the second position a sticker with an order quantity of 1000 and shipping type express is created.

6.9 order_differentdeliveryaddresses.php

In this example an order with several purchase order items is created and different delivery addresses are entered for the items.

The possible fields are filled with sample data.

```
$pos = 0;
$args -> orderPositions[$pos] -> orderInformation -> customerProductId = "default_flyer";
$args -> orderPositions[$pos] -> orderInformation -> orderQuantity = 250;
$args -> orderPositions[$pos] -> orderInformation -> shippingType = "Economy";
$args -> orderPositions[$pos] -> orderInformation -> orderName = "myOrder";
$args -> orderPositions[$pos] -> orderInformation -> customerComment = "CustomerComment";
$args -> orderPositions[$pos] -> orderInformation -> customerCopy = 0;

$pos = 1;
$args -> orderPositions[$pos] -> orderInformation -> customerProductId = "default_aufkleber";
$args -> orderPositions[$pos] -> orderInformation -> orderQuantity = 1000;
$args -> orderPositions[$pos] -> orderInformation -> shippingType = "Express";
$args -> orderPositions[$pos] -> orderInformation -> orderName = "myOrder_P2";
```

```
$args -> orderPositions[$pos] -> orderInformation -> customerComment = "CustomerComment_2";
$args -> orderPositions[$pos] -> orderInformation -> customerCopy = 0;

$pos = 0;
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses = array();
$address = new orderAddressType();
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[] = &$address;

$adr = 0;
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> company = "Firma";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> salutation = "Herr";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> firstName = "FirstName";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> lastName = "Lastname";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> street = "Street";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> zipcode = "90403";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> city = "City";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> country = "DE";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> partialQuantity = $args >
orderPositions[$pos] -> orderInformation -> orderQuantity ;

$address 2 = new orderAddressType();
$pos = 1;
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[] = &$address_2;

$adr = 0;
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> company = "Abw_Firma";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> salutation = "Herr";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> firstName = "Abw_Vorname";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> lastName = "Abw_Nachname";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> street = "Abw_Street";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> zipcode = "90429";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> city = "Abw_Ort";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> country = "DE";
$args -> orderPositions[$pos] -> shippingInformation -> shippingAddresses[$adr] -> partialQuantity = $args >
orderPositions[$pos] -> orderInformation -> orderQuantity ;
```

6.10 Field descriptions

type	field	selection	comment

CreateOrderRequestType	accountInformation		
	orderPositions		
	paymentInformation		
	validate	0, 1	Verification or execution of the order
	messageLevel	all, medium, silent	Auswahl der übertragenen Nachrichten (alle, wichtige)
accountInformation	password		from UP defined
(accountType)	customerNumber		from UP defined
	portalName		from UP defined
	languageCode		DE, Format ISO 639-1, 2 Zeichen
	countryCode		DE, Format ISO 639-1, 2 Zeichen
orderPositions	orderInformation		
(orderPositionsTypes)	dataSourceInformation		
	shippingInformation		
orderInformation	customerProductId		Product number defined in the catalog
(orderDataType)	orderQuantity		Quantity
	deliveryType	delivery, self_collect	Shipping, bulk shipping
	shippingType	economy, standard, express, priority	Possible delivery methods (Economy, Standard, Express, Priority)
	orderName		Name of the order, e.g. order number of the partner
	customerComment		Customer comment
	productAlias		Product description oft he partner
dataSourceInformation	dataSourceHost		Name of FTP-Server
(dataSourceDataType)	dataSourcePort		Port of FTP-Server
	dataSourceUser		Username of FTP-Server
	dataSourcePassword		Password of FTP_Server
	dataTransmission	upload, download	Transmissingtype for the product-PDF
shippingInformation	deliveryType	delivery, self_collect	Deliverytypes: shipping, collect shipping
(shippingDataType)	shippingAddresses		
	senderAddress		
shippingAddresses	Company		Company
(orderAddressType)	Salutation	Herr, Frau	Salutation
	firstName		Prename
	lastName		Lastname
	Street		Street
	houseNumber		Housenumber
	Zipcode		Zipcode
	City		City
	County		County
	Country		Country
	phonePrefix		Prefix phone number

	phoneNumber		Phone number
senderAddress	Company		
(shortAddressType)	Salutation	Herr, Frau	
	firstName		
	lastName		
	Street		
	City		
	Zipcode		
	County		
	Country		Country, format ISO 3166, 2 digits
senderAddress	Company		
(shortAddressType)	Salutation	Herr, Frau	
	firstName		
	lastName		
	Street		
	City		
	Zipcode		
	County		
	Country		Country, format ISO 3166, 2 digits
paymentInformation	paymentMethod	INV	Payment type
(paymentDataType)	invoiceAddress		
invoiceAddress	Company		
(longAddresType)	Salutation	Herr, Frau	
	firstName		
	LastName		
	Street		
	housenumber		
	Digicode		Door code
	City		
	countyCode		
	countryCode		Country, format ISO 3166, 2 digits

6.11 Return codes

In order to give the customer a confirmation of the order, the SOAP response return a wide range of message codes.

The digits of the total –digit return code correspond to the following meaning:

The answer in the

„response“

area applies in principle:

- response
  - responseCode
    For responseCode:
    - 0 = not successful
    - 1 = succesful

For responsecode 0:

1. Ziffer	2.- 6. Ziffer
1 – important	Parameter code (see overview return codes)
2 – missing parameter	
3 – invalid parameter	
4 – notice	

Numbers 2-6 of the return values each encode a key word which describes a particular data field (for example order quantity) or a situation (order).

If a keyword describes a data field, it is also determined whether the absence of this value for UP is a termination criterion (required), whether the value accepts only certain predefined values or whether a default value is set for a missing / incorrect default value)

Overview return values:

Message code	Keyword	Values	Required	Default
20001	aspect_ratio	landscape format | panel format	1	
20002	availability_type	1 | 2	0	1
20003	customer_copy	5 | 10 | 15	0	
20004	customer_item_id		0	
20006	item_id		1	
...				
30002	customer_number		1	
30003	data_source_host		0	
30006	data_source_port		0	
30007	data_transmission	ftp | download | e-mail | upload	1	download
30008	delivery_type	delivery | self collection	0	
30010	order_name		0	
30011	order_quantity		1	
30013	payment_method		0	
30015	shipping_type		0	
30016	partial_quantity		1	
30017	OrderCommitSucessful		0	
30018	OrderCommitFailed		0	
30019	email		1	
30020	password		1	
40005	address_id		1	
40014	address_data_search		0	
40015	address_ambiguous		0	
40016	address_new_successful		0	
40017	address_new_failed		0	
40018	party_group_code_failed		0	
40019	address_id_search		0	

7 Status request, tracking

The status of a purchase order is not queried (!) via the interface. Instead information is returned to the partner via the interface.

For status changes in the UP system, a script can be called at the partner via a

“http callback“

This is possible for any relevant status change.

Possible status changes are:

- Data processing
- Pooling
- Processing
- Prepare shipping
- Shipping
- Cancellation

For each script the transfer of different values is possible. Possible fields are:

- General information
  - Order number
- Position information
  - Status
  - Delivery date
  - Position number
- Delivery information
  - Delivery service
  - Trackingnumber
  - Partial amount

Return codes /mapping delivery service

- 2 DPD (www.dpd.com)
- 26 DHL Express National (www.dhl.de)
- 27 DHL (www.dhl.de)
- 31 Schenker (www.schenker.com)
- 35 Schenker (www.schenker.com)
- 49 DPD (www.dpd.com)
- 50 DPD (www.dpd.com)
- 52 GEL (www.gel-express.de)
- 77 DHL Paket (www.dhl.de)

Return codes /mapping status values

- order_stati_1750_4 Data processing
- order_stati_2000_4 Pooling
- order_stati_5000_4 Processing
- order_stati_5400_4 Prepare shipping
- order_stati_6000_4 Shipping
- order_stati_7000_4 Cancellation

UP must enable the customer to authorize the tracking information.

The registration of the scripts and selection of the fields to be transferred must be made by UP.

The called script on the customer server can evaluate the transferred information and handle it with the ERP- or shop system of the partner.