DX Despatch API - Requirement document

CONTENTS

Overview

DX have introduced an additional method to its tracking label production software suite, which gives our customers an alternative way to produce addressed tracking labels. The following document outlines the requirements for this development, called 'DX Despatch API'.

The DX Despatch (DXD) API allows customers to place Courier (B2B) and Secure (B2C) orders, using a single integration and also print the required tracking labels for each item to be sent.

DXD API is primarily developed for customers that have multiple users, over a number of different locations, e.g. a High Street chain of stores. However it can also be used by customers that have multiple packing stations in a single warehouse.

DXD API allows customers to generate and print DX tracking labels on their own systems; as part of their development, the customer systems needs to be developed to allow it to decide at which locations the relevant tracking labels are printed.

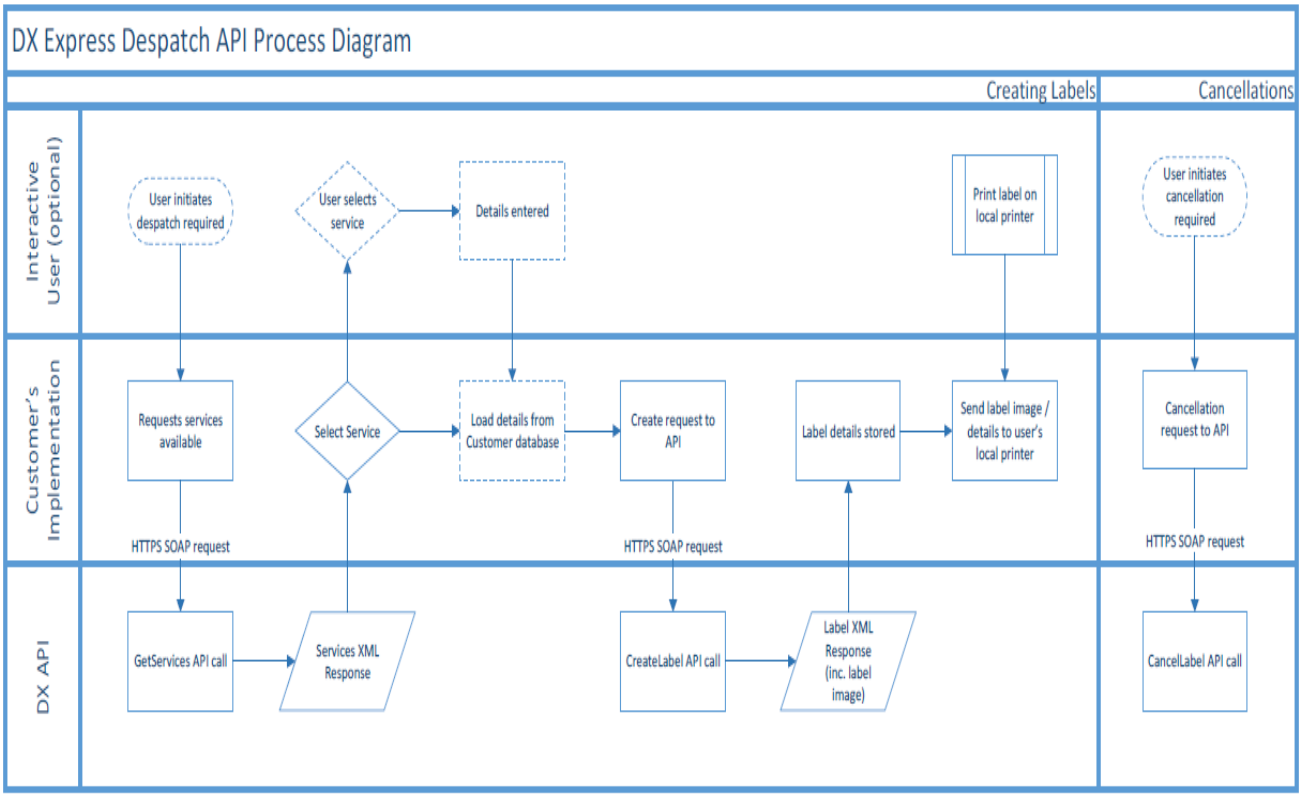
This document provides details to enable a DX customer to develop their systems to use the DXD API utility, whilst conforming to DX requirements.

Item tracking is not available through DX API. However the transactions created using API will be available to view immediately on DX Despatch; similar to how manually created DX Despatch labels can be viewed. There are alternative methods available to obtain tracking event details, which can be discussed with each customer as required.

Overview diagrams

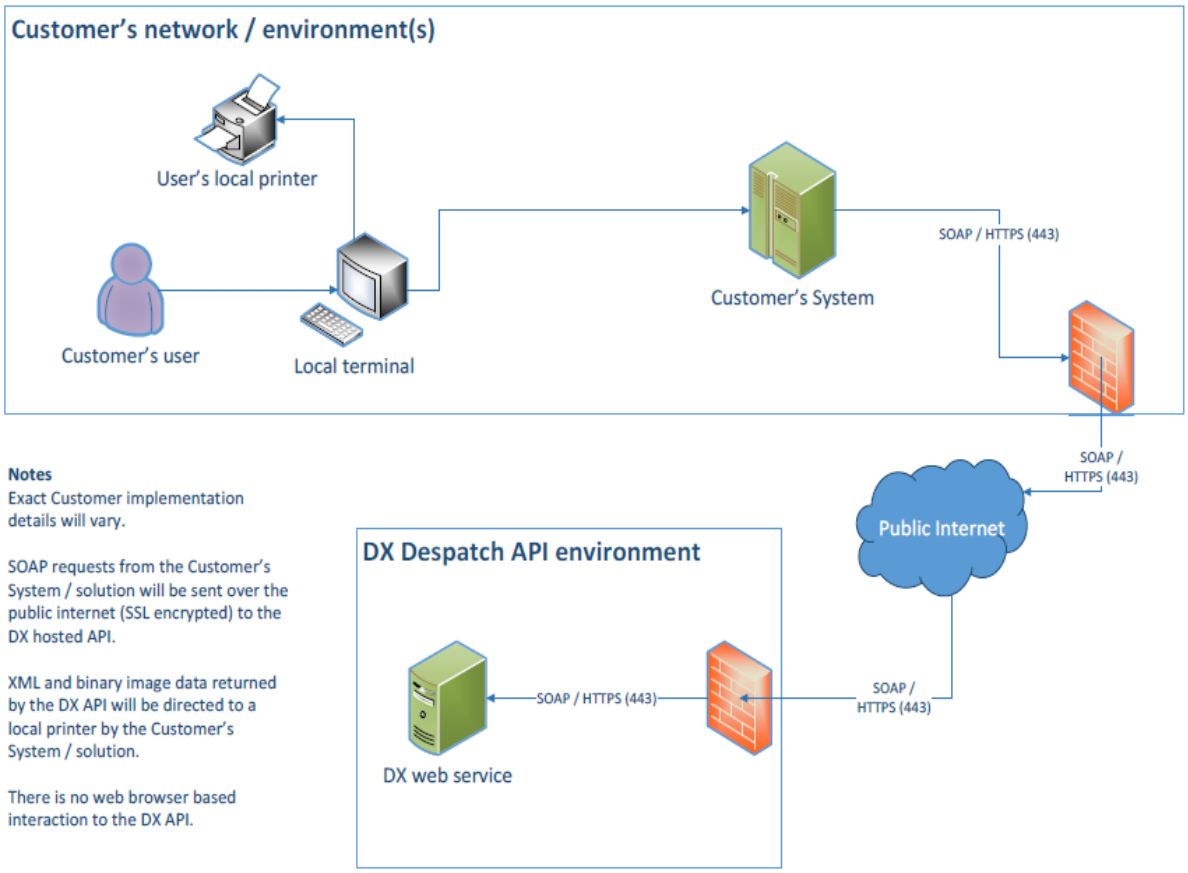
API Process diagram

The process diagram below summarises the API process:



API Solution diagram

The following diagram summarises the overall solution implementation:



API Integration

The API integration is a 3-step process:

- Customers system requests the required DX services, which have been set up in the DX systems, against the customers DX account numbers.
- Customer selects service and submits order details (address, delivery date etc.)
- DX replies with tracking number and label image (label printer & PDF supported)

Note: The list of service options available, user id and password will be supplied by DX.

GetServices

a. The request consists of the collecting location and the delivery address
b. The response contains the valid services available based on the input provided in the request.

- NOTE: Some DX services are linked to the delivery postcode and as such may result in different services codes being used. GetService should be used for each shipment being created, to ensure the correct code is used in the LabelCreation call.

API GetService Schemas:

Request:

Request

Structure					Occurs	Data Type	Description
getServicesRequest					1		XML definition
	serviceHeader				1	Node	This section deals with authentication.
		userID			1	String	UserID is the contact's username of the client. The contact will have been created solely for the use of the API.
		password			1	String	Password for the user. The expiry date of the password will NOT be checked.
	customerID				1	String	DX ERP assigned Customer Account Number for the site or account that should be charged for this order
	serviceFeatures[n]				0..1	Node List	Optional array of Key Value Pair to denote service features required. If "RESADD" is provided, return Secure only. If absent or "BUSADD" is provided, return Courier and Secure.
		name			1	String	Fixed value - "deliveryLocationType"
		value			1	String	One of: "BUSADD" - Business address "RESADD" - Residential address
	deliverTo				1	Node	This Section denotes the Details of the Entity where the Delivery is to be performed
		address			1	Node	This section details the Address Information of the Delivery Location
			@type		1	String	Fixed value - "ORG_CUST". This will not be checked.
			@primary		1	boolean	Fixed value - 1
			postalcode		0..1	String	Postcode of the location where the delivery is to be performed. Mandatory for GB addresses.
			country		1	Node	
				countryCode	1	String	The ISO 3166 Country Code of the country where the delivery is to be performed

For an example GetService request, see appendix 1

Success response:

Success Response

Structure			Occurs	Data Type	Description
getServicesResponse			1		XML definition
	status		1	Integer	One of: 1 - Success 0 - Error
	service[n]		0..n	Node List	Array of services applicable to the request. Each unique combination of legacy service attributes should be presented as a different service.
		ServiceFeatures[1]	2	Node List	Array of Key Value Pair to denote service features. For Courier, one deliveryLocationType and one maxCompensation will be returned. For Secure, two deliveryLocationType nodes and no maxCompensation will be returned.
		name	1	String	One of: "deliveryLocationType" "maxCompensation"
		value	1	String	If name="deliveryLocationType", then one of: "BUSADD" - Business address (for Courier and Secure) "RESADD" - Residential address (for Secure) If name="maxCompensation", then value of the maximim compensation available, ie. 2500.00 or 0 if no compensation is avalable. Up to 2 decimals.
		legacyService[n]	1..n	Node List	Array of Triplets to denote the Corresponding Service Details.
		name	1	String	The name of the legacy service attribute. One of: "serviceLevel" (for Courier) "serviceType" (for Courier) "productServiceCodeID" (for Secure) "productDeliveryMethodID" (for Secure) "timeSlotID" (for Secure)
		partyID	1	String	The legacy service attribute value.
		partyType	1	String	The partyType of the legacy service attribute. One of: "HITS" (for Courier) "TRINITY" (for Secure)
		description	1	String	This should represent as much as possible about the service. For Courier, service name probably is enough. For Secure, this needs to take into account the available options of product, service, delivery method, timeslot, weight range, compensation, delivery days.

For an example GetService success response, see appendix 2

Error response:

Error Response

Structure		Occurs	Data Type	Description
getServicesResponse		1		XML definition
	status	1	Integer	One of: 1 - Success 0 - Error
	errorCode	1	String	Error code
	errorDescription	1	String	Error description

For example GetService error response, see Error code section.

Label Creation

a. The request is made per consignment (piece), and specifies the details of the delivery, including address and contact details.

b. The response contains the tracking number and the label, multiple label formats are available.

API CreateLabel Schemas:

Request

Request

Structure	Occurs	Data Type	Description
createLabelRequest	1		XML definition
serviceHeader	1	Node	This section deals with authentication.
userID	1	String	UserID is the contact's username of the client. The contact will have been created solely for the use of the API.
password	1	String	Password for the user. The expiry date of the password will NOT be checked.
labelType	1	String	One of: "ZPL" "EPL" "PDF" "SVG"
order	1	Node	This is the root node for an Order.
customerID	1	String	DX ERP assigned Customer Account Number for the site or account that should be charged for this order
orderType	0..1	String	Fixed value - "Cons". This will not be checked.
sourceSystem	0..1	String	Fixed value - "DXAPI". This will not be checked.
orderStatus	0..1	Node	State of the order. This will not be checked.
attributeList[0]	1	Node List	Customer needs only send a single node.
name	1	String	Fixed value - "Current"
value	1	String	Fixed value - "Active"
dates	0..1	Node	For Courier only. This section deals with the list of the various date values that may be required for processing. Mandatory for Courier.
date[0]	1	Node List	Customer needs only send a single date node.
*body	1	String	Date requested for Collection – business day date when item will be available for collection. String format "yyyy-MM-dd HH:mm:ss".
@type	1	String	Static string value of "requestedCollectionDate"
@format	1	String	Fixed value - "yyyy-MM-dd HH:mm:ss"
sourceSystemReference	0..1	String	Typically would contain a Unique value used to identify this request in the Source System. The Customer can populate this with their Internal Order Number or Order Id. It is advisable to provide an unique id used in the Source System for support purposes. This will not be checked.
customerReference	0..1	String	This is the free text entered by the Customer for their order reference. Can be left blank . Not to be confused with consignment reference which is printed on the label and can be used to obtain tracking against some services. This will not be checked.
orderLines	1..n	Node	Maximum one orderLines is allowed.
consignment	1	Node	
pieces	0..n	Node	For Courier only. Maximum one piece is allowed.
dimensions[0]	1	Node List	Customer needs only send a weight.

		lastName	0..1	String	Name/last name of the contact. Optional
		phone	0..1	String	For secure only. Full international style telephone number of the contact. Optional
		mobile	0..1	String	For secure only. Mobile telephone number of the contact. Optional
		email	0..1	String	For secure only. Email address of the contact. Optional
consignmentReferences[n]			0..n	Node List	Array of Key Value Pair to denote additional details required by certain customers at a consignment level.
	name		1	String	One of: "customerReference" - this will be used in the label print and can be tracked against on some services "collectionInstructions" - instructions to support the collection activity "deliveryInstructions" - Delivery comments / instructions to support the delivery activity (eg. a suitable leave safe location) "contents" - required for customs "thirdPartyReference" - a third party reference "additionalReference1" - an additional reference "additionalReference2" - an additional reference
	value		1	String	Value of the reference
		value	1	Double	Decimal value of the dimension. Up to 2 decimals
		type	1	String	Denotes what the above value means. Fixed value - "cdlWeight"
		UOM	1	String	Denotes the unit of measurement of the numeric value. Fixed value - "KG"
	pieceReferences		0..1	Node	Array of Key Value Pair to denote the supplementary information with regard to the piece.
		attributeList[1]	1..3	Node List	Reference from the sender before the label is printed. A maximum of 3 can be sent.
		name	1	String	One of: "insuredAmount" - Value of cover "isInsured" - Cover required "valueOfGoods" - Required for customs
		value	1	String	The value of the name-value pair of additional piece information .e.g. 150.00, 1 or 0. Up to 2 decimals for values
legacyService[n]			1..n	Node List	Array of Triplets to denote the Corresponding Service Details. These details will be based on the response from the GetServices API method. Legacy services require multiple attributes to be passed to identify the correct service.
	name		1	String	The name of the legacy service attribute from the GetServices response.
	partyID		1	String	The legacy service attribute value from the GetServices response.
	partyType		1	String	The partyType of the legacy service attribute from the GetServices response.
deliverTo			1	Node	This Section denotes the Details of the Entity where the Delivery is to be performed
	address		1	Node	This contains the address of the delivery location.
		@type	1	String	Fixed value - "ORG_CUST". This will not be checked.
		@primary	1	boolean	Fixed value - 1
		organisationName	0..1	String	The Organisation Name where the delivery is to be performed
		addressLine1	0..1	String	1st line of the address where the delivery is to be performed. One of the four address lines must be present.
		addressLine2	0..1	String	2nd line of the address where the delivery is to be performed. One of the four address lines must be present.
		addressLine3	0..1	String	3rd line of the address where the delivery is to be performed. One of the four address lines must be present.
		addressLine4	0..1	String	4th line of the address where the delivery is to be performed. One of the four address lines must be present.
		postalcode	1	String	Postcode of the location where the delivery is to be performed
		country	1	Node	
		countryCode	1	String	The ISO 3166 Country Code of the country where the delivery is to be performed
	contact		0..1	Node	This section details the contact information of the person who should be contacted while performing the delivery.
		title	0..1	String	For secure only. Salutation of the contact. Optional
		firstName	0..1	String	For secure only. Initials/first name of the contact. Optional

For example CreateLabel request see appendix 3.

Notes on the CreateLabel request:

- The Secure B2C service is primarily a business to consumer product. DX can deliver B2C items to business addresses, providing the following criteria are followed:
    - Each item must have the name of the person the item is being delivered to, using the title, first name and last name fields. The B2C service can't be used without these details.
    - The company name should be sent in the address line1 field, with the remainder of the address in address lines 2, 3 and 4 fields.

- The B2C service can only be used on the day the items are to be collected. Only B2B courier items can be created using a future collection date.

Success response:

Success Response

Structure		Occurs	Data Type	Description
createLabelResponse		1		XML definition
	status	1	Integer	One of: 1 - Success 0 - Error
	trackingNumber	1	String	Tracking number for the created job
	label	1	String	Base64 label

For example CreateLabel success response, see appendix 4

Error response:

Error Response

Structure		Occurs	Data Type	Description
createLabelResponse		1		XML definition
	status	1	Integer	One of: 1 - Success 0 - Error
	errorCode	1	String	Error code
	errorDescription	1	String	Error description

For example CreateLabel error response, see Error code section

Label Output Format (CreateLabel response)

The CreateLabel request must specify the format of label that is required. The following formats are available:

1. Label Printer language, either:
   a. ZPL
   b. EPL
   c. SVG image format.
   d. PDF files

Label codes to use on the request should be one of the following:

ZPL
EPL
SVG
PDF

ZPL and EPL require a label printer to print labels; DX can supply the required 4x4 labels.
SVG and PDF would normally print on a laser type printer on A4 paper (depending what paper stock the customer has loaded), these may require folding before attaching to the item. Use of clear plastic self-adhesive Document Enclosed type wallets is recommended. The label must be attached to the item with the barcode remaining flat. For example label, see appendix 5

Error codes

Invalid data sent in requests will result in error codes being returned in a Failure response file, rather than the label.

There are a number of possible failure response error codes, which include, in no particular order:

- **Invalid delivery method- error code E300**, where the data returned will show:

  < errorCode>E300</errorCode>

  <errorDescription>Invalid delivery method data.</errorDescription>

- **Invalid delivery address – error code E103,** where the data returned will show:

  <errorCode>E103</errorCode>

  <errorDescription>Invalid delivery address.</errorDescription>

- **Invalid login details – error code E102,** where the data returned will show:

  <errorCode>E102</errorCode>

  <errorDescription>Invalid login details.</errorDescription>

,

- **Invalid product data – error code E301,** where the data returned will show:

  <errorCode>E301</errorCode>

  <errorDescription>Invalid product data.</errorDescription>

- **Invalid label type – error code E213**, where data returned will show:

  <errorCode>E213</errorCode>

  <errorDescription>Invalid label type.</errorDescription>

- **Invalid Date – error code E205**, where data returned will show:

  <errorCode>E205</errorCode>

  <errorDescription>Invalid date.</errorDescription>

- **Invalid Job limit – error code E207**, where data returned will show:

  <errorCode>E207</errorCode>

  <errorDescription>Invalid Job limit.</errorDescription>

- **Invalid Job request – error code E220**, where data returned will show:

  <errorCode>E220</errorCode>

  <errorDescription>Invalid Job request.</errorDescription>

- **Invalid Service – error code E208**, where data returned will show:

  <errorCode>E208</errorCode>

  <errorDescription>Invalid Service.</errorDescription>

- **Invalid tracking – error code E211**, where data returned will show:

  <errorCode>E211</errorCode>

  <errorDescription>Invalid tracking.</errorDescription>

- **Invalid weight – error code E212**, where data returned will show:

  <errorCode>E212</errorCode>

  <errorDescription>Invalid weight.</errorDescription>

For an example Failure response file, see appendix 6

Access control

As stated above, access to the DX API is controlled via user id and password, which is at customer (or location) level. DX will supply account numbers, user name and password for each location where the customer requires the labels to be printed.

In addition, all IP addresses accessing the API are whitelisted and will need to be provided as part of the API implementation.

The live Web Service URL will be supplied for testing and go live, after development sign off.

Field lengths

The following is the maximum number of characters allowed in the relevant field:

Description	Length, chars
Customer reference / order number	50
Title	20
First name	50
Surname	50
Address line 1	50
Address line 2	50
Address line 3	50
Address line 4	50
Postcode	8
Telephone (B2C only)	20
Mobile (B2C only)	20
Email address (B2C only)	50
Delivery comment (B2C only)	60
Weight (B2B only)	2, plus decimal point, plus 2 decimal chars
Third party reference (B2C only)	30
Additional reference 1 (B2C only)	30
Additional reference 2 (B2C only)	30

Testing

Before approval can be given to commence using DX Despatch API, full testing is required; the tests normally consist of the stages shown.

Testing can only be completed once the customer has been set up on DX systems and account numbers have been issued. Account numbers will be required for each location for customers with multiple sites:

1. Customer completes the required development on their systems.

2. DX supply credentials to be used for testing, this will be the same information that will be used for live transactions, as testing will be completed on the live API system. Details suppled will be: account number/s, user names and passwords, service codes, API URL etc.

3. Customer creates test items, one for each different service to be used and different locations if applicable, whilst DX are monitoring the systems.

4. Any bugs or issues are identified, resolved and re-tested. This may require multiple test cycles before all items are created correctly.

5. Any live test items in the DX systems are cancelled, development is signed off and customer is ready to go live.

DX will supply the URL to connect to the WSDL if required after the customers IP addresses have been whitelisted and system log on details have been issued.

Assistance

This document is intended to provide you with all the information required to give your organisation the ability to create DX tracking labels in the appropriate formats. You will be given details of a DX contact should you require any further information, or assistance and to help complete testing.

Appendix

Note:

- PDF versions of the schemas should have also been sent with this document. Please request these if they have not been suppled.

Appendix 1: GetService request:

Get Services Request.txt

Appendix 2: GetService success response:

getServices Response - Success.txt

Appendix 3: CreateLabel request:

createLabel Request.txt

Appendix 4: CreateLabel success response:

createLabel Response_EPL - Success.txt

createLabel Response_SVG - Success.txt

createLabel Response_PDF - Success.txt

createLabel Response_ZPL - Success.txt

Appendix 5: Example DX Tracking label:

Appendix 6: Example Failure response file:

The content of the file will be different depending on the error.

Failure Response - Invalid delivery address.txt