Author	**Steve Baker**
Creation Date	**16th May 2014**
Created For	**General**

iMetal Design Document

Sales Document Import

Change Control

Issue	Date	By	QC	Issue	Summary
1	16/05/2014	SB			First Release
2	20/05/2014	SB			Updated based on initial reviews
3	22/05/2014	SB			Add order Change and Deletion details
4	27/04/2018	SB			Updated with recent changes

Table of Contents

1. Introduction

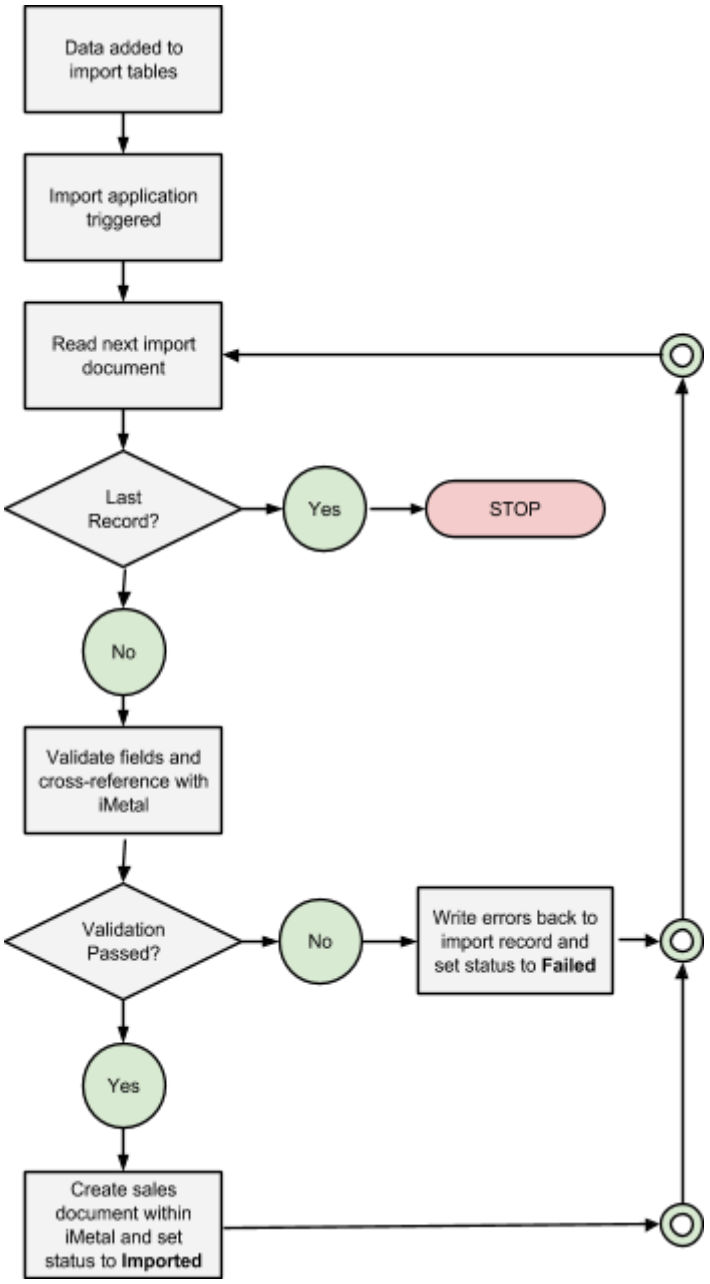
This document describes a generic Sales Document Import application that will provide the ability to import sales documents into iMetal from external sources. The import will provide the following features:

- Simple import table structure within SQL database
- Fully de-referenced data fields (i.e. no ID's used)
- Validation of all import fields
- Common import database can feed multiple iMetal systems
- Screen-based and Command-line import options
- All iMetal sales document types are covered, including:
    - Sales Enquiries
    - Sales Orders
    - Blanket Orders
    - Call Off Orders
    - Order Templates
    - Transfer Orders
    - Production Orders
    - Manual Invoices
    - Manual Credit Notes
- Ability to Delete orders

2. Core Flow

Although the sales document import could be used in various different scenarios, the common core functionality is as follows:

1. New sales document data is added to the sales document import tables
2. The iMetal import application is triggered
3. Each sales document in the import data is validated, cross-referencing it with iMetal tables
4. Any errors during the validation are written back to the import record(s) and the import of that sales document is abandoned. An *import status* field on the import sales header is updated as 'Failed'
5. If the validations pass, then a new sales document is created within iMetal and the import status on the header is set to 'Imported'



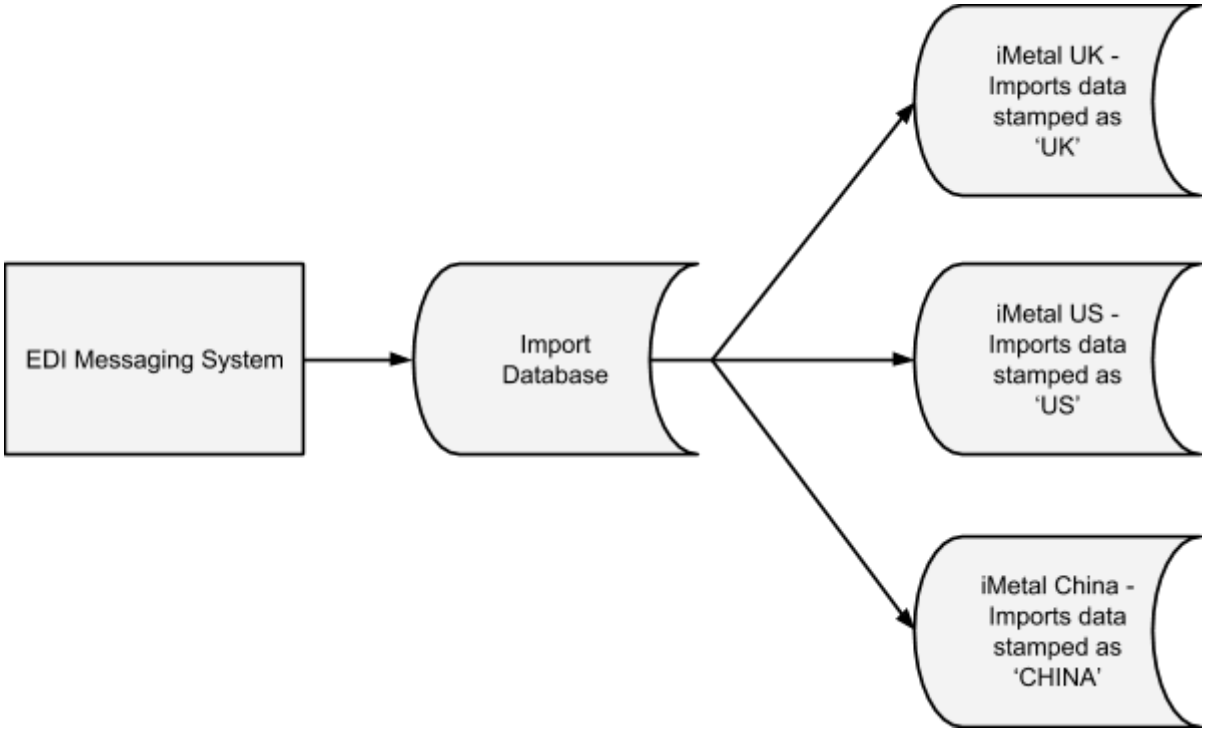
3. Import Database

Generally, all of the data tables required by a single iMetal installation are stored in a common database (e.g, Company A and Company B running on the same server would each have their own iMetal databases). However, the iMetal importing applications will be developed so that they can point to a common database in order to access the import data.

Whilst this won't be necessary in most situations, it does simplify the development of external systems that export data to multiple iMetal sites. It also provides an additional layer of security, in that external systems can be given direct write access to the import database alone, without also giving them access to the full iMetal database itself.

3.1 EDI Example

In the following example an EDI messaging system, which might receive data that is destined for three different iMetal systems within the same organisation, doesn't have to open three different database connections when passing it on to iMetal. It can simply write to the central Import database, specifying a 'destination' against each record. Each different iMetal installation will then be responsible for processing the import data



3.2 iMetal Data Exchange Example

The use of a common import database like this also enables the exchange of data between two different iMetal systems. For example, a module could be developed whereby a Purchase Order raised on one iMetal system could be 'sent' to a different iMetal system by converting it into a Sales Document Import structure in the common import database, and the remote system would then import that order into its own local database.

3.3 New Database Connection

Several developments will be required in order to connect to the new import database:

3.3.1 Additional Properties

The ‘emetal.properties’ file will need to include a new section as follows:

```
importdb.driver=org.postgresql.Driver
importdb.url=jdbc:postgresql://host:11061/imetal_imports
importdb.username=base_emetal
importdb.password=password
importdb.hibernatedialect=org.hibernate.dialect.PostgreSQLDialect
importdb.showsql=false
```

This is the standard configuration, which assumes that the database will use Postgres, but it could be configured to use other supported database systems too.

3.3.2 New Service

In order to communicate with this new database, the iMetal server daemon will need an additional service, which will be set up in the same way as the existing ‘AccpacService’. This will involve two new classes:

```
Interface:
com.metalogicplc.emetal.client.services.ImportService

Implementation:
com.metalogicplc.emetal.client.services.impl.parallel.ImportServiceImpl
```

3.3.3 Spring/Hibernate Configuration

Once the new services have been created, the Spring configuration files will need to be updated to include the necessary wiring for them.

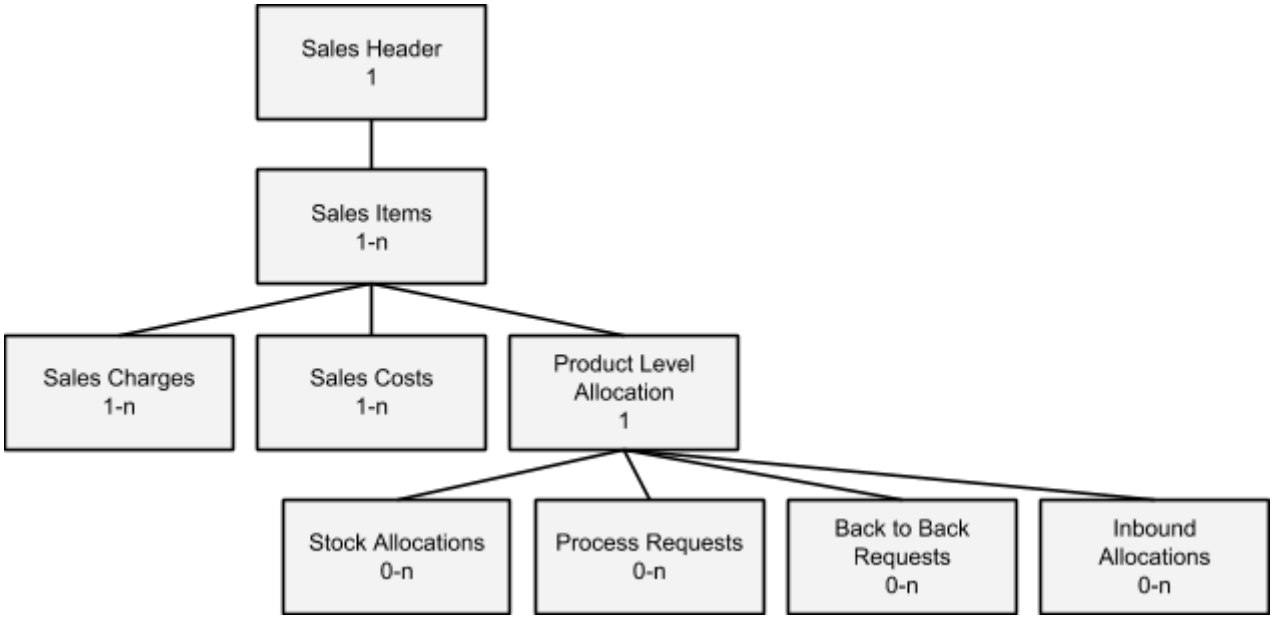
Again, this will be similar to the recently added AccpacService, so rather than detailing all of the configuration sections here, the best approach when developing it is to copy those sections and re-point them at the ImportService and at the ‘importdb’ section of the properties file.

The two configuration files involved are:

```
emetal.server\emetal.serverdaemon\src\main\resources\application.xml
emetal.server\emetal.httpservlet\src\main\resources\httpservlet-servlet.xml
```

4. Sales Document Table Structure

Each sales document that is created within iMetal is stored in a multi-layered table structure, so before detailing the individual sales document import tables, this section provides an overview of that structure:



In this chart we can see that each sales document has a single Sales Header record. Beneath that there can be one-to-many Sales Items, and beneath that there can be one or more Sales Charges and Sales Costs, and also a single Product Level Allocation. At the lowest level, the product level allocation can contain Stock Allocations, Process Requests, Back to Back Requests or Inbound Allocations (although it may contain no records at this level at all).

In order to keep the import process as transparent as possible, the import tables will use exactly the same structure for the first three levels. The Stock Allocations, Process Requests, Back to Back Requests and Inbound Allocations are not covered as part of the import, except for the automatic creation of Process Requests where Part Specifications are used or where process requests are automatically created for every item.

4.1 Import Header Fields

At the top of the main Sales Header import table is a header section containing a number of key fields:

Field Name	Type	V/M	Notes
import_company_reference	varchar(30)	M	e.g. UK, US, CHINA
import_batch_number	integer	M	Reference to import batch
import_status	character(1)	VM	(E)ntered, (F)ailed, (I)mported
import_notes	text		Leave blank
import_user_name	varchar(50)	MV	
import_source	varchar(50)		Source of import data
import_date	date		Leave blank
import_action	char	MV	(A)dd, (C)hange, (D)elete

4.1.1 Import Company Reference

This field is primarily used when multiple iMetal systems are configured to use the same import database. Each iMetal system will use the following server preference to determine its own company reference, which will then be used to find only the import data that is relevant to it:

```
Preference  : ImportCompanyReference
Default     : null

Description :
Determines the Company Reference for this iMetal system when looking for data in
the import database.
```

4.1.2 Import Batch Number

This numeric reference can be used to group multiple orders together into a single common ‘import batch’. The reference will also be stamped onto any orders created within iMetal and made available for searching and/or viewing in selected applications.

4.1.3 Import Status

This field determines the current status of each sales document being imported. It will start as Entered for each newly created sales header, but will then be set to either Failed or Imported during the update.

For some import failures, the problem may be resolved by either creating/amending reference data within iMetal or by altering the import records directly. The user may then alter the status back to Entered so that the sales document becomes eligible for importing again.

Note that if a record with a status of Imported is changed back to Entered, the system will attempt to import it again, creating a duplicate copy of the sales document. See the selection on Deleting Successful Imports.

4.1.4 Import Notes

This field is available on each of the import tables and is used by the import validation routine whenever one or more fields fail their validations. It is an unlimited size text field, and will contain details in the following format:

```
branch_code: "COV" not found in branches table.
customer_code: "CLA002" not found in companies table.
exchange_rate_type_code: "6" must be one of "1", "2"
```

The import program will automatically clear down the field at the start of each validation, in case it is being re-processed after data has been corrected.

4.1.5 Import User Name

This field will be validated against the 'personnel' table in iMetal and can be used as a selection on the import application.

It is important to note that this field is *not* used by the import program to read in group preferences. They are based on the user that is running the import itself.

The field will, however, be used as the Salesperson on the import if no salesperson is provided.

4.1.6 Import Source

This optional field can be used to define the source of the import data (e.g. "iMetal Company A", "Customer B EDI", etc). It will be stamped onto the iMetal sales_headers table.

Although this is a free-format field, any implementations with multiple import sources should carefully consider the format to be used so that reporting/querying of the data is straightforward.

4.1.7 Import Date

The import date field will be stamped onto the import header once the sales document has been successfully imported. The primary purpose behind this is to allow future archiving utilities to delete all imported sales documents older than a certain date.

4.1.8 Import Action

This field will determine whether the import program treats the sales document data as a new record (Add), a change to an existing record (Change) or a deletion of an existing record (Delete).

5. Import Table Definitions

This section describes each of the import tables that need to be populated with data before iMetal can import a sales document.

In each table, the ‘V/M’ column indicates whether the field is Validated (V) and/or Mandatory (M) or Defaultable (D). Where fields are mandatory in iMetal, but can be defaulted based on other data, these will be marked as ‘D’.

5.1 Sales Header

The sales header table contains all of the information that is common to the entire sales document, for example Customer, Delivery Address, Sales Branch, etc.

Table Name: import_sales_headers

Index 1 (Unique):
import_company_reference, import_batch_number, import_number

Index 2 (Not Unique):
import_company_reference, import_batch_number, import_status

Index 3 (Not Unique):
import_company_reference, import_status

Field Name	Type	V/M	Notes
import_company_reference	varchar(30)	M	e.g. UK, US, CHINA
import_batch_number	integer	M	Reference to import batch
import_status	character(1)	VM	(E)ntered, (F)ailed, (I)mported
import_notes	text		Leave blank
import_user_name	varchar(50)	MV	
import_source	varchar(50)		Source of import data
import_date	date		Leave blank
import_action	character(1)	MV	(A)dd, (C)hange or (D)elete
imetal_document_number	integer		Written back to DB by iMetal following successful import
branch_code	varchar(6)	VM	Linked to ‘branches’ table
type_code	Varchar(3)	VM	Linked to ‘sales_types’ table
import_number	integer	M	Unique sales document identity (1)
customer_order_number	varchar(30)		
job_number	varchar(30)		
release_number	varchar(30)		
customer_code	varchar(10)	VM	Linked to ‘companies’ table
customer_name_override	varchar(50)	D	Defaults to customer name

customer_address	text	D	Multi-line address (2)
customer_town	varchar(60)	D	(2)
customer_county	varchar(60)	D	(2)
customer_postcode	varchar(60)	D	(2)
customer_country_code	varchar(3)	VMD	Linked to ‘country_codes’ table (2)
customer_transport_area_code	varchar(16)	VMD	Linked to ‘transport_areas’ table (2)
deliver_to_address_code	Varchar(10)	VMD	Linked to ‘company_sub_addresses’ table
deliver_to_address	text	D	Multi-line address (3)
deliver_to_town	varchar(60)	D	(3)
deliver_to_county	varchar(60)	D	(3)
deliver_to_postcode	varchar(60)	D	(3)
deliver_to_country_code	varchar(3)	VMD	Linked to ‘country_codes’ table (3)
deliver_to_transport_area_code	varchar(16)	VMD	Linked to ‘transport_areas’ table (3)
salesperson_name	varchar(50)	VD	Linked to ‘personnel’ table
inside_salesperson_name	varchar(50)	VD	Linked to ‘personnel’ table
delivery_branch_code	Varchar(6)	V	Linked to ‘branches’ table. Defaults from ‘branch_code’ if blank
delivery_warehouse_code	Varchar(6)	V	Linked to ‘warehouses’ table. Defaults to shipping warehouse on Delivery Branch if blank
delivery_name	varchar(50)	D	Defaults from delivery branch
delivery_address	text	D	Multi-line address (4)
delivery_town	varchar(60)	D	(4)
delivery_county	varchar(60)	D	(4)
delivery_postcode	varchar(60)	D	(4)
delivery_country_code	varchar(3)	VD	Linked to ‘country_codes’ table (4)
delivery_transport_area_code	varchar(16)	VD	Linked to ‘transport_areas’ table (4)
sale_date	date	D	Defaults to today’s date
due_date	date	D	If left blank, the system will calculate a due date based on the sale date
manual_date	varchar(16)		
transport_type_code	Varchar(6)	V	Linked to ‘transport_type_codes’ table. Defaults using same rules as in sales entry
delivery_point	varchar(60)		
carrier_code	Varchar(10)	V	Linked to ‘companies’ table
transport_cost_rate	numeric(12,4)		
transport_cost_rate_unit_code	Varchar(3)	VD	Linked to ‘units_of_measure’ table
transport_cost_amount	numeric(12,2)		
transport_exchange_rate	numeric(14,8)	D	Defaults from carrier currency
transport_exchange_rate_type_code	character(1)	VD	1 = Fixed, 2 = Variable

exchange_rate	numeric(14,8)	D	Defaults from customer currency
exchange_rate_type_code	character(1)	VD	1 = Fixed, 2 = Variable
terms_code	Varchar(6)	VD	Linked to 'terms' table. Defaults from customer
settlement_discount	numeric(5,2)		
chemical_cert	boolean	D	(5)
mechanical_cert	boolean	D	(5)
mill_cert	boolean	D	(5)
compliance_cert	boolean	D	(5)
delivery_copies	integer	D	(5)
invoice_copies	integer	D	(5)
header_text	text	D	Defaults based on sales entry rules
internal_text	text	D	Defaults based on sales entry rules
despatch_text	text	D	Defaults based on sales entry rules
payment_due_date	date	D	For invoices only
credited_invoice_branch_code	Varchar(6)	V	Linked to 'branches' table (9)
credited_invoice	integer		Valid final invoice number (9)
credit_reference	varchar(30)		
order_branch_code	varchar(6)	V	Linked to 'branches' table (6)
order_number	integer	V	Linked to 'sales_headers' table (6)
deliver_to_name_override	varchar(50)	D	Defaults from deliver to name
transport_charge_rate	numeric(12,4)		
transport_charge_rate_unit_code	Varchar(3)	VD	Linked to 'units_of_measure' table
transport_charge_amount	numeric(12,2)		
despatch_branch_code	varchar(3)	V	Linked to 'branches' table (7)
despatch_number	integer	V	Linked to 'despatch_headers' table (7)
followup_date	date		
update_hold	boolean	D	False
use_minimum_grade	boolean	D	False
credited_despatch_branch_code	Varchar(6)	V	Linked to 'branches' table (9)
credited_despatch	integer	V	Valid despatch note number (9)
credited_order_branch_code	Varchar(6)	V	Linked to 'branches' table (9)
credited_order	integer	V	Valid sales order number (9)
credit_release_notes	text		
require_proforma	boolean	D	Defaults from customer
transfer_type	character(1)	VD	(D)elivery, (C)ollection, (U)ndecided
tax_group_code	varchar(12)	D	Defaults from customer
tax_class_1	integer	D	Defaults from customer
tax_class_2	integer	D	Defaults from customer
tax_class_3	integer	D	Defaults from customer

tax_class_4	integer	D	Defaults from customer
tax_class_5	integer	D	Defaults from customer
transfer_to_branch_code	Varchar(6)	VM	Linked to ‘branches’ table (10)
transfer_to_warehouse_code	Varchar(6)	VM	Linked to ‘warehouses’ table (10)
tax_authority1	varchar(12)	D	Defaults from customer
tax_authority2	varchar(12)	D	Defaults from customer
tax_authority3	varchar(12)	D	Defaults from customer
tax_authority4	varchar(12)	D	Defaults from customer
tax_authority5	varchar(12)	D	Defaults from customer
tax_exempt1	boolean	D	Defaults from customer
tax_exempt2	boolean	D	Defaults from customer
tax_exempt3	boolean	D	Defaults from customer
tax_exempt4	boolean	D	Defaults from customer
tax_exempt5	boolean	D	Defaults from customer
document_delivery_type_id	integer	VD	NotApplicable(1), DespatchNote(2), CODInvoice(3), NoteAndInvoice(4)
contact_forename	varchar(50)	V	(11)
contact_surname	varchar(50)	V	(11)
contact_telephone_override	varchar(20)	D	Defaults from contact
contact_mobile_override	varchar(20)	D	Defaults from contact
contact_fax_override	varchar(20)	D	Defaults from contact
contact_email_override	varchar(255)	D	Defaults from contact
contact_web_address_override	varchar(255)	D	Defaults from contact
consignment_order	boolean	D	Defaults based on customer delivery address
footer_internal_text	text		
footer_external_text	text		
blanket_header_branch_code	varchar(3)	V	Linked to ‘branches’ table (8)
blanket_header_number	integer	V	Linked to ‘sales_headers’ table (8)
transport_charged	boolean	D	?
fix_date	date		
expiry_date	date		
no_fixed_date	boolean	D	True
credited_delivery_date	date		(9)
sales_group_code	varchar(3)	D	(12)
transfer_to_branch_code	varchar(6)	V	See FS-IM0038
transfer_to_warehouse_code	varchar(10)	V	See FS-IM0038

Notes:

(1) The ‘import_number’ field will be assigned by the external system and must provide a unique identity for this sales document header when combined with the ‘import_company_reference’ and ‘import_batch_number’ fields. The field’s key purpose is to tie together the sales document header with all of the items, costs, charges, etc that are related to it.

The import program will automatically assign the next document number (e.g. Sales Order Number) for the sales header's Branch and Sales Type.

(2) These fields will be stored in the 'addresses' table, which will be linked to via SalesHeader.customerAddress.

(3) These fields will be stored in the 'addresses' table, which will be linked to via SalesHeader.deliverToAddress.

(4) These fields will be stored in the 'addresses' table, which will be linked to via SalesHeader.deliveryAddress.

(5) These fields will be stored in the 'certification_requirements' table, which will be linked to via SalesHeader.certificationsId.

(6) These fields only need to be populated for Credit Notes or Invoices. If present, they must point to an existing Sales Order on the system.

(7) These fields only need to be populated for Credit Notes or Invoices. If present, they must point to an existing Despatch Note on the system.

(8) These fields only need to be populated for Call Off Orders. If present, they must point to an existing Blanket Sales Order on the system.

(9) These fields only need to be populated for Credit Notes. If present, they must point to a valid Sales Invoice on the system that has already been updated.

(10) These fields are mandatory for Transfer Orders. They represent the Branch and Warehouse that the transfer order will be delivering to.

(11) The contact Forename and Surname fields (along with the customer) are used to find a unique record in the 'contacts' table. This will then be linked to using the SalesHeader.contactId. If no contact is found using these details, then the validation will fail and the sales document won't be imported.

(12) If sales group code is not set, will attempt to default from the Sales Person's default sales group, the logged in user's default sales group, and customer sales group, in that order. If a sales group is not set on any of those records, validation will fail and the sales document won't be imported.

5.1.1 Automatically Populated Internal Sales Header Fields

The following internal fields will be populated automatically by the import program. These are fields in the iMetal sales_headers table and not the import table itself:

Field Name	Type	V/M	Notes
number	integer		Next document number
status_id	integer		1 = Entered
acknowledged	boolean		false
acknowledged_date	date		null
printed	boolean		false
currency_id	integer		From companies.currencyId
completed	boolean		false
completed_date	date		null
invoice_number	integer		null
batch_id	integer		New batch id (for invoice/credit note)
lost_reason_id	integer		null
lost_date	date		null
lost_description	varchar(60)		null
sales_total_id	integer		Created sales totals (1)
printed_date	date		null
credit_hold	boolean		false
credit_hold_date	date		null
credit_hold_reason	integer		null
credit_release_date	date		null
credit_release_user_id	integer		null
credit_release_amount	numeric(12,2)		null
gateway_batch_id	integer		null
exported_date	date		null
enquiry_id	integer		null
sales_entry_total_id	integer		Created entry totals (1)
balance_total_id	integer		Created entry totals (1)
proforma_printed_date	date		null
cutting_header_id	integer		null
credit_held_at_despatch	boolean	D	False

Notes:

(1) Each of the three *total id* fields will point to new entries in the ‘sales_totals’ table, which represent total order quantities and values for all items.

5.2 Sales Items

The Sales Items table must contain at least one record against each sales header. It contains details such as the sold Product and/or Part, along with dimensions, quantities and descriptions.

Table Name: import_sales_items

Index 1 (Unique):
import_company_reference, import_batch_number, import_number, item_number

Field Name	Type	V/M	Notes
import_company_reference	varchar(30)	M	e.g. UK, US, CHINA
import_source	varchar(50)		Source system of the import data
import_batch_number	integer	M	Reference to import batch
import_notes	text		Leave blank
import_number	integer	M	Unique sales document identity (1)
import_item	integer	M	Stored as item_number in iMetal
part_customer_code	varchar(10)	V	Linked to 'companies' table (2)
part_specification_code	varchar(35)	V	Linked to 'part_number_specifications' (2)
product_code	varchar(24)	VM	Linked to 'products' table (2)
dim1	numeric(9,4)		(3)
dim1_negative_tolerance	numeric(9,4)		(3)
dim1_positive_tolerance	numeric(9,4)		(3)
dim2	numeric(9,4)		(3)
dim2_negative_tolerance	numeric(9,4)		(3)
dim2_positive_tolerance	numeric(9,4)		(3)
dim3	numeric(9,4)		(3)
dim3_negative_tolerance	numeric(9,4)		(3)
dim3_positive_tolerance	numeric(9,4)		(3)
dim4	numeric(9,4)		(3)
dim4_negative_tolerance	numeric(9,4)		(3)
dim4_positive_tolerance	numeric(9,4)		(3)
dim5	numeric(9,4)		(3)
dim5_negative_tolerance	numeric(9,4)		(3)
dim5_positive_tolerance	numeric(9,4)		(3)
mark	varchar(16)		(3)
description	text	D	
works_notes	text		
production_notes	text		
invoice_notes	text		

required_pieces	integer	M	(4)
required_quantity	numeric(12,3)	M	(4)
required_weight	numeric(10,3)	M	(4)
original_quantity	numeric(12,3)	M	(4)
original_quantity_unit_code	varchar(3)	MV	Linked to 'units_of_measure' (4)
weight_units_code-ignore-not used in iMetal	varchar(3)	VD	Linked to 'units_of_measure' table
sales_group_code	varchar(3)	VD	Linked to 'sales_groups' table (5)
due_date	date	D	Will default from header if not present
manual_date	varchar(16)	D	Will default from header if not present
specification_value1	varchar(30)	D	Will default from product (8)
specification_value2	varchar(30)	D	Will default from product (8)
specification_value3	varchar(30)	D	Will default from product (8)
specification_value4	varchar(30)	D	Will default from product (8)
specification_value5	varchar(30)	D	Will default from product (8)
specification_value6	varchar(30)	D	Will default from product (8)
specification_value7	varchar(30)	D	Will default from product (8)
specification_value8	varchar(30)	D	Will default from product (8)
specification_value9	varchar(30)	D	Will default from product (8)
specification_value10	varchar(30)	D	Will default from product (8)
credited_item	integer		(6)
credited_order_number	integer		(6)
credited_order_item	integer		(6)
credited_despatch_item	integer		(6)
credited_customer_order	varchar(30)		(6)
despatch_item_branch_code	varchar(3)		Linked to 'branches'. (7)
despatch_item_header_number	integer		(7)
despatch_item_number	integer		(7)
order_item_branch_code	varchar(3)		Linked to 'branches'. (11)
order_item_header_number	integer		(11)
order_item_number	integer		(11)
delivery_branch_code	varchar(6)	VD	Linked to 'branches'. Defaults from header
delivery_warehouse_code	varchar(6)	VD	Linked to 'warehouses'. Defaults from header
show_prices	boolean	D	False
use_minimum_grade	boolean	D	Defaults from header
margin_type	character(1)	VD	(M)argin or Mar(K)up. Default = Margin
transfer_type	character(1)	VD	(D)elivery, (C)ollection, (U)ndecided
tax_class_1	integer	D	Defaults from customer
tax_class_2	integer	D	Defaults from customer
tax_class_3	integer	D	Defaults from customer
tax_class_4	integer	D	Defaults from customer

tax_class_5	integer	D	Defaults from customer
tax_exempt1	boolean	D	Defaults from customer
tax_exempt2	boolean	D	Defaults from customer
tax_exempt3	boolean	D	Defaults from customer
tax_exempt4	boolean	D	Defaults from customer
tax_exempt5	boolean	D	Defaults from customer
part_number	varchar(35)		(9)
outside_diameter	numeric(9,2)		
outside_diameter_minimum	numeric(9,2)		
outside_diameter_maximum	numeric(9,2)		
pack_height	numeric(9,4)		
pack_height_minimum	numeric(9,4)		
pack_height_maximum	numeric(9,4)		
pack_weight	numeric(10,3)		
pack_weight_minimum	numeric(10,3)		
pack_weight_maximum	numeric(10,3)		
inside_diameter	numeric(9,2)		
inside_diameter_minimum	numeric(9,2)		
inside_diameter_maximum	numeric(9,2)		
pack_count_minimum	integer		
pack_count_maximum	integer		
working_specification_code	varchar(30)	V	Links to ‘specification_header’
invoice_costing_weight	numeric(10,3)	D	Defaults to required_weight. Ignore-gets automatically calculated by iMetal.
blanket_item_number	integer		**(12)**
mixture_price	numeric(12,4)		(10)
fabrication_price	numeric(12,4)		(10)
adjustment_price	numeric(12,4)		(10)
yield_percentage	numeric(5,2)		(10)
mechanical_cert	boolean	D	False
show_country_of_material_origin	boolean	D	False
show_country_of_primary_processing	boolean	D	False
show_country_of_final_processing	boolean	D	False
acknowledgement_notes	text	D	
despatch_notes	text	D	
invoice_packing	boolean	D	Defaults from customer

Notes:

(1) The Import Number field must match the import number in exactly one of the sales header records that has been imported. This sales item will then be attached to that sales header when the document is created in iMetal.

(2) The Part and Product fields both trigger special logic which is described in detail below.

(3) Each of these fields will be used by the import program to create an entry in the 'dimension_values' table, which will then be linked to via the 'dimensions_id' field on the sales item.

(4) The import allows either the Required Pieces, Quantity and Weight fields to be specified, *or* the Original Quantity and Original Quantity Unit fields to be specified. It will attempt to calculate any missing values using the standard iMetal quantity calculation logic and rules.

For example, if a Bar product item is imported and just a Required Pieces is specified, the program will use the length of the bar and the product density to calculate the Quantity (e.g. total length) and Weight.

If the Original Quantity is entered, along with an Original Unit, the program will first convert the quantity into a unit that matches the product set up (e.g. converting Metres to Inches), then calculate the Required Pieces, Quantity and Weight accordingly.

(5) This field will default from the Product or Part if left blank.

(6) These fields are only used on Credit Note Items. They are used for reference only, so don't strictly have to refer to valid records within iMetal. They can be left blank.

(7) These fields are only used on Credit Note or Invoice items. If populated, they must point to a valid despatch note item. The import program will convert the reference into the 'despatch_item_id' field.

(8) If the Validate Specification(1-10) fields on the product category are set, then the import program will validate the relevant values against the 'stock_specification_values' table.

If the Mandatory Specification(1-10) fields on the category are set, then the import will validate to make sure that a specification value has been provided.

(9) This optional field is for reference only and can be used to reference a Part Number even if the full Part Specification system is not being used.

(10) These fields are only required if the Alloy Pricing module is enabled.

(11) These fields are only used on Credit Note or Invoice items. If populated, they must point to a valid sales order item. The import program will convert the reference into the 'order_item_id' field.

(12) The Blanket Item Number field is only valid when importing Call Off orders. It must then be populated and must point to a valid blanket item when combined with the Blanket Branch and Blanket Order fields from the header. The import program will convert the reference to the 'blanket_item_id' field.

5.2.1 Automatically Populated Internal Sales Item Fields

The following internal fields will be populated automatically by the import program. These are fields in the iMetal sales_items table and not the import table itself:

Field Name	Type	V/M	Notes
delivered_pieces	integer		0
delivered_quantity	numeric(12,3)		0
delivered_weight	numeric(10,3)		0
balance_pieces	integer		0
balance_quantity	numeric(12,3)		0
balance_weight	numeric(10,3)		0
status_id	integer		Entered
completed	boolean		False
completed_date	date		null
lost_reason_id	integer		null
lost_date	date		null
lost_description	varchar(60)		null
sales_total_id	integer		(1)
original_due_date	date		Same as item due date
transport_pieces	integer		0
transport_quantity	numeric(12,3)		0
transport_weight	numeric(10,3)		0
stock_location	varchar(16)		Default using existing logic
invoiced_pieces	integer		0
invoiced_quantity	numeric(12,3)		0
invoiced_weight	numeric(10,3)		0
balance_total_id	integer		(1)
allocated_pieces	integer		Calculated after creating allocations
allocated_quantity	numeric(12,3)		Calculated after creating allocations
allocated_weight	numeric(10,3)		Calculated after creating allocations
product_level_allocation_id	integer		Set automatically
trim_required	boolean		False
shotblasting_required	boolean		False
paint_description	varchar(255)		null
group_price_unit_id	integer		null
group_total_value	numeric(12,2)		0
group_total_price	numeric(12,4)		0
allocated_coil_pieces	integer		Set at the Domain level when allocating
work_order_printed_quantity	numeric(12,3)		0
call_off_pieces	integer		0
call_off_quantity	numeric(12,3)		0
call_off_weight	numeric(10,3)		0

paint_type_id	integer		null
stock_available	boolean		(2)
fully_allocated	boolean		(2)
standard_price	numeric(12,4)		Set automatically at Domain level
cutting_group_reference	varchar(64)		null
complete_after_next_invoice	boolean		false
completion_balance_quantity	numeric(12,3)		0
item_credited	boolean		false
group_cutting_cost	numeric(12,4)		0
group_cutting_cost_unit_id	integer		null
group_drilling_cost	numeric(12,4)		0
group_drilling_cost_unit_id	integer		null
group_painting_cost	numeric(12,4)		0
group_painting_cost_unit_id	integer		null
group_shotblasting_cost	numeric(12,4)		0
group_shotblasting_cost_unit_id	integer		null
product_sub_group_id	integer		Set automatically
group_consumption_price	numeric(12,4)		0
product_description	text		null
credit_reference	varchar(30)		(3)
calculate_pieces	boolean	D	Based on Category quantity calculation rules

NOTES:

(1) The ‘sales_total_id’ field will point to a new entry in the ‘sales_totals’ table, which represents the total order quantities and values for the item.

(2) If the automatic stock allocation preferences are switched on, the import will attempt to automatically allocate stock to the sales item and then update these fields accordingly.

(3) For credit note items only, the ‘credit_reference’ field will be automatically populated with the full ‘invoice item reference’ of the invoice item that is being credited (if it is specified).

5.2.2 Product Code Functionality

If no Part Specification has been specified in the import record (see below) the ‘product_code’ field must be specified.

This will be validated to ensure that it refers to an *active* product code within iMetal and then turned into a ‘product_id’ reference on the sales item.

The product itself will be used to default the following fields, although in each case the value passed in the import record will take precedence *except* for any *locked dimension* fields (e.g. thickness):

Field Name	Type	V/M	Notes
dim1	numeric(9,4)		
dim1_negative_tolerance	numeric(9,4)		
dim1_positive_tolerance	numeric(9,4)		
dim2	numeric(9,4)		
dim2_negative_tolerance	numeric(9,4)		
dim2_positive_tolerance	numeric(9,4)		
dim3	numeric(9,4)		
dim3_negative_tolerance	numeric(9,4)		
dim3_positive_tolerance	numeric(9,4)		
dim4	numeric(9,4)		
dim4_negative_tolerance	numeric(9,4)		
dim4_positive_tolerance	numeric(9,4)		
dim5	numeric(9,4)		
dim5_negative_tolerance	numeric(9,4)		
dim5_positive_tolerance	numeric(9,4)		
description	text		Built using description formula
weight_units_code	varchar(3)		Defaults from product control
sales_group_code	varchar(3)		
specification_value1	varchar(30)		
specification_value2	varchar(30)		
specification_value3	varchar(30)		
specification_value4	varchar(30)		
specification_value5	varchar(30)		
specification_value6	varchar(30)		
specification_value7	varchar(30)		
specification_value8	varchar(30)		
specification_value9	varchar(30)		
specification_value10	varchar(30)		

5.2.3 Part Specification Functionality

If the Part Specification or Part Specification *and* Part Customer fields have been specified on the

import record, the program will validate that they relate to an *active* part specification within iMetal.

If they do, then a large number of sales item fields will be defaulted in from the part along with Costs, Charges and Process Plans if they are available.

The following table shows the fields that will be defaulted from the part (details of the Cost, Charge and Process Plan defaults are covered in the relevant sections later):

Field Name	Type	V/M	Notes
product_id	integer		
dim1	numeric(9,4)		(1)
dim1_negative_tolerance	numeric(9,4)		(1)
dim1_positive_tolerance	numeric(9,4)		(1)
dim2	numeric(9,4)		(1)
dim2_negative_tolerance	numeric(9,4)		(1)
dim2_positive_tolerance	numeric(9,4)		(1)
dim3	numeric(9,4)		(1)
dim3_negative_tolerance	numeric(9,4)		(1)
dim3_positive_tolerance	numeric(9,4)		(1)
dim4	numeric(9,4)		(1)
dim4_negative_tolerance	numeric(9,4)		(1)
dim4_positive_tolerance	numeric(9,4)		(1)
dim5	numeric(9,4)		(1)
dim5_negative_tolerance	numeric(9,4)		(1)
dim5_positive_tolerance	numeric(9,4)		(1)
description	text		Built using the Part description formula (2)
works_notes	text		
production_notes	text		
invoice_notes	text		
required_pieces	integer		
required_quantity	numeric(12,3)		
required_weight	numeric(10,3)		
original_quantity	numeric(12,3)		
original_quantity_unit_code	varchar(3)		
sales_group_code	varchar(3)		
specification_value1	varchar(30)		
specification_value2	varchar(30)		
specification_value3	varchar(30)		
specification_value4	varchar(30)		
specification_value5	varchar(30)		
specification_value6	varchar(30)		
specification_value7	varchar(30)		
specification_value8	varchar(30)		
specification_value9	varchar(30)		

specification_value10	varchar(30)		
show_prices	boolean		
use_minimum_grade	boolean		
outside_diameter	numeric(9,2)		
outside_diameter_minimum	numeric(9,2)		
outside_diameter_maximum	numeric(9,2)		
pack_height	numeric(9,4)		
pack_height_minimum	numeric(9,4)		
pack_height_maximum	numeric(9,4)		
pack_weight	numeric(10,3)		
pack_weight_minimum	numeric(10,3)		
pack_weight_maximum	numeric(10,3)		
inside_diameter	numeric(9,2)		
inside_diameter_minimum	numeric(9,2)		
inside_diameter_maximum	numeric(9,2)		
pack_count_minimum	integer		
pack_count_maximum	integer		
mechanical_cert	boolean		
show_country_of_material_origin	boolean		
show_country_of_primary_processing	boolean		
show_country_of_final_processing	boolean		
acknowledgement_notes	text		
despatch_notes	text		

Notes:

(1) The dimension and tolerance fields can be overridden by the import record, but only if the ‘Override Allowed’ flag on the part specification itself is ticked.

(2) The main item description will be built from the part’s description formula, but if a description is present in the import file then that will be used instead.

5.3 Sales Charges

The Sales Charges import table is optional (see below for default details) but if data is present it must relate to a specific import sales header and sales item.

Table Name: import_sales_charges

Index 1 (Unique):
import_company_reference, import_batch_number, import_number, import_item, item_no

Field Name	Type	V/M	Notes
import_company_reference	varchar(30)	M	e.g. UK, US, CHINA
import_source	varchar(50)		Source system of the import data
import_batch_number	integer	M	Reference to import batch
import_notes	text		Leave blank
import_number	integer	M	Unique sales document identity (1)
import_item	integer	M	Sales item identity (1)
item_no	integer	M	Charge item no.
cost_group_code	varchar(3)	MV	(2)
description	varchar(50)	D	Will default from sales group charge
charge	numeric(12,4)		
charge_unit_code	varchar(3)	MV	Links to ‘units_of_measure’ table
exchange_rate	numeric(14,8)	D	Defaults from currency
charge_fix_status	character(1)	VD	(F)ixed, (V)ariable (default F)
charge_visibility	character(1)	VD	(S)tand Alone, (I)ncluded, (A)dded (default S)
confirm_at_invoicing	boolean	D	False

Notes:

(1) The Import Number field must match the import number in exactly one of the sales header records that has been imported. The import_item must match exactly one of the sales item records that belongs to that sales header. This sales charge will then be attached to that sales item when the document is created in iMetal.

(2) The cost group code must be one of the following:

- MAT = Material
- TRN = Transport
- PRD = Production
- MSC = Miscellaneous
- SUR = Surcharge

The import program will validate this and use it to read in the Cost Group Code record. The id of the cost group code, along with the Sales Group id from the sales item, will then be used to read in a Sales Group Charge record. The id of the sales group charge will then be stored in the sales_group_charge_id field on the sales charges record.

5.3.1 Automatically Populated Internal Sales Charge Fields

These are fields in the iMetal sales_charges table and not the import table itself:

Field Name	Type	V/M	Notes
sales_charge_type_id	integer		Not used - null
item_id	integer		Set to sales item id
charge_quantity	numeric(12,3)		Set to item quantity using usual rules
charge_quantity_unit_id	integer		Set based on charge unit
value	numeric(12,2)		Calculated from charge and quantity
show_customer	boolean		Defaults based on Visibility flag
system_charge	boolean		True for main charge, false for others
base_charge	numeric(12,4)		Calculated from charge and exchange rate
base_value	numeric(12,2)		Calculated from value and exchange rate
charge_item	integer		Assigned sequentially within each sales item
discount_item	integer		Not used - null
created_from_part	boolean		True if charge came from part (see below)

5.3.2 Default Product Charges

If a product, but no part specification is specified on the import sales item, and no Material charge item has been specified in the import table, the program will attempt to create a default charge based on the Guide Price. The program will first look to see if there is a guide price on the Sub Group for the sold product and dimensions. If this is null or zero, then it will look for a guide price on the sold product.

5.3.3 Default Part Specification Charges

If a part specification has been specified on the import sales item, the program will first create sales charges based on the charges in the part. It will then *also* create charge records for any additional charges in the import table.

5.3.4 Header Level Transport Charges

If the imported sales header contains values in the Transport Charge Rate and Transport Charge Unit fields, the standard iMetal functionality for dealing with header level transport charges will be triggered. The charges will then be used to automatically add a Transport Charge record against each sales item.

5.4 Sales Costs

The Sales Costs import table is optional (see below for default details) but if data is present it must relate to a specific import sales header and sales item. The cost records can be either Internal or External costs.

Table Name: import_sales_costs

Index 1 (Unique):
import_company_reference, import_batch_number, import_number, import_item, item_no

Field Name	Type	V/M	Notes
import_company_reference	varchar(30)	M	e.g. UK, US, CHINA
import_source	varchar(50)		Source system of the import data
import_batch_number	integer	M	Reference to import batch
import_notes	text		Leave blank
import_number	integer	M	Unique sales document identity (1)
import_item	integer	M	Sales item identity (1)
item_no	integer	M	Cost item number
internal_cost	boolean	M	True = Internal, False = External
cost_group_code	varchar(3)	MV	(2)
cost	numeric(12,4)		
cost_unit_code	varchar(3)	MV	Links to 'units_of_measure' table
po_branch_code	varchar(3)	V	Links to 'branches' (3)
po_number	integer		(3)
po_item	integer		(3)
supplier_code	varchar(10)	MV	Links to 'companies' (4)
billing_reference	varchar(50)		
exchange_rate	numeric(14,8)		
exchange_rate_type	integer	VD	1 = Fixed, 2 = Variable (default = 1)
description	varchar(50)	D	Defaults from cost group
purchase_group_code	varchar(3)	MV	Links to 'purchase_groups' (5)
cost_fix_status	character(1)	VD	(F)ixed, (V)ariable, (M)anual (default = V)
visibility	character(1)	D	For future use - defaults to S

Notes:

(1) The Import Number field must match the import number in exactly one of the sales header records that has been imported. The import_item must match exactly one of the sales item records that belongs to that sales header. This sales cost will then be attached to that sales item when the document is created in iMetal.

(2) The cost group code must be one of the following:

- MAT = Material

- TRN = Transport
- PRD = Production
- MSC = Miscellaneous
- SUR = Surcharge

The import program will validate this and use it to read in the Cost Group Code record, the id of which will be stored in the cost_group_id field.

(3) These fields are only used for External costs. They aren't mandatory, but if present will be used to attempt to read in an iMetal purchase item. If an item is found, its id will be recorded in the cost_purchase_item_id field (see below)

(4) This field is Mandatory for External costs and ignored for Internal costs. It must refer to a valid Supplier code, which the import program will convert to the supplier_id field.

(5) This field is Mandatory for External costs and ignored for Internal costs. It must refer to a valid Purchase Group record, which the import program will convert to the purchase_group_id field.

5.4.1 Automatically Populated Internal Sales Cost Fields

These are fields in the iMetal cost_items table and not the import table itself:

Field Name	Type	V/M	Notes
cost_quantity	*numeric(15,6)*		Set to item quantity using usual rules
cost_quantity_unit_id	*integer*		Set based on cost unit
value	*numeric(12,2)*		Calculated
item_type	*character(1)*		S
item_id	*integer*		Set to sales_item's id
actual_cost	*boolean*		nulll
system_cost	*boolean*		false (see below)
quantity_type_code	*integer*		0
base_cost	*numeric(12,4)*		Calculated from cost and exchange rate
base_value	*numeric(12,2)*		Calculated from value and exchange rate
cost_purchase_item_id	*integer*		See (3) above
work_centre_id	integer		Not used - null

5.4.2 Default Material Cost

The standard sales item creation routine will automatically create an Internal Material Cost record against each sales item, representing based on the product costing rules.

5.4.3 Default Part Specification Costs

If a part specification has been specified on the import sales item, the program will first create sales costs based on the Internal and External costs in the part. It will then *also* create cost records for any additional costs in the import table.

5.4.4 Header Level Transport Costs

If the imported sales header contains values in the Transport Cost Rate and Transport Cost Unit fields, the standard iMetal functionality for dealing with header level transport costs will be triggered. The costs will then be used to automatically add a Transport Cost record against each sales item.

5.5 Allocations

This version of the sales document import will not include the ability to import allocation details. However, two existing sets of functionality will be available to automatically create allocations. Also, every item will have a generic Product Level Allocation created for it as part of the import routine.

5.5.1 Automatic Stock Allocations

If the SalesEntryAutoAllocateStock group preference is set to Y, the sales item save routine will attempt to automatically create stock allocations for each new sales item.

This will only be the case for the following sales document types:

- Sales Order
- Call Off Order

5.5.2 Automatic Process Requests

If the SalesAutoCreateProcessRoutes group preference is set to Y, the sales item creation routine will automatically attempt to create a process request for every sales item whether it has a Part Number or not.

If the preference is set to P, the save routine will still automatically create a process request if the sales item has a part number and that part has a process plan defined on it.

If the preference is set to Y and the “DefaultProcessRequestProcessTypeId” is set to a valid ‘processes’ id, then a process request will automatically be created containing a process plan with a single step.

If the preference is set to N, the save routine will never create process requests automatically.

This will only be the case for the following sales document types:

- Sales Enquiry
- Sales Order
- Call Off Order

6. Import Validation

There are two key goals to the validation step:

1. To avoid importing incorrect data into iMetal.
2. To provide information on all of the import failures back to the exporting system.

6.1 Field Validation

Each of the import tables in this document highlights which fields need to be validated and under what circumstances. In all cases, a validation failure will result in the import of the entire order being abandoned, but not before all of the fields on all of the import tables have been validated. For example, if the validation reads in the sales header and finds that the Customer doesn't exist, the import program will continue to validate all of the other fields on the header, as well as all of the fields on any sales item, charges or costs. In that way, a user examining the results of the validation will be able to resolve all of the problems at once, without having to keep trying the import after each correction.

When a field validation fails, the import program will set the 'import_status' field on the header to Failed and will add a description of the failure to the 'import_notes' field of the table being validated. This is an unlimited size text field, so the result will be something like the following:

```
branch_code: "COV" not found in branches table.
customer_code: "CLA002" not found in companies table.
exchange_rate_type_code: "6" must be one of "1", "2".
```

The import program will always use the same format for the message:

```
Table Validations:
field_name: "import value" not found in reference_file table.

Enumeration Validations:
field_name: "import value" must be one of "options list".
```

6.2 Batch Validation

The validation will take place against each sales document *within* a batch, which means that if ten documents are received in the batch and one of them fails the validation, the remaining nine documents will still be imported.

6.3 Successful Import

If a sales document is successfully imported into iMetal, the 'import_status' field will on the header will be set to Imported. It will also update the 'import_notes' field on the header with a reference to the sales document that has been created in iMetal:

```
Sales Document Created: 99999999
```

7. Triggering Imports

Two different interfaces will be provided for triggering the import of sales documents; a Selection Panel and a Command Line Interface.

7.1 Sales Document Import Panel

The Sales Document Import panel provides users with the ability to manually trigger the validation and/or importing of sales documents and will be available as a normal application on the iMetalImports sub menu.

7.1.1 Import Selection Panel

The import panel will provide the following selections:

- Import Company Reference (defaults from ImportCompanyReference preference and is not editable by the user)
- Import Batch Number Range
- Import Status (dropdown with Choose..., Entered and Failed as options). The import will never select documents with a status of Imported.
- Customer Code (with lookup)
- Import User (dropdown selection using Personnel table)

7.1.2 Buttons

The following buttons will be available:

- Validate Only - perform a validation pass on import data matching the selections
- Import - validate and import data matching the selections
- Clear - clear the selections
- Help - link to the help wiki page

Validate
The validate button will loop around all of the import sales documents matching the selection criteria and perform an import validation on each one. If any of the validations fail, the program will flag the sales header as Failed and write any failure details to the import record(s) involved.

Once the validation is complete, a dialogue will be displayed showing the results:

```
                    Sales Document Validation Results

Documents Failing Validation:   99999
Documents Passing Validation:   99999

Total Documents:                99999
```

Import

The import button will loop around all of the import sales documents matching the selection criteria and perform an import validation on each one. If any of the validations fail, the program will flag the sales header as Failed and write any failure details to the import record(s) involved. If all of the validations on a document pass, the program will then import the document into iMetal.

Once the import routine is complete, a dialogue will be displayed showing the results:

```
                        Sales Document Import Results

Documents Failing Validation:   99999
Documents Imported:             99999

Total Documents:                99999
```

7.2 Command Line Import

A command line connection to the import routine will be developed, which will allow an import run to be performed either manually on the command line or via automated scripts or ‘crontab’ entries.

The command line interface will call the same routine as the manual screen defined above, but with just three optional parameters:

```
BatchNumberFrom=999999
BatchNumberTo=999999
ImportUser=Steve_Baker
RunType=VALIDATE or IMPORT

NOTES:
If neither BatchNumber parameter is present, the import will find all imports
regardless of batch number.

If only BatchNumberFrom is present, it will find all imports with that batch
number or higher.

If only BatchNumberTo is present, it will find all imports with that batch
number or lower.

The ImportUser option must have any spaces replaced by underscores (e.g. “Steve
Baker” appears as “Steve_Baker”) in order to successfully pass the parameter
into iMetal.  iMetal will then turn the underscores back into spaces.

The RunType parameter will determine whether the program simply validates the
data or validates and imports it.  If the parameter is not present, it will
default to IMPORT.
```

Internally, the import program will only select imports with a status of Entered. It will also only select imports where the import_company_reference field matches the value of the “ImportCompanyReference” server preference.

After completion, the import routine will display the results as a simple text output:

VALIDATE:

```
VALIDATION
FOUND   : 99999
FAILED  : 99999
PASSED  : 99999
TOTAL   : 99999
```

IMPORT:

```
IMPORT
FOUND   : 99999
FAILED  : 99999
IMPORTED: 99999
TOTAL   : 99999
```

Appendix A - Technical Details

A.1 - Sales Document Creation Options

There are two possible approaches to developing the import routine, each of which has its own pro's and con's:

1. Use the existing Sales Service methods that are called by the sales document entry panel.
2. Develop a brand new application to create the sales document, its items and allocations.

A.1.1 Existing Methods

There are a number of methods in the Sales Service that get called by the sales document entry application. They each pass one or more DTO objects (e.g. Sales Header, Sales Item, etc) which are then used to create the Domain and Database records.

An import application could be written which behaves like an automated Sales Document Entry application; populating each relevant DTO in turn using the data from the import table, then calling the relevant service methods to persist that part of the sales document. It would first validate all of the data, then progress through the Header, Items, Costs, Charges and Allocations, calling each of the services necessary to save them.

The key benefits of this approach are:

- Less development required
- Future changes to the normal order save process will also be incorporated in the import

Downsides are:

- May not run as fast as a dedicated import routine
- The process will not be quite as simple as described above. The code will need to perform various operations on each imported object in order to fully populate the DTO objects (e.g. performing quantity calculations). However, this is also true of a dedicated import routine

The rest of this document assumes that the 'Existing Methods' technique will be used to develop the import.

A.1.2 New Application

Rather than making multiple calls to existing services using DTO's, a brand new application could be developed, which converts the imported data directly into Sales Document domain objects and then persists them on the database.

The key benefits of this approach are:

- May provide a more tailored application, which runs more efficiently and quickly
- Building new domain-based sales document creation logic could potentially benefit the existing save routines if it is propagated back at some point

Downsides are:

- Development and testing will require a significantly greater amount of time than reusing existing code
- The import routine and standard sales document creation routine are at risk of divergence, as new features may need to be added to both

Appendix B - Add, Change, Delete

This section describes the behaviour of the import routine as it processes documents for each *Action* type; Add, Change or Delete.

B.1 Add Document

The descriptions in the rest of this document and particularly in Appendix A, are centered around the addition of brand new sales documents to iMetal via the import program, so no further details are required here.

B.2 Change Document

When an import sales header is found with an import_action of 'C' for Change, the import routine will still validate the data in exactly the same way as is does when adding new documents. Instead of creating a new sales document in iMetal, however, the program will attempt to find the order and then make (limited) changes to it based upon the data in the import tables.

In order to find the order, the import program will do the following:

1. Initially, the program will attempt to find an existing sales document using the Branch, Type Code, Import Number, Customer and Customer Order Number.
2. If this fails, then a second attempt will be made using the Branch, Type Code, Customer and Customer Order Number. If this finds more than one order, the import status will be set to Failed and the import notes updated with a reason description: "Couldn't change sales document - More than one record was found for the Branch, Type, Customer and Customer Order Number."
3. If both attempts to find the sales document fail, the import status will be set to Failed and the import notes updated with a reason description: "Couldn't change sales document - document not found using any references."

If a unique sales document *is* found, then the program will proceed to make changes to it based upon the import data.

IMPORTANT HOWCO IMPLEMENTATION NOTE: Currently, the STELPLAN side of the Baker/STELPLAN link simply ignores change requests, although they are logged in the BizTalk system. The first release of the iMetal link will behave in the same way, therefore, and won't attempt to make any changes to iMetal sales documents based on a change request.

B.3 Delete Document

When an import sales header is found with an import_action of ‘D’ for Delete, the import routine won’t validate the data in the same way as is does when adding new documents. Instead, it will only validate the following fields on the ‘import_sales_headers’ table:

Field Name	Type	V/M	Notes
import_action	character(1)	MV	(A)dd, (C)hange or (D)elete
import_number	integer	M	
branch_code	varchar(6)	VM	Linked to ‘branches’ table
type_code	Varchar(3)	VM	Linked to ‘sales_types’ table
customer_code	varchar(10)	VM	Linked to ‘companies’ table

If these validations pass, the program will then attempt to find an existing order as follows:

1. Initially, the program will attempt to find an existing *open* sales document using the Branch, Type Code, Import Number, Customer and Customer Order Number.
2. If this fails, then a second attempt will be made using the Branch, Type Code, Customer and Customer Order Number. If this finds more than one *open* order, the import status will be set to Failed and the import notes updated with a reason description: “Couldn’t delete sales document - More than one record was found for the Branch, Type, Customer and Customer Order Number.”
3. If both attempts to find the sales document fail, the import status will be set to Failed and the import notes updated with a reason description: “Couldn’t delete sales document - document not found using any references.”

If a unique sales document *is* found, the program will then validate to decide whether it can be deleted, using the same validations that are present in the Sales Document Entry application. These include:

- Are there any Stock Allocations, Process Requests, Production Commitments, Back to Back Requests or Inbound Allocations on *any* of the items?
- Do any items already have a status of Complete?
- Are any items already on a Transport Plan?
- Are any items on a Despatch Note or Transfer Note?
- Are any items on an open Invoice (if this is a sales order)?
- Have any items already been partially invoiced (if this is a sales order)?

If any of these checks fail, then the import status will be set to Failed and the import notes will be updated with a description of the failure.

If the checks all pass, then the import program will perform the standard Sales Document Deletion routine, which will set the status of all of the items and the header to Cancelled. The values and quantities of the items will also be removed from any summary or totals tables (e.g. customer open order values, product balances, etc).

Appendix C - Existing Database Changes

The following changes will be required to existing iMetal tables.

C.1 Sales Headers

The following new fields will be added to the sales_headers table:

```
import_batch_number integer;
import_source varchar(50);
```

Both fields will be populated by the import routine after successfully creating a sales header in iMetal.

The fields will be added as optional columns to the Sales Header Enquiry application.

They will also be added as view-only fields to the References tab of the Sales Document Entry application, to the right of the existing Contract and Release Number fields.

Appendix D - Data Import Options

There are two options for populating the import tables with data.

D.1 Direct Database Connection

Using either a direct database or ODBC connection, an application could write directly to the import tables using standard SQL commands.

This is a relatively straightforward process, but with one important point to bear in mind: If the import of data is automated in any way (e.g. using 'cron' jobs) then the Import Sales Header record should not be created until all of the sub records (sales items, costs and charges) have been created. Otherwise, the import program may start processing the import before all of the data has actually been created.

There are two ways of achieving this:

1. Wrap the creation of the import records in a Database Transaction. This will mean that the records only become visible to the import program once the entire transaction has been committed. There are two further advantages to this method: If a batch of sales documents are included in the transaction, none of them will be visible until they've all been written; and if any failures occur during the creation of the import data, the whole transaction can be rolled back without creating only part of the data.
2. Ensure that the Import Sales Header record is the *last* record to be written to the import tables, after any items, costs and charges have been written. Since the import program only looks for Header records, it will then never get a chance to process a partially created sales document.

D.2 CSV Import

This method can be used for exporting systems that have no ability to write to SQL databases. The import table layouts described above are still used, except that they are flattened into comma-separated format. All of the notes above are still relevant, therefore.

D.2.1 Basic Rules

The basic rules of the CSV import file are as follows:

- One CSV file is required for each import table (Sales Headers, Sales Items, Costs and Charges)
- Each field is separated by a comma.
- Where text fields contain single or double quote characters, they must be preceded with a backslash. For example; \" or \'
- Where a comma is present in the text, a backslash escape character must precede it: \,
- Empty fields can be left completely empty, so "1,,3" would import values of '1', null and '3'.
- Each record is separated by a single line feed character (ASCII 10, hex 0a).
- Linefeed characters within text are represented by the \n escape sequence.
- Each field in the import file must be presented in the exact order defined by the layout above.
- Every field in the above layout must be present in each line of the import file.
- Date fields must be in the format CCYY-MM-DD (e.g. 2014-08-28).

D.2.2 Import Process

In order to import CSV data into the iMetal import tables, the following steps are required *for each import table*. The import_sales_headers table is used in this example:

```
Copy the import file to /usr2/base_emetal on the destination server

Log on to to the server as base_emetal

psql import_imetal    (name of import database)

Type:
COPY import_sales_headers FROM '/usr2/base_emetal/import_file_name DELIMITER ','
null AS ;

If the command is successful, then a response should be given saying "COPY n"
where n is the number of records imported.  Note that the only validations
performed at this stage are to make sure that all the fields are present and
that each field is of the correct type.  The import will stop at the first
failure.

To list the imported records:
select * from import_sales_headers;
```

Appendix E - Test Plan

In order to comprehensively test the import functionality, the following tests will need to be performed:

- Basic sales order creation
- Creation with the *absolute minimum* of import data - i.e. only the Mandatory fields
- Creation using *every single field* to make sure they are all recorded in iMetal
- Using Products only (i.e. no Part)
- Using Parts only (i.e. no Product)
- With fully populated parts (process plan, internal costs, external costs, charges, etc)
- With parts that have minimal data
- Using all standard combination of stock and service products, covering each type of Quantity and Cost calculations (e.g. Bars, Sheets, Coils, Plates, Services, etc, etc)
- Every different document type (e.g. Enquiries, Orders, Transfers, Invoices, Credit Notes, etc)
- Every validated field must be tested using data that should fail to ensure that the import is blocked and that the reason is reported correctly

Appendix F - Functionality Not Covered

The following functionality is *not* covered in this development:

- In the initial release, Change requests will not be processed
- Imports of Stock, Process Request, Back to Back or Inbound allocations
- Enquiries or reports listing import table data
- Maintenance programs for each import table
- Deletion of successfully imported data [(1)]

[(1)] The indexes on each import table have been designed to allow import data to be stored indefinitely without impacting on the performance of the import search routines. However, the 'import date' field which gets stamped onto the import sales header will allow for the development of a deletion routine that can remove records older than a certain date.