IMMS-ILS Interface specification

Intelligent Material Management System (34296-000)

CONTENTS

1 INTRODUCTION.. 4
2 CONTROLLING ITEMS (FLOW ETC.).. 5
2.1 Data from ILS.. 5
2.2 The intelligence in IMMS.. 5
2.3 Feedback to ILS.. 6
3 SCENARIOS/PROCESSES.. 7
3.1 Delivering reserved items.. 7
3.2 Returning "external" items.. 7
3.3 Ordering, followed by picking.. 7
3.4 Requisition with subsequent taking at item return.. 7
3.5 Setting up on pick-up shelf.. 8
3.5.1 Reserve notes via ILS.. 8
3.5.2 Without reservation notes.. 8
3.6 Checking out ordered items.. 8
3.7 Uncollected or cancelled orders/reservations.. 9
3.8 Returning and sorting in self-service or sorting machines.. 9
3.9 Returning via ILS.. 9
3.10 Replenishment of stock.. 9
3.11 Reception.. 9
3.12 Logging new IMMS status.. 9
3.13 Setting up items.. 10
3.14 Purchasing new items.. 10
3.15 Discard order from ILS.. 10
3.16 Item care.. 10
3.17 Discarding via mobile equipment.. 10
3.18 Taken requisition fulfilled in ILS.. 10
3.19 Requisition is temporarily set on hold.. 10
4 DATA EXCHANGED BETWEEN THE TWO SYSTEMS.. 12
4.1 Scope of data.. 12
4.2 Notes on orders/reservations.. 13
4.3 Handling a double taking/picking.. 15
4.4 Dealing with changes to an item's status.. 15
4.5 Notes on sorting points and chutes.. 15
5 PRINCIPLES FOR COMMUNICATION BETWEEN THE TWO SYSTEMS.............. 17
5.1 Communication form.. 17
5.2 Dealing with time.. 17
5.3 Basic principle for synchronised data.. 18
5.4 Initial data.. 18
5.5 Repeating initial data transfer / full synchronisation.. 18
5.6 Notifications.. 19
5.7 Requests.. 19
5.8 Security.. 19
6 WEB SERVICE FUNCTIONS.. 21
6.1 Initial data (REST) call to ILS from IMMS.. 21
6.1.1 Get metadata.. 21
6.1.2 Get all float codes.. 21
6.1.3 Get all branches.. 22
6.1.4 Get all departments.. 22
6.1.5 Get all locations.. 22
6.1.6 Get all sub-locations.. 22

6.1.7 Get all item collections 23
6.1.8 Get all discard reasons 23
6.1.9 Get all sorting points........ 23
6.1.10 Get all chutes 23
6.1.11 Get all titles........ 24
6.1.12 Get all items 24
6.1.13 Get all requisitions........ 24
6.1.14 Get all taken requisitions........ 24
6.2 SOAP web service operations called by ILS from IMMS 25
6.2.1 Receive notifications 25
6.2.2 Initial data processed 25
6.2.3 Ping 25
6.3 SOAP web service operations call to IMMS from ILS 26
6.3.1 Receive notifications 26
6.3.2 Assignment of item to branch 27
6.3.3 Initial data ready........ 27
6.3.4 Ping 27
7 DATA LIST FROM ILS TO IMMS........ 28
7.1 Float codes 28
7.2 Branches........ 28
7.3 Departments 29
7.4 Locations 29
7.5 Sub-locations........ 29
7.6 Item collections........ 29
7.7 Reasons for discarding........ 30
7.8 Sorting points........ 30
7.9 Chute 30
7.10 Titles 31
7.11 Items 32
7.12 Requisitions 33
7.13 Taken requisitions 33
7.14 Requisition cancelled/deleted........ 34
7.15 Item checked out 34
7.16 Item discarded........ 34
7.17 Item returned and sorted to chute 35
7.18 Order created/updated 35
7.19 FAUST number changed 36
7.20 Item ID changed 36
7.21 Order deleted........ 36
8 DATA LIST FROM IMMS TO ILS........ 37
8.1 Item updated........ 37
8.2 Item taken for a requisition 37
8.3 Item discarded........ 37
8.4 Item placed on reserve shelf 38

1 INTRODUCTION

This document defines the overall communication between Intelligent Material Management System (IMMS) and an Integrated Library System (ILS) connected.

This specification will not determine which fields in a given ILS will be used for what. Instead, it will define communication between IMMS and ILS, leaving it up to the customer and/or an ILS supplier to decide which fields are to be used.

> **NB:**
> IMMS is a central system that can be used by multiple authorities/customers, each with their own ILS. Data in the central IMMS is segregated, such that data belonging to different customers cannot be mixed up. "IMMS" is regarded in this document as the central system for a single customer.
> As such, there will be one IMMS for each ILS in the overall presentation in relation to the ILS interface.

We deliberately chose not to facilitate transfer of personal data from ILS to IMMS, thus avoiding any problems with data protection legislation.

2 CONTROLLING ITEMS (FLOW ETC.)

In relation to the existing functionality in an ILS, one of the major contributions of IMMS is to assign a suitable branch to an item when returned.

IMMS will assign a branch based on the following:

- Details on the item returned
- Details on the placement of all the other items
- Rules set up in IMMS

IMMS will also be able to gather logs of the item's movements using mobile units, to obtain a better picture of where they are between branches and within a given branch.

2.1 Data from ILS

ILS informs IMMS via an initial data set and subsequent change notifications such details on the items as follows:

Data concerning an individual copy:

- Item ID
- Float code, (if the item is part of a "floating" set)
- Branch
- Department
- Location
- Sub-location
- Item collection

(Profile values: Branch, department, location, sub-location and item collection can be found as a fixed value and a current value).

Data from the library MARCpost

- FAUST number, item type, title, author, DK5 group, pagination etc.

See sections 7.10 and 7.11 to follow for a complete list of fields.

IMMS can then use a set of predefined rules to make its calculations.

2.2 The intelligence in IMMS

When an item is returned and the ILS should not handle it itself (external loan/reserved/ordered/...), it will ask IMMS which branch the item should be sent to. IMMS will respond with branch and department codes.

> **NB:**
> Regardless of whether the item belongs to a given branch, IMMS can always assign it to an item depot.

ILS can ask IMMS about branch several times, e.g. initially for returning the item in a self-service machine and subsequently during central sorting. The underlying data can have changed in the meantime, so IMMS will not necessarily give the same response each time it is asked.

2.3 Feedback to ILS

When the item is registered as moved or in some other way changes its IMMS status, IMMS will notify ILS accordingly:

- Current branch
- Current department (if the item has a float code, otherwise remains blank)
- Placement text (name of carrier and/or designation of the specific shelf)
- Availability (yes/no) (implicitly selected by the user when selecting new status in IMMS)
- Status text (e.g. "Ready for setting up")

With regard to the item's "department", IMMS will inform ILS of the new branch and department when an item with a float code is moved to a new branch.
IMMS will not be concerned with a department for items with no float code, as that is controlled by ILS.

> **NB:**
> An item can be placed at any branch in IMS, regardless of any rules applied for ownership and float. This does not apply only to item depots.
> If an item with no float code is placed at another branch than its home branch, the right department will not be known to IMMS.

IMMS does not want to be notified of the changes in ILS arising from such feedback.

3 SCENARIOS/PROCESSES

This chapter looks at various scenarios related to the relationship between ILS and IMMS.

3.1 Delivering reserved items

- Reserved items are returned at the desk or from a self-service machine.
- ILS will detect that the item is reserved and decide which lender is to have it.
- ILS can notify the desk, self-service machine or sorting robot itself, to ensure that the item is sent to the right place.
- **ILS will notify IMMS** that the item (item ID) has been taken for a requisition and where it should be picked up.
- Any enquiries to IMMS will now also send the item to the pick-up branch.

3.2 Returning "external" items

These are items which have been loaned from a third party lender (another authority) or belong to a branch excluded from IMMS (e.g. a hospital).

- External items are returned at the desk or from a self-service machine.
- ILS can notify the desk, self-service machine or sorting robot itself, to ensure that the item is sent to the right place.

3.3 Ordering, followed by picking

- ILS: A requisition is created.
- **ILS notifies IMMS** of the requisition (list of item IDs + optional picking branch + pick-up branch).
- IMMS: Item picked (now with item ID) and registered in a transport unit.
- **IMMS notifies ILS** that the item has been taken for the requisition.
- **IMMS notifies ILS** of the item's new location.
- The item may be sorted in a sorting machine.

Process "Setting up on pick-up shelf" then follows.

If the item is not found:

- IMMS: Failure to find the item is logged on the mobile pick list.
- IMMS registers all examples of the same location at the branch as "not found".
- **IMMS notifies ILS** that the items have been discarded with the reason belonging to "not found".
- If there is no other suitable item at the branch, ILS will find another branch to pick the item, and change the requisition to a reservation if there are no items available.
- **ILS notifies IMMS** of a change in the requisition (if the item has to be picked from another branch), or that the requisition is cancelled/inactive (requisition becomes a reservation).

3.4 Requisition with subsequent taking at item return

- ILS: A requisition is created.
- **ILS notifies IMMS** of the requisition (list of item IDs + optional picking branch + pick-up branch).

- An item is returned that corresponds to the requisition. IMMS has not yet notified ILS that an item has been taken for the requisition by picking it.
- ILS takes the item for the requisition and sends the item to the pick-up branch.
- **ILS notifies IMMS** that the original requisition has now been taken (with item ID).
- The item may be sorted in a sorting machine.

Process "Setting up on pick-up shelf" then follows.

3.5 Setting up on pick-up shelf

There are several ways of setting items up on pick-up shelves. In particular, there are several models in use for creating reserve notes (slips).

The following 2 sub-sections describe 2 different scenarios.
Both presume starting with a transport unit with items taken for requisitions.

3.5.1 Reserve notes via ILS

In this scenario, a staff member scans the item with an ILS scanner one at a time, placing a slip inside the item.

The staff member can then choose to either set up each item on the reserve shelves one at a time as soon as a slip has been inserted and then scan the items set up one by one, or set up all the items from the box on the reserve shelf and scan the box set up, whereby all the items from the box are given the status of ready for pick-up.

Once one or all the items from a box are scanned and set up, IMMS will notify ILS that they are on the reserve shelves. ILS will then notify the patron that the item can be picked up along with expiry date and slip number.

- ILS: Item scanned and slips printed.
- IMMS: An item or transport unit with items is registered as set up on the reserve shelf.
- **IMMS notifies ILS** that the item is on the reserve shelf.
- ILS notifies the patron that the item is ready to be collected (with slip number).

3.5.2 Without reservation notes

In this scenario, no slips are printed, but all items are scanned individually on the reserve shelves, e.g. rack 3, shelf 7, compartment d.
When an item is scanned set up in a specific location, ILS notifies the patron that it is ready to be picked up, including expiry date and precisely where it can be found (e.g. rack 3, shelf 7, compartment d).

- IMMS: The item is registered as on a specific reserve shelf.
- **IMMS notifies ILS** that the item is on the specific reserve shelf ready for pick-up.
- ILS notifies the patron that the item is ready to be picked up (with shelf number).

3.6 Checking out ordered items

- ILS: The item is picked up and checked out.
- **ILS notifies IMMS** that the item is checked out and that the taken requisition is deleted.

3.7 Uncollected or cancelled orders/reservations

- ILS: The requisition (taken or not) is cancelled.
- **ILS notifies IMMS** that the requisition is cancelled (with reason if possible).

The items are then processed as if they were returned.

3.8 Returning and sorting in self-service or sorting machines

Self-service and sorting machines communicate with ILS only.

- **ILS will ask IMMS** (unless the item is reserved/ordered/external loan/...), where the item is to be sent. IMMS replies with a branch assigned based on its computations.
- ILS replies to self-service machine/sorting machine.
- The patron (or sorting machine) places the item in a transport unit.
- **ILS notifies IMMS** of which sorting point and chute the item is (ordered to be) placed in.

3.9 Returning via ILS

- ILS: The item is registered as returned.
- **ILS will ask IMMS** (unless the item is reserved/ordered/external loan/...), where the item is to be sent (basically the same as for a machine).
- ILS informs the staff member how the item is to be handled.
- The staff member will place the item in a transport unit.
- **ILS notifies IMMS** of which sorting point and chute the item is (ordered to be) placed in.

3.10 Replenishment of stock

- IMMS decides that an item should be moved from one location to another.
- IMMS generates a pick order (item ID and pick location).
- IMMS: Item is picked and registered in a transport unit.
- **IMMS notifies ILS** that the item has changed IMMS status and location.
- The item can be sorted in a sorting machine where IMMS distributes it as normal.

3.11 Reception

- A transport unit with items is unloaded at a branch.
- IMMS: The transport unit is registered at the new branch.
- **IMMS notifies ILS** that each item is now at the branch along with new location text.
- The items might be manually sorted into new transport units, depending on type/location as preparation for setting up the items.
- IMMS: The item is registered as placed in a new transport unit.
- **IMMS notifies ILS** that the item has changed placement text.

3.12 Logging new IMMS status

- IMMS: A transport unit or individual item is logged with new IMMS status.
- **IMMS notifies ILS** that the item is changed (new status text and possibly also new availability).

3.13 Setting up items

- IMMS: A transport unit is registered as set up or an item is registered at a specific location (e.g. on show or item depot).
- **IMMS notifies ILS that** the item is "in place" at the branch or states a specific placement text for the item.

3.14 Purchasing new items

This scenario also covers instances of "reviving" discarded items.

- ILS: A new item is registered in ILS.
- **ILS notifies IMMS** of a new title if a new FAUST number is involved.
- IMMS: If IMMS does not have a title with the relevant FAUST number, it will be created in IMMS
- **ILS notifies IMMS** of the new item.
- IMMS: If IMMS does not have an item with the relevant FAUST number, it will be created in IMMS

The items are then processed as if they were delivered.

3.15 Discard order from ILS

- ILS logs that an item is to be discarded (automatic discard or manual handling).
- **ILS notifies IMMS** that the item is to be discarded along with the reason (if found in ILS).
- IMMS: Items to be discarded can be seen on the mobile equipment.

Process "Discarding via mobile equipment" then follows.

3.16 Item care

- ILS: A staff member extracts a list of items for item care.
- **ILS notifies IMMS** of each item on the list.
- IMMS: Items to be evaluated can be seen on the mobile equipment.

Process "Discarding via mobile equipment" or "Setting up items" can then follow.

3.17 Discarding via mobile equipment

- IMMS: The user registers an item as discarded and selects the discard reason.
- **IMMS notifies ILS that** the item is discarded and the reason.

3.18 Taken requisition fulfilled in ILS

- ILS: A staff member scans a reserved item that ILS then logs as fulfilled.
- **ILS notifies IMMS** that the taken requisition is now fulfilled.

3.19 Requisition is temporarily set on hold

In this scenario a requisition is temporarily set on hold, e.g. because there no longer is an available item to be picked or because the patron has specified a holiday period.

- ILS: A requisition is created.

- **ILS notifies IMMS** of the active requisition (list of item IDs + optional picking branch + pick-up branch).
- IMMS: A picking order is generated.
- ILS: The requisition is set on hold.
- **ILS notifies IMMS** that the requisition is now inactive or that the requisition is deleted.
- IMMS: The picking order is cancelled.
- ILS: The requisition is reinstated.
- **ILS notifies IMMS** that the requisition is now active.
- IMMS: A new picking order is generated.

The process continues with picking, putting on reservation shelf and checkout as usual.

4 DATA EXCHANGED BETWEEN THE TWO SYSTEMS

IMMS needs a local copy of some of the data in ILS:

Basic data:

- Float codes
- Branches
- Departments
- Locations
- Sub-locations
- Item collections
- Reasons for discarding
- Sorting points
- Chutes

Item data:

- Titles
- Items

Orders:

- Requisitions
- Taken requisitions

Information is also exchanged between the two systems on various events:

Events in ILS notified to IMMS:

- Data for the above types created/updated/deleted
- Item checked out
- Item discarded
- Item returned and sorted to chute
- Order for discarding/item care/ad hoc created
- Order deleted
- FAUST number changed
- Item ID changed

Events in IMMS notified to ILS:

- Initial data processed
- Item updated with new values for {branch, department, availability, IMMS status text, placement text}
- Item discarded
- Item taken for a requisition
- Item placed on reserve shelf

ILS can also ask IMMS about:

- Request assignment of an item to a branch

The specific fields in each piece of data and events are specified in the data lists later in this document.

4.1 Scope of data

The customer can decide (in consultation with the ILS supplier if relevant) which items ILS should inform IMMS of. For example: historical items (e.g. discarded) will be left out of the initial data.

External loan items can be transferred to IMMS as for any other item, or can be kept out of IMMS. Similarly, certain branches and their items can be excluded from IMMS if required.

None of these decisions have any influence on specification of the interface, as they only affect *the scope* of data, and not *how* that data is transferred.

However, it is important that ILS does not notify IMMS of events for items that are excluded.

Items on the way from the supplier are excluded from IMMS until they are received. Alternatively, they can be transferred with ILS status "Lent out" or some other indication of unavailability.

Titles with no items in stock should not be transferred to IMMS from ILS.

4.2 Notes on orders/reservations

IMMS needs to know the requisitions to be picked at the branches, and those requisitions that have already been taken upon return.

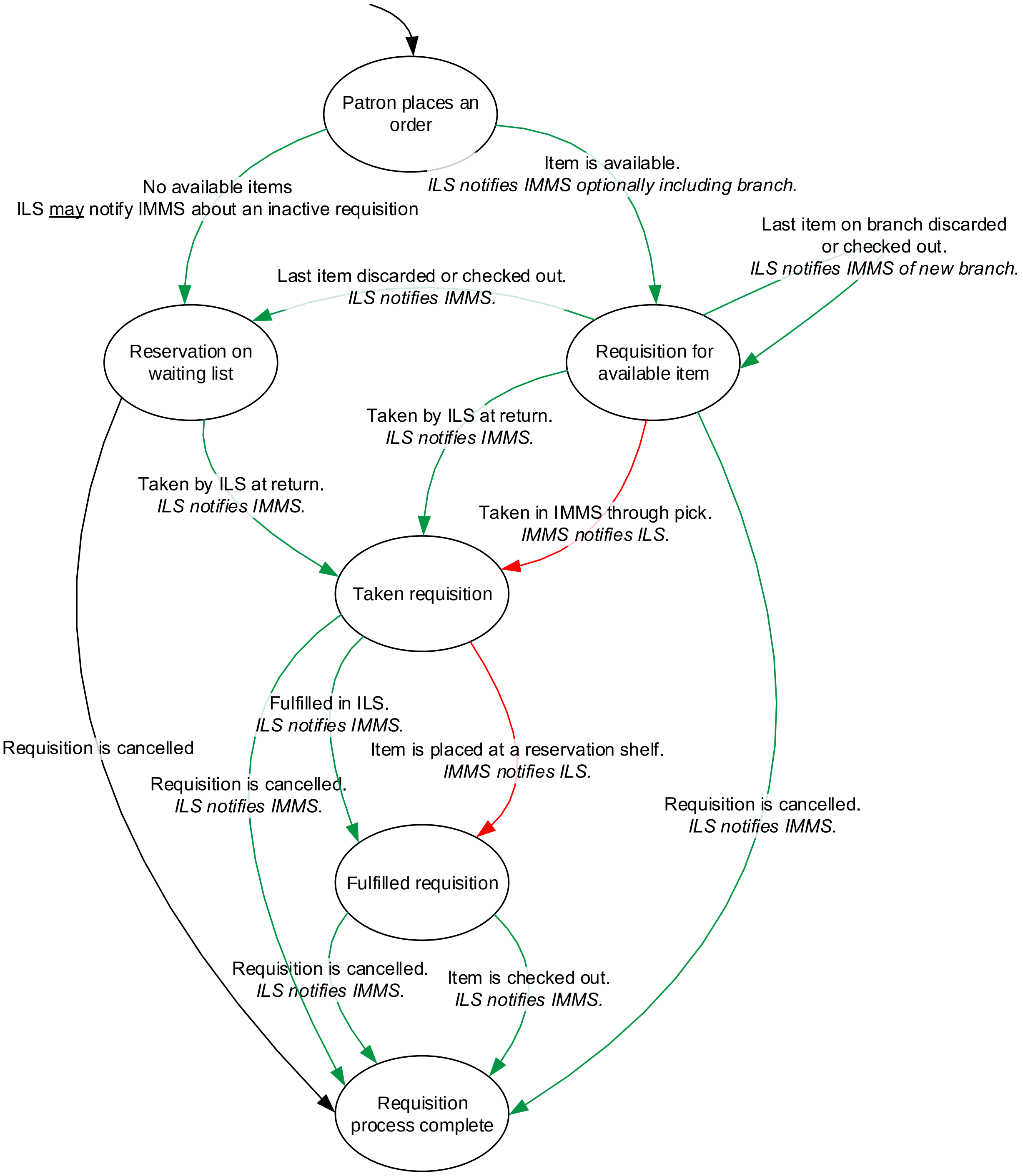
As long as there are no available items for a reservation, IMMS needs not be aware of the reservation. ILS may notify IMMS of the requisition, but in this case the requisition must be marked inactive.
When an item becomes available, ILS must notify IMMS that the requisition is now active.
Conversely, if there no longer is any available item or the requisition for any other reason is not to be picked now (e.g. patron is on vacation), ILS must notify IMMS that the requisition is inactive or delete the requisition entirely.
If the requisition is no longer relevant (e.g. the patron has cancelled it or has checked out the item), ILS must in any case notify IMMS that the requisition is deleted. This is regardless of whether the requisition was active or inactive at the time.

It is up to ILS to ensure that there are sufficient items available (at the designated branches) to meet the requirement for the total number of requisitions transferred to IMMS.

When picking for a requisition, ILS may decide which branch to pick from. ILS can also choose not to specify a pick branch, in which case IMMS will choose a suitable branch based on the placement of the specified items.

The following diagram indicates the various events that can occur.

The *green* arrows are events that ILS notifies IMMS of.
The *red* arrows are events that IMS notifies ILS of.
The *black* arrows are events that do not cause communication between ILS and IMMS.


Patron places an order
No available items
ILS may notify IMMS about an inactive requisition
Item is available.
ILS notifies IMMS optionally including branch.
Last item on branch discarded or checked out.
ILS notifies IMMS of new branch.
Last item discarded or checked out.
ILS notifies IMMS.
Reservation on waiting list
Requisition for available item
Taken by ILS at return.
ILS notifies IMMS.
Taken by ILS at return.
ILS notifies IMMS.
Taken in IMMS through pick.
IMMS notifies ILS.
Taken requisition
Fulfilled in ILS.
ILS notifies IMMS.
Requisition is cancelled
Item is placed at a reservation shelf.
IMMS notifies ILS.
Requisition is cancelled.
ILS notifies IMMS.
Requisition is cancelled.
ILS notifies IMMS.
Fulfilled requisition
Requisition is cancelled.
ILS notifies IMMS.
Item is checked out.
ILS notifies IMMS.
Requisition process complete

IMMS basically needs to be kept updated with the requisitions to be picked and those requisitions already taken. IMMS *can* also be kept informed about reservations not to be picked yet (inactive).

This gives rise to the following notifications from ILS to IMMS:

- Requisition created as active (to be picked)
- Requisition created as inactive (not to be picked yet)

- Requisition changed from active to inactive or vice versa
- Requisition taken (requisition updated with a specific item ID)
- Taken requisition created (with a specific item ID)
- Requisition fulfilled (taken requisition now tagged as fulfilled)
- Requisition cancelled

4.3 Handling a double taking/picking

A situation can arise where a returned item being taken for a requisition in ILS when at the same time another item is being picked and registered as taken for the same requisition in IMMS.

The two systems will notify each other:

- ILS sends a notification to IMMS that item X has been taken for requisition R.
- IMMS sends a notification to ILS that item Y has been taken for the same requisition.

Regardless of which notification is sent first, the result must be that the two systems will agree which item belongs to the requisition.

The problem is dealt with using an "*ILS always wins*" strategy:

- ILS ignores notification from IMMS if an item was already taken for the requisition in ILS.
- IMMS overrules its own registration if conflicting data is received from ILS.

4.4 Dealing with changes to an item's status

In relation to the interface, an item will have one of the following ILS statuses at any given time:

NotCheckedOut: Not checked out and not discarded.
CheckedOut: Checked out.
Discarded: Discarded. No longer part of the stock.

A situation can arise in which details of an item are updated in IMMS and in ILS, and the two systems do not manage to notify each other in time. Both systems simply being updated with the other system's new data should be avoided, as it will result in the two systems no longer having the same details of current status.
The problem is particular to item status (checked out/in stock/discarded), as the other types of data exchanged go either one way or the other.

It is dealt with by time-stamping data and ignoring a notification when received if older than the existing data for the item. Notifications are time-stamped with event time already.

In the event of discarding an item, a situation can also arise in which a patron has lent an item in ILS, after a staff member has generated a picking list that includes the item in IMMS. The staff member could wrongly conclude that the item is lost, and register the fact in IMMS.
It is important that ILS does not accept discarding as "not found" from IMMS if the item has already been registered as checked out in ILS. Discard notification from IMMS to ILS explicitly includes this information.

If IMMS sends notification of an item being discarded and ILS has noted the item as taken (fulfilled) for a requisition, discarding should be ignored by ILS.

4.5 Notes on sorting points and chutes

Sorting point is the name used for the places where items are distributed into one or more transport units, e.g. for returning. Examples of sorting points are self-service machines, sorting robots, desks and offices.

Each sorting point has a number of *chutes*. For a self-service machine or desk/office, chutes are where the patron/staff member can subsequently place the item (e.g. carts for children's books, adult books, ...).

Once an item is logged in ILS as returned, its new location is identified in IMMS by { branch + sorting point + chute }.

ILS can transfer lists of possible sorting points and chutes to IMMS, making it possible to perform the additional configuration in IMMS in advance.
If ILS has not transferred the sorting point and/or chute to IMMS before notification is sent (see section 7.17) of an item being returned at the relevant sorting point and sorted to the relevant chute, IMMS will create the missing sorting point or chute itself.
It is therefore not strictly necessary that (all) sorting points and chutes are explicitly transferred from ILS to IMMS, either as initial data or ongoing notifications.
However, when initial data is processed, IMMS will always ask for all the specified files (including sorting points and chutes). ILS should therefore reply with minimum an empty file.

5 PRINCIPLES FOR COMMUNICATION BETWEEN THE TWO SYSTEMS

The interface between ILS and IMMS consists of three types of data exchange:

- Data exchanged upon startup
- Ongoing transfer of changes to keep the data in both systems updated
- Direct queries from one system to the other

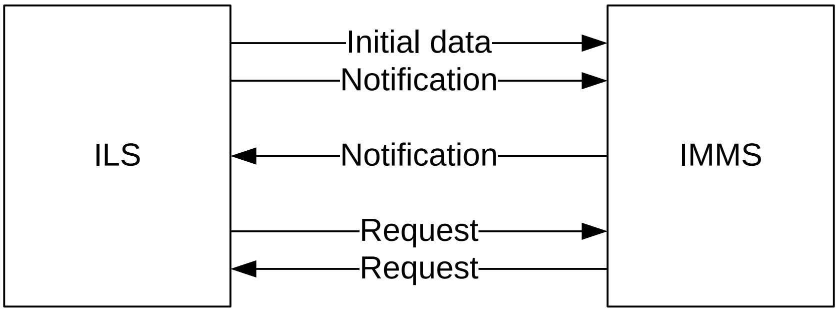


Figure 1: Illustration of the different types of data exchange between the systems. The arrows indicate the direction of data flow.

5.1 Communication form

At the technical level, the two systems communicate via web services over HTTPS. Calls are made in both directions.

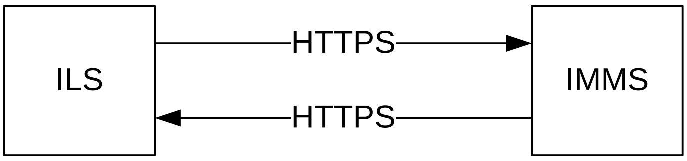


Figure 2: Illustration of the basic communication routes. The arrows indicate the direction for setting up network connection (call direction).

In terms of networks, communication can if necessary take the form of traffic directed through the authority's network. If so, it will be transparent in relation to the HTTPS connection – although a slightly longer delay in the network connection can be expected.

Both parties can regularly test whether the connection is OK by calling a "no-content" web service at the other party (described later).

5.2 Dealing with time

To avoid confusion and problems with time zones and seasonal time changes, all time stamps are exchanged in UTC.

Both servers use NTP independently of each other to keep their clocks correctly set.
A couple of seconds of time difference is regarded as insignificant.

5.3 Basic principle for synchronised data

Some of the data has to be available in both systems. This is primarily data from the existing ILS to be used regularly in IMMS, but ILS also receives data from IMMS.

Data is exchanged at start up, by IMMS asking ILS for initial data which IMMS then imports. Subsequent changes in data are communicated as quickly as possible (almost "live") in both directions between the two systems, so that both always have the same impression of status.

To make this possible, a set of common rules are technically defined for data:

- Each record has a unique identification (primary key) that never changes.
- If just one of the fields in the record is changed, the entire record is sent again (as if it was new).
- If a record is deleted, its identification is sent.
- An identification can only be reused if the same logical unit is reinstated.
- Notifications are sent in the original order of the events.

5.4 Initial data

The purpose of initials data is to feed IMMS with all necessary data from ILS when starting up. The systems are then kept updated with changes using notifications from each other.

Actual transfer of the initial data is made as a set of (REST)queries system to system via HTTPS. Data is returned as CSV. The character set used is ISO-8859-15.

Chapter 7 defines field lists for each data set.

Because the initial data set is quite large and can require a lot of resources to generate, IMMS cannot initiate its generation itself.
The following procedure is used:

1. In ILS a user initiates generation of the initial data.
2. ILS will then put notifications of data changes in a queue without sending them to IMMS.
3. Once generation of the initial data set is complete, ILS will notify IMMS.
4. IMMS retrieves and processes the initial data which is available on the ILS server.
5. Once IMMS is finished processing the initial data, it notifies ILS which starts sending notifications of data changes to IMMS.

> **NB:**
> It is important that subsequent change notifications are queued in ILS right from the point at which generation of the initial data set is initiated. Otherwise, there is a risk that the two systems can become out of synch.

5.5 Repeating initial data transfer / full synchronisation

Transferring an initial data set can be repeated if deemed necessary at a later date.

Given that IMMS already has a lot of details, data from the new initial data set is merged with existing data.
For example, IMMS has details of the item's more precise location (in a chaos stockroom, display shelves, in transport units etc.), which must of course not be overwritten with data from ILS.
In principle, doing a new initial data set transfer can be regarded as a "mass correction".

When loading an initial data set, IMMS keeps track of the existing data that may not have been included in the new data set and that therefore may need to be deleted or specially tagged.

If it becomes necessary to post-synchronise data from IMMS to ILS, it can be done via notifications.

5.6 Notifications

After initial data has been loaded, the systems are kept mutually updated using change notifications.

Change notifications have to be sent as soon as possible (a delay of a few seconds is normally acceptable). If notifications cannot be delivered because of technical problems (e.g. network breakdown or problems with the receiving system), they should be put into a queue in the sending system, and regular checks made for resumption of communication (e.g. every minute).

Each system should send change notifications to the other in the same sequence as generated.

Notifications are sent via SOAP web service call via HTTPS.

Several change notifications can be collected into one web service call to reduce network overheads. Up to 1000 notifications can be sent in one call.

No reply is sent to a notification. If there is a problem preventing the receiving system from accepting a notification, an error message in the form of a SOAP fault is sent. If several notifications are combined in a call and an error occurs with just one of them, the call will fail and the error message back to the sender will indicate the index of the first erroneous notification in the call. The error message will also provide details of the problem in 'technical language'. The sender cannot consider the previous notifications in the call as delivered, and will have to send them again.

Receiving the same notification (or the same call with collected notifications) multiple times must not cause problems. This also means that creating and updating records use the same type of notifications.

When the sending system has received an error-free (empty) reply to a web service call, the notifications sent can be deleted from the outbound queue.

5.7 Requests

The two systems can send requests to each other via SOAP web service calls via HTTPS, in common with notifications.

The calling system will usually need a response, e.g. when ILS asks IMMS about assignment of a branch for a returned item. An enquiry is made "live" and not put into a queue.

A special "ping" request has also been defined which both systems can use to call the other. This has no function other than that the calling system can check that there is a connection and that the other system is alive. The response is empty.
There is no requirement that ILS shall use this type of enquiry actively, but it must be able to receive and respond to it. IMMS use it to monitor its operation. The function will be called by ILS, e.g. once every minute.

5.8 Security

As communication between the two systems can be made in principle over the open Internet, access has to be protected accordingly.

Actual communication is secured against eavesdropping and modification by use of HTTPS.
Using certificates signed by a generally trusted certificate authority (CA) is recommended.

Access to call the web services on the two systems is restricted by the requirement for a set of authorisation credentials to accompany any call in the form of user name and password. Each system uses one set of user name/password to communicate with the other.

Communication can be further protected by restricting which IP addresses that can connect from the network. This lies outside the scope of this specification.

6 WEB SERVICE FUNCTIONS

This chapter describes the web service functions that make exchange of the above data possible.

6.1 Initial data (REST) call to ILS from IMMS

The initial data set consists of a number of files in CSV format that IMMS retrieves from ILS.
The initial data must have been generated in advance on the ILS server. This is started manually from ILS.

The files are downloaded via HTTP(S). The URL for each file consists of a fixed element (server name and path prefix) agreed for each installation, plus a file name defined in the following sub-section.
There is no actual input to HTTP calls with the current interface. The HTTP method 'GET' is used.

6.1.1 Get metadata

Returns metadata that describes the initial data set.

File name: `Meta.csv`

Input:
- (none)

Output:
- Metadata. Result delivered in CSV format.

Data consists of one row with following content:

Field no.	Field name	Data type	Remarks
1	InitialDataTime	Timestamp	Start time for generation of the initial data set. All notifications generated before the generation of the initial data is initiated must be time-stamped before this time. All notifications generated after generation of the initial data is initiated must be time-stamped after this time. The time stamp is in UTC.

See also section 5.4.

6.1.2 Get all float codes

Returns a list of all float codes in ILS.

File name: `FloatCodeRecord.csv`

Input:
- (none)

Output:

- All float codes (see section 7.1 for data list with fields). Result delivered in CSV format.

6.1.3 Get all branches

Returns a list of all branches in ILS.

File name: `Branch.csv`

Input:
- (none)

Output:
- All branches (see section 7.2 for data list with fields). Result delivered in CSV format.

6.1.4 Get all departments

Returns a list of all departments in ILS.

File name: `Department.csv`

Input:
- (none)

Output:
- All departments (see section 7.3 for data list with fields). Result delivered in CSV format.

6.1.5 Get all locations

Returns a list of all locations in ILS.

File name: `Location.csv`

Input:
- (none)

Output:
- All locations (see section 7.4 for data list with fields). Result delivered in CSV format.

6.1.6 Get all sub-locations

Returns a list of all sub-locations in ILS.

File name: `Sublocation.csv`

Input:
- (none)

Output:
- All sub-locations (see section 7.5 for data list with fields). Result delivered in CSV format.

6.1.7 Get all item collections

Returns a list of all item collections in ILS.

File name: `Collection.csv`

Input:

- (none)

Output:

- All item collections (see section 7.6 for data list with fields). Result delivered in CSV format.

6.1.8 Get all discard reasons

Returns a list of reasons for discards in ILS.

File name: `DiscardReason.csv`

Input:

- (none)

Output:

- All item discard reasons (see section 7.7 for data list with fields). Result delivered in CSV format.

6.1.9 Get all sorting points

Returns a list of all the sorting points used for sorting items.
These can be self-service machines, sorting machines and desks/offices.

File name: `SortingPoint.csv`

Input:

- (none)

Output:

- All sorting points (see section 7.8 for data list with fields). Result delivered in CSV format.

The list of sorting points contains the places items can be delivered.

If ILS cannot supply an exhaustive list of all possible sorting points, a portion can be supplied or none at all. If ILS subsequently notifies IMMS of sorting of an item at a sorting point that has not been transferred from ILS in advance, IMMS will create the sorting point itself.

6.1.10 Get all chutes

Returns a list of all the chutes used for sorting points.

File name: `Chute.csv`

Input:

- (none)

Output:

- All chutes (see section 7.9 for data list with fields). Result delivered in CSV format.

If ILS cannot supply an exhaustive list of all possible chutes, a portion can be supplied or none at all. If ILS subsequently notifies IMMS of sorting of an item to a chute that has not been transferred from ILS in advance, IMMS will create the chute itself.

6.1.11 Get all titles

Returns a list of all titles (bibliographic records) in ILS.
Refer to section 4.1 for scope of data.

File name: `BibliographicRecord.csv`

Input:

- (none)

Output:

- All titles (see section 7.10 for data list with fields). Result delivered in CSV format.

6.1.12 Get all items

Returns a list of all current items in ILS.
Refer to section 4.1 for scope of data.

File name: `Item.csv`

Input:

- (none)

Output:

- All items (see section 7.11 for data list with fields). Result delivered in CSV format.

6.1.13 Get all requisitions

Returns a list of all current (non-taken) requisitionsin ILS.

File name: `Requisition.csv`

Input:

- (none)

Output:

- All requisitions (see section 7.12 for data list with fields). Result delivered in CSV format.

6.1.14 Get all taken requisitions

Returns a list of all current requisitions (including fulfilled but not checked out) in ILS.

File name: `TakenRequisition.csv`

Input:

- (none)

Output:

- All taken requisitions (see section 7.13 for data list with fields). Result delivered in CSV format.

6.2 SOAP web service operations called by ILS from IMMS

ILS provides these functions that IMMS can call.
The web service is defined in WSDL: **`Ils4Ims.wsdl`**

6.2.1 Receive notifications

Function called when IMMS wants to notify ILS of changes.

Input:

- A list of the following sub-types:
    - `ItemUpdatedNotification`: Item updated (new branch, department, IMMS status etc.)
    - `ItemTakenToRequisitionNotification`: Item is picked/taken for a requisition
    - `ItemDiscardedNotification`: Item discarded
    - `ItemReadyForPickupNotification`: Item placed on reserve shelf

Output:

- (none)

See data lists for content of the individual type notifications.

6.2.2 Initial data processed

IMMS calls this function to indicate that the initial data set has been loaded and that ILS can therefore begin to send change notifications.

Input:

- (none)

Output:

- (none)

6.2.3 Ping

Used to test the connection between the two systems.

Input:

- (none)

Output:

- (none)

6.3 SOAP web service operations call to IMMS from ILS

IMMS provides these functions that ILS can call.
The web service is defined in WSDL: **`Ims4Ils.wsdl`**

6.3.1 Receive notifications

Function called when ILS wants to notify IMMS of changes.

Input:

- A list of the following sub-types:
  - `FloatCodeCreatedOrUpdated`: Float code created/updated
  - `FloatCodeDeletedNotification`: Float code deleted
  - `BranchCreatedOrUpdatedNotification`: Branch created/updated
  - `BranchDeletedNotification`: Branch deleted
  - `DepartmentCreatedOrUpdatedNotification`: Department created/updated
  - `DepartmentDeletedNotification`: Department deleted
  - `LocationCreatedOrUpdatedNotification`: Location created/updated
  - `LocationDeletedNotification`: Location deleted
  - `SublocationCreatedOrUpdatedNotification`: Sub-location created/updated
  - `SublocationDeletedNotification`: Sub-location deleted
  - `CollectionCreatedOrUpdatedNotification`: Item collection created/updated
  - `CollectionDeletedNotification`: Item collection deleted
  - `DiscardReasonCreatedOrUpdatedNotification`: Discard reason created/updated
  - `DiscardReasonDeletedNotification`: Discard reason deleted
  - `SortingPointCreatedOrUpdatedNotification`: Sorting point created/updated
  - `SortingPointDeletedNotification`: Sorting point deleted
  - `ChuteCreatedOrUpdatedNotification`: Chute created/updated
  - `ChuteDeletedNotification`: Chute deleted
  - `BibliographicRecordCreatedOrUpdatedNotification`: Title created/updated
  - `BibliographicRecordDeletedNotification`: Title deleted
  - `ItemCreatedOrUpdatedNotification`: Item created/updated
  - `ItemDeletedNotification`: Item deleted
  - `RequisitionCreatedOrUpdatedNotification`: Requisition created/updated
  - `TakenRequisitionCreatedOrUpdatedNotification`: Taken requisition created/updated
  - `RequisitionDeletedNotification`: Requisition cancelled/deleted
  - `ItemCheckedOutNotification`: Item checked out
  - `ItemDiscardedNotification`: Item discarded
  - `ItemSortedNotification`: Item returned and sorted to chute
  - `OrderCreatedNotification`: Order created
  - `BibliographicRecordIdChangedNotification`: FAUST number changed
  - `ItemIdChangedNotification`: Item ID changed
  - `OrderDeletedNotification`: Order deleted

Output:

- (none)

See data lists for content of the individual notification types. Delete notifications only contain identification of the record.

6.3.2 Assignment of item to branch

Used when ILS wants IMMS to assign a branch for an item.

Input:

- Branch code (where the item is now)
- Item ID

Output:

- Branch code for proposed branch
- Department code for proposed department

6.3.3 Initial data ready

ILS calls this function to indicate that an initial data set has been generated and is ready to be retrieved by IMMS. ILS will then withhold change notifications until IMMS reports that the initial data has been loaded (see section 6.2.2).

6.3.4 Ping

Used to test the connection between the two systems.

Input:

- (none)

Output:

- (none)

7 DATA LIST FROM ILS TO IMMS

This chapter states for each record type (in an initial data set or notification) which fields a record contains.

The underlined fields should be regarded as the record's unique and permanent identification.

Data types:

Name	**Description**
String [a..b]	Text string with minimum length *a* and maximum length *b*.
Date	A date. yyyymmdd (year, month, day) in CSV. A `date` in XML.
Timestamp	Date and time. CSV uses yyyymmddhhmmss (year, month, day, hours, minutes, seconds) in UTC. XML uses `dateTime` (remember correct designation of time zone). Whole second precision is sufficient, but millisecond registrations are also accepted.
Boolean	A Boolean value, true or false. CSV uses the value "false" and "true". XML uses `boolean`.

Notes on CSV:

- The character set used is ISO-8859-15.
- A semicolon ; (0x3B) is used to separate fields.
- Escaping of strings with double quotes " (0x22) as in RFC 4180, although with semicolon as field separator instead of comma. Escaping of all fields is permitted.
- Lines are terminated with CR+LF (0x0D + 0x0A) or LF (0x0A).
- No header line.

7.1 Float codes

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	**Field name**	**Data type**	**Remarks**
1	<u>FloatCode</u>	String [1..20]	Identification of the float group
2	DisplayName	String [0..100]	Name to be displayed
3	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

7.2 Branches

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	**Field name**	**Data type**	**Remarks**
1	<u>BranchCode</u>	String [1..20]	Identification of the branch

2	DisplayName	String [0..100]	Name to be displayed
3	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

7.3 Departments

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	DepartmentCode	String [1..20]	Identification of the department
2	DisplayName	String [0..100]	Name to be displayed
3	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

7.4 Locations

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	LocationCode	String [1..20]	Identification of location
2	DisplayName	String [0..100]	Name to be displayed
3	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

7.5 Sub-locations

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	SublocationCode	String [1..20]	Identification of the sub-location
2	DisplayName	String [0..100]	Name to be displayed
3	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

7.6 Item collections

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	CollectionCode	String [1..20]	Identification of the item collection
2	DisplayName	String [0..100]	Name to be displayed
3	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

7.7 Reasons for discarding

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	DiscardReasonCode	String [1..20]	Identification of the reason for discard
2	DisplayName	String [0..100]	Name to be displayed
3	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

7.8 Sorting points

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	BranchCode	String [1..20]	The branch to which the sorting point belongs
2	SortingPointCode	String [0..20]	Identification of the sorting point
3	DisplayName	String [0..100]	Sorting point name to be displayed
4	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

Sorting points are where items are sorted into one of a number of transport units. Examples include self-service machines, sorting robots, desks and offices.

A sorting point always belongs to a specific branch. The same code can be used for sorting points at different branches, as identification of sorting point is a combination of the branch code and the sorting point code.

7.9 Chute

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	BranchCode	String [1..20]	The branch to which the chute belongs
2	SortingPointCode	String [0..20]	The sorting point to which the chute belongs
3	ChuteCode	String [0..20]	Identification of the chute
4	DisplayName	String [0..100]	Chute name to be displayed
5	ShortName	String [0..20]	Short name to be displayed. If omitted, the identification will be shown

Chutes are where an item ends up when it passes a sorting point.

A chute always belongs to a specific sorting point. The same code can be used for chutes at different sorting points, as identification of chute is a combination of the branch code, the sorting point code and the chute code.

7.10 Titles

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	Field name	Data type	Remarks
1	BibliographicRecordId	String [1..20]	FAUST number
2	Classification	String [0..100]	• For reference books, the field contains the DK5 group + any sub-division (e.g. danMARC2 field 652 *moahb) • For fiction the field contains the value "sk". • For music the field contains the 3 letter code for the music location (danMARC2 field 039 *ab) • For children's books the field contains the location code (danMARC2 field 038 *a) The above are expected and recommended, but IMMS accepts all values within the length constraint. A reduced character set is used.
3	Alphabetisation	String [1..1000]	States the alphabetisation used on the shelves. Reduced character set.
4	ItemTypeCode	String [0..20]	Code for the item type.
5	ItemTypeText	String [1..100]	Name of the item type.
6	ClassificationDisplay	String [0..100]	As for the "Classification" field, but with full character set – for display only.
7	MusicClassificationText	String [0..100]	Written description of the music location. danMARC2 field 039 *ab. Omitted if no music involved.
8	Author:	String [0..1000]	Author. Can be omitted.
9	Title	String [1..1000]	Title
11	PageCount	String [0..100]	danMARC2 field 300 *a. Description of pagination. Omitted for non-book items.
10	Edition	String [0..100]	danMARC2 field 250 Designates edition no. Can be omitted.
12	PhysicalDescription	String [0..100]	danMARC2 field 300 *nd Description of the item's scope. Can be omitted.
13	Series	String [0..100]	Designation of series.

			E.g. danMARC2 field 440 *anopv Can be omitted.
14	RecordLabelNumber	String [0..100]	danMARC2 field 538 *fg record label and number for musical recordings. Can be omitted.
15	FirstBibliographicRecordId	String [0..20]	danMARC2 field 017 *a. Can be omitted.
16	InitialCategory	String [0..100]	Code for the category the title should initially have in IMMS. Can be omitted.

NB: If the MARC record is enhanced with local data, e.g. an alternative DK5 group in field G52, ILS should take the value from the correct field.

7.11 Items

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	**Field name**	**Data type**	**Remarks**
1	<u>ItemId</u>	String [1..20]	Item ID
2	BibliographicRecordId	String [1..20]	FAUST number
3	StatusCode	Indicates one of the following values: • NotCheckedOut: • CheckedOut • Discarded.	Code for the item's status. Changes in this status will be notified separately (sections 7.15, 7.16 and 7.17) and should <u>not</u> be notified as a change in the item. If the item is notified as changed for some other reason, the current status should be included in the notification.
4	FloatCode	String [0..20]	The float code. Can be blank/omitted.
5	FixedBranchCode	String [0..20]	Code for fixed branch
6	CurrentBranchCode	String [0..20]	Code for current branch.
7	FixedDepartmentCode	String [0..20]	Code for fixed department
8	CurrentDepartmentCode	String [0..20]	Code for current department.
9	FixedLocationCode	String [0..20]	Code for fixed location
10	CurrentLocationCode	String [0..20]	Code for current location.
11	FixedSublocationCode	String [0..20]	Code for fixed sub-location
12	CurrentSublocationCode	String [0..20]	Code for current sub-location.
13	FixedCollectionCode	String [0..20]	Code for fixed item collection
14	CurrentCollectionCode	String [0..20]	Code for current item collection.
15	AccessionDate	Date	Accession date. Can be omitted.
16	DiscardReasonCode	String [0..20]	Code for discard reason if the item is discarded, otherwise blank/omitted
17	PeriodicalYear	String [0..20]	Year (for periodical)
18	PeriodicalNumber	String [0..20]	Number/issue (for periodical).
19	PeriodicalVolume	String [0..20]	Volume (for periodical)
20	InterLibrary	Boolean	If the item has been borrowed from another library or authority.

NB: "Current" means the value valid at this time in ILS.

7.12 Requisitions

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	**Field name**	**Data type**	**Remarks**
1	RequisitionId	String [1..20]	Identifies the requisition
2	ItemId	String [1..20] Can be omitted	Item ID. **See remark below!**
3	PickBranchCode	String [0..20] Can be omitted	Code for the branch where the item should be picked. If IMMS should decide the pick branch, it is omitted.
4	PickupBranchCode	String [0..20]	Code for the branch the patron wants to pick up the item from.
5	WebOrder	Boolean. Can be omitted.	If the requisition has been made by the patron online. If ILS does not have this information, it is omitted.
6	RequisitionTime	Timestamp	Time at which the patron created the requisition
7	RequisitionTypeCode	String [0..20]	Code for requisition type
8	RequisitionTypeText	String [0..100]	Name of requisition type
9	SpecialHandling	Boolean	Tag to indicate whether the item should be specially handled.
10	Note	String [0..1000]	Omitted if not relevant
11	Active	Boolean Can be omitted.	Specifies whether the requisition should be picked or not. If omitted, the requisition is regarded as active.

NB:

There is one line for each item ID the requisition contains in CSV format (the other fields are repeated).
There is one order with an embedded list of item IDs in XML format (one item ID element for each item).
If a requisition is not active, it is allowed not to specify any items. In CSV format the field is left blank. In XML format the list of items is empty.

7.13 Taken requisitions

This is data kept synchronized between the two systems. The field list covers both initial data and notifications in the event of create/change.

Field no.	**Field name**	**Data type**	**Remarks**
1	RequisitionId	String [1..20]	Identifies the requisition
2	ItemId	String [1..20]	The item taken

3	PickupBranchCode	String [0..20]	Code for the branch the patron wants to pick up the item from.
4	WebOrder	Boolean. Can be omitted.	If the requisition has been made by the patron online. If ILS does not have this information, it is omitted.
5	RequisitionTime	Timestamp	Time at which the patron created the requisition
6	RequisitionTypeCode	String [0..20]	Code for requisition type
7	RequisitionTypeText	String [0..100]	Name of requisition type
8	SpecialHandling	Boolean	Tag to indicate whether the item should be specially handled.
9	Note	String [0..1000]	Omitted if not relevant
10	Fulfilled	Boolean Can be omitted.	States whether the requisition is fulfilled, i.e. ready for collection. If omitted, regarded as "False"

If the requisition was previously transferred to IMMS (as non-taken), the same RequisitionID is used, but now as taken requisition.

7.14 Requisition cancelled/deleted

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

Field no.	**Field name**	**Data type**	**Remarks**
1	EventTime	Timestamp	Time of event
2	RequisitionId	String [1..20]	Identifies the requisition
3	CancelReason	String [0..100]	Reason for cancellation (collection date exceeded, patron changed his/her mind, item picked up by patron, ...). Omitted if unknown

7.15 Item checked out

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

Field no.	**Field name**	**Data type**	**Remarks**
1	EventTime	Timestamp	Time of event
2	ItemId	String [1..20]	Identifies the item
3	RequisitionId	String [1..20]	Omitted if not checked out on the basis of a requisition
4	CheckoutBranchCode	String [0..20]	Checkout branch

7.16 Item discarded

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

Field	**Field name**	**Data type**	**Remarks**

no.			
1	EventTime	Timestamp	Time of event
2	ItemId	String [1..20]	Identifies the item
3	DiscardReasonCode	String [1..20]	Reason for discarding code

7.17 Item returned and sorted to chute

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	ItemId	String [1..20]	Identifies the item
3	BranchCode	String [0..20]	The branch at which the item was returned/sorted
4	SortingPointCode	String [0..20]	Indicates which self-service machine, sorting robot or office/desk is involved
5	ChuteCode	String [0..20]	Identification of chute. If there is no information on a specific chute an item has been sorted to, a bin number can be given instead.

7.18 Order created/updated

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

These orders are transferred from ILS as they are created in ILS. Some (e.g. auto-discard) are created by ILS itself, whilst others (e.g. item care) are generated explicitly by a staff member.
Each line in ILS' traditional lists represents a separate order. Orders can be linked in a list using a list name.

The following types of orders are supported

- Discard
- Item care
- Ad hoc

"Ad hoc" is included to cover future needs. Which search results/lists in ILS can be sent to IMMS as ad hoc orders is not specified in this document.

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	OrderType	String [1..20]	One of the following: "Discard", "ItemCare", "AdHoc"
3	OrderId	String [1..20]	Unique identification of an order. Can be omitted.
4	ItemId	String [1..20]	Identifies the item
5	ArchiveOnDiscard	Boolean	Whether the item should be delivered to the National Library. Only relevant for discard orders
6	DiscardReasonCode	String [1..20]	Reason for discarding code. Omitted if not a discard order or the

			reason is unknown.
7	ListName	String [0..100]	The name of the list the order is part of. Omitted if not relevant
8	Note	String [0..1000]	Omitted if not relevant

Orders can be updated by sending a new notification with the same OrderId.
OrderId can be omitted, but then the order cannot subsequently be updated/deleted.

7.19 FAUST number changed

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	OldBibliographicRecordId	String [1..20]	The title's previous FAUST number
3	NewBibliographicRecordId	String [1..20]	The title's new FAUST number

7.20 Item ID changed

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	OldItemId	String [1..20]	The item's previous item ID
3	NewItemId	String [1..20]	The item's new item ID

7.21 Order deleted

This is an event and is therefore not part of the initial data set. The field list is used for notifications.

This notification indicates that a previously transferred order should be deleted.

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	OrderId	String [1..20]	Identification of the order

8 DATA LIST FROM IMMS TO ILS

8.1 Item updated

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	ItemId	String [1..20]	Identifies the item
3	BranchCode	String [0..20]	The branch at which the item is now placed.
4	DepartmentCode	String [0..20]	The department in which the item is now placed. Omitted if the item does not have a float code.
5	PlacementText	String [0..1000]	Placement text. Textual description of the item's placement within the branch. If the item is in its designated place, this field is blank.
6	ImsStatusCode	String [1..20]	IMMS status code
7	ImsStatusText	String [1..100]	IMMS status text e.g. "Ready for set up"
8	Available	Boolean	If the item is "available" for picking

8.2 Item taken for a requisition

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	ItemId	String [1..20]	Identifies the taken item
3	RequisitionId	String [1..20]	Identifies the requisition

8.3 Item discarded

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	ItemId	String [1..20]	Identifies the item
3	DiscardReasonCode	String [1..20]	Reason for discarding code
4	NotFound	Boolean	If discard is a result of "not found" or other loss, which may not override the fact that the item is registered as being checked out in ILS.

8.4 Item placed on reserve shelf

Depending on whether a reservation note has previously been printed, ILS can opt to send the placement text on to the patron.

Field no.	Field name	Data type	Remarks
1	EventTime	Timestamp	Time of event
2	ItemId	String [1..20]	Identifies the item
3	RequisitionId	String [1..20]	Identifies the requisition
4	BranchCode	String [0..20]	Code for the branch the item is now placed at.
5	DepartmentCode	String [0..20]	Code for the department the item is now placed at. Omitted if the item does not have a float code.
6	PlacementText	String [0..1000]	Placement text. Textual description of the item's placement within the branch.