# LogiSoft Host Link

Build 2446

## Table of Contents

# LogiSoft Host Link

This document describes how to integrate LogiSoft with an overlying Host System. This system will typically be an ERP system, but it may also be another warehouse management system (WMS).

The Host interface is called LogiSoft Link and is an application, used to provide an interface to external Host/ERP systems for on-line handling of orders and exchange of other information like article master data, inventorying requests, etc. Orders can be transferred from the Host system and LogiSoft will return an acknowledge back, when the order has been processed. LogiSoft Link might also return stock adjustments, when the actually number of articles on stock does not fit with the expected count or when an order could not be fulfilled completely. In general, the purpose is, that LogiSoft and the host system can be used as one homogeneous system.

Some issues that must be clarified to make a successfully integration are:

| Requirement | Transaction Type | Notes |
|---|---|---|
| To pick by orders from the Host | Order Picking | |
| Only store articles manually | Master Data. | If put-away is done manually without orders from a Host system, LogiSoft should as a minimum know the article numbers and article description in advance. It is possible to create new articles directly by entering the number, but dangerous if the operator makes a mistake. |
| Store articles by orders from the Host | Order Put-Away / Master Data | |
| Picking confirmation to the host | Order Confirm | This will confirm the pick is done and if there was any deviations from the requested quantity |
| Put-Away confirmation to the host | Order Confirm | This will confirm the store is done and if there was any deviations from the requested quantity |
| Message to the host when the stock is changed without a Host order | Order Confirm | When the stock is adjusted due to a manual pick, store or adjustment, LogiSoft generates an internal order to track this, the order can be confirmed back to the Host. |
| Is order cancel needed? | Order Cancel | If an order is not yet started, fit can be cancelled. |
| Will orders be modified after they are sent? | Order Pick / Put-Away | This is not recommended, but possible with some limitation. Orders are expected to be send in Ã³ne file, it is not possible to append to an existing order, so if the order is modified, resend the whole order again. The reason for this limitation, is that errors are easily generated if appending to an order was allowed. For example if the same file for some reason is read twice, this would result in duplicated order lines. |
| Will the Host generate stock counting requests | Stock Count | |
| Is stock reports needed? | Stock Report | |

| Requirement | Transaction Type | Notes |
|---|---|---|
| Is serial numbers used? | | |
| Is batch numbers used? | | |

The native format, for the interchange of data between the Host system and LogiSoft is XML files, alternatively flat ASCII files (CSV Files). A folder (directory) accessible on the network for both systems is used for data exchange. If the native format is used, this format must be implemented by the host system. XML files is the format with the most extended support.

Other alternatives are Web Services, direct Database Access, SAP and Message Queues. They will typical require some modifications.

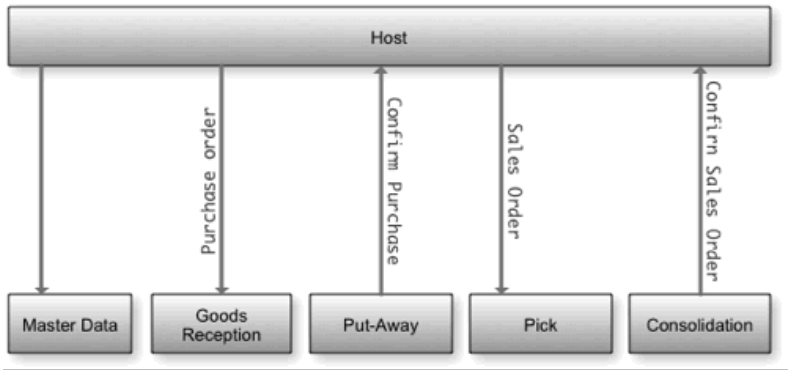
LogiSoft Link supports a number of different type of information, that can be exchanged between the two systems, the most common are:

**Article Master** Data. LogiSoft has a local database for articles. This database can either be maintained manually or by receiving the information from the ERP system. The New Article format creates a new entry or updates the information for an existing article and thereby free the user for the double work of maintaining two databases.

**Order Request**. By sending an Order Request, the ERP system requests the specified items to be picked from the storage. There are different type of orders, like picking orders and put-away orders.

**Order Acknowledge**. When an order has been picked the LogiSoft software acknowledge the order by an Order Acknowledge telegram. The purpose is both to indicate that the order processing has been completed, but also to adjust the order in case it could not be completely fulfilled, for example if not all the requested articles are available on the stock.

**Inventorying control**. LogiSoft can send stock adjustments to the host and the host can request stock counting for a specific article.




There are 2 basic types of orders:

- Picking orders.
- Put-away orders.

But there can be many sub order types. A unique order is identified by:

- The Order Number (50 characters).
- The Delivery Number.
- The Order Type ID, a number from 1 - 65000.

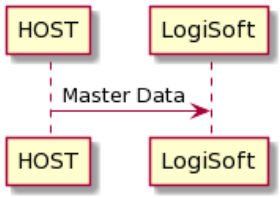
# General Rules

## Numbers

Numbers are typical quantities. Both whole numbers and signed numbers are allowed. The decimal separator is always US style “.” . In most cases the sign is positive and follows the in/outbound direction of the order, so for an inbound order the quantity is positive and the same for an outbound order. The exception is for stock adjustments confirmation where the adjustment quantity is positive if the stock increase and negative if the stock is decreased.

# Master Data

Master data is basic information about the articles. The most important information is the article number and description, but other information can be transferred as well. LogiSoft will create new articles and update the description, when met in an imported order, but the Master Data allows more information and will also work if LogiSoft is used with the Manual Transactions (transactions with no host-orders), this could be the case in an initial store process, where goods are moved from old locations into LogiSoft without orders.




*Master data is send by the host, there is no response back from LogiSoft*

# Inbound Orders

The Host may send an order when inbound goods are expected. When the goods are received, it will be matched against the order. Typical reasons to receive goods are:

- Purchase orders.
- Returned goods from customers.
- Produced items from own production.
- Replenishment from other storage locations.

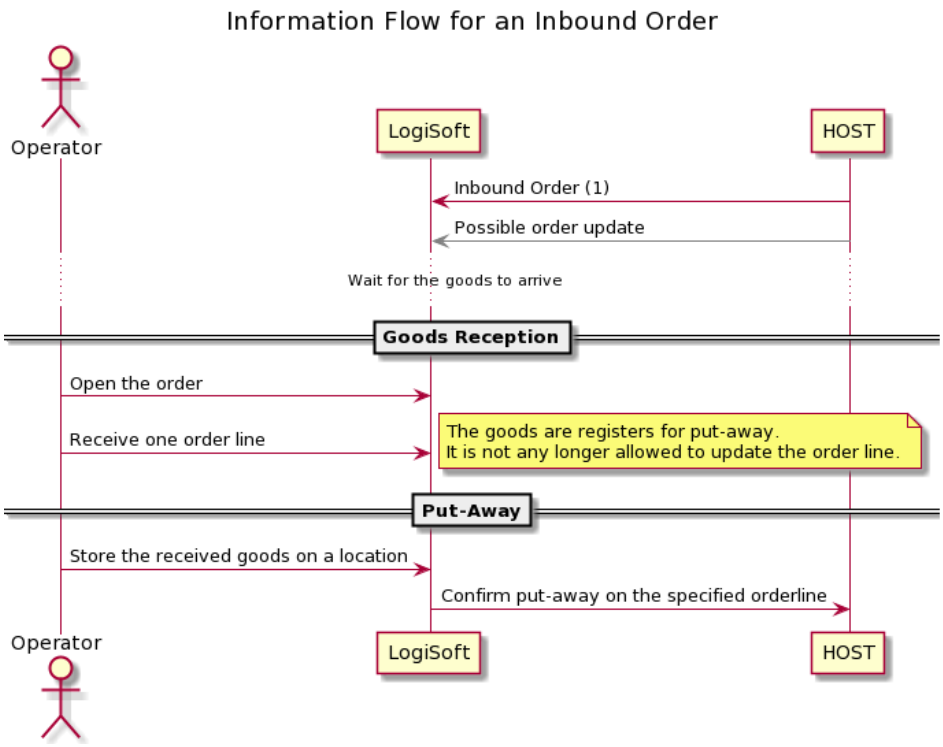
When the goods are stored on the final location, it is reported back to the host. To tell the host that the goods are ready to be picked and an invoice for the goods can be accepted. This is typical done per stored transaction.

It is recommended that the file is created with a temporary extension and renamed when the file is ready. This to prevent the file to be read, before it is finalized by the host.

The file is moved to a temporary folder. This to ensure that nobody else uses the file and if the "Import" folder is on an external drive, to speed up the processing speed.

LogiSoft Link process the file from the temporary folder.

If the processing was successful the file is moved to the log folder else the file will be moved to the "Error" folder. ng of incoming goods can be received in several deliveries and the put-away process may take a while.




*Basic flow of goods reception with acknowledgement back to the host for each received line*

(1) The Host might send the inbound order as soon as possible (when it is created) or delayed until it is needed. It is also possible to make the actually registration in the Host System and then let the Host send an in-bound order with the actually received goods. This might end in a two step process, but can have other advantages.

## Partly Deliveries

Inbound orders are often partly delivered, this means that the expected quantity to be received must be registered at several times. To make the Host aware of the received goods, it can be practical to confirm *by transactions*, this means that the same order line can be confirmed several times.

## Updating Purchase (put-away) Orders

It can be required to modify an already sent order. This is possible with the limitation, that when the first goods are registers on one order line, it can no longer be changed. When changing an order the whole order is send again.

## Special Requirements

Some special requirements can optionally be supported like:

- Reservation of articles for a specific picking order.
- Inspection of incoming goods.
- Registration of batch/lot and serial numbers.

### Reserved Purchases

A feature that support purchases reserved for a specific sales order.

### Incoming Inspection

Used to flag purchases that must be inspected before they are released for picking.

### Batch Number Registration

The batch number of incoming goods must be registered. The batch number can be reported back to the Host.

### Serial Number Registration

Serial numbers of incoming articles must be registered. The serial number can be reported back to the Host.

# Outbound Orders (Picking)

The Host sends orders to request goods to be picked. This can for example be sales orders to customers or production orders.

Confirmation back to the host can be:

- When the order is finalized and confirmed.
- When the order is picked completely (no confirmation).
- Line by line.

## Import Rules for Orders

The following table shows the configuration parameters and how LogiSoft reacts in various situations when importing picking orders.

| Parameter | Value | Note | |
|---|---|---|---|
| AutoCreateItem | True | False | If true, unknown articles will be created when needed. |
| CancelOrderLinesWithoutStock | True | False | If true the stock must be present else the orderline will be deleted. |
| ItemNumberUnique | True | False | The same article number must only appear once per order. |
| ConfirmPickPerLine | True | False | Acknowledge generated immediately for each pick line. |
| ConfirmPutPerLine | True | False | Acknowledge is generated immediately for each put-away line. |
| LinkDeleteZeroLines | True | False | If a line is received with a 0 count, it will be deleted. Used to cancel lines existing orders. |
| ReleaseOrderMode | Auto | Manually | Orders must be released before they can be picked. Auto mode will release the order immediately when received. |
| ReplaceOrders | True | False | If the same order is receved 2 or more times, the order will replace the previous one. If false the lines will be appended. |

## Updating Pick Orders

Some times it might be required to update an existing pick order. This should be avoided if possible, but if not, here is the rules and behavior when it happens.

To start, it is important to understand the life-cycle of an order.

When an order is resend, with the same order number/delivery note number, it will replace the existing order and delete the previous order!

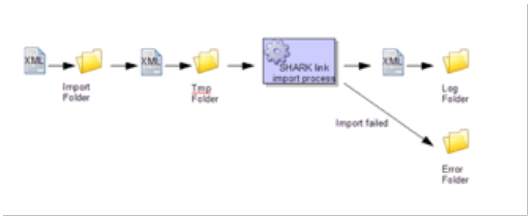
| Order State | Result |
|---|---|
| Not released/not started | The old order will be deleted and a new created. |
| Not released/partly picked | The old order will be deleted and a new created. |
| Released | The order update will fail and rejected. Recall the order first. |
| OK (finished) | The old order will be deleted and a new created. |

# File Exchange

The standard way of connecting a Host System and LogiSoft is by means of file exchange. The exchange of files between the host and LogiSoft is done by shared folders or by FTP. Other strategies may be used as well.

## Importing files

The folder used for transaction exchange can freely be selected on the local network. It is defined in the LogiSoft registry. The import strategy is:




*The folders used for importing files.*

The definition of the import folders are done from a LogiSoft client, in the registry setup, an example is shown below:

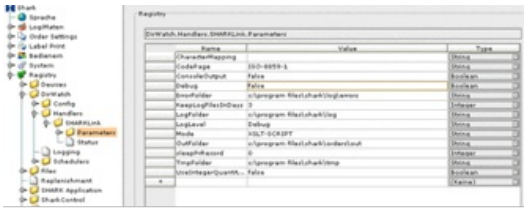


*Figure 1 - The registry setup in LogiSoft defining folders used for file import.*

## Exporting Files

Acknowledge information back to the host is typical done by exporting files.

## Cleaning up the Log Folders

All imported files will as default be saved in the "Log" folder. This folder will after a short period be filled up with files if nothing was done. To avoid this LogiSoft Link includes a folder cleaning function that automatically delete the files when they are older than a specified number of dates. For larger system 3 days can be appropriated, for smaller installation 30 days can be OK. The nice thing about the log folder is, that files can be reimported in case something goes wrong and it is easy to pin-point a problem because the original data can easily be found.

The error folder is not automatically cleaned.

## Files Names

To some extend, it is possible to customize the file names for written files. This is done using placeholders in the export file definition that will be replaced by the actually value, when the file is written.

| Placeholder | Description |
|---|---|
| %DocType% | Name of the document as defined in LogiSoft |
| %OrderNumber% | Order number from LogiSoft |
| %Index% | Document index number. This is a running unique number. |
| %Date% | Date when the file is written. The format is YYYYMMDD, for example is the august 2th 2012, written as 20120802 |
| %Time% | Time as HHMMSS, 17:25 and 15 seconds is written as 172515. |

# The XML File Interface

XML is the native format of LogiSoft and is the recommended way of implementing an interface to a host.

XML provides a standard way of representing hierarchical, structured data. XML make use of mark-up tags, similar to HTML, indicating how data should be interpreted by the application.

Support transactions:

- Master Data
- Orders
- Cancel an Order
- Order Acknowledge
- Inventory Count Request

For configuration of the importer, look here.

All the XML files must have the extension .xml

General comments:

- The element names are case sensitive.
- New lines and indentation is only used to make the XML readable, they are not required and can be used freely.
- XML files can be displayed in a Web Browser (nicely formatted, for example in Internet Explorer).
- The formats can be combined, so article information can for example be sent in the same XML file as the order and orders and put-away requests can be combined in the same file. This makes the format very flexible.
- In general it is recommended to use UTF-8 as encoding of the XML files.Â  The XML header should look like:
  <?xml version="1.0" encoding='UTF-8'?>
- Unrecognised fields will be ignored. This means it is possible to add fields for future use.

Please note that some characters have a special meaning in XML:

| Character | Can be encoded asÂ â€¦ |
|---|---|
| & | & |
| < | < |
| > | > |
| " | " |
| ' | ' |

Character references provide a way to insert Unicode characters that are identified by a number pointing to a Unicode code point. Code points can be identified using either decimal or hexadecimal notation.

&#value; Syntax used for decimal references.

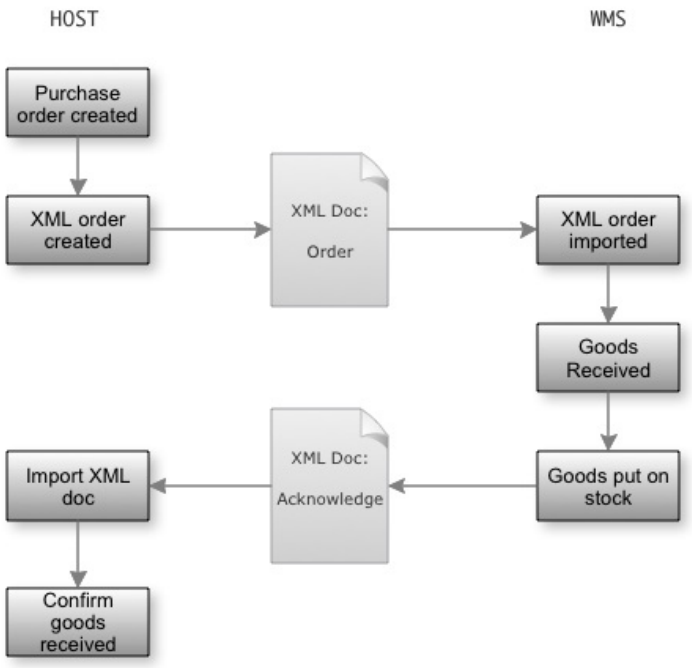
&#xvalue; Syntax used for hexadecimal references.

# Work Processes

Description on the required communication to the Host for various typical work flows.

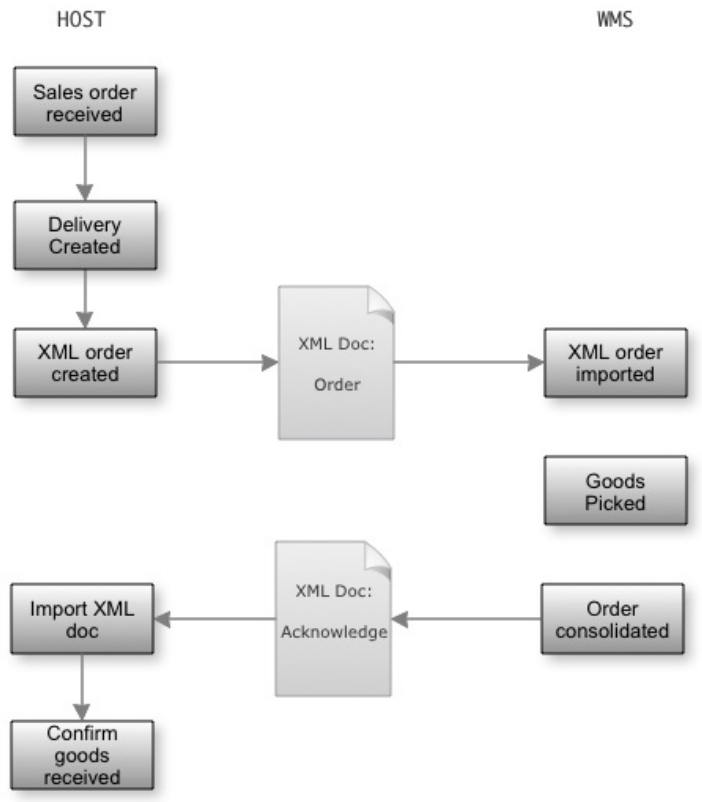
## Put-away Orders

Simple flow when goods have to be stored. Typical this is the result of a purchase order.




## Picking Orders

Flow of goods picked by orders. Typical as a result of a sales or production order.




Â ## XML Formats

In the following, the XML formats for the supported transaction types, will be described.

## General File Format

All XML files generated by LogiSoft have a fixed â€œDocumentâ€? header that defines an envelope for the document. The header includes a unique document number, the time when it was generated and type of document. The information can be ignored by the host, but can be used to track the document in LogiSoft.

```
 <?xml version="1.0" ?>
<LogiSoft-ERP>
 <Document>
   <Type>ACKNOWLEDGE</Type>
   <Date>2011-02-17 21:58.08</Date>
   <Sender>LogiSoftLINK</Sender>
   <Receiver>HOST</Receiver>
   <DocNumber>20</DocNumber>
 </Document>
  â€¦ content
</LogiSoft-ERP>
```

### Segment Document

| Element | Type | Description |
|---|---|---|
| *DocNumber* | ^String | The Document Number is a unique id, which identifies the document. Same as DocumentID in the database Document table. |
| *Type* | ^String | Type of this document. |
| *Sender* | ^String | Who that generated the document. |
| *Receiver* | ^String | The expected receiver of this document |
| *Date* | ^String | Date when the document was generated. |

# Article Master Data

File creating new articles or modifying existing ones, in the LogiSoft database. Sent by the Host system to the LogiSoft system.

It is recommended to have a clear strategy how the data is maintained. If the Master Data is maintained both in LogiSoft and from an external system using the XML features, there is a potential risk of inconsistent data.

For most system very little information is needed, only article number and description are required information, the rest to used for supporting various features. Most of the information can also be maintained in the LogiSoft system itself and there is no need to send the information unless it already is provided by the host system.

Rules:

- If the article already exists, the information will be updated with the new content.
- File extension must be .xml.
- Filename is freely selected.
- No response (acknowledge) is generated by this transaction.
- If a box type is specified, this will be the default box type for that article, until a another box type is specified.
- If a storage strategy is specified (by <LocationType>), this will be the strategy used for future articles of that type.
- If new Package Sizes, Replenishemt Information and Location Types are specified, old information will as default be deleted, unless the attribute Clear=â€™falseâ€™ atttribute is specified on the Packages tag.

Example 1 â€“ simple master data file:

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
 <Article>
  <ArticleNumber>Article number</ArticleNumber>
  <Description>Article description</Description>
</Article>
</LogiSoft-ERP>
```

Text in italics is optional.

A more complex example of creating master data for an article ART001 that should be stored in Zone â€œAâ€? on the location type â€œSmall Boxâ€?, in each of the locations of type â€œSmall Boxâ€?, 200 pieces will fit.

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
 <Article>
  <ArticleNumber>ART001</ArticleNumber>
  <Description>Article 1</Description>
  <Packages>
   <Package>
    <Replenishments>
     <Replenishment>
      <Zone>A</Zone>
     </Replenishment>
```

```
      </Replenishments>
      <LocationTypes>
        <LocationType>
          <Name>Small Box</Name>
          <Qty>200</Qty>
        </LocationType>
      </LocationTypes>
    </Package>
  </Packages>
 </Article>
</LogiSoft-ERP>
```

Â Example with most of the possible information added:

```
 <?xml version='1.0' encoding='UTF-8'?>
<LogiSoft-ERP>
 <Article>
  <ArticleNumber>Article3</ArticleNumber>
  <Description>Article 3</Description>
Â  <ABC>B</ABC>
  <Packages>
   <Package>
    <Name>Pcs</Name>
    <Height>100</Height>
    <Width>200</Width>
    <Depth>300</Depth>
    <PackageSize>1</PackageSize>
    <Replenishments>
     <Replenishment>
      <Zone>A</Zone>
      <LocationStrategy>Floating</LocationStrategy>
      <MinQty>10</MinQty>
      <MaxQty>450</MaxQty>
      <ReplenishmentQty>150</ReplenishmentQty>
      <ReplenishmentFrom>B</ReplenishmentFrom>
     </Replenishment>
    </Replenishments>
    <LocationTypes>
     <LocationType>
      <Name>UK2</Name>
      <Qty>150</Qty>
     </LocationType>
    </LocationTypes>
   </Package>
  </Packages>
 </Article>
</LogiSoft-ERP>
```

## Segment Article

Basic information about the article.

| Element | Type | Description |
|---|---|---|

| Element | Type | Description | |
|---|---|---|---|
| ABC | String | Set the picking rate class for the this article. Default names are ‘A’, ‘B’ or ‘C’, but this can be changed in the configuration. The name must be an existing on | |
| ArticleNumber | String | Unique article number defining the item, despite the name, it is a string that can contain both numbers and letters. | |
| ArticleGroup | String | Optional article group/class value. | |
| Description | String | Name of the article | |
| Description1 | String | 1. extra description of the article | |
| Description2 | String | 2. extra description of the article | |
| AlternativeArticleNumber1 | String | An alternative to the article number. Visibility in the Article Browser is optional. | |
| AlternativeArticleNumber2 | String | An alternative to the article number. Not visible in the Article Browser. | |
| EAN | String | EAN number | |
| ERPCount | Number | The Host systems stock quantity, used to compare LogiSoft stock with the Host stock. | |
| LocationStrategy | String | FIFO | Fixed Floating |
| Misc1 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc2 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc3 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc4 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc5 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc6 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc7 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc8 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc9 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| Misc10 | String | Optional field for customized information. The first 5 are visible in the Article Browser. | |
| MarkForDeletion | True/False | If true the article will be deleted if possible else it will marked for later deletion. It cannot be used in new orders or put-away operations. The article cannot be deleted immediately, because it might have database references. | |
| BatchNumberRequired | True/False | If true a batch number must be registered when picked or stored. | |
| SerialNumberRequired | True/False | False: Never (0), True:Always(1) | |
| SerialNumberRequiredInbound | True | Inbound(2) | |
| SerialNumberRequiredOutbound | True | Outbound(3) | |

| Element | Type | Description |
|---|---|---|
| SerialNumberRequiredTracked | True | Tracked(4), (not released yet, from build 2428) |

## Segment Article/Packages/Package

This Segment is OPTIONAL.

This segment defines one or more package size(s) for the article.

The Package tags must be specified, even only one package size is used for the article.

| Element | Type | Description |
|---|---|---|
| Name | String | Name of the package, for example “PCS”. “Stuck” or “Pallet”. Default is “P1” |
| Height | Float | Height in mm. |
| Width | Float | Width in mm. |
| Depth | Float | Depth in mm. |
| PackageSize | Float | Number of items in basic unit (pieces) for this package size. Default is “1”. |

*Only text in bold are required.*

## Segment Article/Packages/Package /LocationTypes/LocationType

This Segment is OPTIONAL.

This segment defines a location type where the article should be stored.

The Location Type must exists as locations before this can be specifiied.

| Element | Type | Description |
|---|---|---|
| Name | string | Name of the location type |
| Qty | float | Quantity to store on this location type |

## Segment Article/Packages/Package /Replenishments/Replenishment

This Segment is OPTIONAL.

This segment defines where the article should be stored and from where the zone should be replenished. More than one Replenishment segment can be defined.

| Element | Type | Description | | |
|---|---|---|---|---|
| Zone | string | Zone name where the articles can be stored. The zone must exists with locations. | | |
| ReplenishmentFrom | string | Name of zone to replenish from. | | |
| MinQty | float | Reorder level of automatic replenishment (if supported) | | |
| MaxQty | float | Maximum quantity to store in the zone | | |
| ReplenishmentQty | float | Quantity to use for replenishment orders | | |
| LocationStrategy | String | Fixed | Floating | FIFO (default is floating) |

*Only text in bold are required.*

Example: Defines that maximum 850 pieces will fit into the location type â€œSmall Boxâ€? of the article

â€œART001â€?.

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
 <Article>
  <ArticleNumber>ART001</ArticleNumber>
  <Packages>
Â    <Package>
     <LocationTypes>
      <LocationType>
       <Name>Small Box</Name>
       <Qty>850</Qty>
      </LocationType>
     </LocationTypes>
Â    </Package>
Â   </Packages>
 </Article>
</LogiSoft-ERP>
```

# Rename an Article Number

Rename an article number. All references will remain. The transaction log will show the old number. Orders will use the new numbers, even if they have been send with old numbers.

It is only allowed to change articler number in the same owner group.

| Element | Type | Description |
|---|---|---|
| ArticleNumberOld | String | This is the existing article number |
| ArticleNumberNew | String | This is the new order number |
| Owner | String | Optional owner field, if the warehouse has multiple owners. Skip this field if not used. |

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
  <ArticleRename>
    <ArticlerNumberOld>ART001</ArticleNumberOld>
    <ArticlerNumberNew>ART345</ArticleNumberNew>
  </ArticleRename>
</LogiSoft-ERP>
```

*This will rename article ART001 to ART345.*

# CreateÂ anÂ Order

The "Order" is generated by the Host system to create a new order. Orders can be both pick and put-away (goods in/goods out), the OrderTypeID defines the type and direction of the order.

Example:

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
 <Order>
  <OrderNumber>order number</OrderNumber>
  <OrderTypeID>order type id</OrderTypeID>
  <Owner>company name</Owner>
  <Priority>priority</Priority>
  <Customer>
   <Number>Customer Number</Number>
   <Name>Customer Name</Name>
  </Customer>
  <Address>
   <Street>Street</Street>
   <City>City</City>
   <ZipCode>zip code</ZipCode>
   <Country>Country</Country>
  </Address>
  <DeliveryAddress>
   <AddressLine1>Street</AddressLine1>
   <AddressLine2>Street line 2</AddressLine2>
   <AddressLine3>City</AddressLine3>
   <AddressLine4>zip code</AddressLine4>
   <AddressLine5>Country</AddressLine5>
  </DeliveryAddress>
  <DeliveryDate>date</DeliveryDate>
  <Reference>Reference</Reference>
  <Shipment>Name</Shipment>
  <OrderLine>
   <LineNumber>linenumber</LineNumber>
   <ArticleNumber>Article Number</ArticleNumber>
   <Qty>Number of items</Qty>
   <Note>Item note or description</Note>
   <BatchNumber>batch number</BatchNumber>
   <BatchDate>date</BatchDate>
   <BoxType>location type</BoxType>
   <Misc1>misc 1</Misc1>
  </OrderLine>
 </Order>
</LogiSoft-ERP>
```

Minimum example for aÂ LogiSoft WCS pick list where stock is controlled by the host system:

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
 <Order>
  <OrderNumber>1000</OrderNumber>
```

```
  <OrderTypeID>1</OrderTypeID>
  <OrderLine>
   <ArticleNumber>ART001</ArticleNumber>
   <Qty>5</Qty>
   <Location Zone='A' Module='1' Tray='5'>A1-5-01</Location>
  </OrderLine>
 </Order>
</LogiSoft-ERP>
```

This will generate a pick from the system with order number = "1000" of 1 line with article "ART001" from module 1,Â in zone A and tray number is 5. The displayed location adddress will be "A1-5-01". The host must here explicit specify the physical location by the attributes in the element, because LogiSoft is not necessarily able to interprete the locationÂ address.

## Segment Order

| Element | Type | Description |
|---|---|---|
| **OrderNumber** | string | The order number is a reference number, that in combination with the OrderType and the DeliveryNoteNumber defines a unique order. |
| **OrderTypeID** | int | OrderTypeID defines the type of the order. The default types are: 1=pick, 2=put-away. Other types can be defined as needed. |
| Owner | string | Optional owner (company) of this order. In case an owner is specified, all articles in the order must be owned by the specified owner. |
| DeliveryNoteNumber | string | A delivery number used when an order is delivered in more trunks. |
| Priority | string | Priority is a number between 0 and 255. Default is 127, 0 is lowest and 255 is express orders. |
| DeliveryDate | string | Date the order should be delivered. The default date format is yyyy-mm-dd. The date format can be modified. |
| Note | string | Free order note. |
| Shipment | string | Type of shipment. Information to the consolidation and shipment area and to an optional freight system. |
| Carrier | String | Name of the transport company that will pick up a pick order. |
| Tour | string | Orders can be picked to different tours. This parameter can be used to tell the system, that the order must be delivered with the specified tour. |
| Gate | string | Where the order will be picked up, when leaving the warehouse. |
| Misc1 | string | Miscellaneous field for customized functionality. |
| Misc2 | string | Miscellaneous field for customized functionality. |
| Misc3 | string | Miscellaneous field for customized functionality. |
| Misc4 | string | Miscellaneous field for customized functionality. |
| Misc5 | string | Miscellaneous field for customized functionality. |
| Misc6 | string | Miscellaneous field for customized functionality. |
| Misc7 | string | Miscellaneous field for customized functionality. |
| Misc8 | string | Miscellaneous field for customized functionality. |
| Misc9 | string | Miscellaneous field for customized functionality. |
| Misc10 | string | Miscellaneous field for customized functionality. |

*Text in bold is required*

The OrderTypeID defines the type of the order. Custom types can be created, but there is a number of standard

values as listed here:

| OrderTypeID | Description | Qty |
|---|---|---|
| 1 | Standard outbound order (Picking) | Qty is always positive |
| 2 | Standard inbound order (putaway) | Qty is always positive |
| 8 | Pick for internal replenishment | Qty is always positive |
| 9 | Store for internal replenishemnt | Qty is always positive |
| 10 | Relative adjustment. Used when the stock is adjusted directly | Qty reflects the stock change and is negative when the stock is decreased. |
| 11 | Absolut adjustment. Used when the stock is adjusted directly | |
| 15 | Picking without an order | Qty is negative |
| 16 | Store without an order | Qty is always positive |
| 17 | Setting the location to an absolut value | Qty reflects the stock change and is negative when the stock is decreased. |
| 18 | Stock counting relative adjustment | Qty reflects the stock change and is negative when the stock is decreased. |
| 19 | Qty is always positive | |

## Segment Order/Customer (optional)

| Element | Type | Description |
|---|---|---|
| Number | string | Customer number is a unique number for the customer. For purchase orders this is the supplier of the goods. |
| Name | string | The name of the customer |

## Segment Order/Address (optional)

Customer address information.

| Element | Type | Description |
|---|---|---|
| Street | string | Address (street) related to the Customer |
| City | string | Address (city) related to the Customer |
| ZipCode | string | Address (zip code) related to the Customer |
| Country | string | Address (country) related to the Customer |

## Segment Order/DeliveryAddress (optional)

Delivery address for the shipment.

| Element | Type | Description |
|---|---|---|
| AddressLine1 | string | Line 1 of the delivery address |
| AddressLine2 | String | Line 2 of the delivery address |
| AddressLine3 | String | Line 3 of the delivery address |
| AddressLine4 | String | Line 4 of the delivery address |
| AddressLine5 | String | Line 5 of the delivery address |
| AddressLine6 | String | Line 6 of the delivery address |

| Element | Type | Description |
|---|---|---|
| AddressLine7 | String | Line 7 of the delivery address |
| AddressLine8 | String | Line 8 of the delivery address |
| AddressLine9 | String | Line 9 of the delivery address |
| AddressLine10 | string | Line 10 of the delivery address |

## Segment Order/OrderLine

| Element | Type | Description |
|---|---|---|
| ArticleNumber | String | The article number is a unique identification of an article in combination with the Owner. |
| ArticleDescription | String | The article number is a unique identification of an article in combination with the Owner. |
| BatchNumber | String | Optional batch number for the articles. |
| BatchDate | String | Batch date is typical the latest date the article can be sold. Default date format: yyyy-mm-dd. |
| SerialNumber | String | Optional serial number for the article. If specified quantity must be 1. |
| LineNumber | String | An integer that will be used in the returned acknowledge and thereby creating a link between order and acknowledge lines.<br>If not specified, a running number starting from 1 for each order, will be used. |
| Location | String | Location address used to force picking from specific location or to select specific tray/module. The element has 3 optional attributes: Zone, Module, Tray. This element is only support by LogiSoft WCS. |
| Qty | Float | Article quantity. Default decimal separator is ".". |
| Unit | String | The article unit is the package size in LogiSoft. |
| Note | String | Optional note that will be displayed to the operator when the order line is picked. |
| QualityInspection | True/False | Optional flag used with the Quality Inspection feature. It will force inspection at goods reception. |
| TargetZone | String | Used for put-away orders to request the articles to go to a specific zone. |
| Reservation | String | Used for purchase orders, to reserve the article for a specific sales order. Use the sales order numeber as reservation number. |
| Misc1 | String | Miscellaneous field for customized functionality. |
| Misc2 | String | Miscellaneous field for customized functionality. |
| Misc3 | String | Miscellaneous field for customized functionality. |
| Misc4 | String | Miscellaneous field for customized functionality. |
| Misc5 | String | Miscellaneous field for customized functionality. |
| Misc6 | String | Miscellaneous field for customized functionality. |
| Misc7 | String | Miscellaneous field for customized functionality. |
| Misc8 | String | Miscellaneous field for customized functionality. |
| Misc9 | String | Miscellaneous field for customized functionality. |
| Misc10 | String | Miscellaneous field for customized functionality. |

*Text in bold are required elements.*

Segment for each line in the order. Repeat for each line. Minimum one line must be present, orders without any lines are not allowed. Only article number and quantity is mandatory.

### Reservations

Reservations are used for allocation a specific sales order to a purchase. Use the same reservation number on the inbound and outbound order.

How it works:

1. When the inbound (purchase) order line is received it will create a new article with the article number article/reservation. This ensures that it cannot be picked by other orders.
2. When the outbound (sales) order is received, it looks for the same new article, it it is not yet reserved, the order will be in a back order status.

Note that this reserved article will appear in the stock report as a special item. Also there is no automatic cleanup if the sales order is cancelled after the inbound order is received.

## Segment Order/OrderLine/SerialNumbers (optional)

Define an optionalÂ list of serial numbers, either to store or to pick. The serial number count must match the quantity for order line.

| Element | Type | Description |
|---|---|---|
| SerialNumber | string | Serial number. |

## Segment Order/OrderLine/PartList

| Element | Type | Description |
|---|---|---|
| PartListNumber | string | This is the name of the part list. |

The XML file importer supports creation of parts list from a master order. This feature allows a single order line to be broken down into multiple picks of different articles and thereby supporting a setup where the articles stored in the warehouse are sub components of a structure. The partlist is supplied with each order from the host and is not stored in LogiSoft, this means all maintenance of the part list is done in the Host system.

Each order line that contains a part list will create a separate sub-order. The sub order with the part list has to be picked before the master order and must be stored on a location first. This is usefull if picking the part list also involves some small production, like for example assembly of the parts or just packing of the product.

In the master order, the part list will be seen as an article with the name:

```
<master order number>.<part list number>
```

This means the picked part list order will be stored with a unique article number, to ensure it goes to the correct master order.

The Master Order will have a back order status, until all part lists in the order have been picked.

** Examples **

The example below includes only the required information for picking of two items for an order:

```
<? xml version="1.0" ?>
<LogiSoft-ERP>
  <Order>
    <OrderNumber>1234567</OrderNumber>
```

```
      <OrderTypeID>1</OrderTypeID>
        <OrderLine>
          <LineNumber>1</LineNumber>
          <ArticleNumber>000001</ArticleNumber>
          <Qty>100</Qty>
      </OrderLine>
      <OrderLine>
          <LineNumber>2</LineNumber>
          <ArticleNumber>000007</ArticleNumber>
          <Qty>200</Qty>
      </OrderLine>
   </Order>
</LogiSoft-ERP>
```

Example of a put-away order:

```
 <? xml version="1.0" ?>
<LogiSoft-ERP>
   <Order>
      <OrderNumber>1234567</OrderNumber>
      <OrderTypeID>2</OrderTypeID>
        <OrderLine>
          <LineNumber>1</LineNumber>
          <ArticleNumber>000001</ArticleNumber>
          <Qty>100</Qty>
      </OrderLine>
      <OrderLine>
          <LineNumber>2</LineNumber>
          <ArticleNumber>000007</ArticleNumber>
          <Qty>200</Qty>
      </OrderLine>
   </Order>
</LogiSoft-ERP>
```

## Segment Order/CustomFields

Custom fields are used for adding key/value pairs to the order which is not supported by the general order format.

| Element | Type | Description |
|---|---|---|
| Field | | Use Attribute Name and Value |

```
 <? xml version="1.0" ?>
<LogiSoft-ERP>
   <Order>
      <OrderNumber>1234567</OrderNumber>
      <CustomFields>
        <Field Name='Address.DropPoint.1' Value='Lyngsoe Alle 3A'\>
        <Field Name='Address.DropPoint.2' Value='Hoersholm'\>
        <Field Name='Address.DropPoint.3' Value='2970'\>
      </CustomFields>
      <OrderTypeID>2</OrderTypeID>
```

```
        <OrderLine>
          <LineNumber>1</LineNumber>
          <ArticleNumber>000001</ArticleNumber>
          <Qty>100</Qty>
        </OrderLine>
    </Order>
</LogiSoft-ERP>
```

# Order Cancel

Use this format to Cancel an existing order.

Note this may fail if cancel is not possible, for example if the order is already being processed.

Syntax:

Elements written in bold are required. The rest is optional.

```
 <?xml version=\"1.0\" ?>
<LogiSoft-ERP>
   <OrderCancel>
      <OrderNumber>order number</OrderNumber>
      <OrderTypeID></OrderTypeID>
  <DeliveryNoteNumber></DeliveryNoteNumber>
   </OrderCancel>
</LogiSoft-ERP>
```

## Segment Cancel

| **Element ** | **Type ** | **Description ** |
|---|---|---|
| OrderNumber | string | |
| OrderTypeID | int | OrderTypeID defines the type of the order. The default types are: 1=pick, 2=put-away. Other types can be defined as needed. |
| DeliveryNoteNumber | String | A delivery number used when an order is delivered in more trunks. |

# Order Acknowledge

The order acknowledge is generated by LogiSoft when processing of an order is finished.

Generation of acknowledge files are an optional feature, that can be enabled/disabled in the configuration setup. If the files are not used, the feature should be disabled, in order to conserve disk space.

Rules:

- For each line in the original order, a similar line is returned, but containing the actually picked number of articles.
- The default name of the order acknowledge file is: ACK-<documentnumber>.xml . This can be configured.
- The file is put in the subfolder Out under the shared exchange folder.
- The generated files are not deleted by LogiSoft Link, the Host System has the responsibility for this.
- â€˜Manual Transactionsâ€™ are acknowledged as one line orders.

Syntax:

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
  <Document>
    <Type>ACKNOWLEDGE</Type>
    <Date>2011-02-17 21:58.08</Date>
    <Sender>LogiSoftLINK</Sender>
    <Receiver>HOST</Receiver>
    <DocNumber>20</DocNumber>
  </Document>
  <Acknowledge>
    <Status>Status</Status>
    <OrderNumber>order number</OrderNumber>
    <OrderTypeID>200</OrderTypeID>
    <OrderLine>
    <ArticleNumber>Article Number</ArticleNumber>
    <Qty>Count of actually picked item</Qty>
    </OrderLine>
  </Acknowledge>
</LogiSoft-ERP>
```

## Segment Acknowledge

| Element | Type | Description |
|---|---|---|
| OrderNumber | String | Order number. |
| OrderTypeID | Integer | Manual Transactions are acknowledged with the OrderTypeIDâ€™s:<br>15 ~ Outbound<br>16 ~ Inbound<br>17 ~ Relative adjustments. Order numbers for manual transactions are generates using a prefix defined in the OrderType table and a sequence number based on the order id< |
| Owner | String | Owner of the order (empty if this feature is not used) |

| Element | Type | Description |
|---|---|---|
| Status | String | Status: OK/ERROR/CANCEL/PARTLY<br>OK: The order was carried out successfully.<br>ERROR: An unspecified error occurred, for example that the specified article was not found in the system.<br>CANCEL: The order was cancelled from LogiSoft link or by the operator.<br>PARTLY: The operator has indicated that the order could not be fulfilled because of missing articles. This happens if the actually number of articles does not fit the expected count. |
| FreightCost | Float | |
| Shipment | String | Shipment type set at consolidation. |
| Carrier | String | Transporter used for the order |
| Misc1 | String | Optional text field |
| Misc2 | String | Optional text field |
| Misc3 | String | Optional text field |
| Misc4 | String | Optional text field |
| Misc5 | String | Optional text field |

## Segment: Acknowledge/OrderLine

| Element | Type | Description |
|---|---|---|
| Status | String | OK: The order line was processed as requested. Partly: The request could not be completely fulfilled. Cancelled: The line was cancelled. |
| LineNumber | Integer | Order line number |
| ArticleNumber | String | Article number |
| QtyOrdered | Float | |
| Qty | Float | |
| CostCenter | String | |
| CostCenterText | String | |
| User | String | |
| Time | String | eg. 2020-10-29 16:46:57 |
| Misc1 | String | Optional text field |
| Misc2 | String | Optional text field |
| Misc3 | String | Optional text field |
| Misc4 | String | Optional text field |
| Misc5 | String | Optional text field |
| Misc6 | String | Optional text field |
| Misc7 | String | Optional text field |
| Misc8 | String | Optional text field |
| Misc9 | String | Optional text field |
| Misc10 | String | Optional text field |

The <OrderLine> section is repeated for each line in the original order.

An error status response will be generated in case the order could not successfully be imported by LogiSoft. Possible reasons are: specified articles did not exist in the database, order number is already used, etc. Use the log file for further debugging. Note that syntax errors or other problems that results from an order file that could

not be read, will not generate an acknowledge file! This is because the order number canâ€™t be known, in case the order file canâ€™t be read correctly by LogiSoft.

## Segment Acknowledge/OrderLine/Transactions/Transaction

The order line may be split up in transactions, for example if it is stored or picked from different locations. The transaction segment contains specific information related to this, it might be ignored

| Element | Type | Description |
|---|---|---|
| Qty | Float | |
| LocationAddress | String | Location address in the warehouse from where this transaction was handled. |
| User | String | Operator initial for this transaction |
| BatchNumber | String | Batch number |
| DeliveryNote | String | For inbound orders this field may contain a delivery note number entered at goods reception. |
| Locked | Integer | 0: The stock is released for picking. 1: The stock is locked, for example if it is stored for quality inspection. |

## Segment Acknowledge/OrderLine/Transactions/Transaction/SerialNumbers

| **Element ** | **Type ** | **Description ** |
|---|---|---|
| SerialNumber | string | Serial number recorded for this order line. There may be 1 to many serial numbers. |

The serial number segment is used to report back serial numbers recorded while the order line is picked or stored. It will only be visible if serial numbers are supported by installation and if any serial numbers are present. The number of serial numbers should match the picked or stored quantity.

# Order State

Optional telegram send from LogiSoft to the Host System to indicate the current state of an order. Used for example by the Host System to check if the order can be changed.

For pick-orders LogiSoft only allows orders to be updated or replaced when the order is “idle”. This means it cannot be released or in work, in that case an attempt to update the order will fail. If it for some reason is required to update a released order, it must first be recalled. Orders that are finished can be replaced by a new order with the same order number (although not recommended).

Syntax:

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
 <OrderState>
  <OrderNumber>order number</OrderNumber>
Â  <OrderTypeID>200</OrderTypeID>
Â  <Locked>true|false</Locked>
  <State>Waiting|Released|Working|Picked|Shipped</State>
  <Status>PartlyPicked</Status>
  <Time></Time>
  </OrderState>
</LogiSoft-ERP>
```

## Segment OrderState

| Element | Type | Description |
|---|---|---|
| OrderNumber | String | The order number. |
| OrderTypeID | Integer | The order type. |
| Locked | Boolean | False: The order can be changed. True: The order cannot be changed. |
| State | String | Waiting/Released/Working/Picked/Shipped |
| Status | String | General order status: Waiting/PartlyPicked/Cancelled/OK |
| Time | String | Time stamp when the state change occured. |

This telegram is an optional feature of the interface.

# Stock Count Request

It is possible to send an order to LogiSoft that requests specific articles to be counted. The format order the order is similar to the normal pick and store orders with the exception that the OrderTypeID is "19".

The example below will request a count for article ART001 and ART002. The request has the order number S1000.

```
<?xml version="1.0" ?>
<LogiSoft-ERP>
 <Order>
  <OrderNumber>S1000</OrderNumber>
  <OrderTypeID>19</OrderTypeID>
  <OrderLine>
   <ArticleNumber>ART001</ArticleNumber>
  </OrderLine>
  <OrderLine>
   <ArticleNumber>ART002</ArticleNumber>
  </OrderLine>
 </Order>
</LogiSoft-ERP>
```

## Stock Count Report

At the end of a stock counting, a report file is generated.Â

The result of the inventorying will be returned in a standard acknowledge document with same order number and OrderTypeID. The fields will have this meaning:

| Field | Type | Description |
|---|---|---|
| QtyOrdered | Float | This is the expected stock quantity. |
| Qty | Float | Actually counted quantity. |

> The quantity is always the total stock quantity for this article and not only the specific counted location. Thereby the Host System does not have to care about articles stored on multiple locations.

## Stock Counting Without Host Orders

It is also possible to generate local inventorying orders in LogiSoft. This will be reported back as well, but the OrderTypeID will be 18.

# Stock ReportÂ

Report of the total stock send to the Host. This is typically generated when the two systems are in an "idle" state â€" in other words, when no picking are in progress. It can either be send automatically when an inventory counting order is finished or manually on request.

## Segment StockReport\OrderÂ (One instance)

| Element | Type | Description |
|---|---|---|
| OrderID | Integer | |
| OrderNumber | String | |
| OrderTypeID | Integer | |
| Misc1 | String | |
| Misc2 | String | |
| Misc3 | String | |
| Misc4 | String | |
| Misc5 | String | |

## Segment StockReport/ArticleÂ Â (one or more instances)

| **Element ** | **Type ** | **Description ** |
|---|---|---|
| ID | integer | Line number from the Stock Order being reported |
| ArticleNumber | string | |
| Owner | string | Optional |
| BatchNumber | string | Optional |
| SerialNo | string | Optional |
| Unit | string | Optional |
| StockDate | date | |
| CountedBy | string | Optional |
| QtyExpected | int | |
| QtyCounted | int | |
| QtyDiff | int | QtyCounted - QtyExpected |
| QtyInGoodsReception | float | Store Qty not yet reported to ERP |
| QtyInConsolidation | float | Pick Qty not yet reported to ERP |
| LocationCount | int | Number of locations the article can be found at |

```
<StockReport>
  <Order>
    <OrderID>14</OrderID>
    <OrderNumber>CC  -3-3</OrderNumber>
    <OrderTypeID>18</OrderTypeID>
  </Order>
  <Article>
    <ID>2533</ID>
    <ArticleNumber>31969</ArticleNumber>
    <BatchNumber>74</BatchNumber>
```

```
      <CountedBy>System(sys)</CountedBy>
      <QtyExpected>3</QtyExpected>
      <QtyCounted>3</QtyCounted>
      <QtyDiff>0</QtyDiff>
      <QtyInGoodsReception>0</QtyInGoodsReception>
      <QtyInConsolidation>0</QtyInConsolidation>
      <NumberOfLocations>1</NumberOfLocations>
   </Article>
   <Article>
      <ID>581</ID>
      <ArticleNumber>20420</ArticleNumber>
      <BatchNumber>86</BatchNumber>
      <CountedBy>System(sys)</CountedBy>
      <QtyExpected>2</QtyExpected>
      <QtyCounted>2</QtyCounted>
      <QtyDiff>0</QtyDiff>
      <QtyInGoodsReception>0</QtyInGoodsReception>
      <QtyInConsolidation>0</QtyInConsolidation>
      <NumberOfLocations>1</NumberOfLocations>
   </Article>
</StockReport>
```

# Configuration

The Automatic report is generated by the database stored procedure Document_MakeStockCountedReport_OrderID.

The actually format of the report can be modified in this procedure.

The Automatic report is generated to the Document Port "StockReport". The document system must be configured to output port "StockReport" to the requested target location.

The OrdertypeID 19 (used for Automatic report) must be configured:

| OrderTypeID | Name | Description | OrderNumberPrefix | ConfirmMode | ConfirmPort |
|---|---|---|---|---|---|
| 18 | Inventory-Rel | Inventory(Releative) | IRE# | NULL | NULL |
| 19 | Inventory-Full | Inventory(Full) | IFU# | NULL | NULL |

Set ConfirmMode = 1 and ConfirmPort = "StockReport"

If user created inventory reports should be reported, then configuer OrdertypeID 18, the same way as OrderTypeId 19

Registry: Scheduler.jobs.StockReport

| Registry | Type | Value | Description |
|---|---|---|---|
| ClassName | String | dk.logiware.LogiSoftlink.JobStoredProcedure | Job module. |
| Name | String | StockReport | |
| TriggerType | String | RunAt | |
| TriggerValue | String | 01:00 | This will generate the report each night at 1 o'clock. |
| Enabled | Boolean | true | Set to true to enable the task. |

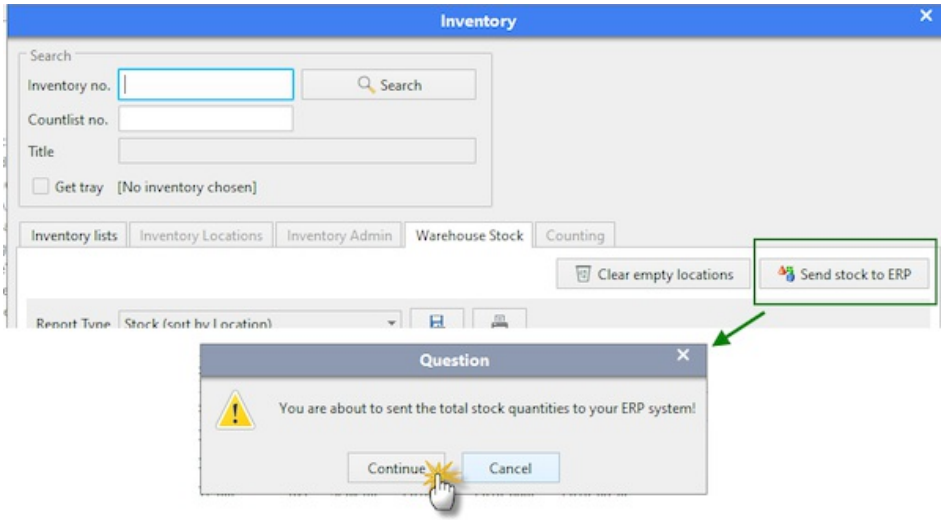
Registry: Scheduler.jobs.StockReport.Parameters

| Registry | Type | Value | Description |
|---|---|---|---|
| StoredProcedure | String | Document_MakeStockCountedReport_OrderID | Name of the stored procedure to execute. |

## Sending the stock to ERP when required

Set this Registry value to "true" : SharkLink.Export.InventoryFull.InventoryID

Click on button [Send stock to ERP]


Inventory
Search
Inventory no.
Search
Countlist no.
Title
Get tray
[No inventory chosen]
Inventory lists
Inventory Locations
Inventory Admin
Warehouse Stock
Counting
Clear empty locations
Send stock to ERP
Report Type
Stock (sort by Location)
Question
You are about to sent the total stock quantities to your ERP system!
Continue
Cancel

# XSLT Transformations

XSLT transformations are used to transform between the LogiSoft XML format and external XML formats.

# Customizations

Here are some “code snippets” for XSLT that can be useful to implement transformations.

## XML to XML transformations

The normal use of XSLT is for transformation between two different XML formats. LogiSoft supports both transformation of imported and exported XML files.

Some usefull constructions:

## Remove preceding zeroes (here from a SAP material number):

```
<xsl:if test="string(number(MATNR)) = 'NaN'"><xsl:value-of select="MATNR"/></xsl:if>
```

## XML to ASCII transformation

With XSLT it is also possible to transformXML into flat ASCII files. This can be a convenient way of making special formatted ASCII files. Â  Example of an XSL transformation that convert the LogiSoft XML stock report to a flat ASCII file.

```
 <?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
	<xsl:output method="text"/>
	<xsl:strip-space elements="*"/>
	<xsl:template match="/LogiSoft-ERP/StockReport">
		<xsl:for-each select="Article">
			<xsl:call-template name="OneLine"/>
			<xsl:text> </xsl:text>
		</xsl:for-each>
	</xsl:template>
	<xsl:template name="OneLine">
		<xsl:value-of select="ArticleNumber"/>
		<xsl:text> </xsl:text>
		<xsl:value-of select="Qty"/>
		<xsl:text> </xsl:text>
		<xsl:value-of select="Unit"/>
		<xsl:text> 0.0</xsl:text>
		<xsl:text> 0.0</xsl:text>
		<xsl:text> </xsl:text>
		<xsl:value-of select="LastCountedTime"/>
	</xsl:template>
	<!-- ************************************* -->
	<!-- Default match                         -->
	<!-- ************************************* -->
	<xsl:template match="@*|node()">
		<xsl:apply-templates/>
	</xsl:template>
</xsl:stylesheet>
```

## Convenient Examples (code snippets)

XSLT can be difficult to master, here are some examples of code, usefull to solve typical issues.

### Fixed length string right justified

```
<xsl:param name="article" select="Article"></xsl:param>
<xsl:value-of select="substring(concat($article,'                    '), 1, 24)"></xsl:value-of>
```

### Fixed length string left justified

```
<xsl:param name="qty" select="Qty"></xsl:param>
<xsl:value-of select="substring(concat('00000000',$qty), string-length($qty)+1, 8)"></xsl:value-of>
```

### Change decimal separator

Change the decimal separatorÂ “.” to “,”

Qty is the field to translate

```
<xsl:value-of select="translate(Qty,'.',',')"/>
```

### Changing thousand separator

Changing thousand separator and/or decimal separator.

First remove thousand separators (.) then, in the result, replace decimal separator (,) with (.)

precondition: thousand separator (if any) must be (.) decimal separator (if any) must be (,)

```
<xsl:value-of select="translate(Qty,'.',',')"/>
<xsl:template match="/LogiSoft-ERP/StockReport/Stock/Qty">
 <xsl:variable name="qty" select="."/>
 <xsl:variable name="qtyW_OThousandSeparator" select="translate($qty,'.','')"/>
 <xsl:variable name="qtyWDecimalPoint" select="translate($qtyW_OThousandSeparator,',','.')"/>
 <Qty><xsl:value-of select="$qtyWDecimalPoint"/></Qty>
</xsl:template>
```

### Insert a new line

Insert a new line in text files using platform style new lines.

Create a variable “newline” (with a new line) and refer later to that variable.

This method will use the new line format for the current platform

```
<xsl:variable name="newline"><xsl:text>
</xsl:text></xsl:variable>
..
..
<xsl:value-of select="$newline" />
```

**Insert a new line in text files using hardcoded new lines.**

Carriage return: 

New Line: 


```
 <xsl:variable name="newline"><xsl:text>&#10;</xsl:text></xsl:variable>
..
..
<xsl:value-of select="$newline" />
```

Can be used to insert all kind of special characters.

# Configuration

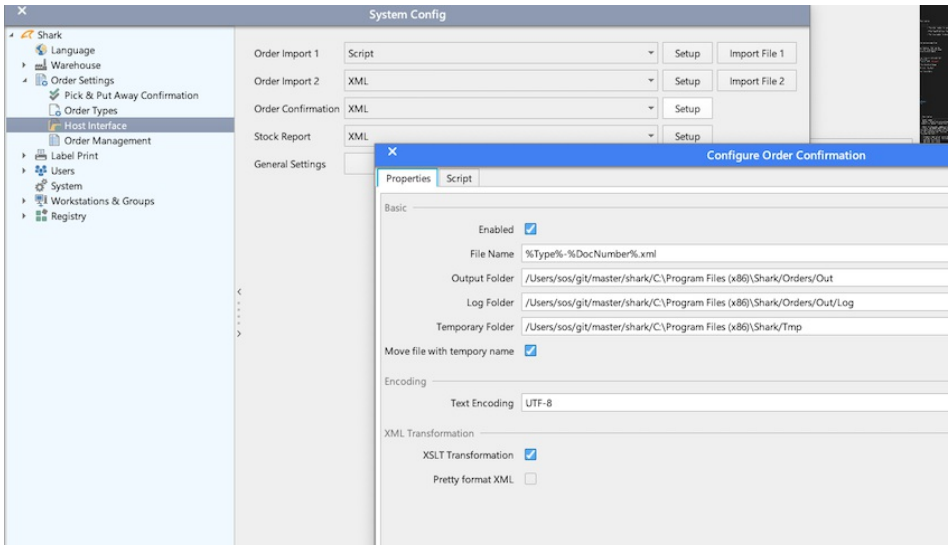
The XML adapter uses two interface methods for import and export of data.

Import of files is done by one or more jobs running under the LogiSoft Scheduler.

Export of XML files is done through the DocumentExport job also running under the Scheduler.

## Configuration of the XML Exporter

The file exporter is configured from the System Configuration menu (Maintenance -> System Configuration).




Configure the parameters:

**Enabled**: Must be selected to make the interface run.

**File Name**: File pattern for the generated output file, use *place holders* to define the name (se below).

**Output Folder**: The folder where the file will be stored. See the section Using FTP to send the file by FTP.

**Log Folder**: A copy of the generated file will be stored here.

**Temporary Folder**: Working folders used when transforming and generating the file. To avoid working in an external network drive.

**Move file with temporary name**: The file is written with an tmp extension and renamed to the final name after created.

**Text Encoding**: Default is UTF-8.

**XSLT Transformation**: Activate the transform the generated output using an XSLT transformation (the XSLT is edited on the "Script" tab). This might both generate XML or ASCII output (like csv files).

**Pretty Format XML**: Nicely format the XML, to make it human readable. Please note this might change the format in an undesirable manner.

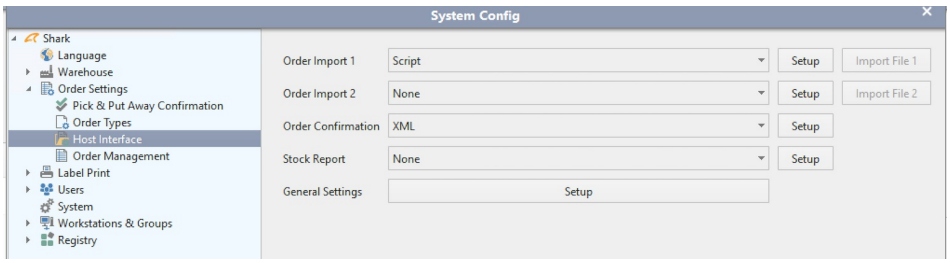
Place holders used for the file name:

| Parameter | Description |
|---|---|
| %DocNumber% | Current Document ID. This is a unique running number, that can be used to trace the document. |
| %Type% | Type of document. |
| %Copy% | If the document has been resend, this number will be incremented each time. Is “1” the first time. |
| %Date% | Current date as yyyyMMdd |
| %Time% | Time formatted as HHmmss |
| %Arg1% | Argument 1. A parameter where the content depends on the rest of the configuration. Typical it is the order number. |
| %Arg2% | Optional document parameter. |
| %Arg3% | Optional document parameter. |

# Configuration of the XML Importer

The â€œImporterâ€? is the job that imports the XML files. It is running as a job in the LogiSoft Scheduler and must be configured to work. The setup will in most cases be handled by the installation and system initialization wizard. But if configured later, it must be done by hand.

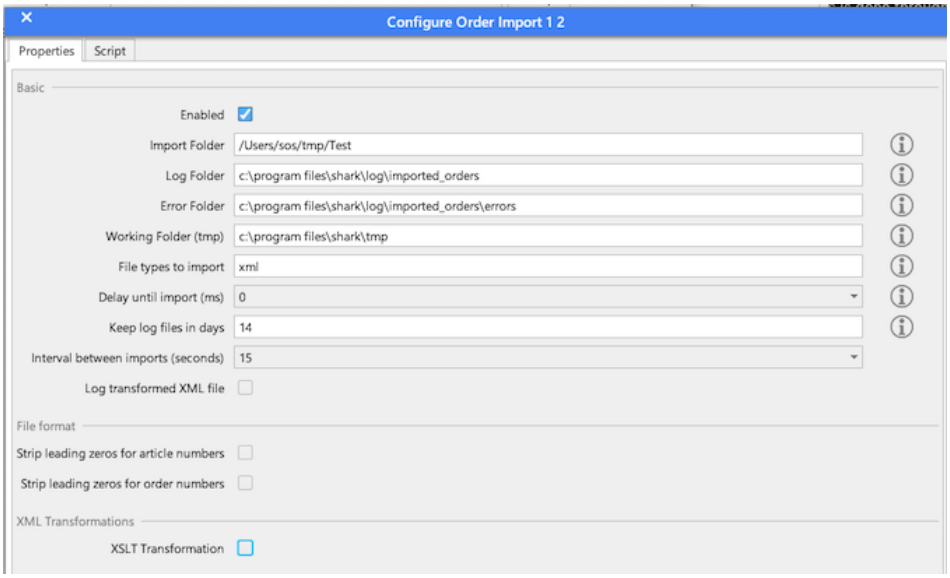
The file importer is configured from the System Configuration menu (Maintenance -> System Configuration).




Two importers are available. Normally only one is needed, but due to the fact only one file is imported at a time it might slow the import if files, if a large file is under import. The typical scenario is that large master data files are sent during working hours and they might block for more important pick orders. The solution is to setup two importers, using two different folders, one for large, less important imports and another “fast lane”.

Select “XML” as the import type (other types may be available, but they are not described here).

Press “Setup” to open the configuraton menu.




Now configure the parameters:

**Enabled**: Must be selected to make the interface run.

**Import Folder**: This is the folder from where the files are imported. Remember that if the folder is an external folder, it must be a UNC path (do not use mapped drives, they are typical not visible from services). All check

the user access rights to the folder, if the importer is running as a Windows service, the user running the service must have access rights to the folder. See the section Using FTP to import from a FTP folder.

**Log Folder**: When the file is imported successfully, it is moved to this folder.

**Error Folder**: If the import failed, the file will go here (so this folder should be empty).

**Working Folder**: If this folder is defined, it will be used as a temporary folder. Network folders can be slow and therefore, the importer can move the file to an internal folder and do the importing from this folder.

**File types to import**: File extension of files to import.

**Delay until import**: The importer will first import the file, when it can get write access to the file. Some host systems might open/close the file for each line, this might start an import before the file is finished. A work-around to this is to specify an import delay. When the file is seen by the importer, it will wait the specified number of ms.

**Keep Log files in days**: The imported files are moved to the log folder and automatically deleted after some days, as specified here. Note that it will not delete in the error folder.

**Interval between Imports**: Time between the import folder is polled. Minimum time is 5 seconds. Do not set the time lower than required.

**Log transformed XML file**: If an XSLT transformation is used, this wil also log the output of the transformation.

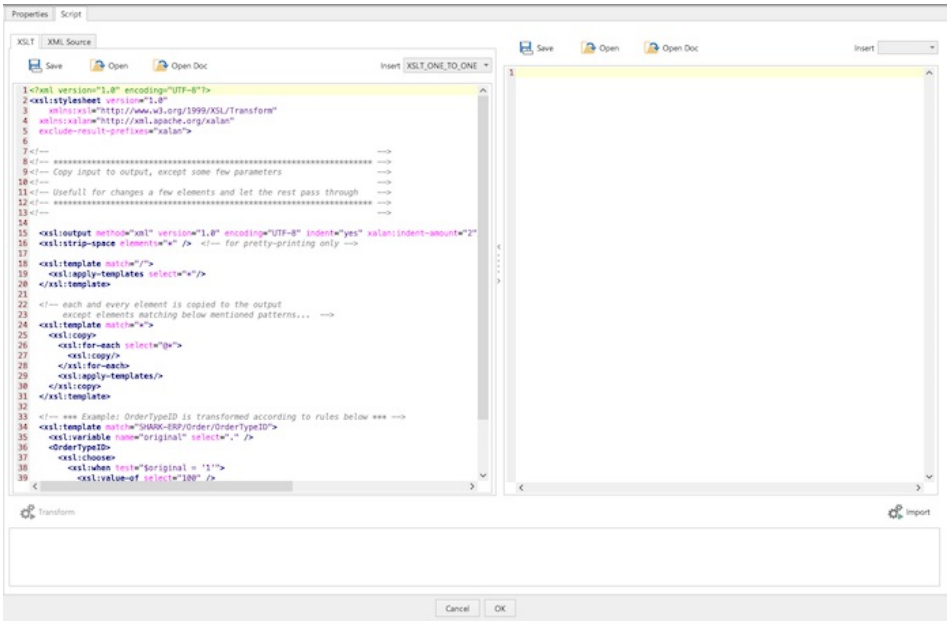
**Strip leading zeros from article numbers**: Some Host Systems uses integer numbers as article numbers (LogiSoft uses strings), this might generate long numbers with zeros in front. Use this option to strip of the zeros. Check this will work with confirmation back to the host.

**Strip leading zeros from order numbers**: Some Host Systems uses integer numbers as order numbers (LogiSoft uses strings), this might generate long numbers with zeros in front. Use this option to strip of the zeros. Check first this will work with confirmation back to the host.

**XSLT Transformation**: Enable this to use an XSLT transformation when importing. The script is edited on the "Script" tab.

## Define XSLT Configuration for the XML Importer

First select the "script" tab to open the script editor.



Here you find 2 tabs:

**XSLT**: This is the XSLT script. There is an "Insert" menu from where code snippets for basic XSLTs can be pasted into the XSLT window.

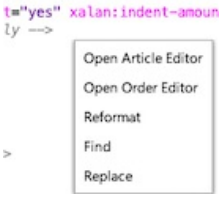
**XML Source**: XML test data. Either copy/paste the XML source or import a file.

To the right is a window showing the result from the transformation.

To transform without import, press the "Transform" button.

To import the transformed XML, press the "Import" button.

Right clicking the mouse will show a menu with the following features:




**Open Article Editor**: Select an article number and choose this, the Article Editor will be opened with this article number.

**Open Order Editor**: Select an order number and choose this, the Order Editor will be opened with this order number.

**Reformat**: Uses Tidy to format the XML/XSLT.

**Find**: Search for text.

**Replace**: Replace text in file.

## Using FTP

The file importer and exporter supports also FTP and secure FTP as file transfer protocols. The login information can't be set using the normal configuration, but must be configured in the *registry*.

LogiSoft must be the FTP client.

The import or export folders must be setup like this:

| Transfer type | Format | Description |
|---|---|---|
| FTP (active mode) | ftp://host/folder | Basic FTP using active mode |
| FTP (passive mode) | ftp://host/folder | Basic FTP using active mode. The registry parameter ".FTP.PassiveMode" must be set "true" |
| SFTP | sftp://host/folder | |

# REST API

LogiSoft WMS supports integration to other systems, using a REST based web service API. It is used to provide an interface for external Host/ERP systems for on-line handling of orders and exchange of other information like article master data, inventorying requests, etc. Orders can be transferred from the Host system and LogiSoft may return an acknowledge back, when the order has been processed. Might also return stock adjustments, when the actually number of articles on stock does not fit with the expected count or when an order could not be fulfilled completely. In general, the purpose is, that LogiSoft and the host system can be used as one homogeneous system.

The reference documentation, for the latest version (1.1.1) can be found here

The REST API has some great advantages compares to exchanging information by files:

- It is fast, the order is created immediately in LogiSoft when it is sent by the Host.
- It is reliable. The Host get an immediately confirmation back if the order was created successfully as a status code and possible error message from the REST call.
- It works over the Internet and does not need any shared folders or ftp servers.

The return of confirmations and other asynchronous events, can be done in 3 ways:

- Using Webhooks. This is the preferred method, it requires no development in LogiSoft and has a low overhead and fast response. The user species an URL to call, when a certain event occurs. It requires the host is able to provide an URL for a web server.
- Using polling. Might be easier to implement in some systems, especially if the Host system does not have its own rest server. It requires more resources, since it has to ask at fixed intervals, if any confirmations are available.
- If the host system has a web service API itself, the confirmation can be done using that, but it requires implementation of the API in LogiSoft, typical done by a Groovy script.

It is also possible to work without acknowledge back, this might be OK for some setups.

The REST API is supported both for cloud based installations and on-prim (own server).

For the Cloud solution, use this URL: https://restapi.sharkwms.com/1.1/, a token is required for authentization. This can be obtained from the Shark PC client.

For on-premise installations, the URL depends on where the software is installed.

# Typical Integration to an ERP/Host System

What is needed for a typical implementation? There is no standard answer and the integration can be simple or complex, depending on the requirements.

This is the proposed priority of integration phases. Only level 1 is required to have the basic functionality for a working system.

## Level 1 - Basic Support for Picking by Orders

A minimum solution, for an integration that support picking of orders from the warehouse. At this level storing of goods is supposed to be done "manually", without involving the Host System. Typical most of the time is spend picking, so this first level may be an easy way to get a lot of the benefits from an integration.

This is a one-way communication and requires a minimal of effort at the Host side.

| Implement | Notes |
|---|---|
| post/article | Article Master Data is sendt from the ERP system to LogiSoft. It includes article number, description and perhaps EAN code. The inbound order can also include article description and new articles are automatically created in LogiSoft when an inbound order is received with a new article and the Article Master Data telegram could be skipped if very little master data is required. |
| post/order | To create an outbound order from the systemm (picking). |

## Level 2 - Get Confirmation back to the Host System

For the simple integration, the Host had no ideas if the order actually was successfully carried out. The next step could be to add a confirmation for a picked order back to the host.

| Implement | Notes |
|---|---|
| get/confirm | Get confirmation back about processed orders and transactions that effect the warehouse stock. The simplest way is to use polling. |

## Level 3 - Add support for Inbound Orders

The next step, could be to integrate support for inbound orders. This does actually not require any new implemenation of end-points. The same post/order format is used, used with another order type parameter and perhaps some other fields.

## Level 4 - Add support for Stock Adjustment

Telegram when the stock is changed without an order from the ERP system. This could be the result of a stock error that is adjusted by an operator or other operations that change the stock.

# General Formatting

## Date-Time

Date/time values are formatted as defined by RFC 3339, for example 2017-07-21T17:32:28Z.

# Master Data

Master data is basic information about the articles. The most important information is the article number and description, but other information can be transferred as well. LogiSoft will create new articles and update the description, when met in an imported order, but the Master Data allows more information and will also work if LogiSoft is used with the Manual Transactions (transactions with no host-orders), this could be the case in an initial store process, where goods are moved from old locations into LogiSoft without orders.

## Packages

Packages are using for handling different package sizes for the articles. It is optional information that can be specified with the article master data.

# Processing Orders

This section shows the flow of information in different possible configurations.

## Inbound Orders

The Host may send an order when inbound goods are expected. When the goods are received, it will be matched against the order. Typical reasons to receive goods are:

- Purchase orders.
- Returned goods from customers.
- Produced items from own production.
- Replenishment from other storage locations.

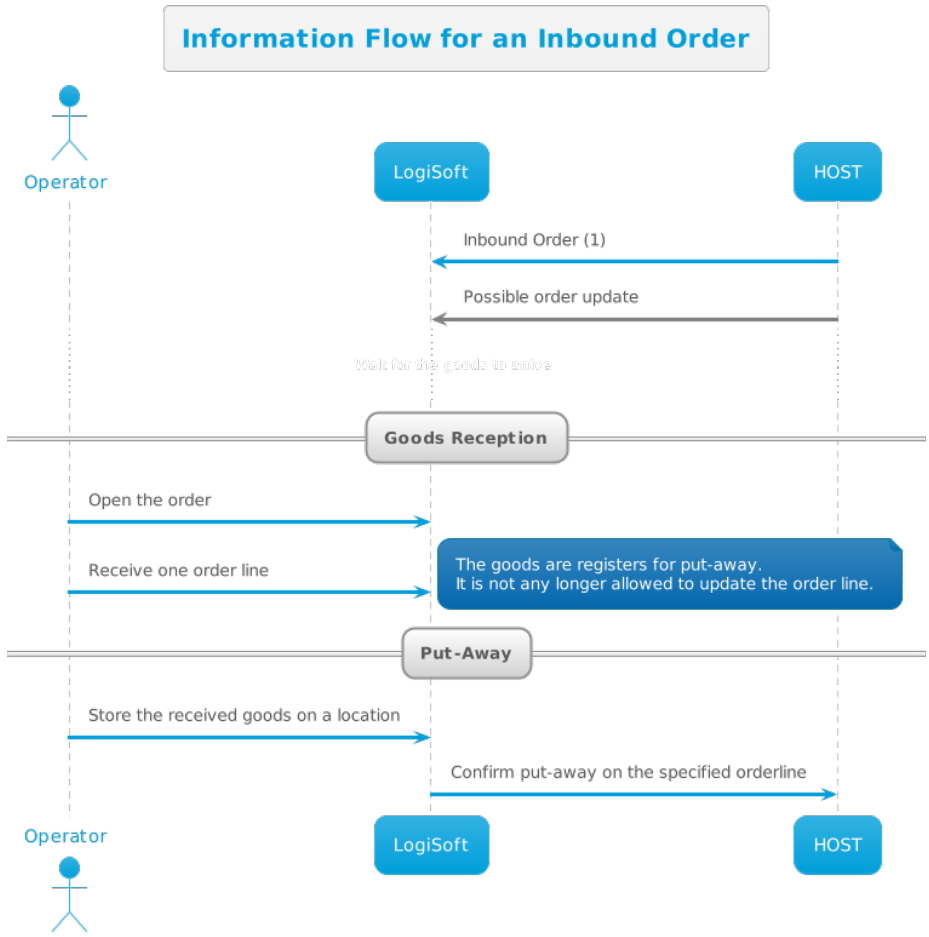
When the goods are stored on the final location, it is reported back to the host. To tell the host that the goods are ready to be picked and an invoice for the goods can be accepted. This is typical done per stored transaction.

It is recommended that the file is created with a temporary extension and renamed when the file is ready. This to prevent the file to be read, before it is finalized by the host.

The file is moved to a temporary folder. This to ensure that nobody else uses the file and if the "Import" folder is on an external drive, to speed up the processing speed.

LogiSoft Link process the file from the temporary folder.

If the processing was successful the file is moved to the log folder else the file will be moved to the "Error" folder. ng of incoming goods can be received in several deliveries and the put-away process may take a while.




*Basic2 flow of goods reception with acknowledgement back to the host for each received line*

(1) The Host might send the inbound order as soon as possible (when it is created) or delayed until it is needed. It is also possible to make the actually registration in the Host System and then let the Host send an in-bound order with the actually received goods. This might end in a two step process, but can have other advantages.

## Outbound Orders (Picking)

The Host sends orders to request goods to be picked. This can for example be sales orders to customers or production orders.

Confirmation back to the host can be:

- When the order is finalized and confirmed.
- When the order is picked completely (no confirmation).
- Line by line.

### Import Rules for Orders

The following table shows the configuration parameters and how LogiSoft reacts in various situations when importing picking orders.

| Parameter | Value | Note | |
|---|---|---|---|
| AutoCreateItem | True | False | If true, unknown articles will be created when needed. |
| CancelOrderLinesWithoutStock | True | False | If true the stock must be present else the orderline will be deleted. |
| ItemNumberUnique | True | False | The same article number must only appear once per order. |
| ConfirmPickPerLine | True | False | Acknowledge generated immediately for each pick line. |
| ConfirmPutPerLine | True | False | Acknowledge is generated immediately for each put-away line. |
| LinkDeleteZeroLines | True | False | If a line is received with a 0 count, it will be deleted. Used to cancel lines existing orders. |
| ReleaseOrderMode | Auto | Manually | Orders must be released before they can be picked. Auto mode will release the order immediately when received. |
| ReplaceOrders | True | False | If the same order is receved 2 or more times, the order will replace the previous one. If false the lines will be appended. |

## Updating Outbound Orders

Some times it might be required to update an existing pick order. This should be avoided if possible, but if not, here is the rules and behavior when it happens.

To start, it is important to understand the life-cycle of an order.

When an order is resend, with the same order number/delivery note number, it will replace the existing order and delete the previous order!

| Order State Â | Â Result |
|---|---|
| Not released/not started | Â The old order will be deleted and a new created. |
| Not released/partly picked | The old order will be deleted and a new created. |
| Released | The order update will fail and rejected. Recall the order first. |
| OK (finished) | The old order will be deleted and a new created. |

### Partly Deliveries

Inbound orders are often partly delivered, this means that the expected quantity to be received must be registered at several times. To make the Host aware of the received goods, it can be practical to confirm *by transactions*, this means that the same order line can be confirmed several times.

## Updating Purchase (put-away) Orders

It can be required to modify an already sent order. This is possible with the limitation, that when the first goods are registers on one order line, it can no longer be changed. When changing an order the whole order is send again.

## Special Requirements

Some special requirements can optionally be supported like:

- Reservation of articles for a specific picking order.
- Inspection of incoming goods.
- Registration of batch/lot and serial numbers.

## Auto processing of orders

For the order line, there is an "auto" flag that can be used to automatically process the order line immediately. It does also require the location field to be defined.

## Reserved Purchases

A feature that support purchases reserved for a specific sales order.

## Incoming Inspection

Used to flag purchases that must be inspected before they are released for picking.

When an order line with incoming goods are marked for inspection, it is decided at the reception time, if the articles are accepted or stored for inspection. In case all or some articles are stored for inspection, they will be put on a location where the stock is locked and cannot be picked with ordinary orders. The location is also marked with the incoming order number.

The Host can get a list of all items waiting for inspection using the GET /Inspection endpoint.

The Host can change the status of items waiting for inspection using the PUT /Inspection endpoint.

## Batch Number Registration

The batch number of incoming goods must be registered. The batch number can be reported back to the Host.

## Serial Number Registration

Serial numbers of incoming articles must be registered. The serial number can be reported back to the Host.

# Outbound Orders (Picking)

The Host sends orders to request goods to be picked. This can for example be sales orders to customers or production orders.

Confirmation back to the host can be:

- When the order is finalized and confirmed.
- When the order is picked completely (no confirmation).
- Line by line.

## Updating Pick Orders

Some times it might be required to update an existing pick order. This should be avoided if possible, but if not, here is the rules and behavior when it happens.

To start, it is important to understand the life-cycle of an order.

When an order is resend, with the same order number/delivery note number, it will replace the existing order and delete the previous order!

| Order State Â | Â Result |
|---|---|
| Not released/not started | Â The old order will be deleted and a new created. |
| Not released/partly picked | The old order will be deleted and a new created. |
| Released | The order update will fail and rejected. Recall the order first. |
| OK (finished) | The old order will be deleted and a new created. |

## Part List functionality

LogiSoft supports creation of parts list from a master order. This feature allows a single order line to be broken down into multiple picks of different articles and thereby supporting a setup where the articles stored in the warehouse are sub components of a structure. The part list is supplied with each order from the host and is not stored in LogiSoft, this means all maintenance of the part list is done in the Host system.

Each order line that contains a part list will create a separate sub-order. The sub order with the part list has to be picked before the master order and must be stored on a location first. This is useful if picking the part list also involves some small production, like for example assembly of the parts or just packing of the product.

In the master order, the part list will be seen as an article with the name:

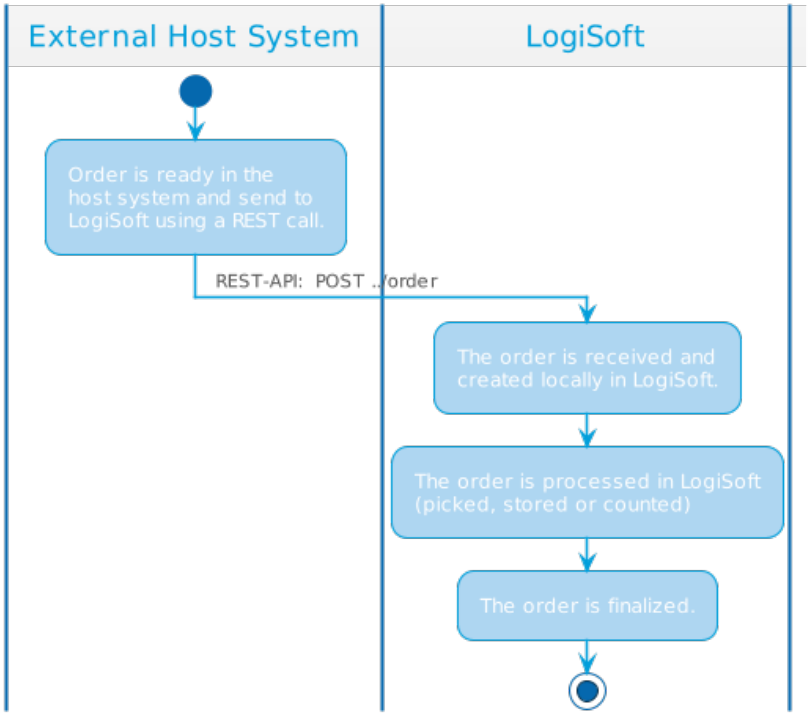
```
<master order number>.<part list number>.<master order orderline linenumber>
```

This means the picked part list order will be stored with a unique article number, to ensure it goes to the correct master order.

The Master Order will have a back order status, until all part lists in the order have been picked.

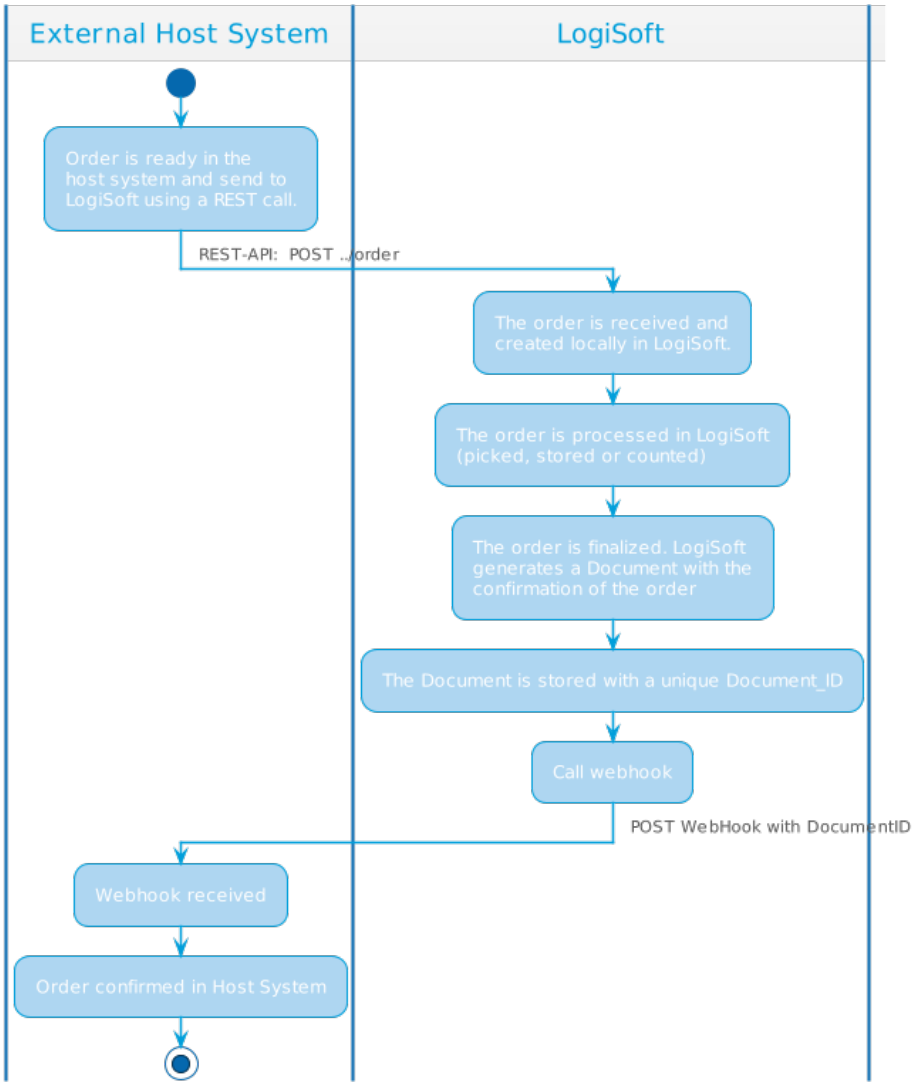
## Order Processing without Confirmation

This is the flow if information, when the integration has no acknowledge back. This is by far the most simple way to make an integration.



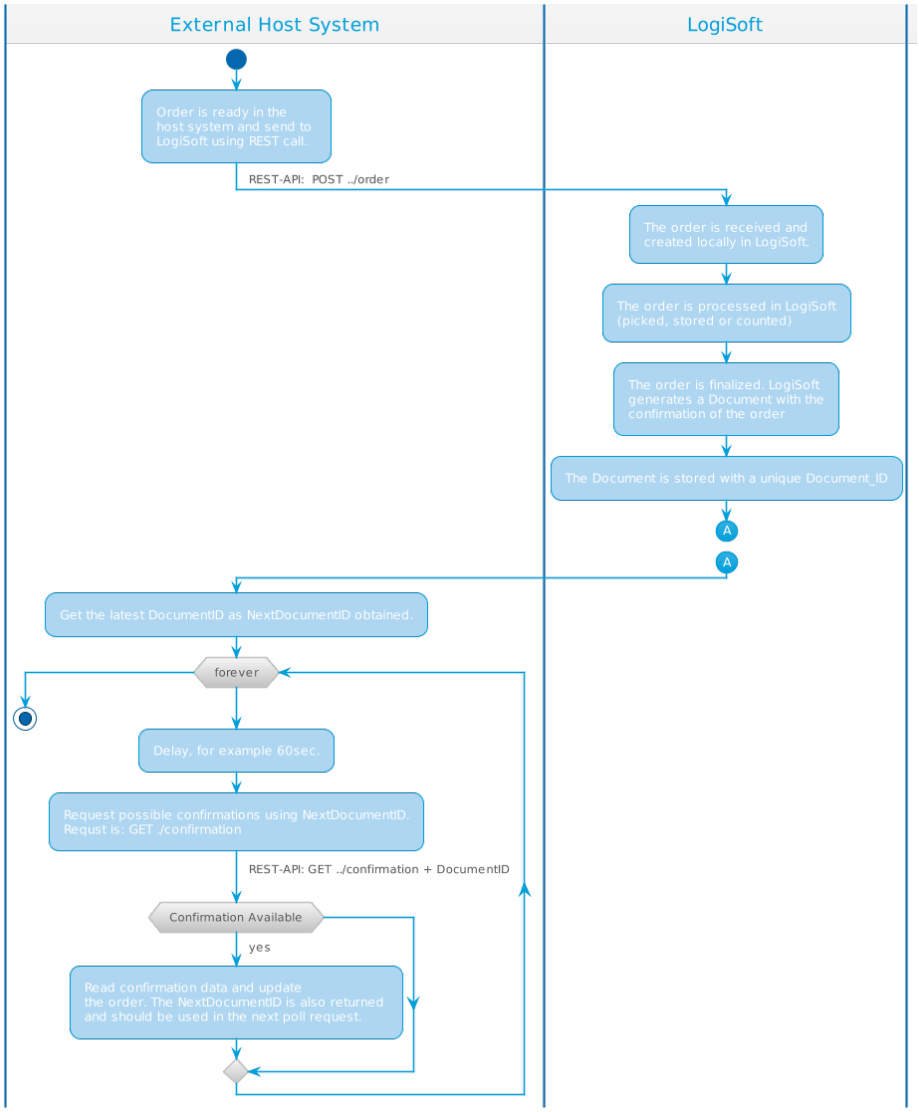

## Order Processing using Webhooks

This is the flow if information, when the integration is based on webhooks.




## Order Processing using Polling

This flow is an example of how a polling based integration can be made.


External Host System
LogiSoft
Order is ready in the host system and send to LogiSoft using REST call.
REST-API: POST ../order
The order is received and created locally in LogiSoft.
The order is processed in LogiSoft (picked, stored or counted)
The order is finalized. LogiSoft generates a Document with the confirmation of the order
The Document is stored with a unique Document_ID
A
A
Get the latest DocumentID as NextDocumentID obtained.
forever
Delay, for example 60sec.
Request possible confirmations using NextDocumentID. Requst is: GET ./confirmation
REST-API: GET ../confirmation + DocumentID
Confirmation Available
yes
Read confirmation data and update the order. The NextDocumentID is also returned and should be used in the next poll request.

# Stock Taking Orders

A stock taking order can be created as any other order, just use OrderTypeID=19.

The only parameters that are used are order number, article number, batch number and owner. The rest is ignored.

After the order is executed (counted), it will be acknowledge in the usual order acknowledge format, with the original order number and OrderTypeID=19. The following fields have special meaning:

| Field | Type | Description |
|---|---|---|
| QtyOrdered | Float | This is the expected stock quantity. |
| Qty | Float | Actually counted quantity. |

> The quantity is always the total stock quantity for this article and not only the specific counted location. Thereby the Host System does not have to care about articles stored on multiple locations.

Example that will create an inventory order "INV004", requesting a count of two articles.

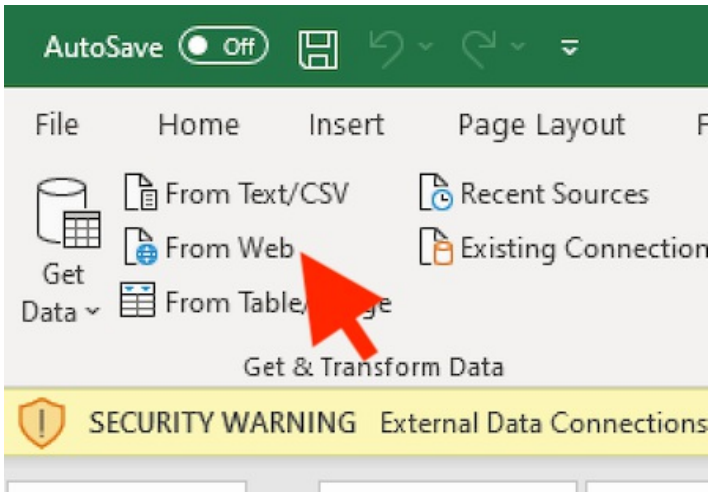
```
{
 "orderNumber": "INV004",
 "orderTypeID": 19,
 "orderline": [
   {
    "articleNumber": "67547"
   },
   {
    "articleNumber": "20087"
   }
 ]
}
```

# Reports

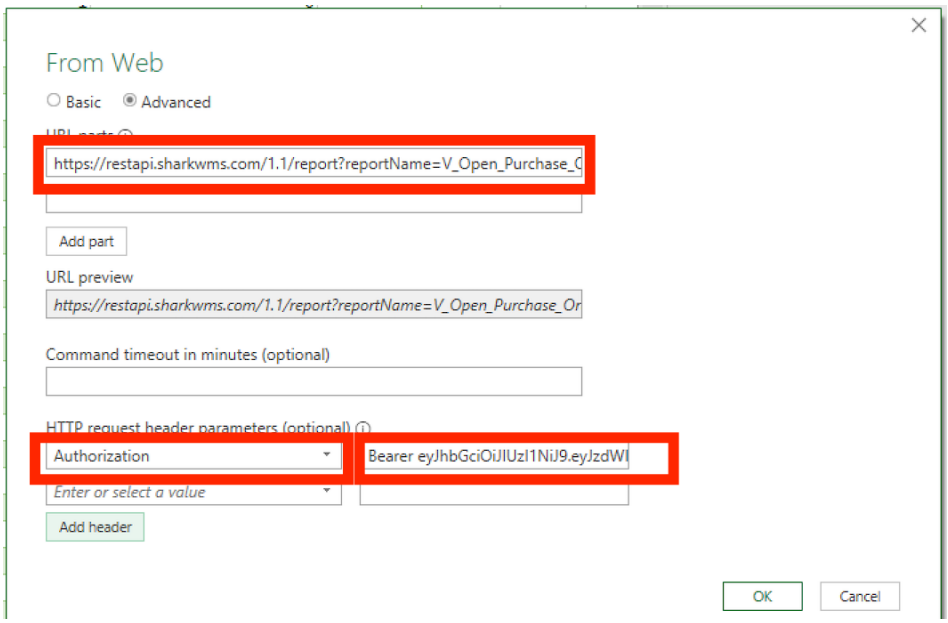
The integration allows reports defined in the database to be returned. There are standard reports and it is also possible to have user defined reports.

Data is returned as json and is compatible with Excel, thereby it is possible to use it for extraction of data, directly into Excel.

**Step 1 - Open the Connection Dialog for the data source.**




**Step 2 - Enter the required connection information.**



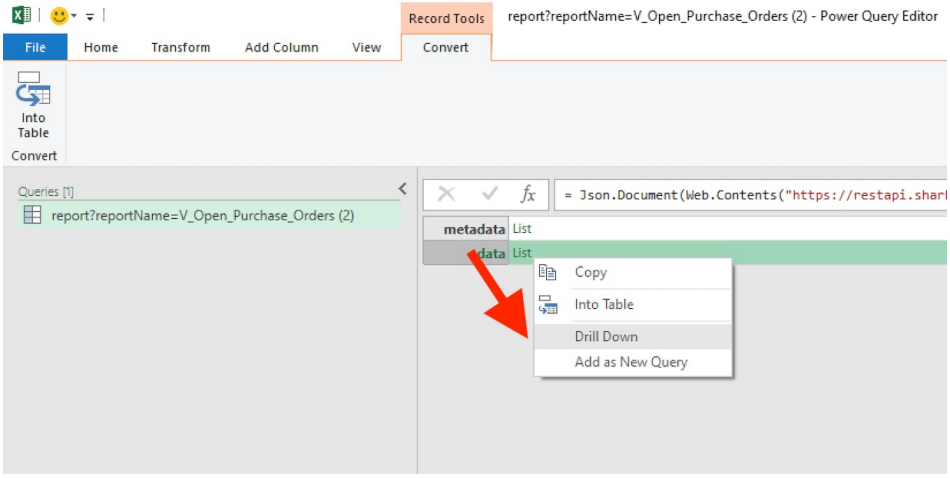

Enter the URL, it is

- https://restapi.sharkwms.com/1.1/report?reportName= .*
- Add a new header "Authorization".
- Add the value for Authorization, it is "Bearer + ". The token is a unique token for your installation.
- Remember to press *Add header*, when the info is entered.
- Press *OK*

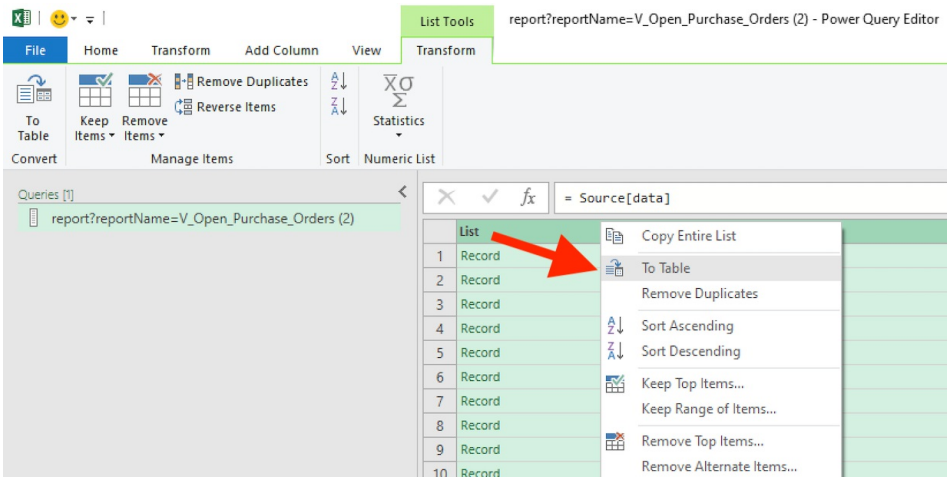
**Step 3 - Select the data, using the Power Query Editor**

When the connection is entered successfully, Excel will fetch the data and open the *Power Query Editor*.

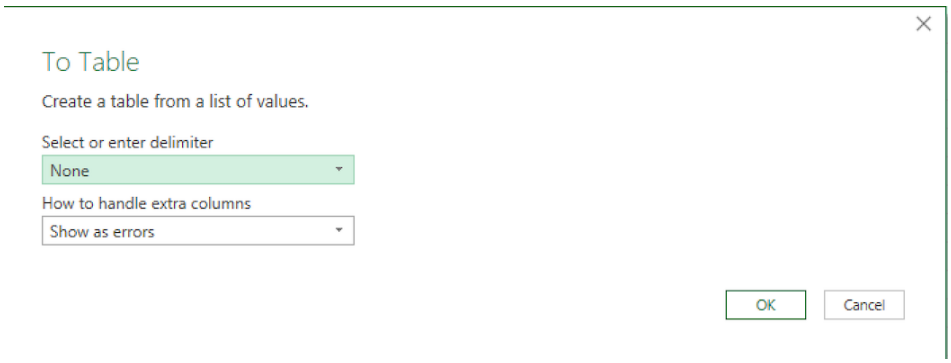
Do the following steps:



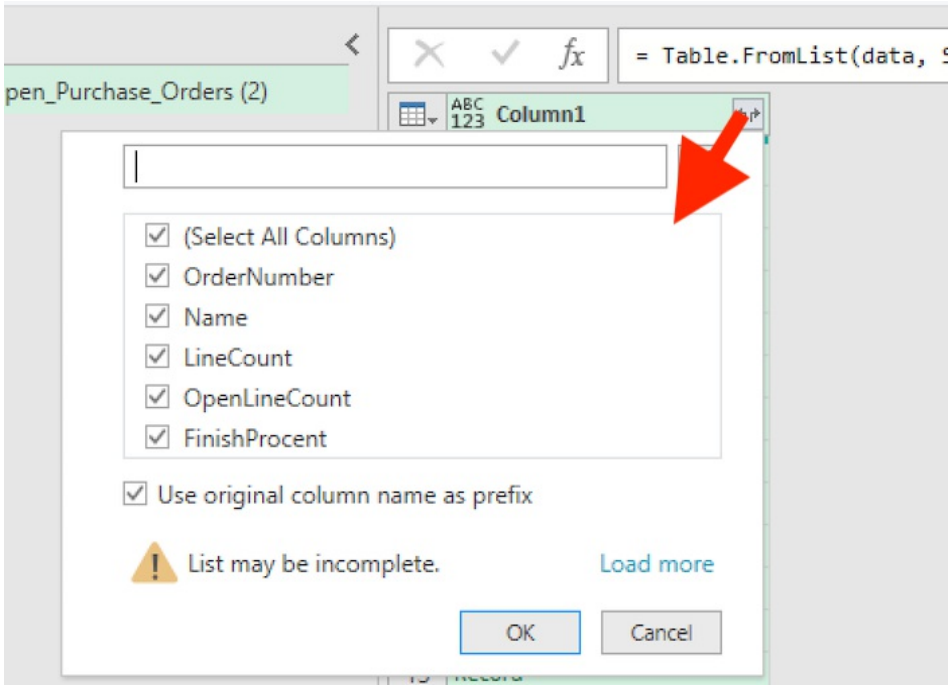

*Select and right-click the data-List and Drill Down the data (get rid of the metadata)*



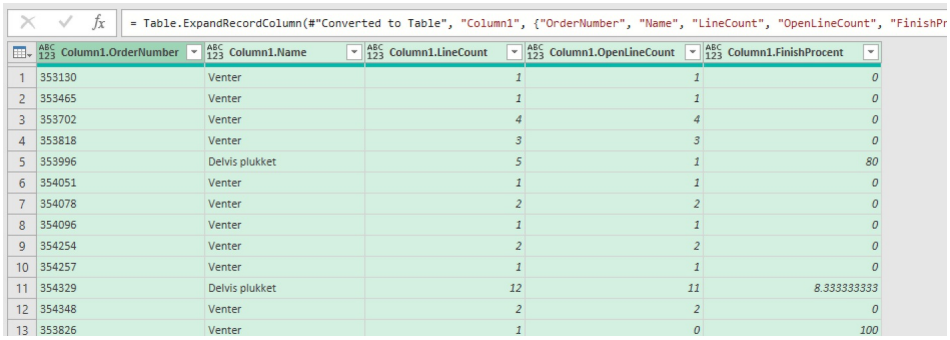

*Select List, right-click and convert the list to a table*




*This will ask for delimiters, use the default "None" and press OK*




*Then unpack the columns by pressing the small icon in the column, choose the default "all" or select specific columns*



= Table.ExpandRecordColumn(#"Converted to Table", "Column1", {"OrderNumber", "Name", "LineCount", "OpenLineCount", "FinishPr

| | Column1.OrderNumber | Column1.Name | Column1.LineCount | Column1.OpenLineCount | Column1.FinishProcent |
|---|---|---|---|---|---|
| 1 | 353130 | Venter | 1 | 1 | 0 |
| 2 | 353465 | Venter | 1 | 1 | 0 |
| 3 | 353702 | Venter | 4 | 4 | 0 |
| 4 | 353818 | Venter | 3 | 3 | 0 |
| 5 | 353996 | Delvis plukket | 5 | 1 | 80 |
| 6 | 354051 | Venter | 1 | 1 | 0 |
| 7 | 354078 | Venter | 2 | 2 | 0 |
| 8 | 354096 | Venter | 1 | 1 | 0 |
| 9 | 354254 | Venter | 2 | 2 | 0 |
| 10 | 354257 | Venter | 1 | 1 | 0 |
| 11 | 354329 | Delvis plukket | 12 | 11 | 8.333333333 |
| 12 | 354348 | Venter | 2 | 2 | 0 |
| 13 | 353826 | Venter | 1 | 0 | 100 |

*The selecteddata is now visible*

Save the Excel sheet for later use. Press refresh to update the data.

# Authentication

The REST API is using a token based authentication system called Bearer authentication.

Bearer authentication (also called token authentication) is an HTTP authentication scheme that involves security tokens called bearer tokens.

The client must send this token in the Authorization header when making requests to protected resources: Authorization: Bearer

To improve security, it is recommended to use HTTPS (SSL), this will encure encryption of the token.

The token is delivered in the header of the request i the authorization field. Looks like the example below.

```
POST /1.0/order HTTP/1.1
Host: api.sharkwms.com
User-Agent: curl/7.54.0
accept: */*
Authorization: Bearer LSDFIE24LDFD4
Content-Type: application/json
Content-Length: 19
```

# WebHook

Webhooks are user defined web callbacks using HTTP POST.

LogiSoft supports webhooks to notify to an external system, when certain events occurs. These events are typical when LogiSoft has data ready to be read by the external system. The big advantage is that polling thereby can be avoided.

The body of the webhook is a json data structure with all the required information.

In version 1.1, this structure for an order confirmation looks like this:


```
{
 "nextID": 0,
 "documentID": 0,
 "status": "string",
 "orderTypeID": 0,
 "orderNumber": "string",
 "deliveryDate": "2019-08-24T14:15:22Z",
 "owner": "string",
 "deliverNoteNumber": "string",
 "carrier": "string",
 "weight": "string",
 "colli": "string",
 "misc": {
  "misc1": "string",
  "misc2": "string",
  "misc3": "string",
  "misc4": "string",
  "misc5": "string",
  "misc6": "string",
  "misc7": "string",
  "misc8": "string",
  "misc9": "string",
  "misc10": "string"
 },
 "orderline": [
  {
   "status": "string",
   "lineNumber": 0,
   "articleNumber": "string",
   "qtyOrdered": 0,
   "qty": 0,
   "costCenter": "string",
   "costCenterText": "string",
   "user": "string",
   "time": "2019-08-24T14:15:22Z",
   "misc": [
   ],
   "serialNumbers": [
   ],
   "transactions": [
   ]
  }
```

```
  ],
  "partlist": [
    {
      "partListNumber": "string",
      "orderTypeID": 1,
      "lines": [
      ]
    }
  ]
}
```

## Order State

It is possible to get a post on a webhook when an order change it state.

**Possible Inbound Order States:**

| State | Can be modified | Notes |
|---|---|---|
| DONE | No | The order is received and stored completely. |
| ERROR | Yes | Something went wrong. |
| PARTLY | No | The order has been received partly and stored. |
| READY | Yes | The inbound order is created, no goods have have been received. The order can be modified by the host. |
| WORKING | No | Some goods have been received, but not yet stored on a storage location. |

**Possible Outbound Order States:**

| State | Can be modified | Notes |
|---|---|---|
| BACKORDER | Yes | The order cannot been picked, because not all stock is available. |
| CONSOLIDATED | No | The order is consolidated (packed) and shipped. |
| DONE | No | The order is picked, not yet consolidated/shipped. |
| ERROR | Yes | Something went wrong. |
| READY | Yes | The outbound order is created, the order is idle and nothing has been done yet. The order can be modified by the host. |
| RECALLED | Yes | The order has been released, but is recalled (unrelased). It can now be modified by the Host System. |
| RELEASED | No | The order is released and has allocated stock. Ready for picking to start. |
| WORKING | No | The order is being picked. |

# Reference Documentation

The API reference documentation is found here:

Reference documentation

The documentation is made using Swagger. It support generation of example codes and source code generations for some languages.

Reference documentation at Swagger

Note that most of the parameters in the REST calls are optional and you should skip them if not needed. If they are included either with default content or empty, they will be used and is and sometimes with unexpected results.

Client source code can also be generated by tools from the OpenAPIproject. Download the json OpenAPI specification for the LogiSoft API and use the tools for making code templates.

# Tools for Testing

You can test the REST API with different tools.

For Windows and Mac *postman* is a great tool. You can download it from [https://www.getpostman.com]

From the command line curl can be used. Download curl from [https://curl.haxx.se].

# SAP Connector

The LogiSoft Link SAP Connector supports interface to SAP, using the XML IDOC file format for receiving and confirmation.

The described setup requires the SAP WM module to be available. If the SAP WM module is not an option, an alternative solution is to interface to SAP at a more basic level, using WMMBID and DELVRY IDOCs.

The interface supports both a configuration where the stock is maintained in SAP and in LogiSoft.

- If the stock is in LogiSoft, the controlled stock is seen as one BIN. So SAP does not worry about where exact in LogiSoft the material is stored.
- If the stock is maintained by SAP, each location (Bins) are known by SAP and send with each transfer order.

The following transaction types are supported as basic functionality:

| IDOC | Message Type | Direction SAP -> LogiSoft | LogiSoft format | Description |
|---|---|---|---|---|
| MATMAS02 | MATMAS | -> | Article | Update Article |
| WMTOID02 | WMTORD | -> | Order | Receiving transfer orders from SAP. |
| WMCAID01 | WMCATO | -> | >Cancel | Cancel a transfer order. |
| WMIVID01 | WMINVE | -> | Inventory | Create a inventory order in LogiSoft (Requires the optional inventory module) |
| WMTOCO | WMTOCO | <- | Confirmation Transfer order | Send to SAP when an order line has been processed in Shark. |
| | | | Stock Adjustment | Used when stock is adjusted, stored or picked in LogiSoft without an order from SAP. |

*The version numbers may change.*

The interface can be customized, extended and modified, as needed.

The SAP Connector does not include a standard setup for handling:

- Special Stock in SAP
- Units

As alternatives to the XML IDOCs, the following methods are partly supported, but will require some customization:

- RFC (SAP Java Connector) for confirming orders. Using the RFC call L_TO_CONFIRM.

# Exchange of IDOCs from SAP

A shared folder is defined from where LogiSoft will read the IDOC XML files. LogiSoft first copy the file to a temporary folder, transform the file using an XSLT style sheet from SAP to the native XML format of LogiSoft and then import the file. Afterwards the file is deleted or moved to a log folder if successfully imported or to an error folder if the import failed.

This figure shows the flow of an IDOC from SAP and the import process performed by LogiSoft Link.

LogiSoft has also build-in support for FTP/SFTP and may poll an external FTP server for files to import.

> As an alternative to using a shared folder, SAP XI can be used in combination with the LogiSoft Link HTTP Interface.

## XSLT file for reading SAP IDOCs

The IDOC documents are read by transforming the SAP format to LogiSoft internal XML format. This is done using a XSLT stylesheet. The LogiSoft Link SAP Connector has a default XSLT stylesheet called sap-to-LogiSoft.xslt. The stylesheet is typical  modified to fit the actually requirement.

## Exchange IDOCs to SAP using XML files

IDOC files will be written to SAP, using the following procedure:

1. The IDOC will be generated in a local folder for temporary files.
2. When the file is generated, it will be moved to the destination folder, from where SAP will read the file. Normal file system folders and FTP folders are supported.
3. A copy will also be made to a log folder, where it will reside for a specified number of days (default is 14 days).
4. SAP reads the IDOC file and move or delete the file afterwards.

The file name of the IDOC file can somehow be configured and consist of one of more fields:

- A fixed text.

- The LogiSoft document number (unique).
- The data and time.
- The order type id.
- The order number.
- The file extension will always be .xml .

## IDOC Control Segment

This section represents the header of the IDoc, it contains an identifier of the IDOC, along with data concerning the sender system and the receiver system.

Fields in the Control Segment:

| SAP Element | LogiSoft | Note |
|---|---|---|
| DOCNUM | DocumentID | LogiSofts internal document number. |
| MESTYP | | Message Type |
| IDOCTYPE | | Basic document type, eg. WMTCID |
| SNDPOR | | |
| SNDPRT | | |
| SNDPRN | LOGISOFT | |
| RSVPOR | SAPVME | Port |
| RSVPRT | LS | Partner Type |
| RSVPRN | VME700 | Partner Number |

Some of these fields are hard coded in the XSLT, used for creating the IDOC and can be defined by modifying the XSLT.

## XSLT file for writing SAP IDOCs

The IDOC is generated from the standard LogiSoft XML, using a XSLT transformation file. This file can be modified, to adjust the IDOC format to the actually requirements.

## Exchange of information to SAP using RPC

To return information back to SAP, LogiSoft can as an alternative use the SAPs RPC interface (WMS-WCU).

## Login information to the SAP RPC Interface

To allow LogiSoft to logon to SAP to use RFT, the following information must be provided:

- Host (Name of the SAP Server).
- Client (Name of the SAP client, eg. an empty string).
- Language (eg. "EN").
- SystemNumber (eg. "00").
- Password
- WarehouseNumber (eg."191").

| Parameter | Description |
|---|---|
| SAP Host Name | Name of the SAP server |
| SAP client | For example "100" |

| Parameter | Description |
|---|---|
| | |
| SAP User ID | User with authorization for RFC |
| SAP System Number | eg “00” |
| Password | Password for the specified user |
| Warehouse Number | eg “191” |
| Language | eg. “EN” |

This covers the supported transaction types and IDOCs involved.

- Master Data.
- Put-Away goods.
- Picking goods.
- Cancel Transfer Orders.
- Stock Adjustments.
- Stock Report.

## Mapping SAP to LogiSoft

| SAP | LogiSoft | Notes |
|---|---|---|
| Handling Unit (HU) | Picking Box | |

# Master Data - Message MATMAS

Master data is imported by the MATMAS document.

Note that if Master Data is updated by SAP, the data will override information edited directly in LogiSoft . Especially  if any kind of replenishment data is specified, for example like quantity that fits into a box type and similar information, this data could also be modified directly in LogiSoft, these data will be overwritten and lost when updated by SAP.

## IDOC MATMAT

Segment: E1MARAM

| Element | Type | LogiSoft Name | Description |
|---|---|---|---|
| MATNR | string | Article/ArticleNumber | Unique article number defining the item, despite the name, it is a string that can contain both numbers and letters. |
| E1MAKTM/MAKTX | string | Article/Description | Description of the article |
| EAN11 | string | Article/EAN | Article EAN Code |
| MEINS | string | Article/Unit | Default unit for this article |

This is the basic information in the file, more fields can be added as needed.

Below is the section of the sap-to-LogiSoft.xsl file, that transform the the SAP IDoc to XML format used by LogiSoft.

```
<!-- ************************************ -->
<!-- Read Item Master Data -->
<!-- ************************************ -->

<xsl:template match="/MATMAS01/IDOC">
   <LogiSoft-ERP>
      <xsl:apply-templates/>
   </LogiSoft-ERP>

</xsl:template>
<xsl:template match="E1MARAM">
   <Article>
      <ArticleNumber><xsl:value-of select="MATNR"/></ArticleNumber>
      <Description><xsl:value-of select="E1MAKTM/MAKTX"/></Description>
     <EAN><xsl:value-of select="EAN11"/></EAN>
     <LocationType>Floating</LocationType>
     <Unit><xsl:value-of select="MEINS"/></Unit>
     <Packages>
        <Package>
           <Unit><xsl:value-of select="MEINS"/></Unit>
          <LocationTypes>
             <LocationType>
                <LocationName><xsl:value-of select="E1MLGNM/LETY1"/></LocationName>
             </LocationType>
          </LocationTypes>
        </Package>
     </Packages>
```

```
    </Article>
  </xsl:template>
```

# Inbound Transfer Order (Goods Reception) - Message WMTORD

SAP creates transfer orders (TOs) for the goods movements.

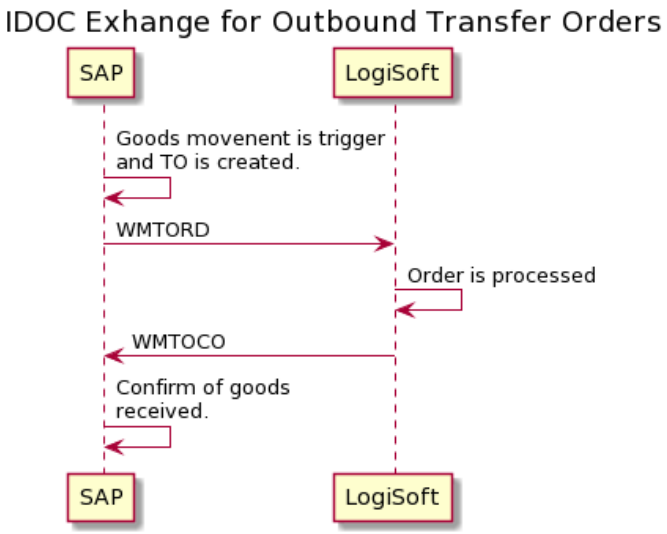
SAP transfers the transfer orders to LogiSoft system using the message type WMTORD.

LogiSoft executes the goods movements in the warehouse on the basis of the transferred data.

LogiSoft reports  back to the WMS via message type WMTOCO, that the warehouse movements for the TO have been executed.

The SAP confirms the transfer order on the basis of the data that was transferred using message type WMTOCO

Orders are received using the IDOC WMTORD.



# IDOC WMORD

WMORD IDOC contains the order from SAP. It is mapped to the LogiSoft type by an XML-XSLT transformation file.

## Segment: E1LTORH

| Element | Type | LogiSoft Name | Description |
|---|---|---|---|
| TANUM | string | OrderNumber | Transfer order number in SAP is mapped to a LogiSoft order number. |
| BWLVS | string | Order type | The movement type in SAP is called OrderTypeID in LogiSoft. The mapping can either be done by having a one-to-one mapping (BWLVS=>OrderTypeID) or the XSLT file is used to convert the SAP type to another type. |

Please note that if an undefined movement type is received. The only action taken, is to generate an error log. LogiSoft cannot see if it is a pick or put order will not generate any orders from the received document.

#### Segment: E1LTORI

Repeated for each order line in the transfer order.

| Element | Type | Type | Description |
|---|---|---|---|
| MATNR | string | Article number | Unique article number defining the item, despite the name, it is a string that can contain both numbers and letters. |
| NSOLM | float | Qty | Description of the article |
| MEINS | string | PackageSize | If defined, this must match an package size in LogiSoft . |
| TAPOS | integert | LineNumber | This is the position or line number for the order line. |
| CHARG | string | BatchNumber | Optional batch number. |

# Order line Confirmation

When a put is done in LogiSoft it is reported back to SAP using an XML IDOC alternatively an RPC call.

## RPC Call

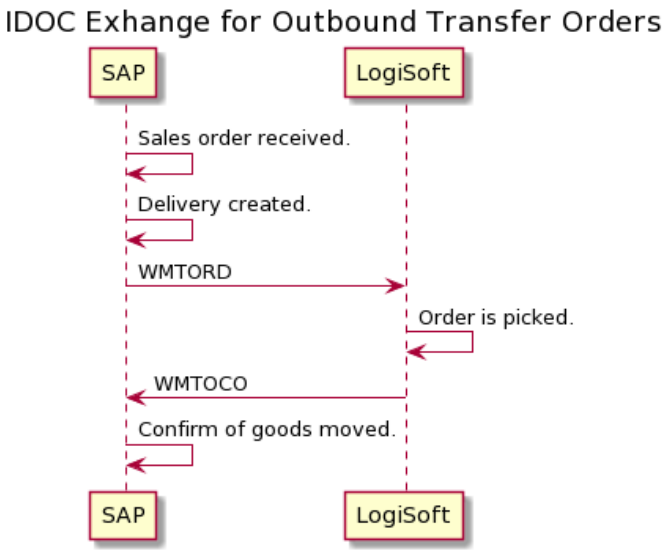
These calls can fail if a SAP user has opened the order LogiSoft is trying to confirm, in that case will SAP put a lock on the order and the confirmation will fail. LogiSoft will in this situation log a warning and try again to commit until it succeeds. Other errors will log an error in the system log and LogiSoft will not try again. Typical errors can for example be if a user as confirmed an order manually that LogiSoft is trying to confirm.

# Outbound Transfer Order (Picking) - Message WMTORD

The outbound transfer order is used for picking sales, production or other types of outbound delivery of goods.

1. SAP creates transfer orders (TOs) for the goods movements.
2. SAP transfers the transfer orders to LogiSoft system using the message type WMTORD.
3. LogiSoft executes the goods movements in the warehouse on the basis of the transferred data.
4. LogiSoft reports back to the WMS via message type WMTOCO, that the warehouse movements for the TO have been executed.
5. The SAP confirms the transfer order on the basis of the data that was transferred using message type WMTOCO

Orders are received using the IDOC WMTORD.




## IDOC WMORD

WMORD IDOC contains the order from SAP. It is mapped to the LogiSoft type by an XML-XSLT transformation file.

### Segment: E1LTORH

| Element | Type | LogiSoft name | Description |
|---|---|---|---|
| TANUM | string | OrderNumber | Transfer order number in SAP, is mapped to a LogiSoft order number. |
| BWLVS | string | OrderTypeID<br><br>O.Misc5 | The movement type in SAP is called OrderTypeID in LogiSoft . The mapping can either be done by having a one-to-one mapping (BWLVS=>OrderTypeID) or the XSLT file is used to convert the SAP type to another type. |
| TAPRI | string | Priority | Transfer Order Priority |
| LGNUM | string | - | Storage Number in SAP. |
| TRART | string | OrderType | Typical the movement type is used to identify the direction of the transfer. |

| Element | Type | LogiSoft name | Description |
|---|---|---|---|
| BETYP | string | - | Requirement type. |
| BENUM | string | - | Purchase order number |

Please note that if an undefined movement type is received. The only action taken, is to generate an error log. LogiSoft cannot see if it is a pick or put order will not generate any orders from the received document.

### Segment: E1LTORI

Repeated for each order line in the transfer order.

| Element | Type | LogiSoft Name | Description |
|---|---|---|---|
| MATNR | string | Article number | Unique article number defining the item, despite the name, it is a string that can contain both numbers and letters. |
| NSOLM | float | Qty | Description of the article |
| MEINS | string | PackageSize | Unit of measure.<br><br>If defined, this must match an package size in LogiSoft . |
| TAPOS | integert | LineNumber | This is the position or line number for the order line. |
| WERKS | string | - | Plant. Not used in LogiSoft. |
| CHARG | string | BatchNumber | Optional batch number. |
| LETYP | string | | Storage Unit Type |
| KZQUI | string | - | X: Confirmation Required.<br><br>Typical not used in LogiSoft. |
| VLTYP | string | OL.Misc2 | Source Storage Type. |
| VLBER | string | OL.Misc3 | Source Storage Section. |
| VLPLA | string | Location<br><br>OL.Misc4 | Source Storage Bin.<br><br>For WCS solution, this is the location in LogiSoft from where SAP wants to pick to goods. It must exactly match an existing location in LogiSoft. |
| VSOLM | Number | - | Source target quantity in stock keeping units. |
| NLTYP | string | OL.Misc5 | Destination Storage Type.<br>(Not relevant for LogiSoft) |
| NLBER | string | OL.Misc6 | Destination Storage Section.<br>(Not relevant for LogiSoft) |
| NLPLA | string | OL.Misc7 | Destination Storage Bin.<br>(Not relevant for LogiSoft) |
| NSOLM | Number | - | Destination Target Quantity in stock keeping units.<br>(Not relevant for LogiSoft) |
| MAKTX | string | | Material Description |

## Confirmation by RPC

A successful pick or store is reported by:

**RPC function: L_TO_CONFIRM**

| Parameter | Value | Description |
|---|---|---|
| I_LGNUM | Registry: SAP.WareHouse | Warehouse Number |
| TANUM | LogiSoft Order number | |
| TAPOS | LogiSoft Line number | |
| SQUIT=X | If no differences reported | “X” if successfully executed. |
| NDIFA | Subseq diff. qty | Difference qty |
| NISTA | Subseq. Actual qty | Actually delivered qty |
| ALTME | Unit | |
| KZDIF | | |

# Cancel a Transfer Order - Message WMCATO

SAP may transfer an order or an order line in LogiSoft using the message.

| Element | Type | Type | Description |
|---|---|---|---|
| LGNUM | string | | Storage Number. This is a fixed number from SAP , ignored by LogiSoft. |
| TANUM | string | OrderNumber | Transport Order Number. |
| CANRQ | string | | X: Cancel Request |
| TAPOS | integert | LineNumber | This is the position or line number for the order line. |

## Order line Confirmation using RPC – Message WMTOCO

When a pick is done in LogiSoft it is reported back to SAP using RPC calls.

These calls can fail if a SAP user has opened the order LogiSoft is trying to confirm, in that case will SAP put a lock on the order and the confirmation will fail. LogiSoft will in this situation log a warning and try again to commit until it succeeds. Other errors will log an error in the system log and LogiSoft will not try again. Typical errors can for example be if a user as confirmed an order manually that LogiSoft is trying to confirm.

A successful pick or put is reported by:

**RPC function: L_TO_CONFIRM**

| Parameter | Value/source | Description |
|---|---|---|
| TANUM | Order number | |
| TAPOS | Line number | |
| SQUIT=X | If no diff. reported | |

A partical pick or put is reported by:

**RPC function: L_TO_CONFIRM**

| Parameter | Value/source | Description |
|---|---|---|
| TANUM | Order number | |
| TAPOS | Line number | |
| NDIFA | Subseq diff. qty | |
| NISTA | Subseq. Actual qty | Actually delivered quantity |
| ALTME | Unit | |
| KZDIF | | Differences indicator |

## Cancel a Transfer Order in LogiSoft (RFC only)

When the order is cancelled in LogiSoft it will respond to SAP by calling L_TO_CANCEL.

**RPC function: L_TO_CANCEL**

| Parameter | Value/source | Description |
|---|---|---|
| I_TANUM | Order Number | |
| I_LGNUM | Warehouse number | |

# Reporting Stock Adjustments to SAP using RFC

When a pick/put or adjustment is done in LogiSoft, as a "Manuel Operation" (transactions without orders from a host system), it can be reported back to SAP. Because there is no transfer order in SAP that has initiated this operation, we have to create one, this is done by calling the RPC function L_TO_CREATE_SINGLE.

**If the stock is increased in LogiSoft:**

**RFC: L_TO_CREATE_SINGLE**

| Parameter | Value/source | Description |
|---|---|---|
| I_WERKS | XSLT file | Plant |
| I_LGNUM | | Warehouse number |
| I_BENUM | Logimat | Requirement number |
| I_BWLVS | XSLT file | Movement type default is 712 |
| I_MATNR | Article number from LogiSoft | SAP Material Number |
| I_VLTYP | Registry: SAP.Adjustments.SAPStorageType(1) | Storage type in SAP . Default is 999. |
| I_NLTYP | Registry: SAP.Adjustments.LogiSoftStorageType(1) | Storage type in LogiSoft. Default is 20 |
| I_LGORT | Registry: SAP.Adjustments.SAPStorageLocation(1) | Default is 0001 |
| I_ANFME | Quantity from LogiSoft | Quantity adjusted |
| I_SQUIT | X | Execute |

*(1) The registry is relative to the port used for confirmation. The typical placement is Scheduler.Jobs.DocumentExport.Parameters.Ports.OrderAck.SAP*

**If the stock is decreased in LogiSoft:**

**RFC: L_TO_CREATE_SINGLE**

| Parameter | Value/source | Description |
|---|---|---|
| I_WERKS | Plant | Plant |
| I_LGNUM | Warehouse number | Warehouse number |
| I_BENUM | Logimat | Requirement number |
| I_BWLVS | Movement type | Movement type default is 712 |
| I_MATNR | Article number | Article number from LogiSoft. SAP Material Number |
| I_VLTYP | Registry: SAP.Adjustments.SAPStorageType(1) | Storage type in SAP . Default is 999. |
| I_NLTYP | Registry: SAP.Adjustments.LogiSoftStorageType(1) | Storage type in LogiSoft. Default is 20 |
| I_ANFME | Quantity | Quantity adjusted |
| I_LGORT | Registry: SAP.Adjustments.SAPStorageLocation(1) | Default is 0001 |
| I_SQUIT | X | Execute |

*(1) The registry is relative to the port used for confirmation. The typical placement is Scheduler.Jobs.DocumentExport.Parameters.Ports.OrderAck.SAP*

To enable this function in LogiSoft, set the registry parameter:

SAP.ReportStockAdjustments=true

# Absolute Stock Reports

SAP can initiate a complete stock report using absolute qty values. A stock counting list is generated in SAP, LogiSoft reads the file and adds the correct quantities to the file and immediately returns it to SAP. This means that the actually counting in LogiSoft must be done before the SAP requests the stock.

By sending an IDOC file of type WMINVE SAP is asked to read the file back in using the RPC call: EDI_DATA_INCOMING, the call requires a "port" parameter, the file port is used to determine the file type of the input file whether is flat or XML file. If the file type is XML, then FM EDI_DATA_INCOMING will call FM IDOC_XML_FROM_FILE to process inbound IDoc file.

Procedure:

1. SAP output an XML IDOC file in the LogiSoft import folder.
2. LogiSoft reads the XML file generates a new similar file with the quantities added to the MENGA field.
3. The final file is moved to the defined "OutFolder".
4. Optional LogiSoft calls SAPs "EDI_DATA_INCOMING" with the path to the file.

**Fields in the IDOC file**

| Field in Source File | Field in returned file | Note |
|---|---|---|
| SNDPRN | RCVPRN | |
| RCVPRN | SNDPRN | |
| | MENGA | Quantity on stock |
| CHARG | CHARG | Bath number if supplied |
| MATNR | MATNR | Article Number |

The call used to notify SAP about the returned file:

**RFC: EDI_DATA_INCOMING**

| Parameter | Value/source | Description |
|---|---|---|
| PATHNAME | Path the returned LogiSoft IDOC file | |
| PORT | Registry: SAP.PortIncommingData | |

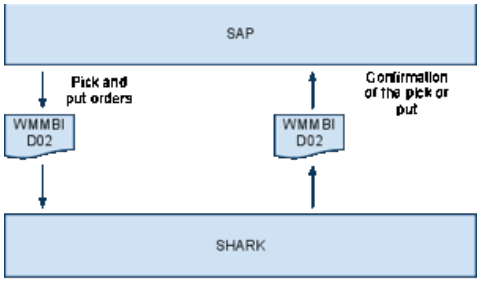
Example of IDOC WMINVE file as send by SAP:

```
<?xml version="1.0" encoding="UTF-8" ?>
<WMIVID01>
        <IDOC BEGIN="1">
                <EDI_DC40 SEGMENT="1">
                        <TABNAM>EDI_DC40</TABNAM>
                        <MANDT>300</MANDT>
                        <DOCNUM>0000000000964029</DOCNUM>
                        <DOCREL>620</DOCREL>
                        <STATUS>30</STATUS>
                        <DIRECT>1</DIRECT>
                        <OUTMOD>2</OUTMOD>
```

```
                    <IDOCTYP>WMIVID01</IDOCTYP>
                    <MESTYP>WMINVE</MESTYP>
          <          <SNDPOR>SAPR3P</SNDPOR>
                    <SNDPRT>LS</SNDPRT>
                    <SNDPRN>R3PCLNT300</SNDPRN>
                    <RCVPOR>XMLSYNC</RCVPOR>
                    <RCVPRT>LS</RCVPRT>
                    <RCVPRN>WM  SUB  001</RCVPRN>
                    <CREDAT>20061130</CREDAT>
                    <CRETIM>081644</CRETIM>
                    <SERIAL>20061130081644</SERIAL>
               </EDI  DC40>
               <E1LINVX SEGMENT="1">
                    <LGNUM>191</LGNUM>
                    <IVNUM>0000000026</IVNUM>
                    <IVPOS>0001</IVPOS>
                    <LGTYP>005</LGTYP>
                    <LGPLA>LOGIMAT</LGPLA>
                    <MATNR>000000000001000058</MATNR>
                    <WERKS>1110</WERKS>
                    <WDATU>20050320</WDATU>
                    <MENGA>0.000</MENGA>
                    <ALTME>ST</ALTME>
                    <LQNUM>0000001958</LQNUM>
                    <NANUM>00</NANUM>
                    <NVERS>00</NVERS>
                    <ISTAT>N</ISTAT>
                    <IDATU>20050614</IDATU>
                    <KZINV>ST</KZINV>
                    <MAKTX>INK ROLL FOR MARKING (LOOSE)</MAKTX>
                    <ISEIT>0000</ISEIT>
                    <VFDAT>00000000</VFDAT>
                    <LGORT>1191</LGORT>
               </E1LINVX>
....
```

# Interface without SAP WM

If the SAP installation does not use SAP WM, an alternative way to interface the 2 systems is to use the interface between LogiSoft and SAP, that can be used when not using the SAP WM module, can be based on the exchange of orders using the IDOCs WMTCID and WMTCID.




```
<xsl:template match="/WMMBID02/IDOC/E1MBXYH/E1MBXYI [BWART='312' or BWLVS='999']">
      <LogiSoft-ERP>
        <Order>
          <OrderNumber>
            <xsl:value-of select="TANUM"/>
          </OrderNumber>
          <DeliveryNote>
            <xsl:value-of select="E1LTORI/NLPLA"/> [<xsl:value-of select="BWLVS"/>]
          </DeliveryNote>
          <xsl:for-each select="E1LTORI">
            <xsl:call-template  name="PICKLINE"/>
          </xsl:for-each>
        </Order>
      </LogiSoft-ERP>
    </xsl:template>

    <xsl:template match="/WMMBID02/IDOC/E1MBXYH/E1MBXYI[(BWART='101' or BWART='311') and SHKZG='S']">
      <LogiSoft-ERP>
        <Article>
          <ArticleNumber><xsl:value-of select="MATNR"/></ArticleNumber>
        </Article>
        <PutAway>
          <OrderNumber><xsl:value-of select="EBELN"/></OrderNumber>
          <DeliveryNoteNumber><xsl:value-of select="SMBLN"/></DeliveryNoteNumber>
          <OrderLine>
            <ItemNumber><xsl:value-of select="MATNR"/></ItemNumber>
            <ItemText><xsl:value-of select="MAKTX"/></ItemText>
            <ItemCount><xsl:value-of select="BPMNG"/></ItemCount>
            <BatchNumber><xsl:value-of select="CHARG"/></BatchNumber>
          </OrderLine>
        </PutAway>
      </LogiSoft-ERP>
    </xsl:template>
```

```
<xsl:template match="/WMMBID02/IDOC/E1MBXYH/E1MBXYI[(BWART='102' or BWART='311') and SHKZG='H']">
```

## Pick and Put-Away Orders

Orders are received using the IDOC WMTCID02, Post goods receipt (goods movement), message.

LogiSoft confirms the goods movement, using the IDOC DELVRY05.

Movement types are used to control the direction of material flow.

## Picking Confirmation to SAP

Confirmation is done using the IDOC WMTCID01. LogiSoft will return this document when the order is finalized.

IDOC: WMTCID03

# Configuration

This section covers configuration issues with the SAP interface.

## LogiSoft Scheduler Setup

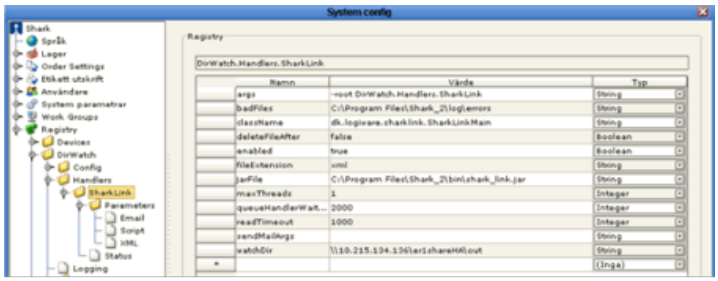
The LogiSoft Scheduler is used to setup automatic import of IDOCs and to confirm by IDOCs and RPC call to SAP.

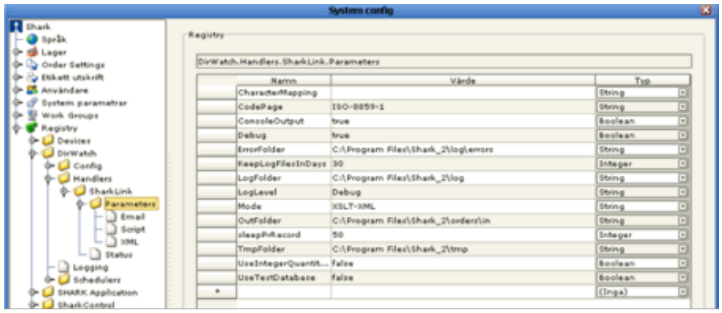
Two jobs are defined in the Scheduler:

An file importer using the class dk.logiware.LogiSoftlink.LogiSoftLinkImportSAP.

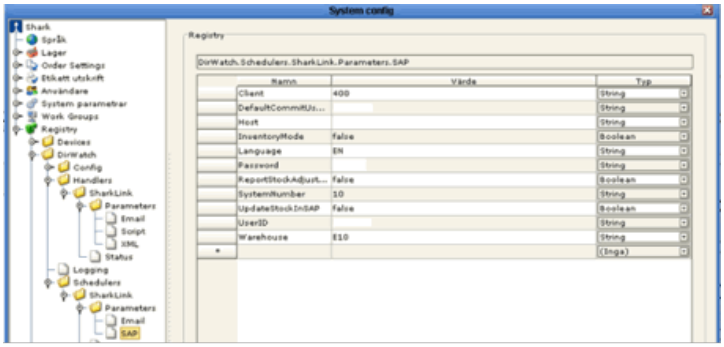
A job that confirms operations to SAP using the class: dk.logiware.LogiSoftlink.LogiSoftLinkExportSAP.

### Configuration of the Importer:



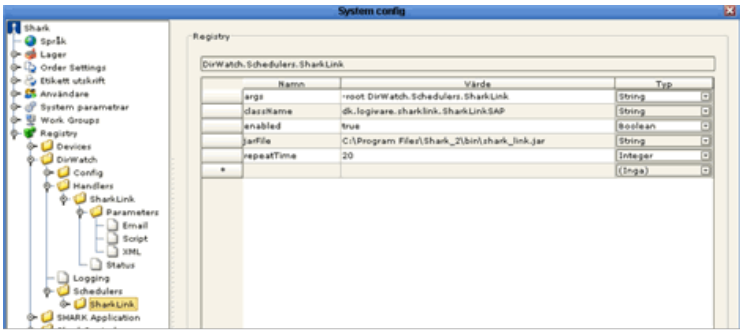



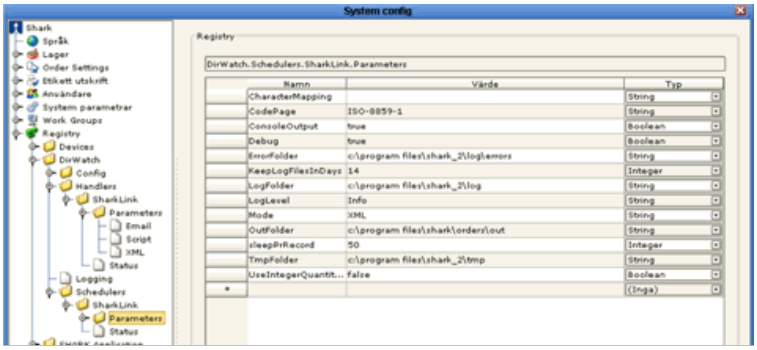

SAP specific parameters for the importer:

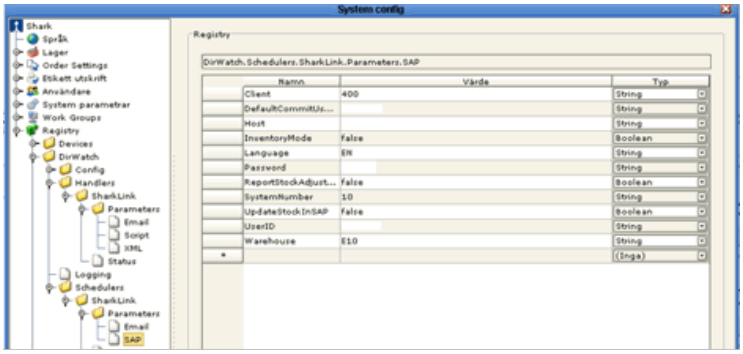



**PortIncommingData:** This is the SAP Port used in the EDI_DATA_INCOMING RPC call for reading the XML inventory file. The port must be an XML file reader entry.

Configuration of the exporter:






**DefaultCommitUser**: This is the default SAP user name used for commitment of transactions. It must be a valid SAP user or leave it empty.

**InventoryMode:** If true the inventory functionality will be enabled.

**ReportStockAdjustment:** If true stock adjustment in LogiSoft will be reported back to SAP using relative adjustment and the L_TO_CREATE_SINGLE RPC call. Not all SAP setups will accept this.

**OutFolder:** This is the SAP path to the folder where LogiSoft deliveres IDOC files for import into SAP. Default is the same path as LogiSoft see, but in case SAP is using another file system than Windows, the path may be different. LogiSoft still uses the normal OutFolder parameter when generating the file, it is only the information that is supplied to SAP.

# SAP Movement Types

SAP uses Movement Types to specify transfer orders. The translation of the Movement Types to LogiSoft order types happens in the XSLT template, default named "sap-to-LogiSoft.xslt". If customized or none supported Movement Types are used, they must be added to the XSLT file. There are two lines in the file, one for picking orders and one for put-away.

Picking orders:

```
\<xsl:template match="/WMTOID01/IDOC/E1LTORH\[BWLVS='319' or
BWLVS='601'  or BWLVS='711' or BWLVS='712' or ....
```

Put-away orders:

```
\<xsl:template match="/WMTOID01/IDOC/E1LTORH\[BWLVS='101' or BWLVS='561'
or BWLVS='501' or BWLVS='312' or ...
```

Add a new BWLVS='<movement type>' to one of those lines.

Please note that if transfer order is received from SAP with an unknown movement type, it will result in an error and the file will not be imported.

# SAP Configuration

This document does not cover configuration of the SAP application. But the following must be configured in SAP:

The Partner Profile must be defined.

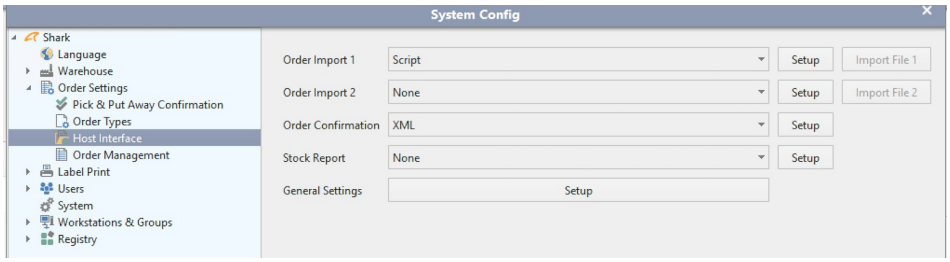
A PORT must be defined to the exchange of IDocs.

# Host Link Scripting Interface

It is possible to interface to LogiSoft using Groovy Scripting. This allows LogiSoft to interface to most systems without customizing the program it self, but it requires some knowledge in programming.

# Importing Order using the Script Importer

Activate the scripting interface from System Configuration menu.



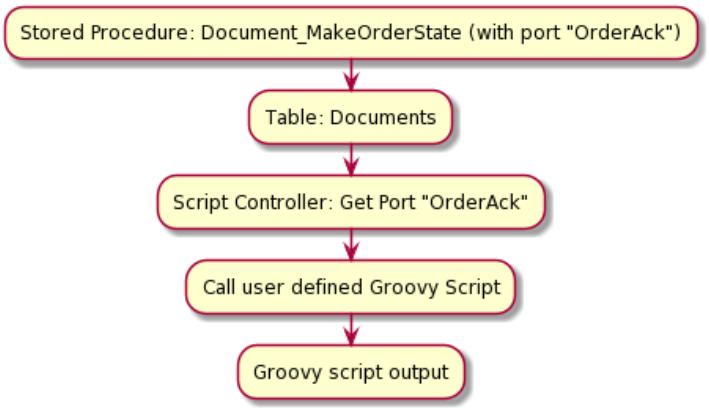
# Order Acknowledge by Scripting

To get full control of how the format of the generated acknowledge response is, a Groovy script can be used for the formatting. Since Groovy is a general script language, it can be used for file generation, web services, database access etc.

The internally LogiSoft always generates an XML data structure for acknowledgement/confirmation when a order has been fulfilled. The scripting interface allows a Groovy script to interpret the XML and do whatever is required.

The configuration can be done by using the standard menu for configuration

Information flow in LogiSoft:




Typical interfaces where scripting fits well:

- Generation of other file formats than XML (an alternative here is also to use XSLT to make the conversion).
- Interface to REST Web Services.
- Interface to SOAP Web Services.
- Integration to databases.

*Example on configuration of the script exporter*

When the script is called, the following variables are defined:

| Name | Java Type | Description |
|---|---|---|
| document | DocumentVO | This is the basic document. Use document.getXMLData() to get the XML code as a String. |
| slogger | SLogger | Logger class used for logging info, errors, etc. to the LogiSoft system log. |
| development | boolean | True if the script is running from the editor in a debug environment. |
| tmpFolder | String | Temporary folder if the script needs a working area |
| receiver | String | Name of the receiver. |

**External classes:**

- dk.logiware.sharklink.models.DocumentVO
- dk.logiware.sharklink.SLogger

**Script Example:**

```
 import java.text.SimpleDateFormat
import java.util.Date

// Folder where files are written
def exportFolder = new File('/Users/sos/tmp')

// Parse the XML document from LogiSoft Document System
def xmldoc = new XmlParser().parse(document.getXMLData())

// Format a datetime string for the filename
def dateString = new SimpleDateFormat("yyyyMMddHHmmss").format(new Date())

// Define the filename
def filename = xmldoc.OrderNumber.text() + dateString + ".csv"

// Get parameters for the file from the XML document
def ordernumber = xmldoc.OrderNumber.text()
def state = xmldoc.State.text()

// Write the file
new File(exportFolder, filename).write("$state,$ordernumber")
```

**Calling a REST or SOAP Web Service**

Libraries good for call web services:

- groovy-wslite - https://github.com/jwagenleitner/groovy-wslite. Supports both REST and SOAP, but the library is only available as standard in newer LogiSoft version.
- Groovy HTTPBuilder . See for example JavaWorld.

## Using the SLogger

The SLogger class is accessible through the instance slogger, available when the script is called. The log will go the general target for the logs, typical the database. Can be viewed in the System Log.

**Examples:**

```
 slogger.logError(String message, int errorCode, Exception exception)
slogger.logError(String message, int errorCode, Exception exception)
slogger.logError(String message, int errorCode, Exception exception, String orderNumber, String detailed)

slogger.logInfo(String message)
slogger.logInfo(String message, int errorNumber)
slogger.logInfo(String message, int errorNumber)
slogger.logInfo(String message, String detailedMessage, String order, int errorNumber)

slogger.logDebug(String message)
slogger.logDebug(String message, String detailedMessage, String order, int errorNumber)

slogger.logWarning(String message, int errorCode)
```

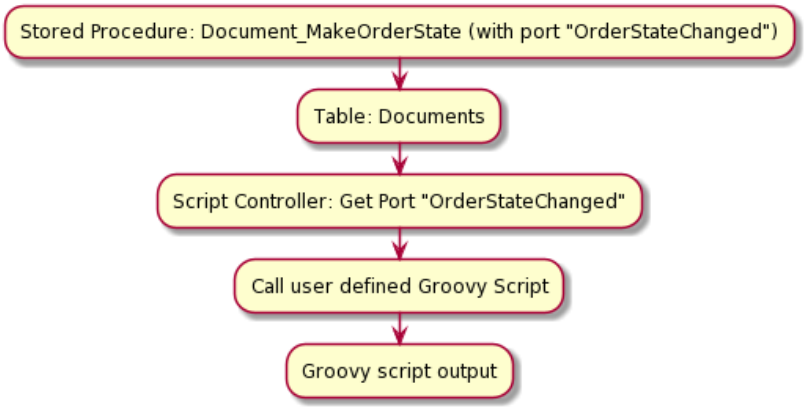
The errorCode is an integer and there is a number of predefined error codes in the ErrorCode object. If you create your own codes, it is recommended to make them in the range 4000-4999.

# Report Order State Changing by Groovy Scripting

This is only partly supported and required manually configuration.

To get full control of how the format of the generated acknowledge response is, a Groovy script can be used for the formatting. Since Groovy is a general script language, it can be used for file generation, web services, database access etc.

Information flow in LogiSoft:

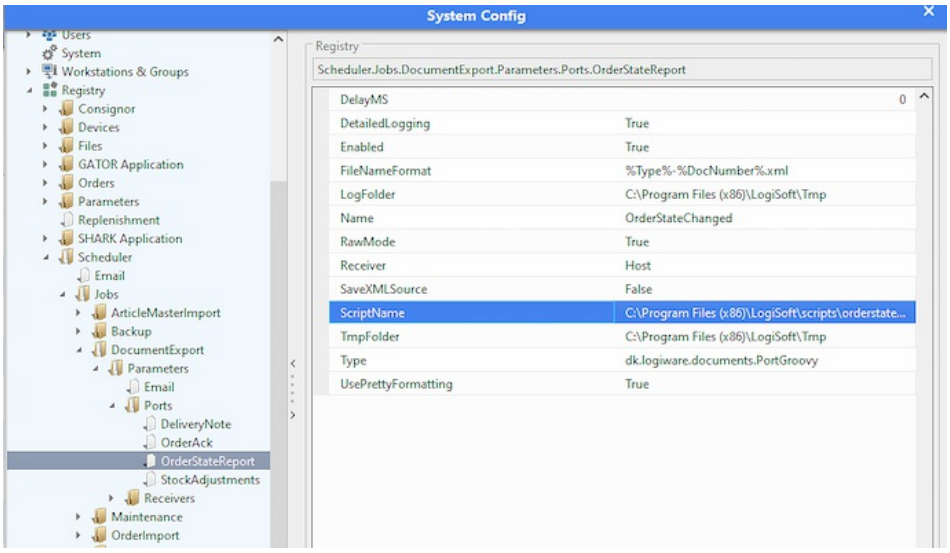



The configuration must be done in the LogiSoft Registry, there is no simple dialog for the configuration.

Define a port using the type: *dk.logiware.documents.PortGroovy*

Define a port name that matches the document to grab (Probably OrderStateChanged)

The script is defined either as file or as a script name, using the parameter: *ScriptName*




*Example on configuration of the script exporter*

When the script is called, the following variables are defined:

| Name | Java Type | Description |
|---|---|---|
| document | DocumentVO | This is the basic document. Use document.getXMLData() to get the XML code as a String. |
| slogger | SLogger | Logger class used for logging info, errors, etc. to the LogiSoft system log. |
| development | boolean | True if the script is running from the editor in a debug environment. |
| tmpFolder | String | Temporary folder if the script needs a working area |
| receiver | String | Name of the receiver. |

**External classes:**

- dk.logiware.sharklink.models.DocumentVO
- dk.logiware.sharklink.SLogger

**Script Example:**

```
 import java.text.SimpleDateFormat
import java.util.Date

// Folder where files are written
def exportFolder = new File('/Users/sos/tmp')

// Parse the XML document from LogiSoft Document System
def xmldoc = new XmlParser().parse(document.getXMLData())

// Format a datetime string for the filename
def dateString = new SimpleDateFormat("yyyyMMddHHmmss").format(new Date())

// Define the filename
def filename = xmldoc.OrderNumber.text() + dateString + ".csv"

// Get parameters for the file from the XML document
def ordernumber = xmldoc.OrderNumber.text()
def state = xmldoc.State.text()

// Write the file
new File(exportFolder, filename).write("$state,$ordernumber")
```