Infor LN Document Output Management User Guide

Release 10.7.x

Important Notices

The material contained in this publication (including any supplementary information) constitutes and contains confidential and proprietary information of Infor.

By gaining access to the attached, you acknowledge and agree that the material (including any modification, translation or adaptation of the material) and all copyright, trade secrets and all other right, title and interest therein, are the sole property of Infor and that you shall not gain right, title or interest in the material (including any modification, translation or adaptation of the material) by virtue of your review thereof other than the non-exclusive right to use the material solely in connection with and the furtherance of your license and use of software made available to your company from Infor pursuant to a separate agreement, the terms of which separate agreement shall govern your use of this material and all supplemental related materials ("Purpose").

In addition, by accessing the enclosed material, you acknowledge and agree that you are required to maintain such material in strict confidence and that your use of such material is limited to the Purpose described above. Although Infor has taken due care to ensure that the material included in this publication is accurate and complete, Infor cannot warrant that the information contained in this publication is complete, does not contain typographical or other errors, or will meet your specific requirements. As such, Infor does not assume and hereby disclaims all liability, consequential or otherwise, for any loss or damage to any person or entity which is caused by or relates to errors or omissions in this publication (including any supplementary information), whether such errors or omissions result from negligence, accident or any other cause.

Without limitation, U.S. export control laws and other applicable export and import laws govern your use of this material and you will neither export or re-export, directly or indirectly, this material nor any related materials or supplemental information in violation of such laws, or use such materials for any purpose prohibited by such laws.

Trademark Acknowledgements

Publication Information

Release: Infor LN 10.7.x
Publication Date: June 29, 2021
Document code: ln_10.7.x_lnttdomug_op_en-us

Contents

About this guide

This guide covers the basic features of Infor LN Document Output Management (DOM) and how this can improve the way your organization delivers documents. You will be guided through the process of sending your first document. The more advanced topics, where you can send documents to your own document management system are also explained.

Intended audience

This guide is intended to assist users and system administrators of Infor LN who want to use Document Output Management to send the report output to their preferred destination.

Related documents

You can find the documents in the product documentation section of the Infor Support Portal, as described in "Contacting Infor".

Contacting Infor

If you have questions about Infor products, go to Infor Concierge at https://concierge.infor.com/ and create a support incident.

The latest documentation is available from docs.infor.com or from the Infor Support Portal. To access documentation on the Infor Support Portal, select **Search > Browse Documentation**. We recommend that you check this portal periodically for updated documentation.

If you have comments about Infor documentation, contact documentation@infor.com.

Chapter 1: Introduction

With Document Output Management (DOM) you can perform these actions:

- Automatically provide each LN report with the corporate identity of your own company.
- Send reports through email to your business partners.
- Store reports in Infor Document Management (IDM) or your own document management system.
- Print reports to one or more printers.

By printing to the DOM device the system automatically detects which document must be sent, for example, an Invoice or Order Acknowledgement. Variables are used on the report, the data values, such as customer name or invoice number. These values are detected from the document, which can be used in a personalized email.

Additionally, a copy of the document can be sent to the printer, and stored in a document management system. The destination that is chosen for a document can be configured in the system.

Each document can have attachments, such as Terms & Conditions. Even documents that were sent earlier can be attached to a new document, for example a Reminder can have the original Invoice attached to it.

System requirements

For printing you must connect to a printer which supports direct printing of PDF files. The printer must be able to handle a byte stream with PDF data which will be sent from LN.

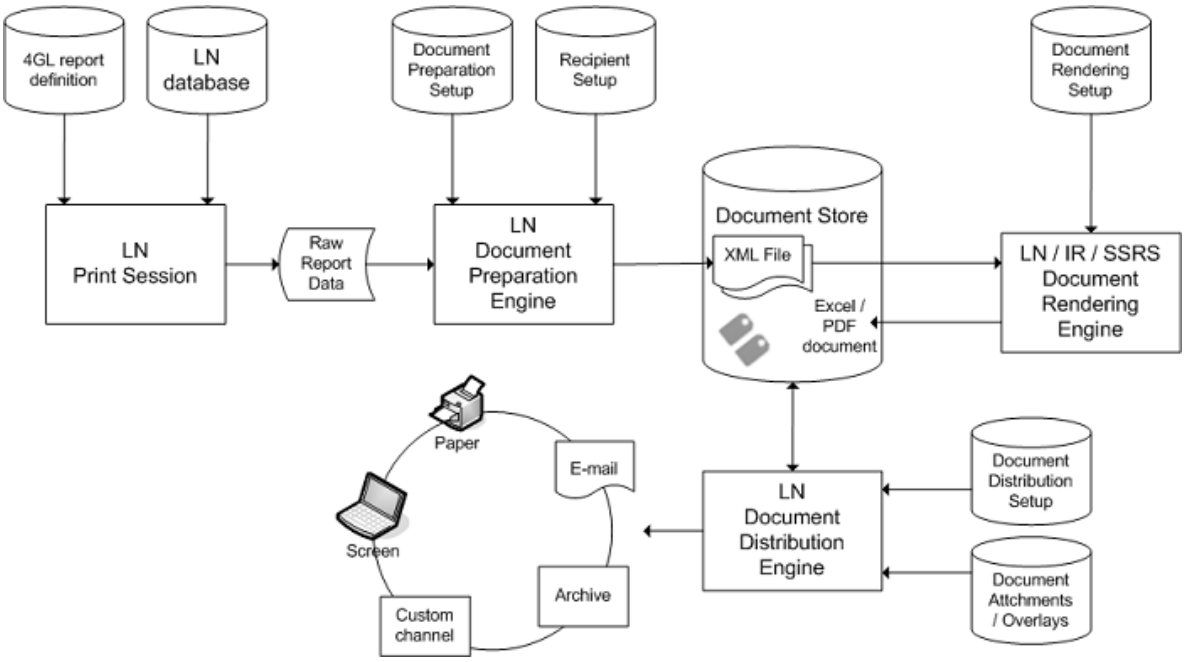
License requirement

Infor LN Document Output Management, with license product ID 7130, is included in the LN package. There is no additional charge for customers who want to use it. To use DOM, ensure that this license ID (7130) is registered in the SLM server and that the server was activated after this ID was added.

Chapter 2: Overview

Document Output Management is an easy and very flexible solution to send the documents, where your customers define how they want to receive their documents.

This diagram shows the Document Output Management architecture:



This table shows the main components in the diagram:

Component	Description
LN Print Session	The LN Print Session represents any LN session which collects data from an LN database and produces a report output (raw report data). Document Output Management is designed in such a way, that these LN sessions do not need any modification to support Document Output Management.
Document Preparation Engine	The LN Document Preparation Engine generates a batch of documents with routing information. These documents are stored as raw document in the Document Store.

Component	Description
Document Store	The Document Store holds all the documents which were generated by the LN Document Preparation Engine. From the Document Store these documents can be viewed including attachments and overlays. If a document is rendered by the LN Document Rendering Engine, the final document is stored in the Document Store, where it can be distributed by the LN Distribution Engine. It is also possible to repeat a distribution for an already distributed document from the Document Store.
Document Rendering Engine	The LN Document Rendering Engine takes the raw document from the Document Store and processes this to a document in PDF or Excel format. The reporting solutions which can be used by the LN Document Rendering Engine are: Infor Reporting, Microsoft SSRS Reporting and Infor LN's own PDF / Excel creation.
Document Distribution Engine	The LN Document Distribution Engine retrieves the document from the Document Store. Applies overlays to it, adds additional attachments, and distributes it to the preferred destination. This destination can be: e-mail, printer, Infor Document Management, or a customer defined destination, or a combination of those destinations.

Terms and definitions

This document uses specific terminologies and definitions.

Document
A document is a file produced by an LN print session or a file provided by an external document provider. Each document is stored in the Document Store from where it can be used for further processing

Batch
A batch is a group of documents produced by one print job.

Document type
A document type describes how a document of a specific category, such as Sales Order Acknowledgement or Invoice, must look when it is distributed to the receiver. A document type contains the email subject and email body text, the overlay page, extra attachments that must be added to the document, and so on.

You can use variables in the parts of the document type. Those variables are replaced by real report values at the moment the document is distributed.

Report rule

A report rule is used to define the report-specific settings, such as how to split up the report in several documents and which report server or report design should be used to render the report. Also the mapping between the report fields and the document type variable is defined in the report rule. You can define multiple Report Rules for one report. Based on the outcome of an expression, one of the report rules is selected. If no expression is valid, the default report rule for that specific report is selected.

A report rule links a report to a specific document type.

Receiver type

A receiver is a person or organization who receives the document. Several types are supported:

- Business Partner
- Contact
- Employee
- User
- Custom

Destination type

A destination is the place where to send the documents to. Several types are supported:

- E-mail
- Print
- Custom
- Fax

Document provider

A document provider provides, documents to be distributed with Document Output Management. The documents are expected to be in a rendered format.

Document provider fields

The document provider specifies the document provider fields with data at the time a document is provided to Document Output Management.

Document provider contexts

A document provider can define zero or more contexts. A context can be used in Document Output Management to differentiate between Document Type Parts to be used when distributing a document. The document provider specifies the context at the time a document is provided to Document Output Management.

Document provider rules

Document provider rules can be defined for a document provider.

The mapping between the provider fields and the document type variable is defined in the document provider rule. You can define multiple Document Provider Rules for one document provider. Based on the outcome of an expression, one of the rules is selected. If no expression is valid, the default document provider rule for that specific document provider is selected.

A document provider rule links a document provider to a specific document type.

Document authorization

Users are only allowed to render, distribute, or view documents from the Document Store if they have print authorization for the LN session that produces the document.

Users are not allowed to perform any action on a document if they do not have company authorization for the company in which the document was created.

If a user is not authorized to perform an action, the action is disabled or a message is displayed.

Receiver independent destinations

Usually the receiver type of the document type determines the destination(s) of the document. For example, the document must be sent to the e-mail address that is specified in the business partner data. Besides that, you can also define one or more destinations, which are receiver independent. You want, for example, also store the documents on your file system. You can specify these destinations on the **Receiver Independent Destinations** tab in the **Document Types (ttrpi2510m000)** session. Here you can specify the destination type (Email, Print, or Custom) and the destination of the documents.

Rendering and distributing documents

Once available in the Document Store, raw documents can be rendered to a document in PDF or Excel format and distributed to the destinations. Rendering and distributing of documents is handled by the **Document Rendering and Distribution (ttrpi3210m000)** session.

This session keeps track of the status of each document and the document sequence. According to the status of a document, the document is rendered and distributed: the document is rendered to a PDF or Excel document, or merged with overlays and attachments and distributed to one or more destinations. Rendering and distribution of documents can be spread over multiple servers (bshells).

This session ensures that a main document is printed together with its subdocuments, and no other documents are printed in between to the same printer.

This session can run in two different modes:

- Run as daemon
- Run for a selection of batches or documents

This session typically runs as daemon in an LN job, started by a system administrator. In that situation, the session continuously, after a configurable number of seconds of waiting time, searches for new documents to be rendered or distributed.

To prevent that the job runs continuously, you can specify a time-out. The job is ended when this time-out passes.

You can also start the session through the Command Line Interface.

See Command Line Interface document rendering and distribution on page 77

Viewing documents

Once available in the Document Store, you can view the documents. Select a document and choose the action **View Document**. This will view the document including overlays. Besides this, there are two options to preview documents.

Previewing documents

Usually, documents are rendered and distributed when choosing the Document Output Management device. Afterwards, you can view the distributed document. To preview the document, before it is distributed to the receiver, set the **Use During Preview** field to **Yes** in the **Document Type Details** session.

When printing the document to the Preview device, a 'Preview' batch will be created. When the preview looks fine, the preview batch can be converted to a real batch to be distributed in the normal way. Select the action **Convert Preview Batch to Distributable Batch** in the **Batches (ttrpi3500m000)** session.

Easy overlaying

You can preview a document with overlays without setting up the distribution for the document type. This is called easy overlaying and enables you to preview your document with your house style in an easy way.

You must create a document type that has the **Split and Distribute Documents** field set to **No**.

When you print the document to the Preview device, the document is stored in the Document Store, but is not distributed to any destination. Once the document is available in the Document Store, you can preview it including overlays and, if desired, send it to a destination using the **Instant Distribution** functionality.

Instant distribution

You can send a document directly to a customer or printer without the intervention of the Rendering and Distribution Engine. This is called instant distribution. To do this, select a document and select the 'View Document' action. The Document Viewer starts. Right-click the document and select one of these actions:

- **Send to other Printer Destination**. Specify a device and click **Send**.
- **Send to other Email Destination**. Specify a valid e-mail address, modify the e-mail body or select a predefined e-mail body, fill the document variables, and click **Send**.
- **Send to other FAX Destination**. Specify a valid Fax number and click **Send**.
- **Send to other Custom Destination**. Specify Type and address and click **Send**.

Restarting failed documents

Sometimes rendering of a document fails because of an outage of the report server or the distribution of a document fails because of a connection failure to the mail server. In those cases the indicator "Failures Present" will be set to true for that document, and further actions will be blocked.

When the problem is solved, the status of the documents can be reset and the documents can be rendered and/or distributed again. You can do this by selecting the document and choosing the **Release Failures for Retry** action. After this action the document will be selected automatically by the rendering and distribution process. Resetting the status for a range of batches or documents can be done by session **Release Failures for Retry (ttrpi3215m000)**. For more information, see the session help of this session.

Repeating and canceling distribution

Once a document is distributed, you can repeat this distribution action. Go to the details of a document, select a distribution or destination and choose the action **Repeat Distributions**.

If a document is not yet distributed, its distribution can be cancelled. And if a distribution was cancelled, the cancellation can be undone.

Cleanup document store

Use the **Cleanup Batches (ttrpi3200m000)** session to remove redundant batches and documents from the document store.

Chapter 3: Getting started

To get you up and running with Document Output Management, some of the basic features are covered and you are guided through sending your first DOM document.

You learn more about advanced features using the online help, or you can continue with Advanced Options on page 24.

As an example the Request for Quotation report is sent to an email recipient.

Note: A list of predefined document types is available. Administrators can import these document types and modify the documents as required.

See Predefined document types on page 78.

Setting up email

Before you can use email facilities from LN, you must complete these steps:

1. Start the **Service Providers (ttcmf0110m000)** session.
2. Check whether the "SMTP" provider exists.
3. If not, create a service provider with these properties:
   - **Provider**: SMTP
   - **Description**: SMTP
   - **4GL Connector**: ttcmfsmtp
4. On the **Specific** menu, select **Provider Parameters (ttcmf0120m000)**.

   Four parameters are generated. Ask your system administrator how to specify these company-specific parameters.
5. Start the **Services (ttcmf0130m000)** session.
6. If not available, create a service with these properties:
   - **Service Name**: SMTP
   - **Description**: SMTP Service
   - **Provider**: SMTP
   - **Enabled**: Yes
   - **Logging**: Yes
   - **Message Storage Path**: `${BSE}/tmp/outbox`
   - **File Type**: Plain Text ASCII

7 Start the **Address Types by Service (ttcmf0140m000)** session.
8 If not available, create an Address Type "SMTP" with these properties:
   - **UI Required**: No
   - **Resolve Capability**: Yes
   - **Service Name**: SMTP
   - **Paper**: A4
9 Specify a valid email address (Email Type: SMTP) for the user(s) in the **User Data (ttaad2500m000)** session.
10 Select **eMessage Connection** in the **Parameters (ttrpi2100s000)** session.

Setting up fax

Before you can use fax facilities from Infor LN, complete these steps:

1 Start the **Parameters (ttrpi2100s000)** session.
2 Select one of these Fax Solution options:

Option	Description
Email Faxing	Faxes are sent as 'normal' e-mail as defined earlier. Specify a correct e-mail address used for faxing. This address can contain variable values, depending on the used fax solution provider. See Faxing on page 76.
Custom Library	You can use an own implementation of a fax solution.

Setting up a simple document type

First step is to create a document type. A document type is used to store the e-mail subject and e-mail body text, the overlay page and extra attachments which must be added to the document. As an example a document type will be created for the Request for Quotation report. Two keywords are used to be able to identify the documents in the Document Store.

To create a document type for the Request for Quotation report:

1 Start the **Document Types (ttrpi2510m000)** session.
2 Create a new Document Type with these properties:
   - **Document Type**: RFQ
   - **Description**: Request for Quotation
   - **Split and Distribute Documents**: Yes
   - **Receiver Type**: User

Note: This distributes the documents to the current user running the Request for Quotation report. This is not the Receiver Type you will use in the final setup. It is used to see the first results of your document type setup.

- Keyword 1, Description: RFQ
- Keyword 1, Zoom Session: tdpur1501m000
- Keyword 1, Zoom Return Field: tdpur105.qono
- Keyword 2, Description: BP
- Keyword 2, Zoom Session: tccom4500m000
- Keyword 2, Zoom Return Field: tccom100.bpid

Save this Document Type record.

3 Go to the **Parts** tab to create a new Part.

This Part will be used as e-mail subject. Specify these properties:

- **Description**: Subject
- **Destination Type**: Email
- **Part Type**: Email Subject
- **Simple Content**: Request for Quotation #rfq#

Variables are placed between hashes (#) and will be replaced at runtime.

Save this record.

4 In the **Parts** tab, create a new Part.

This Part will be used to upload the body text file. Specify these properties:

- **Description**: Body
- **Destination Type**: Email
- **Part Type**: Email Body

Save this record.

5 On your local system create a text file with this content:

```
Dear partner,

Please find attached our Request for Quotation #rfq#.
Our standard Terms & Conditions document is attached as well.

When returning please state #bp#/#rfq#.

With kind regards,

#purchaseoffice#
```

6 Select the action **Upload File** from the just created Document Type Part. Upload and add this text file to the Document Type Part.

Note:

- The **Variables** tab contains the three variables, which were used in the text file. These variables will be used later, where they will be mapped to report input fields.
- You can create an html file instead of a text file. This html file can contain the same variables and some pictures and logos.

7 On the **Parts** tab create a new Part.

This Part will be used as filename of the attached document. Specify these properties:

- **Description**: Filename
- **Destination Type**: All
- **Part Type**: Filename
- **Simple Content**: RFQ_#rfq#.pdf

Save this record.

8 On the **Mail From Addresses** tab add an Address.

This Address will be used as From Address of the e-mail. Specify these properties:

- **Mail From Address**: noreply@<your.domain>

  Replace <your.domain> with your own company domain, such as infor.com.
- **Mail From Name**: Your Company (or any other speaking name for your company)
- **Is Default**: Yes

Save this record.

Setting up a report rule

Create a report rule to define the report specific settings and map variables and keywords to report input fields.

To create a report rule for the Request for Quotation report:

1 Start the **Report Rules (ttrpi2520m000)** session.

2 Add a new Report Rule for the Request for Quotation report with these properties:

- **Report**: tdpur140101002
- **Is Default**: Yes
- **Document Type**: RFQ
- **Report Server**: <empty>
- **Render Format**: PDF

Save this record.

Note that on the **Variables** tab, a number of variables are created with a blank expression.

3 Specify the expressions or browse and select the correct report input field. This table shows the expressions:

Expression	Specification
bp	tdpur105.otbp
keyword1	tdpur105.qono
keyword2	tdpur105.otbp
keyword3	""

Expression	Specification
keyword4	""
purchaseoffice	tdpur012.dsca
receiverfield	logname$
rfq	tdpur105.qono
splitfield	tdpur105.qono & tdpur105.otbp
font.scaling	"9"
show.total.number.of.pages	"No"

4 Click **Validate** to validate the settings for this Report Rule.

Check if the Report Rule is validated and compiled successfully.

Setting up receiver type User

When you specified to use receiver type User, you must define the e-mail addresses, and whether the document must be (blind) copied to other e-mail addresses.

Complete these steps:

1 Start the **User Document Type Settings (ttrpi2551m100)** session.

2 Add a new record with these properties:

- **Specific Document Type**: Yes
- **Specific User**: No
- **Document Type**: RFQ

Save this record.

3 Double-click the just created record. The **User Document Type Settings (ttrpi2551m000)** session is displayed.

4 Add a new record with these properties:

- **Destination Type**: Email
- **From Address Book**: Yes

Save this record.

Ensure that your current user's e-mail address has been filled in the address book.

5 Optionally, double-click the just created record. The **User Document Type Destinations (ttrpi2552s000)** session is displayed.

In this session, you can add extra e-mail addresses to send (blind) copies of the document.

Printing and sending a document

After you completed all the steps mentioned earlier, you can print the document and send it to the email recipient.

Complete these steps:

1. Start the **Request for Quotation (tdpur1501m000)** session.
2. Select an already existing request, or create a new one.
3. Select the option to **Print Request for Quotations**.

   The **Print Request for Quotations (tdpur1401m000)** session is displayed.
4. On the **Device** tab, select the Document Output Management device and click **Print**.

   The Request for Quotation is printed and a document is created in the Document Store.
5. To distribute the document, start the **Document Rendering and Distribution (ttrpi3210m000)** session. Specify this information:

   Run as Daemon
   Select this check box.

   Daemon Time-out (min)
   Specify `0` to run the daemon continuously.
   Specify a value greater than 0 to run the daemon for ‘x’ minutes.

   Number of Additional Servers
   Specify `0`.

   Query Wait Time
   Specify `10`.
6. Click **Continue**. A document in PDF format is produced and distributed to the email recipient.

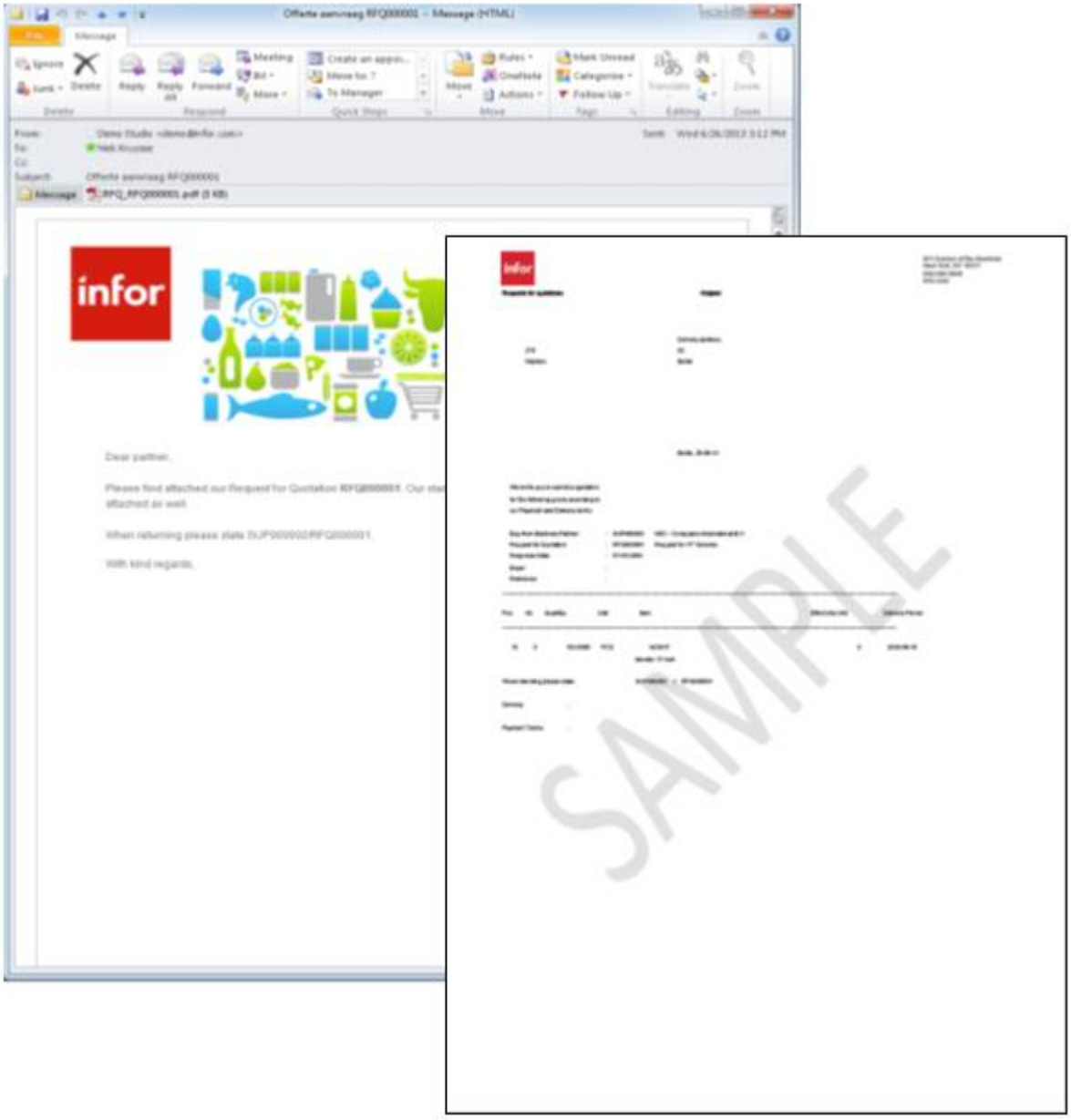
   Usually this Document Rendering and Distribution session is running in an LN Job. It continuously searches for new documents in the Document Store, which are ready to be rendered and distributed. You can use multiple additional servers to spread the work over multiple bshells.
7. Start the **Batches (ttrpi3500m000)** session to view the status of the just created batch and its document.

Using overlays

Overlays are used to place additional information on a document. Overlays are PDF files, which are merged with the document before the document is sent. For example, to ensure that the word COPY is shown on the document if the document is sent to a local printer. If the document is sent to a customer, it must contain your corporate identity.

You can define multiple overlays for one document type. Based on the outcome of an expression, the overlay is selected. All overlays with a valid expression are merged to the document.

This screenshot shows an example:



Adding overlays

To add an overlay page to your document type:

1 Create a one page PDF file containing your company house style, and store it on your local PC.
Note: The orientation of the overlay must match the orientation of the report.

2 Start the **Document Types (ttrpi2510m000)** session, and select and open the RFQ document type.
3 On the **Parts** tab, create a new Part with these properties:
   - **Description**: Housestyle
   - **Destination Type**: All
   - **Part Type**: Overlay

   Save this record
4 Choose the **Upload File** action.
5 Select the newly created PDF file with your corporate house style and click **Open**.
6 Print and send the document and view the result.

Using attachments

You can add attachments to the document type. These attachments can be merged with the document or sent as separate file.

For example, you can add a price list to your document, which will be merged with the document. An extra terms and conditions document can be added to the document as separate file.

You can define multiple attachments for one document type. Based on the outcome of an expression, the attachment will be selected. All attachments with a valid expression will be merged to the document or added as separate file.

To add an attachment to your document type:

1 Create a PDF file containing terms and conditions, and store it on your local PC.
2 Start the **Document Types (ttrpi2510m000)** session, and select and open the RFQ document type.
3 On the **Parts** tab, create a new Part with these properties:
   - **Description**: Terms and Conditions
   - **Destination Type**: Email
   - **Part Type**: File to Attach
   - **Relative Position**: After
   - **Order**: 0
   - **Separate File**: No

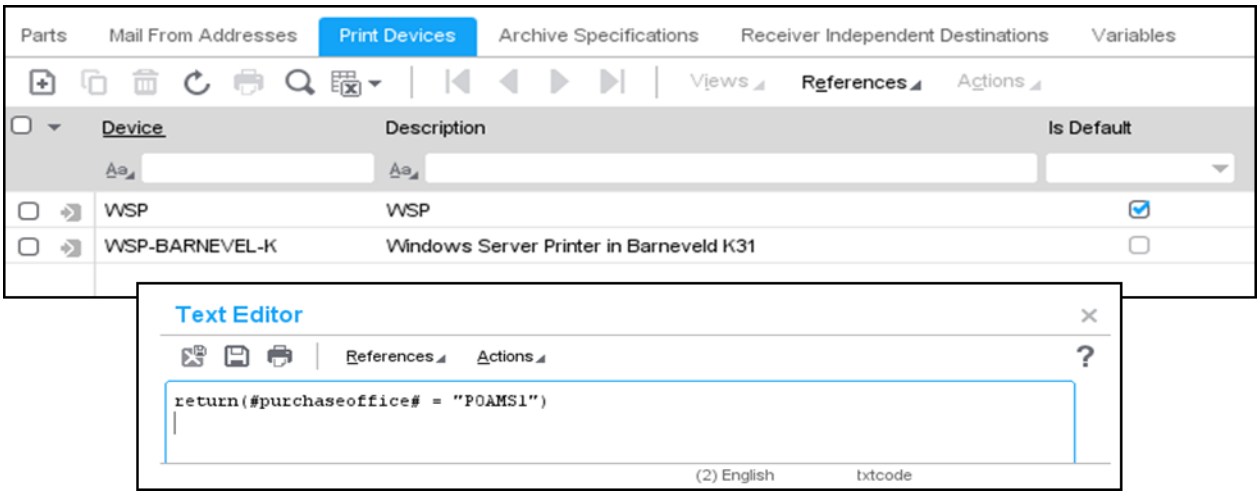
   Save this record
4 Choose the **Upload File** action.
5 Select the just created PDF file with the terms and conditions and click **Open**.
6 Print and send the document and view the result.

Using print devices

You can add Print devices to the document type to send the document to a print device.

For example, if you want to print the document to a specific device for one purchase office. All other documents are printed on a default device.

The user can specify multiple print devices for one document type. The print device is selected based on the outcome of an expression. If no valid expression applies, the document is printed on the default device.



To test if you are printing to the correct printer, add the destination type Print in the **User Document Type Settings (ttrpi2551m100**) session.

Setting up receiver type Business Partner

Until now you have used receiver type User and receiver field `logname$`.

See Setting up a report rule on page 17

In that case you were able to send the e-mail to the current LN user. If you want to send your request for quotation you want to send it to the contact or business partner linked to that request.

To change the receiver type of the document type:

1. Start the **Document Types (ttrpi2510m000)** session.
2. Select and open the RFQ document type.
3. Change the receiver type to Business Partner, and save the record.
4. Click **Validate**.
5. Start the **Report Rules (ttrpi2520m000)** session.
6. Select and open the report rule for report `tdpur140101002`.
7. On the **Variables** tab, change the Expression of the receiverfield variable to: `tdpur105.otbp`

8 Save the record, and click **Validate**.

The Business Partner data must be extended, to specify the destination (e-mail, print or custom) per business partner for each document type.

9 Start the **Recipient Sets (tccom6140m000)** session, extend the business partner data.

10 Add a new Recipient Set for Recipient Type Business Partner and import all business partners or a selection into the recipient set.

For more information, see the session help of this session.

11 Start the **Document Output Management Details (tccom6170m000)** session.

12 Add a new record with these properties:

- **Recipient Type**: Business Partner
- **Recipient Set**: <recipient set>

  Specify the just created recipient set.
- **Document Type**: RFQ
- **Destination Type**: Mail
- **Mail from Master Data**: Yes

13 Save the record.

For more information, see the session help of this session.

After these steps, the document can be sent to the business partner, which is linked to the request for quotation.

See Printing and sending a document on page 19.

Chapter 4: Advanced options

By now, you have learned some of the basic features of Document Output Management and you have created and sent your first Document Output Management document.

Document Output Management also contains advanced features and several options to enhance the reports and documents. For example, you can use condition expressions to use different reports or documents.

Using language dependent distributions

For each document type part, language dependent variants can be defined. In this way you send your document with an e-mail body and e-mail subject in the French language to your customers in France. Customers in the USA receive their documents and e-mails in the English language.

The language that is chosen for the document type parts is the same as the language in which the Infor LN report is printed.

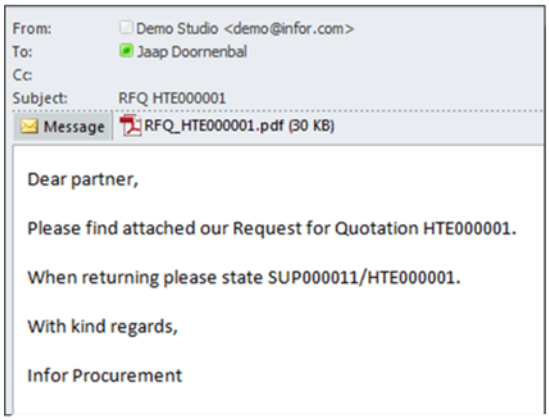
Preparation

- Use the RFQ document type, created earlier.
- You will send an e-mail in the French language to the customers who are using the French language (system language = 4). All other customers receive the default English e-mail.

To add a language dependent variant to your document part:

1. Start the **Document Types (ttrpi2510m000)** session, and select and open the RFQ document type.
2. On the **Parts** tab, select the Email Subject part and go to its details.
3. On the **Language Specific Parts** tab, create a new record with these properties:
   - **Language**: 4
   - **Simple Content**: Demande de devis #rfq#
4. Save this record.
5. Return to the **Document Type Parts**, select the Email Body and open its details.
6. On the **Language Specific Parts** tab, create a new record for language 4, save the record, and upload a text file or html file containing a French e-mail body text.

   This screenshot shows an example:



Additional information

To get the language in which the document is printed, you can use the `dom.document.language` variable in expressions linked to the Document Type.

Using configurable designs

You can use a different report design if the Sales Contract is sent to the US government. Sales Contracts sent to other business partners use the default report design.

Preparation

- You want to create a document type and report rule for the Sales Contract Acknowledgement report. You can use Infor Reporting to create two different report designs. The default report design is saved as 'report'. The specific report design is saved as `dd250`. Add a customer defined field, which indicates whether the Business Partner is the US government, to the Business Partner (`tccom110`) table.
- You will add the two different report designs to the report rule. The specific report design will be used to render the report depending on the outcome of the expression.

Complete these steps:

1 Start the **Report Rules (ttrpi2520m000)** session. Select the Sales Contract Acknowledgement report (tdsls340501000) and open its details.

2 On the **Design Codes** tab, create a new design code called `dd250` with this expression:

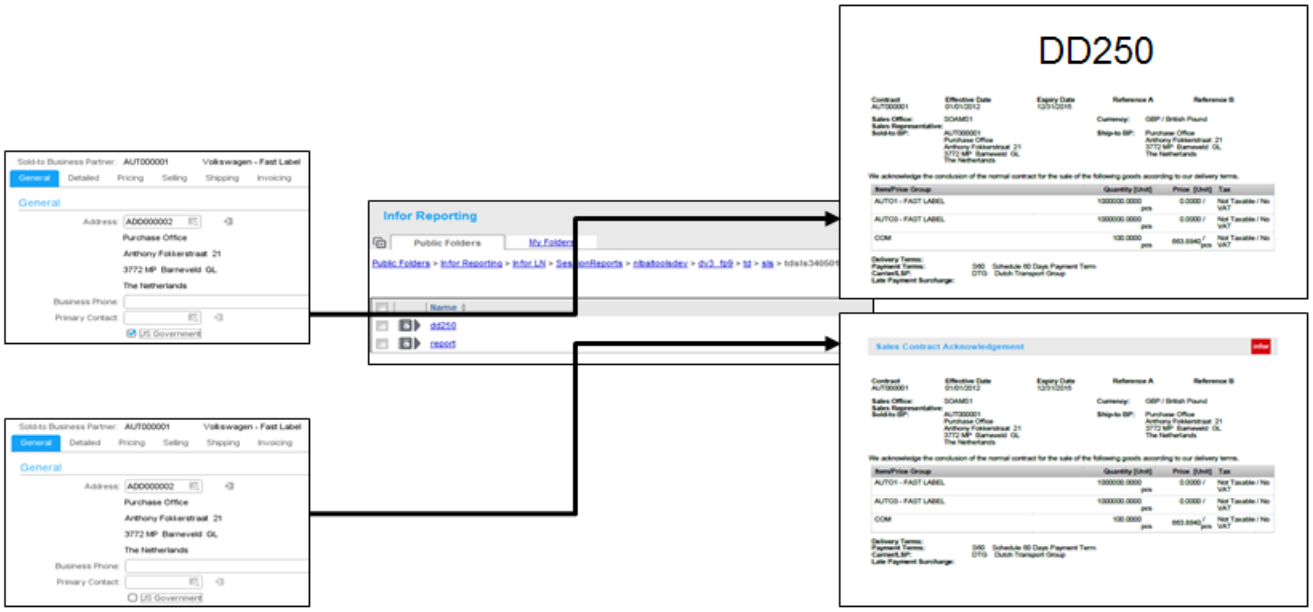
```
table ttccom110

select tccom110.cdf_gov
from tccom110
where tccom110.ofbp = :tdsls300.ofbp
as set with 1 rows
selectdo
         if tccom110.cdf_gov = tccdf_____chk.yes then
              return(true)
        endif
endselect
return(false)
```

3 Save this record.

4 Click **Validate** to validate the report rule.

5 Print the Sales Contract Acknowledgement report for the US Government to the Document Output Management device and view the result.

This screenshot shows what the result will be based on the setting of the US Government field:



Additional information

Different report designs can be used, based on the outcome of a condition.

In the expression linked to the report design you can use report fields and customer defined fields.

You can also use configurable designs for different countries or languages if not everything can be handled by labels.

Using conditions

You can use different rendering options for one report and other overlay for different companies.

Using different rendering options for one report

You can use different rendering options for your report. For example, you can use LN native rendering if RFQ line texts should not be printed on your report. In other cases, you can render your report using Infor Reporting.

Preparation

- Use the RFQ document type, created earlier.
- You will define two different report rules for the Request for Quotation report (tdpur140101002). The default report rule was already created and must be modified to use Infor Reporting.

Complete these steps:

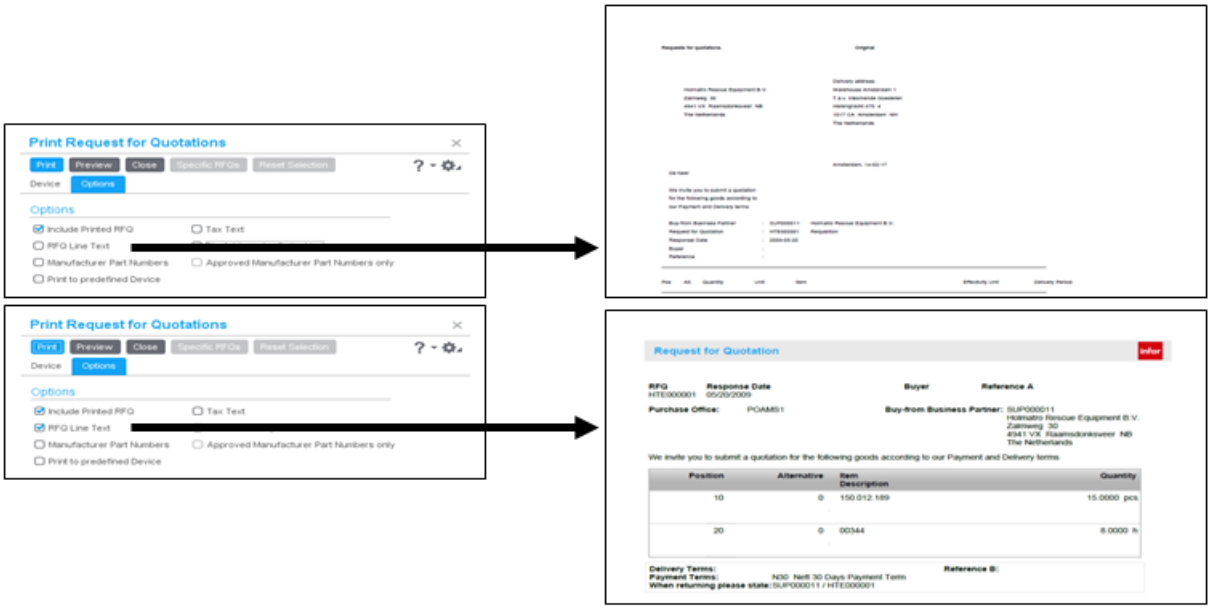
1. Start the **Report Rules (ttrpi2520m000)** session. Select report tdpur140101002, and open its details.
2. In the **Report Server** field select a report server from the available report servers and save this record.
3. Return to the **Report Rules (ttrpi2520m000)** session and add a new rule with these properties:
   - **Report**: tdpur140101002
   - **Is Default**: No
   - **Document Type**: RFQ
   - **Report Server**: <empty>
   - **Render Format**: PDF
4. Add this code to the **Condition Expression** field:

```
domain tcyesno line.text
get.var(dom.batch.session.pid, "prnt.line.txt", line.text)
if line.text = tcyesno.no then
       return(true)
endif
return(false)
```

5. Save this record.

   Note: On the **Variables** tab, nine variables are created. These variables must get a valid expression.
6. Click **Validate** to validate the report rule
7. Run the **Request for Quotations** session without Line Texts and with Line Texts. View the results.

   This screenshot shows the result based on the content of the **prnt.line.text** field on the form:



Additional information

Different report rules can be used, based on the outcome of a condition.

Note: In the expression the predefined variable `dom.batch.session.pid` was used. See appendix B for the complete list of predefined variables.

In the expression you can also use customer defined fields or form fields of the print session.

Using other overlay for different companies

You can use a different overlay for each company in LN. For example, when printing an LN report in company 100, you want to use the house style for that company. When you print an LN report in company 200, you want to use another house style.

Preparation

- Use the RFQ document type, created earlier.
- You will create two different house styles, and add them to the document type. Based on a condition (which LN company is used) one of the two house styles is selected and merged with the document.

Complete these steps:

1. Start the **Document Types (ttrpi2510m000)** session. Select the RFQ document type and open its details.
2. On the **Parts** tab, open the already existing overlay part, which contains the house style of your company.

3 Add this code to the **Condition Expression** field:

```
if dom.document.company = 100 then
       return(true)
endif
return(false)
```

4 Save the record, and return to the Document Type.

5 On the **Parts** tab, add a new record for another overlay part with these properties:
   - **Description**: Housestyle company 200
   - **Destination Type**: All
   - **Part Type**: Overlay
   - **Overlay Type**: All Pages

6 Add this code to the Condition Expression field:

```
if dom.document.company = 200 then
       return(true)
endif
return(false)
```

7 Save this record.

8 Create a pdf file with the house style for company 200 and upload this file to the new document type part.

9 Click **Validate** to validate the document type.

10 Run the **Request for Quotations** session in both company 100 and company 200.

11 Print a Request for Quotation to the Document Output Management device and view the results.

Additional information

For each company you can use the house style of that company, without changing the report.

Using other overlay for draft invoice

You can use a different overlay for draft invoices. The word DRAFT will be added as a watermark to the document.

Preparation

- Create a document type and report rule for the Invoice report, or use an existing document type.
- You will create an extra overlay pdf file, which will be added to the Invoice report for draft invoices. Other invoice reports will not get this overlay. Based on a condition, the draft overlay is selected and merged with the document.

Complete these steps:

1. Start the **Document Types (ttrpi2510m000)** session. Select the Invoice document type and open its details.
2. On the **Parts** tab, add a new record for another overlay part with these properties:
   - **Description**: Overlay Draft
   - **Destination Type**: All
   - **Part Type**: Overlay
   - **Overlay Type**: All Pages
3. Add this code to the **Condition Expression** field:

```
if tolower$(#print.type#) = "draft" then
       return(true)
endif
return(false)
```

4. Save this record.
5. Create a pdf file with a draft watermark and upload this file to the new document type part.
6. Click **Validate** to validate the document type.
   Note that a new variable is added to the document type, called `print.type`
7. Start the **Report Rules (ttrpi2520m000)** session.
8. Select the report rule for report `cisli12001000`, which is linked to the Invoice document type.
   Open the details for this report rule.
9. On the **Variables** tab, click **Refresh Variables**, to refresh the variables from the document type.
10. For the new variable, specify this expression: `r.print.type`
11. Save this record, and click **Validate** to validate the report rule.
12. Print a draft invoice to the Document Output Management device and view the results.

Using other mail from address for different purchase office

You can send requests for quotations from different purchase offices in different locations, where each purchase office uses its own mail-from address. For example use mail-from address `noreply@infor.com` from the purchase office in the USA, and use mail-from address `noreply@infor.fr` from the purchase office located in France.

Preparation

- Use the RFQ document type, created earlier.
- You will create two different mail-from addresses, and add them to the document type. Based on the name of the purchase office one of the two mail addresses is used as mail-from address in the e-mail message.

Complete these steps:

1. Start the **Document Types (ttrpi2510m000)** session. Select the RFQ document type and open its details.

2 On the **Mail From Addresses** tab, add a new Address and specify a valid email address.
3 Open the details of this new address and add this expression:
4 Add this code to the **Condition Expression** field:

```
if #purchaseoffiename# = "PUR001" then
       return(true)
endif
return(false)
```

Replace the PUR001 with your own purchase office name.

5 Click **Validate** to validate the document type.
A new variable `purchaseofficename`, is added, which must be mapped to a report variable.
6 Start the **Report Rules (ttrpi2520m000)** session.
7 Select the report rule for report `tdpur140101002`.
Open the details for this report rule.
8 On the **Variables** tab, click **Refresh Variables**, to refresh the variables from the document type.
9 For the new variable, specify this expression: `tdpur100.cofc`
10 Save this record, and click **Validate** to validate the report rule.
11 Run the LN session **Request for Quotations**. Print two Request for Quotation reports for different purchase offices to the Document Output Management device and view the results.

Additional information

You will notice that different **mail-from** addresses can be used, based on the outcome of a condition.

Using other printer for different purchase office

When your business partners also want to receive a hard-copy of the Request for Quotation document, you can use the printer of the purchase offices in the different locations to print the hard copy.

Preparation

- Use the RFQ document type, created earlier.
- You will create different print devices, and add them to the document type. Based on the name of the purchase office one of the Print Devices is used to create the hard copy of the Request for Quotation report.

Complete these steps:

1 Start the **Document Types (ttrpi2510m000)** session. Select the RFQ document type and open its details.
2 On the **Print Devices** tab, add a default Device and select one of the available Devices.
This default Print Device is used if none of the other Print Devices has a valid expression.

3 Add another Print Device for each of your purchase offices. Add this code to the **Condition Expression** field:

```
if #purchaseofficename# = "PUR001" then
       return(true)
endif
return(false)
```

Replace the PUR001 with your own purchase office name.

4 Save this record.
5 Click **Validate** to validate the document type.
6 Run the **Request for Quotations** session.
7 Print two Request for Quotation reports for business partners who want a hard copy, from different purchase offices to the Document Output Management device. View the results.

Additional information

You will notice that different printers can be used for different office locations.

Note: You can also use different print devices where each printer has its own preprinted stationary.

Using images in the email body text

When you want to include your company logo in the email body text, you must use an html file as email body.

You can include images in the email body text. In these ways:

- Place an image on a web page. Specify the URL with the reference to the image location on your server in the email.
- Attach the image to the email. Add a reference to this attachment in the email body.

Preparation

- Use the RFQ document type, created earlier.
- You will include an image in the email body text, which is located on a local drive.

Complete these steps:

1 Start the **Document Types (ttrpi2510m000)** session.
2 Select the RFQ document type and open its details.
3 Add a new document type part to the document type and specify this information:
   - **Destination Type**: Email
   - **Part Type**: File to Attach
   - **Simple Content**: image1

- **Separate File**: Yes

4 Click **Upload** to upload the local image and link this to the new document type part.
5 Specify the **Simple Content** field, for example: image1.
6 In the HTML file that is uploaded as body, specify the src attribute of the img tag:
`src=cid:image1`
The same name is used as simple content in the document type part, prefixed with `cid:`
7 Upload the html file again in the other document type part that is used as Email Body.
8 Click **Validate** to validate the document type.
9 Run the **Request for Quotations** session.
10 Print a Request for Quotation to the Document Output Management device and view the results.

Additional information

You will notice that the image is not attached to the e-mail as separate document, but is included in the email body. The content-id (`cid`) of the image is used as reference in the html file.

Using existing documents

You can send a reminder document to your customer with the original invoice attached as additional document. The original invoice was sent earlier, and is still available in the Document Store.

Preparation

- Create a document type and report rule for the Reminder report, or use an existing document type.
- You will add an extra part to the Reminder document type, which will be added as additional document, containing the original invoice.

Complete these steps:

1 Start the **Document Types (ttrpi2510m000)** session. Select the Reminder document type and open its details.
2 On the **Parts** tab, create a part with these properties:
- **Description**: Original invoices
- **Destination Type**: All
- **Part Type**: Existing Document(s)
- **Simple Content**: INV_#keyword4#.pdf
- **Relative Position**: After
- **Order**: 0
- **Separate File**: Yes

3 Specify the **Expression** field with this code:

```
table ttfacr200
long node

select tfacr200.ttyp, tfacr200.ninv
from   tfacr200
where  tfacr200.itbp = :#bp#
and    tfacr200.balc > 0.0
and    tfacr200._compnr = :dom.document.company
selectdo
         node = dom.existing.document.add("INVOICE",
                         dom.document.company,
                         "keyword1", #bp#,
                         "keyword2", str$(dom.document.company),
                         "keyword3", tfacr200.ttyp,
                         "keyword4", str$(tfacr200.ninv))
         dom.add.part(node, 4)
endselect
return(true)
```

4 Save this record.
5 Click **Validate** to validate the document type.
6 Run the **Report Rules (ttrpi2520m000)** session to map the new variables to report fields.
7 Click **Validate** to validate the report rule.
8 Print a reminder to the Document Output Management device and view the result.

Additional information

An extra document is attached to the email, which contains the original invoice. In the expression of this new part, the existing document is read from the document store using the keywords. It is therefore very important to fill the keywords with proper values.

In the expression, you are using the `dom.existing.document.add()` function to retrieve the existing invoice document from the document store and add this to your email message.

You are also using the `dom.add.part()` function to add an overlay to the document.

For the syntax of these functions, see Adding existing documents from the DOM document store on page 59.

Addendum documents

Document Output Management supports main and sub documents. This manages the situation when multiple LN reports are sent to the same spooler. These subdocuments are collected based on the document identification, and merged into one PDF before being distributed.

There are also cases where during one print session, different LN reports are sent to different spoolers. When the reports were shown in the UI, you would see different tabs with the different printed report. An example is the Project Contract Acknowledgement with the Installments as possible addendum.

In Document Output Management you can specify to collect the different reports based on the document identification and distribute them in one run. One report acts as the main report, the other reports are distributed as attachments.

Configuring addendum documents

Complete these steps:

1. Select the document type.
2. On the details enable the **Multiple Sub-Documents**.
3. Select a document grouping from the **Group by** drop-down list.

   This indicates how the main document and the addendum documents are grouped together.
4. Specify the distribution options in the report rule details for each report rule contributing to the document type:
   a. Set **Distribute** as the property for the Main Document and the main report rule. Set **Addendum** as property for the addendum report rules.
   b. For each addendum report rule select the **Separate File** check box. When selected the addendum document will be distributed as separate attachment. When cleared the addendum document will be merged into the main document.
   c. For each addendum report rule select the **Relative Position** check box. When set to **before** the addendum document will be distributed before the main document. Either merged into the main document file before the main document content or as attachment added before the main document.
   d. For each addendum report rule select the **Order** check box. With the order you can change the addendum document ordering in the distribution. The ordering of the addendum documents is merged with the ordering of the attachments defined at the document type. When the ordering is the same an addendum document precedes a document type attachment.

Custom destination types and customer receiver types

Besides the currently supported destinations and receiver types you can create your own custom destination and receiver type.

Using custom destinations

As an example a custom destination type is created that stores documents on the server file system.

Preparation

- Use an existing document type, or create a new one.
- You will create a new library, which will store documents on the file system. This library must be registered as custom destination type.

Complete these steps:

1 Create a new library `tccomdomdestfs` with this content:

```
domain tcmcs.str256m g.path
 domain tcmcs.str256m g.folder
 domain tcmcs.str256m g.filenames(1) based
 domain tcmcs.str256m g.displaynames(1) based
 long   g.nr.files


function extern long distribution.new(ref string o.error.msg)
{
 FunctionUsage
 Expl: Initializes distribution.
 Pre: -
 Post: -
 Input: -
 Output: - o.error.msg - The error message if anything went wrong
 Return: 0 if OK, otherwise the error code
 EndFunctionUsage

 g.path = bse.dir$() & "/dom/"

 g.nr.files = 0

 return(0)
}

function extern long distribution.set.document.file(
   const string i.filename,
   const string i.displayname,
   const string i.mime.type,
   ref string o.error.msg)
{
 FunctionUsage
 Expl: Sets the document file to be distributed.
 Pre: -
 Post: -
 Input: - i.filename - The file name which contains the document file
  - i.displayname - The name to be displayed in the distribution
  - i.mime.type - The MIME type of the document
 Output: - o.error.msg - The error message if anything went wrong
 Return: 0 if OK, otherwise the error code
 EndFunctionUsage
```

```
 store.file(i.filename, i.displayname)

 return(0)
}

function extern long distribution.add.attachment(
   const string i.filename,
   const string i.displayname,
   const string i.mime.type,
   ref string o.error.msg)
{
 FunctionUsage
 Expl: Adds additional attachment files to the distribution.
 Pre: -
 Post: -
 Input: - i.filename - The file name which contains the attachment
  - i.displayname - The name to be displayed in the distribution
  - i.mime.type - The MIME type of the document
 Output: - o.error.msg - The error message if anything went wrong
 Return: 0 if OK, otherwise the error code
 EndFunctionUsage

 store.file(i.filename, i.displayname)

 return(0)
}

function extern long distribution.set.destination(
   const string i.address,
   ref string o.error.msg)
{
 FunctionUsage
 Expl: Sets the destination address.
 Pre: -
 Post: -
 Input: - i.address - The address of the destination
 Output: - o.error.msg - The error message if anything went wrong
 Return: 0 if OK, otherwise the error code
 EndFunctionUsage

 g.folder = i.address

 return(0)
}

function extern long distribution.send(ref string o.error.msg)
{
 FunctionUsage
 Expl: Stores the received document and attachment on the filesystem
  in the subfolder defined in destination address.
 Pre: -
 Post: -
 Input: -
 Output: - o.error.msg - The error message if anything went wrong
 Return: 0 if OK, otherwise the error code
 EndFunctionUsage
```

```
 domain tcmcs.str256m path
 long i
 long ret

 path = strip$(g.path) & strip$(g.folder)
 ret = mkdir(path)

 for i = 1 to g.nr.files
  ret = file.cp(g.filenames(1, i), path & "/" & g.displaynames(1, i))
  if ret < 0 then
   o.error.msg = "Failed to store file " & g.displaynames(1, i)
   return (ret)
  endif
 endfor

 return (0)
}

function extern long distribution.end(ref string o.error.msg)
{
 FunctionUsage
 Expl: Finalizes a distribution.
 Pre: -
 Post: -
 Input: -
 Output: - o.error.msg - The error message if anything went wrong
 Return: 0 if OK, otherwise the error code
 EndFunctionUsage

 g.nr.files = 0
 free.mem(g.filenames)
 free.mem(g.displaynames)

 return(0)
}

function extern long distribution.cancel(ref string o.error.msg)
{
 FunctionUsage
 Expl: Cancels and removes a distribution.
 Pre: -
 Post: -
 Input: -
 Output: - o.error.msg - The error message if anything went wrong
 Return: 0 if OK, otherwise the error code
 EndFunctionUsage

 return(distribution.end(o.error.msg))
}

function store.file(
   const string i.filename,
   const string i.displayname)
{
 domain tcmcs.str256m str fixed
```

```
 INC(g.nr.files)
 alloc.mem(g.filenames, len(str), g.nr.files)
 g.filenames(1, g.nr.files) = strip$(i.filename)
 alloc.mem(g.displaynames, len(str), g.nr.files)
 g.displaynames(1, g.nr.files) = strip$(i.displayname)
}
```

2 Start the **Custom Destination Types (ttrpi2554m000)** session and create a custom destination type with these properties:
   - **Description**: Filesystem Destination
   - **Custom Destination Type**: FILESYSTEM
   - **Library**: tccomdomdestfs

3 Use this custom destination type to send the document to the file system. How to do this depends on your receiver type:
   - If you use receiver type Contact, Business Partner, or Employee, start the **Document Management Output Details (tccom6170m000)** session. Add a record for your receiver type, your document type, destination type 'Custom', and custom destination type 'FILESYSTEM'. **Note:** The **Custom Destination Type** field is initially hidden in the **Document Management Output Details (tccom6170m000)** session. You can unhide the field in the **Personalization Workbench**. To start this workbench, click the gear icon in the session's toolbar and select **Personalize Form**.
   - If you use receiver type User, start the **User Document Type Settings (ttrpi2551m100)** session. Open the details of a specific user or all users and your specific document type. Add a record with destination type 'Custom' and custom destination type 'FILESYSTEM'.

4 Print to the Document Output Management device and view the result. Note that the documents are stored on the file system in this directory: `<bse_path>/dom`.

Additional information

You can create your own destination type and use it for different document types and recipients.

Create a library with these functions, which are run by the Document Distribution Engine:

- `distribution.new`
- `distribution.set.document.file`
- `distribution.add.attachment`
- `distribution.set.destination`
- `distribution.send`
- `distribution.end`
- `distribution.cancel`

For the syntax of these functions, see Custom destination type on page 57.

Note: Use this technique to create your own destination or send the documents to your own document management system.

Using customer receiver types

Use the procedure as an example to create a custom receiver type which is the primary contact of the shipment's carrier

Preparation

- Create a new Document Type SHIPMENT with the required parts (e-mail subject, e-mail body, document filename, and mail from address) and create a new Report Rule for the report whinh443511000 with the required expressions.
- You will create a new receiver type, to be able to send the document to the primary contact of the shipment's carrier.

Complete these steps:

1 Create a new library tcmcsdomcarr with this content:

```
table ttccom120 |* Buy-from Business Partners
 table ttccom140 |* Contacts
 table ttcmcs080 |* Carriers/LSP

#include <bic_dom>

function extern long dom.get.destinations(
  const string i.doc.type,
  const string i.receiver.type,
  const string i.custom.receiver.type,
  const string i.receiver.value)
{
 FunctionUsage
 Expl: Returns the list of destinations
 Pre: -
 Post: -
 Input: i.doc.type  - document type
  i.receiver.type - receiver type
  i.custom.receiver.type - custom receiver type, if applicable
  i.receiver.value - receiver value as defined in the report rule
 Output: -
 Return: destinations as returned by get.dom.document()
 EndFunctionUsage

 long destination
 domain tccfrw  carrier
 domain tcmail  email

 dom.init()

 carrier = i.receiver.value
 email = get.email(carrier)

 if not isspace(email) then
  | send email to the primary contact
  destination = dom.destination.new(DOM_DESTINATION_TYPE_EMAIL)
  dom.destination.set.address(destination, email)
 endif
```

```
 return (dom.get.document())
}

function string get.email(const string i.carrier)
{
 domain tcmail email

 email = ""

 select tcmcs080.suno
 from tcmcs080
 where tcmcs080._index1 = { :i.carrier }
 as set with 1 rows
 selectdo
  select tccom120.ccnt
  from tccom120
  where tccom120._index1 = { :tcmcs080.suno }
  as set with 1 rows
  selectdo
   select tccom140.info:email
   from tccom140
   where tccom140._index1 = { :tccom120.ccnt }
   as set with 1 rows
   selectdo
   endselect
  endselect
 endselect

 return (email)
}
```

2 Start the Custom Receiver Types (ttrpi2554m000) session and create a new Custom Receiver Type with these properties:
   - **Custom Receiver Type**: Carrier
   - **Description**: Carrier
   - **Library**: tcmcsdomcarr
3 Start the **Document Types (ttrpi2510m000)** session and open the details of Document Type SHIPMENT.
4 Change these properties:
   - **Receiver Type**: Custom
   - **Custom Receiver Type**: Carrier
5 Save the record and click **Validate** to validate the document type.
6 Start the **Report Rules (ttrpi2520m000)** session and open the details for report whinh443511000.
7 Change the expression of the receiver field variable to:

```
table  twhinh430
select  whinh430.carr
from    whinh430
where   whinh430._index1 = {:whinh431.shpm}
as set with 1 rows
```

```
selectdo
    return (whinh430.carr)
endselect
return(“”)
```

8 Change the expression of the split field variable to: `whinh431.shpm`

9 Save the record and click **Validate** to validate the report rule.

10 Start the **Print Shipping Discrepancies (whinh4435m000)** session. Print to the Document Output Management device and view the result.

Additional information

You can create your own receiver type. Create a library with the `dom.get.destinations` function, which is run by the Document Distribution Engine.

In this library some standard defines and functions can be used by including `<bic_dom>`.

For a detailed description of these defines and functions, see Custom receiver type on page 54.

Using custom email

In addition to the currently supported CMF as email solution, you can also create your own email solution.

Additional information:

To use your own email solution, you must create a library and specify it as custom email solution library in the **Parameters (ttrpi2100s000)** session. In the library, specify these functions, which are run by the Document Distribution Engine:

- `init.mail.message`
- `set.mail.sender`
- `add.mail.recipient`
- `set.mail.subject`
- `set.mail.body`
- `add.mail.attachment`
- `send.mail.message`
- `cancel.mail.message`

For the syntax of these functions, see Custom email solution on page 67.

Using custom fax

You can create your own fax solution in addition to the currently supported faxing solution using a fax provider who processes e-mails.

Additional information:

To use your own fax solution, you must create a library and specify it as custom fax solution in the **Parameters (ttrpi2100s000)** session. In the library, specify these functions, which are run by the Document Distribution Engine:

- `init.fax`
- `add.fax.attachment`
- `set.fax.number`
- `send.fax`
- `cancel.fax`

For the syntax of these functions, see Custom fax solution on page 69.

Defining document providers

Besides printing to the Document Output Management device, documents can be added to Document Output Management through a document provider.

To define a new document provider:

1 Create a new library, for example `tssocdomflpr`, with this content:

```
#include <bic_dom>

function extern domain ttxmlnode get.metadata()
{
 FunctionUsage
 Expl: Provides the document provider meta data in datafields and op
tionally contexts.
  Example implementation which assumes the bic_dom be included:

  domain ttxmlnode meta.node

  meta.node = dom.provider.metadata.init()

  dom.provider.metadata.add.field("filename", "Filename", "ttaud.path",
 meta.node)
  dom.provider.metadata.add.field("docname", "Document name",
"ttaud.path", meta.node)
  dom.provider.metadata.add.field("extension", "Extension", "ttst10",
 meta.node)


  dom.provider.metadata.add.context("new", "New file", meta.node)
```

```
  dom.provider.metadata.add.context("update", "Updated file", meta.node)


  return (meta.node)
 Pre: -
 Post: -
 Input:  -
 Output:
 Return: domain ttxmlnode an XML node containing the meta data describ
ing the
  document provider
 EndFunctionUsage

 domain ttxmlnode meta.node
 domain ttxmlnode duml.node

 meta.node = dom.provider.metadata.init()

 duml.node = dom.provider.metadata.add.field("filename", "Filename",
"ttaud.path", meta.node)
 duml.node = dom.provider.metadata.add.field("docname", "Document name",
 "ttaud.path",
  meta.node)
 duml.node = dom.provider.metadata.add.field("extension", "Extension",
 "ttst10", meta.node)


 duml.node = dom.provider.metadata.add.context("new", "New file",
meta.node)
 duml.node = dom.provider.metadata.add.context("update", "Updated file",
 meta.node)

 return (meta.node)
}
```

2 Start the**Document Providers (ttrpi2556m000)** session.

3 Create a new Document Provider. Specify this information:

Document Provider
Files.

Description
File Provider.

Library
`tssocdomflpr`

4 Save the record.

The `get.metadata` method of the document provider library is called. The fields and contexts are shown in the Document Provider details.

When the document provider library is updated you must refresh the Document Provider data with the refresh option in the Document Provider details.

Setting up a document provider rule

Create a document provider rule to define the provider specific settings and map variables and keywords to document provider fields.

To create a document provider rule for a document provider:

1. Start the **Document Provider Rules (ttrpi2558m000)** session.

   You can start the session directly or through the Document Providers session.

2. Add a document provider rule for the document provider. Specify this information:

   Document Provider
   Specify the provider the rule belongs to.

   Is Default
   The default rule yes or no.

   Condition Expression
   The expression to be validated when this is not the default rule.

   Document Type
   The document type to be used.

3. Save this record.

   On the **Variables** tab, a several variables are created with a blank expression.

4. Specify the expressions or browse and select the correct document provider field.
5. Click **Validate** to validate the settings for the document provider rule.

   Check whether the document provider rule is validated and compiled successfully. When valid you are ready to use the Document Provider to add documents to Document Output Management.

Using document providers

To use a document provider and add documents to Document Output Management you must write some Infor LN code. The code is probably written in the document provider library.

This library completes these steps:

1. Ensure that the provider fields are specified with data. Provider fields are either table fields or external program variables.
2. Call the `dom.add.external.document` method of `bic_dom` to add a document.

   For each call one batch with one document in it is created.

   For the syntax of `dom.add.external.document`, see Document provider on page 71.

Chapter 5: DMS integration

Document Management System (DMS) is used to track, manage and store documents. With Document Output Management (DOM) you can store documents in DMS or attach an already stored document to the distribution of a document.

To communicate with the DMS, DOM uses the document hub, which is a tool to map Applications to Document Management Systems. The document hub currently supports these DMSs:

- Infor Document Management (IDMIDM)
- Object Data Management (ODM)

This section explains how to set up the document hub and how documents can be stored in DMS and how they can be reused. The Request for Quotation document is used as an example.

You can also use your own document solution to store the DOM documents by creating a custom destination. For more information, see Advanced Options on page 24

Setting up document hub

You must initialize the document hub before using it. This initialization process adds these components:

- Applications and its attributes.
- Document Management Systems (DMS) and its document type attributes.

Start the **Initialize Document Hub (ttdms3200m000)** session to initialize the document hub. Run this session once. If a re-initialize is required the DMS document type is changed.

Before a document can pass the document hub, a mapping must be defined between the application attributes describing a document and the document management system attributes describing the same document.

The available document management system attributes depend on the document management system configuration.

For IDM DMS it depends on the attributes defined with the IDM document types.

For ODM DMS it depends on the ODM Object access index, each field of the access index is represented as a DMS attribute.

For each mapping also a DMS attribute called `externalApplication` is created. This attribute must get a Mapping Type fixed value with the value of an external application being defined in ODM. Use

the **External Applications (dmcom3100m000)** session. The external application is only used when uploading documents to ODM. It defines defaults to be used in ODM for the upload.

To print an RFQ document using Document Output Management and store it in the IDM archive.

Preparation

Use the RFQ document type, you created earlier.

To map the application attributes to the document management system (DMS) attributes:

1. Start the **Document Mapping (ttdms3550m100)** session and select application `infor.ln.dom`.
2. Add a new record to the Application Tables. Specify this property:
   - **Table Name**: tdpur105
3. Save the record.
4. In the left bottom part of this session add a new document type. Specify this information:
   - **Document Type**: MDS_GenericDocument
   - **Upload**: Yes
   - **Download**: Yes
5. Save the record.
6. Select the new document type, and notice that the attribute mappings are already filled on the right hand part of the session.
7. If all mandatory attributes are mapped, the status of the document type is changed to **Ready to Use**.

Additional information

When a document type status is **Ready to Use**, the application can use the Document Hub to store documents in the document management system (DMS) or retrieve documents from the DMS. If a document type does not have status **Ready to Use**, use the **Validate** option to search for issues.

Each document type attribute must be mapped to either an application attribute or a fixed value. When mapping is not possible, the mapping type can be switched to Application UI. In that case it is up to the application to have the attribute filled by the application user.

Store documents in DMS

You can archive your sent Request for Quotation documents and store them in the DMS (IDM archive).

Preparation

Use the RFQ document type, created earlier.

To create a receiver-independent destination and link it to the DMS:

1. Start the **Document Types (ttrpi2510m000)** session. Select the RFQ document type and open its details.
2. On the **Archive Specifications** tab, add a new record with these properties to link a DMS document type to the LN document type:
   - **SpecificationID**: dms
   - **Description**: Archive
   - **DMS Document Type**: MDS_GenericDocument
   - **LN table**: tdpur105
3. On the **Receiver Independent Destinations** tab, add a record with these properties to link an archive specification to a destination:
   - **Destination Type**: Document Management System
   - **Address**: dms
4. Open the Archive Specification and on the **Attributes** tab, specify the **Mapping** fields to link the DMS document type attributes to the LN document variables.

   Note:

   The id1 - id2 attributes are reserved for the primary key fields of the LN table. For example table tdpur105 has two primary key fields; therefore only id1 and id2 must be filled.
5. Save this Archive Specification and click **Validate** to validate the document type.
6. Run the **Requests for Quotation** session. Print a request to the Document Output Management device and verify whether the document is stored in DMS.

Additional information

To use multiple archive specifications, you must create multiple receiver independent destinations on the **Receiver Independent Destinations** tab. The condition expression of the receiver independent destination determines which archive specification is used.

Using existing DMS documents

You can send a specific document for the business partner attached to the Request for Quotation document. This specific document is stored in DMS and must be added as additional document to the RFQ document.

Preparation

- You will use the RFQ document type, created earlier.
- Add an extra part to the RFQ document type, which will be added as additional document. This additional document is linked to the business partner and stored in DMS.

Complete these steps:

1. Start the **Document Types (ttrpi2510m000)** session.

Select the RFQ document type and open its details.

2 On the **Parts** tab, create a part with these properties:
   - **Description**: BP specific terms
   - **Destination Type**: All
   - **Part Type**: Existing Document(s)
   - **Expression**:

```
long  nr.of.documents, ret
string path(256) mb

dom.dochub.list.init("tccom400", "MDS_GenericDocument")
dom.dochub.list.add.attribute.filter("id1", #bp#)
dom.dochub.list.set.filename.filter("Terms*.jpg")
nr.of.documents = dom.dochub.list.execute()
if nr.of.documents <> 1 then
 return (false)
endif
ret = dom.dochub.get.document(dom.dochub.get.document.id(1), path)
if ret >= 0 then
 dom.add.document.to.distribution(path,
  dom.dochub.get.document.filename(1),
  dom.dochub.get.document.mimetype(1))
 return (true)
endif
return (false)
```

   - **Simple Content**: x.pdf
   - **Relative Position**: After
   - **Order**: 0
   - **Separate File**: Yes

3 Save this record.

4 Click **Validate** to validate the document type.

5 Run the **Requests for Quotation** session. Print a request to the Document Output Management device and verify the result.

Additional information

An extra document is attached to the email, which was linked to the business partner and stored in the DMS.

In the expression of this new part, the document is read from the DMS using the filename of the document. It is therefore very important to use naming conventions for the file names.

Note: this is a simplified example. In reality there can be multiple documents.

When attaching documents from a DMS, these functions are available:

- `dom.dochub.list.init`
- `dom.dochub.list.add.attribute.filter`

- `dom.dochub.list.set.filename.filter`
- `dom.dochub.list.add.mimetype.filter`
- `dom.dochub.list.execute`
- `dom.duchub.get.document`
- `dom.dochub.get.document.attribute`
- `dom.dochub.get.document.id`
- `dom.dochub.get.document.filename`
- `dom.dochub.get.document.mimetype`
- `dom.add.document.to.distribution`

For the syntax of these functions, see Adding existing documents from any supported DMS through document hub on page 60.

Chapter 6: Troubleshooting

Use this troubleshooting information as a resource to help you solve specific problems you can encounter when using Document Output Management.

Logging

When you are troubleshooting, log files can help you.

To get more logging information, you can set these environment variables:

- `RPI_SERVER_LOGGING=1`
  Logs information from reporting or dom processes in $BSE/log/log.rpi
- `DOM_TRACE=<N>`
  Logs information from scheduler process in $BSE/log/log.rpi
  - N=1: Errors are logged
  - N=2: Informational messages
  - N=3: Debug level

In case of issues also check the `$BSE/tmp/bshell.<pid>` file and the `$BSE/log/log.bshell` file.

Debugging

To detect problems in expressions or custom libraries, you can use Infor LN Studio for debugging purposes.

Custom implementations

You can debug custom implementations, such as custom destination types, custom e-mail implementations, and custom fax implementations, using Infor LN Studio. These custom implementations are libraries that are called by these sessions:

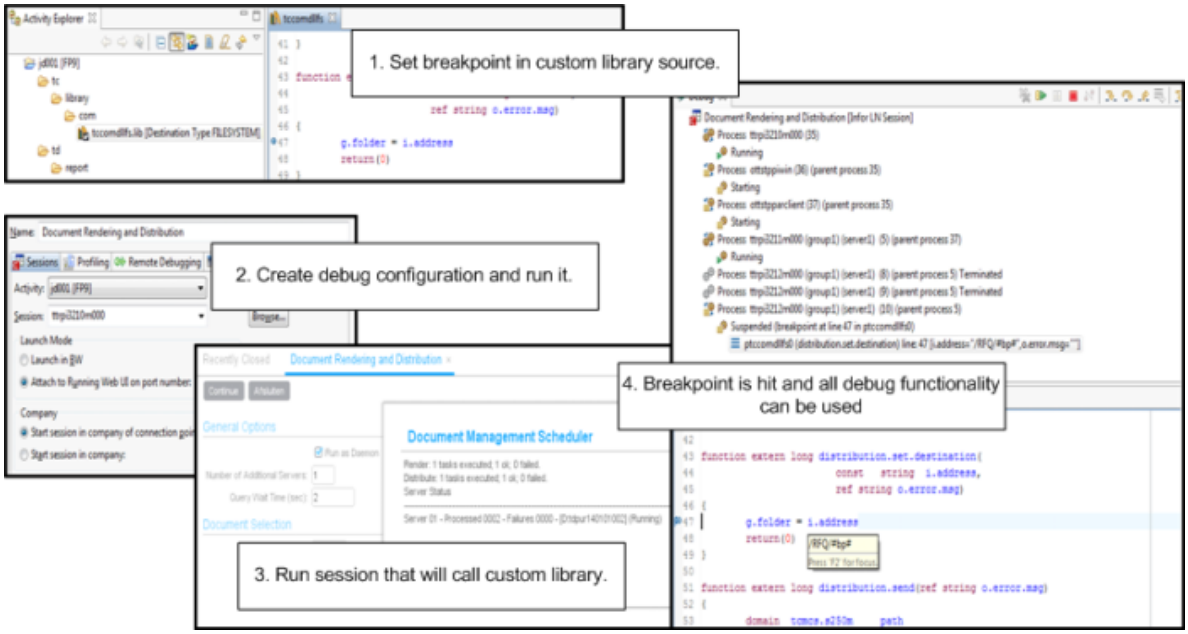
- **Document Rendering and Distribution (ttrpi3210m000)**
- **Documents (ttrpi3510m000)** - Select the View Document action.

The custom receiver type implementation is called during printing of the original LN report to the Document Output Management device.

Complete these steps:

1. Set breakpoint in library.
2. In case of Web UI, create a launch configuration for one of the sessions mentioned earlier and launch that session in debug mode from Infor LN Studio.
3. In case of LN UI, start the Debug and Profile 4GL (ttadv1123m000) session.

   Select your activity and select the **debug** check box. Click **OK**. Start one of the sessions mentioned earlier from LN UI.



Expressions

You can debug the expressions defined for document types and report rules using Infor LN Studio. Actually these expressions are compiled into objects. The generated source code of these document type objects and report rule objects is visible in Infor LN Studio.

On the Overview page of the LN Report in Infor LN Studio the different Document Output Management Sources, which are applicable for this LN Report, are displayed. Click the link related to this source to open the source editor and set a breakpoint.

The document type expressions are called during the preparation phase (printing the LN Report) to determine the printer devices, and during the distribution phase to handle mail from addresses, receiver independent destinations, existing documents, and so on.

Report rule expressions are called during the selection of a device (to select the applicable report rule), and during printing the LN Report to map the correct report values to document variables.

Appendix A: Standard functions

bic_dom

The <bic_dom> include contains these defines:

```
#define DOM_DESTINATION_TYPE_EMAIL  "email"
#define DOM_DESTINATION_TYPE_PRINT  "print"
#define DOM_DESTINATION_TYPE_FAX  "fax"
#define DOM_DESTINATION_TYPE_CUSTOM  "custom"
#define DOM_RECEIVER_TYPE_BUSINESS_PARTNER "bus.partner"
#define DOM_RECEIVER_TYPE_CONTACT  "contact"
#define DOM_RECEIVER_TYPE_EMPLOYEE  "employee"
#define DOM_RECEIVER_TYPE_USER   "user"
#define DOM_RECEIVER_TYPE_CUSTOM  "custom"
```

Custom receiver type

The custom receiver type interface consists of the dom.get.destinations function.

```
Syntax:
     function long dom.get.destinations(
                 string i.doc.type,
                 string i.receiver.type,
                 string i.custom.receiver.type,
                 string i.receiver.value)
Description:
     Use this function to add the destinations to the document.
Arguments:
    i.doc.type: input argument which contains the document type
    i.receiver.type: input argument which contains the receiver type
    i.custom.receiver.type: input argument which contains the name of the
 custom receiver
        type, if applicable
    i.receiver.value: input argument which contains the receiver value as
 defined in the
        report rule
Return value: xml node with the document, including the destinations
```

The <bic_dom> include contains these supporting functions, which can be used in the custom receiver type implementation:

dom.init()

```
Syntax:
     function extern dom.init()
Description:
     Initializes the document.
```

dom.destination.new()

```
Syntax:
     function extern long dom.destination.new (
                         const string i.destination.type,
                         [const string i.custom.code])
Description:
     Creates a new destination and adds it to the document created by
dom.init(). This can
     be repeated to add multiple destinations to the document.
Arguments:
     i.destination.type: input variable, which contains the destination
type, use one of these
        values:
                          DOM_DESTINATION_TYPE_EMAIL
                          DOM_DESTINATION_TYPE_FAX
                          DOM_DESTINATION_TYPE_PRINT
                          DOM_DESTINATION_TYPE_CUSTOM
     i.custom.code: input variable, which contains the custom code, if
applicable (optional)
Return value:
     Returns an xml node containing the newly created destination or 0
(zero) if the
        destination could not be created.
```

dom.destination.add.address()

```
Syntax:
 function extern void dom.destination.add.address(
     long  i.destination,
   domain ttrpi.eatp i.address.type,
  const   string i.address)

Description:
 Adds a recipient for the given destination.
Arguments:
 i.destination - The destination to add the address to, as
      created by dom.destination.new.
 i.address.type - Indicates whether the address is To, CC, or
      BCC. This must be specified for destinations
      with type email. For destinations with other
      types, it is not applicable.
 i.address  - The address to add.
```

dom.destination.set.address()

```
Syntax:
     function extern void dom.destination.set.address(
                          long i.destination,
                          const string i.address)
Description:
     Sets the address for a given destination.
Arguments:
    i.destination: input variable, which contains the destination created
 by
         dom.destination.new()
     i.address: input variable, which contains the address to set
```

dom.destination.set.additional.address()

```
Syntax:
     function extern void dom.destination.set.additional.address(
                          long i.destination,
                          const string i.address,
                          boolean i.is.bcc)
Description:
     Sets the additional address for a given destination. This can be
repeated to add multiple
     additional addresses to the destination.
Arguments:
    i.destination: input variable, which contains the destination created
 by
         dom.destination.new()
     i.address: input variable, which contains the additional address to
 set
     i.is.bcc: input variable (true/false) to set whether the additional
 address is a bcc (blind
          carbon copy) or a normal cc (carbon copy)
```

dom.get.document()

```
Syntax:
     function extern long dom.get.document()
Description:
     Retrieves the document.
Return value:
    Returns an xml node which contains the document. In case of an error,
 the function will
        return an xml node containing the error message.
```

dom.errormessage.set()

```
Syntax:
     function extern void dom.errormessage.set(const string i.error.mes
sage)
Description:
     Creates an error message for a document. Already added destinations
```

```
 will be removed.
Arguments:
     i.error.message: input variable which contains the error message.
```

Custom destination type

The custom destination type interface consists of these functions:

distribution.new

```
Syntax:
     function extern long distribution.new(ref string o.error.msg)
Description:
     Use this function to program the actions to be done before the dis
tribution is created.
Arguments:
     o.error.msg: output parameter, which can be filled with an error
message if anything
       went wrong.
Return value: 0 if processing without errors, otherwise the error code
```

distribution.set.document.file

```
Syntax:
     function extern long distribution.set.document.file(
                       const string i.filename,
                       const string i.displayname,
                       const string i.mime.type,
                       ref string o.error.msg)
Description:
     Use this function to program the actions to be done to add the docu
ment to the
        distribution.
Arguments:
     i.filename: input argument with the file name of the document file.

     i.displayname: input argument with the display name of the document
 file.
    i.mime.type: input parameter with the mime type of the document file.

     o.error.msg: output parameter, which can be filled with an error
message if anything
         went wrong.
Return value: 0 if processing without errors, otherwise the error code
```

distribution.add.attachment

```
Syntax:
     function extern long distribution.add.attachment(
                         const string i.filename,
```

```
                        const string i.displayname,
                        const string i.mime.type,
                        ref string o.error.msg)
Description:
     Use this function to program the actions to be done to add an attach
ment to the
        distribution.
Arguments:
    i.filename: input argument with the file name of the attachment file.

    i.displayname: input argument with the display name of the attachment
 file.
     i.mime.type: input parameter with the mime type of the attachment
file.
     o.error.msg: output parameter, which can be filled with an error
message if anything
         went wrong.
Return value: 0 if processing without errors, otherwise the error code
```

distribution.set.destination

```
Syntax:
     function extern long distribution.set.destination(
                        const string i.address,
                        ref string o.error.msg)
Description:
     Use this function to program the actions to be done to add a desti
nation address to the
        distribution.
Arguments:
     i.address: input argument with the address of the destination.
     o.error.msg: output parameter, which can be filled with an error
message if anything
        went wrong.
Return value: 0 if processing without errors, otherwise the error code
```

distribution.send

```
Syntax:
     function extern long distribution.send(ref string o.error.msg)
Description:
     Use this function to program the actions to be done in case the
distribution is sent.
Arguments:
     o.error.msg: output parameter, which can be filled with an error
message if anything
        went wrong.
Return value: 0 if processing without errors, otherwise the error code
```

distribution.end

```
Syntax:
     function extern long distribution.end(ref string o.error.msg)
Description:
```

```
     Use this function to program the actions to be done before the dis
tribution is finalized.
Arguments:
     o.error.msg: output parameter, which can be filled with an error
message if anything
        went wrong.
Return value: 0 if processing without errors, otherwise the error code
```

distribution.cancel

```
Syntax:
     function extern long distribution.cancel(ref string o.error.msg)
Description:
     Use this function to program the actions to be done before the dis
tribution is cancelled.
Arguments:
     o.error.msg: output parameter, which can be filled with an error
message if anything went wrong.
Return value: 0 if processing without errors, otherwise the error code
```

Existing documents

You can add existing documents in these ways:

- From the DOM document store
- From a DMS through document hub
- From IDM

Adding existing documents from the DOM document store

To add a document from the DOM document store, you can use the `dom.existing.document.add` function. This function has this syntax:

```
Syntax:
    function long dom.existing.document.add(
                  string i.doctype,
                  string i.doc.company,
                  [string i.keyword1,
                   string i.value1,
                   string i.keyword2,
                   string i.value2,
                   string i.keyword3,
                   string i.value3,
                   string i.keyword4,
                   string i.value4])
Description:
 This function retrieves an existing document from the Document Store,
```

```
including
 overlays and adds this to the document currently processed.
Arguments:

  string i.doctype
                   The document type of the existing document.
  string i.doc.company
                   The company of the existing document.
  string i.keyword1, i.value1, i.keyword2, i.value2, i.keyword3, i.value3,
 i.keyword4, i.value4
                  The (optional) string pairs for the keywords and their
 values,
                   to search the document.
Return value:

  This function returns an XML node, which contains the existing document.
```

You can use the `dom.add.part()` function to add an overlay to the document. This function has this syntax:

```
Syntax:
  function dom.add.part(
                       long i.document,
                       long i.sequence)
Description:
  This function adds an extra overlay to the existing document.
Arguments:
  long i.document
                 The xml node which contains the existing document.
  long i.sequence
                 The sequence of the overlay to set
```

Adding existing documents from any supported DMS through document hub

The `<bic_dom>` include contains functions, which can be used to add existing documents from a DMS:

dom.dochub.list.init

```
Syntax:
     function long dom.dochub.list.init(string i.table, i.doc.type)
Description:
     Initializes the List Documents request.
Arguments:
string i.table
                     the LN Table
string i.doc.type
                     the DMS Document Type, can be empty
Return value: 0, or -1 if an error occurred
```

dom.dochub.list.add.attribute.filter

```
Syntax:
     function long dom.dochub.list.add.attribute.filter (string i.appl.at
tr.id, string i.attr.value)
Description:
     Adds an attribute filter to the List Documents request.
Arguments:
     string i.appl.attr.id
                          the application attribute ID
     string i.attr.value
                          the attribute value
Return value: 0, or -1 if an error occurred
```

dom.dochub.list.set.filename.filter

```
Syntax:
     function long dom.dochub.list.set.filename.filter (string i.file
name.filter)
Description:
     Adds a filename filter to the List Documents request.
Arguments:
     string i.filename.filter the filename filter, a regular expression
Return value: 0, or -1 if an error occurred
```

For more information about the regular expressions, see the description of `expr.compile` in the *Infor ES Programmers Guide (Infor Support Portal KB 22924522).*

dom.dochub.list.add.mimetype.filter

```
Syntax:
     function long dom.dochub.list.add.mimetype.filter (string i.mimetype)
Description:
     Adds a MIME type filter to the List Documents request.
Arguments:
     string i.mimetype
                         the MIME type, a regular expression
Return value: 0, or -1 if an error occurred
```

dom.dochub.list.execute

```
Syntax:
     function long dom.dochub.list.execute()
Description:
     Executes the List Documents request.
Return value: the number of documents found, otherwise <0
```

dom.dochub.get.document

```
Syntax:
     function long dom.dochub.get.document (long i.unique.ident ,
                                            ref string o.result.file)
```

```
Description:
     Use this function to retrieve one document from DSM and store it as
 a temporary file.
Arguments:
     long i.unique.ident
                          the internal DMS document ID
     ref string o.result.file
                        the temporary file (including path), which contains
 the downloaded
                               DMS document
Return value: 0 if document retrieved, otherwise -1
```

dom.dochub.get.document.attribute

```
Syntax:
     function string dom.dochub.get.document.attribute (long i.entry,
                                                        string  i.attribute)
Description:
     Use this function to retrieve the value of a document attribute from
 DMS.
Pre:
     Call dom.dochub.list before this method to retrieve the list of
documents.
Arguments:
     long i.entry
                         The index of the DMS document
     string i.attribute
                         The name of the attribute which value should be
retrieved
Return value: the attribute’s value of the document, or empty if an error
 occurred
```

dom.dochub.get.document.id

```
Syntax:
     function string dom.dochub.get.document.id (long i.entry )
Description:
     Use this function to retrieve an internal ID of a document from DMS.
Pre:
     Call dom.dochub.list.execute before this method to retrieve the list
 of documents.
Arguments:
     long i.entry
                         The index of the DMS document
Return value: the ID of the document, or empty of an error occurred
```

dom. dochub.get.document.filename

```
Syntax:
     function string dom. dochub.get.document.filename (long i.entry)
Description:
     Use this function to retrieve the filename of a document from DMS.
Pre:
     Call dom.dochub.list.execute before this method to retrieve the list
```

```
 of documents.
Arguments:
     long i.entry
                       The index of the DMS document
Return value: the filename of the document, or empty of an error occurred
```

dom.dochub.get.document.mimetype

```
Syntax:
     function string dom.dochub.get.document.mimetype (long i.entry)
Description:
     Use this function to retrieve the mime type of a document from DMS.
Pre:
     Call dom.dochub.list.execte before this method to retrieve the list
 of documents.
Arguments:
     long i.entry
                       The index of the DMS document
Return value: the mime type of the document, or empty if an error occurred
```

dom.add.document.to.distribution

```
Syntax:
     function long dom.add.document.to.distribution (string i.tempfile,
                                                      string   i.name,
                                                     string   i.mimetype)
Description:
     This function adds an existing document from IDM to the list of
distributions.
Arguments:
     string i.tempfile
                     The name of the IDM document to add (including path)

     string i.name
                       The display name of the IDM document to add
     string i.mimetype
                       The mime type of the IDM document to add
Return value: 0 in case of success, otherwise -1
```

dom.errormessage.set()

```
Syntax:
     function extern void dom.errormessage.set(const string i.error.mes
sage)
Description:
     Creates an error message for a document. Already added destinations
 will be removed.
Arguments:
     i.error.message: input variable which contains the error message.
```

Adding existing documents from IDM

Before the introduction of the Document Hub, only the IDM DMS was supported. The <bic_dom> include still contains these functions, which can be used to add existing documents from IDM:

dom.idm.list.documents.by.attributes

```
Syntax:
    function long dom.idm.list.documents.by.attributes (string idm.doctype,
 … )
Description:
    Use this function to retrieve a list of documents from IDM using at
tributes.
Arguments:
    string idm.doctype: The IDM document type
    …: A variable number of arguments, containing the name and the value
 of each attribute
Return value: The number of documents found, otherwise -1
```

dom.idm.list.documents.by.xquery

```
Syntax:
    function long dom.idm.list.documents.by.xquery (string i.xquery )
Description:
    Use this function to retrieve a list of documents from IDM using
XQuery.
Arguments:
    string i.xquery: The XQuery string to retrieve a list of documents.
See the IDM documentation for the syntax of the XQuery string.
Return value: The number of documents found, otherwise -1
```

dom.idm.get.document

```
Syntax:
    function long dom.idm.get.document (string i.pid,  string i.filename
 ref string o.tempfile)
Description:
    Use this function to retrieve one document from IDM and store it as
a temporary file.
Arguments:
    string i.pid: The internal IDM document ID
    string i.filename: The filename (display name) of the IDM document
    ref string o.tempfile: (Output) The temporary file (including path),
 which contains the downloaded IDM document
Return value: 0 if document retrieved, otherwise -1
```

dom.idm.get.document.attribute

```
Syntax:
    function string dom.idm.get.document.attribute (long i.entry, string
  i.attribute)
Description:
    Use this function to retrieve the value of a document attribute from
```

```
 IDM.
Pre:
    Call dom.idm.list.documents.by.attributes or dom.idm.list.docu
ments.by.xquery before this method to retrieve the list of documents.
Arguments:
    long i.entry: The index of the IDM document
    string i.attribute: The name of the attribute which value should be
retrieved
Return value: The attribute's value of the document, or empty if an error
 occurred
```

dom.idm.get.document.pid

```
Syntax:
    function string dom.idm.get.document.pid  (long i.entry)
Description:
    Use this function to retrieve an internal ID of a document from IDM.

Pre:
    Call dom.idm.list.documents.by.attributes or dom.idm.list.docu
ments.by.xquery before this method to retrieve the list of documents.
Arguments:
    long i.entry: The index of the IDM document
Return value: The ID of the document, or empty if an error occurred
```

dom.idm.get.document.filename

```
Syntax:
    function string dom.idm.get.document.filename (long i.entry)
Description:
    Use this function to retrieve the filename of a document from IDM.
Pre:
    Call dom.idm.list.documents.by.attributes or dom.idm.list.docu
ments.by.xquery before this method to retrieve the list of documents.
Arguments:
    long i.entry: The index of the IDM document
Return value: The filename of the document, or empty if an error occurred
```

dom.idm.get.document.mimetype

```
Syntax:
    function string dom.idm.get.document.mimetype (long i.entry)
Description:
    Use this function to retrieve the mime type of a document from IDM.
Pre:
    Call dom.idm.list.documents.by.attributes or dom.idm.list.docu
ments.by.xquery before this method to retrieve the list of documents.
Arguments:
    long i.entry: The index of the IDM document
Return value: The mime type of the document, or empty if an error occurred
```

dom.idm.get.document.created.by

```
Syntax:
    function string dom.idm.get.document.created.by(long i.entry)
Description:
    Use this function to retrieve this document attribute: created by.
Pre+Arguments:
    Same as dom.idm.get.document.mimetype
Return value: The created by attribute of the document, or empty if an
error occurred
```

dom.idm.get.document.created.timestamp

```
Syntax:
    function string dom.idm.get.document.created.timestamp(long i.entry)
Description:
    Use this function to retrieve this document attribute: created
timestamp.
Pre+Arguments:
    Same as dom.idm.get.document.mimetype
Return value: The created timestamp attribute of the document, or empty
if an error occurred
```

dom.idm.get.document.last.changed.by

```
Syntax:
    function string dom.idm.get.document.last.changed.by(long i.entry)
Description:
    Use this function to retrieve this document attribute: changed by.
Pre+Arguments:
    Same as dom.idm.get.document.mimetype
Return value: The changed by attribute of the document, or empty if an
error occurred
```

dom.idm.get.document.last.changed.timestamp

```
Syntax:
    function string dom.idm.get.document.last.changed.timestamp(long
i.entry)
Description:
    Use this function to retrieve this document attribute: changed
timestamp.
Pre+Arguments:
    Same as dom.idm.get.document.mimetype
Return value: The changed timestamp attribute of the document, or empty
if an error occurred
```

dom.add.document.to.distribution

```
Syntax:
    function long dom.add.document.to.distribution (string i.tempfile,
string   i.name, string   i.mimetype)
```

```
Description:
    This function adds an existing document from IDM to the list of dis
tributions.
Arguments:
    string i.tempfile: The name of the IDM document to add (including
path)
    string i.name: The display name of the IDM document to add
    string i.mimetype: The mime type of the IDM document to add
Return value: 0 in case of success, otherwise -1
```

Custom email solution

The custom email solution interface consists of these functions:

init.mail.message

```
Syntax:
     function extern long init.mail.message(
                 ref string o.error.message)
Description:
     Use this function to initialize the email message.
Arguments:
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

set.mail.sender

```
Syntax:
     function extern long set.mail.sender (
                 const string i.name,
                 const string i.address,
                 ref string o.error.message)
Description:
     Use this function to add a sender to the email message.
Arguments:
     i.name: input argument which contains the name of the sender
     i.address: input argument which contains the mail address of the
sender
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

add.mail.recipient

```
Syntax:
     function extern long add.mail.recipient(
                 const string i.role,
                 const string i.name,
                 const string i.address,
```

```
                  ref string o.error.message)
Description:
     Use this function to add a mail recipient to the email message. This
 function can be repeated to add more mail recipients.
Arguments:
     i.role: input argument which contains the recipient's role. Possible
 values: to, cc, bcc
     i.name: input argument which contains the name of the recipient
     i.address: input argument which contains the mail address of the
recipient
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

set.mail.subject

```
Syntax:
     function extern long set.mail.subject(
                  const string i.subject.
                  ref string o.error.message)
Description:
     Use this function to add a mail subject to the email message.
Arguments:
     i.subject: input argument which contains the subject of the mail
message
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

set.mail.body

```
Syntax:
     function extern long set.mail.body(
                  const string i.body.text.file,
                  const string i.mimetype,
                  ref string o.error.message)
Description:
     Use this function to add a mail body to the email message.
Arguments:
     i.body.text.file: input argument which contains the file name of the
 body text
     i.mimetype: input argument which contains the mime type of the body
 text
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

add.mail.attachment

```
Syntax:
     function extern long add.mail.attachment(
                  const string i.filename,
                  const string i.displayname,
                  ref string o.error.message)
```

```
Description:
     Use this function to add an attachment to the email message.
Arguments:
     i.filename: input argument which contains the name of the file.
    i.displayname: input argument which contains the name to be displayed
 in the message
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

send.mail.message

```
Syntax:
     function extern long send.mail.message(
                ref string o.error.message)
Description:
     Use this function to send the email message.
Arguments:
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

cancel.mail.message

```
Syntax:
     function extern long cancel.mail.message(
                ref string o.error.message)
Description:
     Use this function to cancel the email message.
Arguments:
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

Custom fax solution

The custom fax solution interface consists of these functions:

init.fax

```
Syntax:
     function extern long init.fax(
                ref string o.error.message)
Description:
     Use this function to initialize the fax message.
Arguments:
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

add.fax.attachment

```
Syntax:
     function extern long add.fax.attachment(
                 const string i.filename,
                 const string i.displayname,
                 ref string o.error.message)
Description:
     Use this function to add an attachment to the fax message.
Arguments:
     i.filename: input argument which contains the name of the file.
    i.displayname: input argument which contains the name to be displayed
 in the message
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

set.fax.number

```
Syntax:
     function extern long set.fax.number (
              const string i.fax.number,
              ref string o.error.message)
Description:
     Use this function to set the fax number.
Arguments:
     i.fax.number: The faxnumber to which the fax must be sent.
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

send.fax

```
Syntax:
     function extern long send.fax(
                 ref string o.error.message)
Description:
     Use this function to send the fax.
Arguments:
     o.error.message: output argument which contains the error message,
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

cancel.fax

```
Syntax:
     function extern long cancel.fax(
                 ref string o.error.message)
Description:
     Use this function to cancel the fax.
Arguments:
     o.error.message: output argument which contains the error message,
```

```
or an empty string
Return value: 0 if processing without errors, otherwise the error code
```

Document provider

The document provider interface consists of this function:

```
Syntax:
    function extern xmlnode get.metadata()
Description:
    Provides the document provider meta data in datafields and optionally
 contexts.
Example implementation which assumes the bic_dom be included:

    domain ttxmlnode meta.node

    meta.node = dom.provider.metadata.init()

   dom.provider.metadata.add.field("filename", "Filename", "ttaud.path",
  meta.node)
    dom.provider.metadata.add.field("docname", "Document name",
"ttaud.path", meta.node)
    dom.provider.metadata.add.field("extension", "Extension", "ttst10",
meta.node)

    dom.provider.metadata.add.context("new", "New file", meta.node)
dom.provider.metadata.add.context("update", "Updated file", meta.node)

    return (meta.node)
    Description: XML node with metadata
```

The <bic_dom> include contains these supporting functions, which can be used in the get.metadata implementation:

dom.provider.metadata.init()

```
Syntax:
     function extern long dom.provider.metadata.init()
Description:
     Initializes the DOM Document Provider meta data structure.
```

dom.provider.metadata.add.field()

```
Syntax:
     function extern long dom.provider.metadata.add.field (string i.id,
string i.description, string i.domain, long io.metadata)
Description:
     Adds a provider data field to the meta data.
Arguments:
     i.id: Name of the provider data field.
```

```
     i.description: Description of the provider data field.
     i.domain: Domain of the provider data field.
     io.metadata: Node created by dom.provider.metadata.init().
```

dom.provider.metadata.add.context ()

```
Syntax:
     function extern long dom.provider.metadata.add.context (string i.id,
 string i.description, long io.metadata)
Description:
     Adds a provider context to the meta data.
Arguments:
     i.id: Name of the provider context.
     i.description: Description of the provider context.
     io.metadata: Node created by dom.provider.metadata.init().
```

dom.add.external.document()

```
Syntax: function extern long dom.add.external.document( const string
i.doc.provider,
                                                        const string
i.doc.provider.context,
                                                        const string
i.file.name,
                                                        const string
i.language,
                                                        boolean
i.own.transaction,
                                                      ref string o.error)
Description:
 Add a rendered document to Document Output Management (DOM). The given
 language is used to select language dependent DOM document type parts.
 A batch is created with one document
Precondition:
 The given document provider exists and a valid document provider rule
exists.
Postcondition:
 The document given in i.file.name is added to DOM as rendered document
 and is ready for distribution.
Arguments:
 i.doc.provider - The DOM Document Provider
 i.doc.provider.context - The Document Provider context used to differ►
entiate in DOM   expressions.
 i.file.name - The rendered document to be added. This includes the path
  on the LN server.
 i.language - Language to be used for selection of language dependent
document type   parts.
 i.own.transaction - whether or not the changes have to be commited to
the database
 o.error - Error message in case of an error. Return value:  long 0 in
case document has been added, <0 when not OK
 -2 Document Provider not found
 -3 Document Provider Context not found
 -4 Document Provider rule problem
```

```
-5 Document type used in document provider rule not found
-6 Batch cannot be created in DOM
-7 Problem in getting keywords from document provider rule object
-8 Document cannot be created in DOM
-9 Document Variables cannot be created in DOM
-10 Problem in initializing document distribution
-11 Most likely given file cannot be attached as BLOB
```

Appendix B: Using variables

You can use these variables during distribution.

You can use them in expressions that are linked to the document type:

- dom.document.language
- dom.document.company
- dom.document.reportcode
- dom.batch.user
- dom.batch.session
- dom.batch.context: The context specified by a Document Provider.
- dom.destination.name: The name specified at the Receiver Independent Destination.
- dom.destination.type: The constant names of the `ttrpi.dest` domain.
- dom.destination.custom
- dom.destination.address: The address of the destination. If the destination has type email, this variable is set only if there is only one to-address.
- You can also use the `##` expression. The text between the `##` is converted to a document type variable and must be mapped to a report variable in the report rule.

To get all email addresses for the destination, you can use this function in expressions that are linked to the Document Type: `function long dom.destination.get.email.addresses(ref string address.types(,), ref string addresses(,) mb)`

The output argument `address.types` is filled with the type of the email recipient, for example: “to”, “cc”, or “bcc”. The output argument `addresses` is filled with the recipient address. Both output arguments must be declared as based, the output argument `addresses` must be declared as multibyte as well. The function returns the number of email addresses present.

You can use these variables in expressions that are linked to the Report Rule:

- dom.batch.session
- dom.batch.session.pid
- dom.report.language

You can use these variables in expressions that are linked to the Document Provider Rule:

- dom.batch.context: The context specified by a Document Provider.
- dom.batch.language: The language specified by a Document Provider.

You can use these functions in expressions that are linked to the Report Rule:

- get.keyword1()
- get.keyword2()

- get.keyword3()
- get.keyword4()
- get.<report rule variable>()

These variables are used as special parameter setting during printing:

- splitfield; When the content or value of the expression in 'splitfield' changes, printing will continue in a new separated document.
- font.scaling; the value of this variable is used for the font scaling. Possible values are:
  - 'blank': No font scaling
  - > '2' font scaling factor.
  - < '2' default font scaling factor '9' is used.
- show.total.number.of.pages; indicate whether the total number of pages must be printed.

  For example: 'Page: 1' will be replaced with '1 - x' where 'x' is the total number of pages.

  The expression can contain:
  - 'true' print total number of pages.
  - 'yes' print total number of pages.
  - > '0' print total number of pages.
  - Any other value. Do not print total number of pages.

Appendix C: Faxing

Depending on the fax solution provider, you must configure the fax settings differently:

- Send an e-mail with attachments.
  - Specify the fax number in the e-mail address or e-mail subject.
  - Examples:
    - #FAXNUMBER#@efaxsend.com
    - #FAXNUMBER#-<security code>@informmaxion.faxservice.nl
    - `youraccountname@faxservice.nl` and mail subject contains <security-code>;#FAXNUMBER#;#subject#
- Implement a web service.
  - XML SOAP web services API

Appendix D: Command Line Interface document rendering and distribution

You can start the **Document Rendering and Distribution (ttrpi3210m000)** session from the command prompt.

To start the session, use this command:

- On UNIX/Linux:
  `bshell6.2 -server ttrpi3210m000 <options>`
- On Windows:
  `ntbshell -server ttrpi3210m000 <options>`

This table shows the options you can use:

Option	Explanation	Default value when not present
-d	Run as daemon.	No. Run once.
-dt x	Daemon running time. The daemon runs for 'x' minutes.	0
-s x	Additional Servers. 'x' contains number of additional servers.	0
-wt x	Query Wait Time 'x' seconds.	0
-mincomp x	Minimal Company number.	0
-maxcomp x	Maximal Company number.	999 or 9999
-ui	Show User Interface if possible.	No UI
-mindoct	x Minimal Document Type	“”
-maxdoct	x Maximal Document Type	“ZZZZZZZZZZZZ”

If no options are specified, the session is started with the default values. Rendering and distribution is processed once. After that the session closes.

Appendix E: Predefined document types

To aid in the modeling of document types, a list of predefined document types is available. These can serve as the basis or starting point in a DOM implementation. Administrators can import these document types and modify the documents as required.

The predefined document types are delivered in a `.zip` file and can be accessed from the **Additional Files (ttadv2570m000)** session.

To import this content, download the `tcgenLN_Document_Type_Templates.zip` file, which is located in the “tc-common” package, from the **Additional Files** session. Then use the **Import** option in the **Document Types (ttrpi2510m000)** session. As part of the import procedure, you must upload the `.zip` file.

This table shows the predefined document types in the `.zip` file:

Document Type	Description	Receiver type
RN	Purchase Order Return Note	Business Partner
POC	Purchase Order Claim	Business Partner
POR	Purchase Order Reminder	Business Partner
PCA	Purchase Contract Acknowledgement	Business Partner
PCT	Purchase Contract Termination Letter	Business Partner
RFQ	Request for Quotation	Business Partner
RFQUB	Letter for Unsuccessful Bidder	Business Partner
RFQR	Letter for Quotation Reminder	Business Partner
SVCON	Service Contract Document	Business Partner
SCA	Sales Contract Acknowledgement	Business Partner
SHIPPS	Shipment Packing Slip	User
SHIPBOL	Shipment Bill of Lading	User
SHIPPL	Shipment Packing List	User
SHIPMF	Shipping Manifest	User
TFREM	Reminder Letter (AR)	Business Partner
INVOICE	Invoice	Business Partner

Document Type	Description	Receiver type
PO	Purchase Order	Business Partner
SVCALL	Service Call	Business Partner
SVCCRMA	Service Customer Claim RMA	Business Partner
SVCCA	Service Customer Claim Acknowledgement	Business Partner
SVSC	Service Supplier Claim	Business Partner
SO	Sales Order	Business Partner
QUOTE	Sales Quote	Business Partner
PROJ_CONTR	Project Contract Acknowledgement	Business Partner