Infor LN Public Interfaces & Process Extensions Reference Guide (On-premises)

Release 10.6

Publication Information

Release: Infor LN 10.6

Publication date: September 9, 2025

Contents

About this guide

Intended audience

This guide is intended for IT professionals working in implementation projects or IT optimization phases for Infor LN. Basic knowledge about the Infor LN software structure and Infor LN's 4GL programming language is a pre-requisite.

Related documents

You can find these documents on docs.infor.com:

- *Infor LN Studio Application Development Guide*
- *Infor LN Studio Integration Development Guide*
- *Infor LN Extensions Development Guide*

You can find the *Infor ES Programmer's Guide* in KB2924522. The content of this guide is also available in the help pages of Infor LN Studio.

Contacting Infor

If you have questions about Infor products, go to Infor Concierge at https://mingle-portal.us2.prd3.inforcloudsuite.com/v2/CONCIERGE_PRD and create a support incident.

For the latest documentation, go to Documentation Central at docs.infor.com. We recommend that you check this website periodically for updated documentation. If you have comments about Infor documentation, contact documentation@infor.com.

Chapter 1 Introduction

Infor LN is a standard ERP application with rich functionality. With its built-in flexibility with parameters, workflows, dynamic processes, it can be adjusted to serve the business processes in the industries Infor LN is designed for. To close the small gaps between the standard functionality and the specific business needs, Infor LN offers a variety of extensibility possibilities. The main goal of extensibility is to develop the last-mile functionality for your organization without changing the core standard software components and using only the public interfaces of the standard application. In this way, you can develop the extensions fully separated from the standard components and upgrading the standard software will therefore not result in additional efforts and costs for upgrading the customizations. Extensions are not influenced by the upgrade process.

Public Interfaces

Public Interfaces are methods with LN application functionality that can be called from extensions.

Public Interfaces explained

The development of extensions can be made easier if methods from the LN Application can be used. This can be done by using the so-called LN Public Interfaces. LN Public Interfaces are functions in the LN Application that are available for anyone who develops extensions on LN. The available LN Public Interfaces are visible in LN Studio and in the Extension Modeler. LN Public Interfaces are generic functions with a certain level of complexity that will likely be used by multiple customers. Simple read actions, for example, needs to be developed by customers or implementation partners themselves. Moreover, LN Public Interfaces will perform something what cannot easily be achieved with one or more standard sessions or processes in LN.

How to request a new Public Interface

Apply the following process if you need a new LN Public Interface:

1. Evaluate if the new LN Public Interface is generic and contains a certain level of complexity. Make sure the required feature cannot be achieved with personalizing a standard session or process.

2. Create an incident and clearly describe the required LN Public Interface. Use the Excel Sheet “Request Template for LN Public Interfaces & Process Extensions)” which is attached to KB2003722 in the Infor Customer Portal.
3. Infor Support will create a defect for this incident.
4. Infor Development will review the requested LN Public Interface and will either develop the new LN Public Interface or reject the request. If the LN Public Interface is developed it will be released via the regular delivery process.
5. After the new solution has been installed the customer can use the new LN Public Interface in the extension modeler and/or in LN Studio.
6. Infor Support will complete the incident.

Public Interface example

Find below an example of one of the LN Public Interfaces. This Public Interface converts an amount to another currency.

```
long Common.ConvertAmount(
           domain    tcncmp        iFinancialCompany,
           domain    tcamnt        iSourceAmount,
           domain    tcccur        iSourceCurrency,
           domain    tcrtyp        iExchangeRateType,
           domain    tcdate        iRateDateUTC,
           domain    tcccur        iTargetCurrency,
    ref    domain    tcamnt        oTargetAmount,
    ref    domain    tcmcs.s999m   oExceptionMessage mb,
    ref              long          oExceptionID)
```

It is very important to use LN Public Interfaces properly and to catch the errors. For this reason, each Public Interface has the output arguments 'oExceptionMessage' and 'oExceptionID'.

If the Public Interface returns a value unequal to zero, then argument oExceptionMessage is filled for sure with the latest exception message and argument oExceptionID is a reference to an XML-object that contains some more information about the exception. This extra information can be retrieved by using the Public Interface 'Exception' in otcextextapi. It is also important to free up memory by calling Public Interface Exception.Delete(...).

If the Public Interface returns zero, arguments oExceptionMessage and oExceptionID could be filled due to information messages. So, in fact only the return value indicates if a public interface is successful or not.

The next example shows how to implement error handling when using LN Public Interfaces, in this case Common.ConvertAmount to convert an amount to another currency.

```
#pragma used dll "otcextextapi"
#pragma used dll "otcextemmapi"

                long          exception.id, i
        domain   tcmcs.s999m   exception.message

if Common.ConvertAmount(..., exception.message, exception.id) <> 0 then
        |* Exception(s) found.
        for i = 1 to Exception.NumberOfMessages(exception.id)
                dal.set.error.message("@"& Exception.GetMessage(i))
        endfor
        Exception.Delete(exception.id)
        return(DALHOOKERROR)
else
        |* Call was successful. Exception messages could exist!
        Exception.Delete(exception.id)
endif
```

To be able to use the Public Interface in your extensions, you need to add a “#pragma used dll” statement for the DLL that contains the Public Interface. The DLL names can be found in the documentation of the available Public Interfaces. Note that the DLL name needs to be preceded by an “o” in the #pragma-statement.

Available Public Interfaces

The next chapters of this document describe the available Public Interfaces for Infor LN (10.6) and what their usage is.

If the Public Interfaces described in this document are not shown in the Extension Modeler or Infor LN Studio in your environment, it may be necessary to apply a Knowledge Base article (KB) that can be found in the Infor Customer Portal. The applicable KB number is mentioned in the Public Interface description.

Public Interface to call BDE methods

BDEs (Business Data Entities) are components in Infor LN that are the base for Infor LN’s web services. A special Public Interface is available to call the BDE methods within extensions: BDE.ExecuteMethod. Note that this will not be a real (SOAP) web service call, but the BDE method will be executed internally in LN.

The available BDEs are shown in session “Business Objects” (ttadv7500m000). The BDEs that are available to be called as Public Interface must satisfy the following conditions:

- The name must not end with “BOD”
- The Type of Business Object must be ‘Public’

- The Maintained in Studio checkbox must be checked
- The action Download WSDL must be available for the BDE.

Proxy DLL

BDE methods are functions that have an XML document as input (the request), and the output is either a response XML document (in case the method succeeded) or a result XML document (in case the method failed). To build the request or to retrieve data from the response or result, you need to use a Proxy DLL that is generated based on the WSDL (Web Service Description Language). To be able to generate a proxy DLL you need to have a Development license (product ID 10146).

Perform the following steps to generate a Proxy DLL:

1. Select the BDE you want to call in your extension in session “Business Objects” (ttadv7500m000).
2. Click **Actions>Download WSDL**.
3. Save the WSDL in a folder on your PC.
4. Start Infor LN Studio. At least version 10.7.0.389 is needed. For more information about Infor LN Studio see the Related Documents section of this document.
5. If you don’t have an Activity yet, create a new one.
6. Create a new software component of type Library. Enter a name and a description and click **Finish**.
7. A new library is created, and the editor is opened for it. Check out the library.
8. Right-click in the editor and select **Generate Source from WSDL**.
9. The “Generate Library from WSDL file” displays. Browse to the WSDL file you saved on your PC in step 3.
10. Click **OK**.
11. The source is generated. Click **Save** button.
12. If you need to use the proxy DLL in another LN Studio Activity or another Extensibility Activity, the proxy DLL must be committed by (partially) ending the current Activity.

Coding example

This program creates and releases a Service Order using the Create and ReleaseOrder methods of the ServiceOrder_v4 BDE. The numbers behind |# refer to remarks after the example program.

```
#pragma used dll "otcextextapi"                                              |# 1
#pragma used dll "otcextbdeapi"                                              |# 2

function create.and.release.service.order()
{
```

```
long    request, response, result
long    so.order, so.act, so.ass
long    ro
long    da
long    ret
long    exceptionid
string  exceptionmessage(1000) mb

domain  tcorno  service.order

db.retry.point()                                                        |# 3

request = Create_CreateRequest.New()                                    |# 4

so.order = Create_ServiceOrder_v4.New("JJD")                            |# 5
Create_ServiceOrder_v4.SetserviceCenter(so.order, "100")                |# 6

da = Create_DataArea.New()                                              |# 7
ret = Create_DataArea.AddServiceOrder_v4(da, so.order)
Create_CreateRequest.SetDataArea(request, da)

so.act = Create_ServiceOrderActivity.New()                              |# 8
ret = Create_ServiceOrder_v4.AddServiceOrderActivity(so.order, so.act)  |# 9
                                                                        |# 6
Create_ServiceOrderActivity.SetserviceOrderActivityPlannedStartTime(so.act, utc.num())
Create_ServiceOrderActivity.SetserviceOrderActivityPlannedFinishTime(
                                                         so.act, utc.num() + 3600)

so.ass = Create_ActivityEngineer.New()                                  |# 10
ret = Create_ServiceOrderActivity.AddActivityEngineer(so.act, so.ass)   |# 11
Create_ActivityEngineer.SetassignmentEngineer(so.ass, "0099101")        |# 6

ret = BDE.ExecuteMethod("ServiceOrder_v4",                              |# 12
                        "Create",
                        request,
                        response,
                        result,
                        exceptionmessage,
                        exceptionid)

if ret <> 0 then                                                        |# 13
    abort.transaction()
    message(exceptionmessage)
    Exception.Delete(exceptionid)
    return
endif

ret = Create_CreateResponse.GetDataArea(response, da)                   |# 14
```

```
so.order = Create_DataArea.GetServiceOrder_v4(da)
service.order = Create_ServiceOrder_v4.GetserviceOrderCode(so.order))

xmlDelete(request)
xmlDelete(response)

request = ReleaseOrder_ReleaseOrderRequest.New()                        |# 15

ro = ReleaseOrder_ServiceOrder_v4.New(service.order)

da = ReleaseOrder_DataArea.New()
ret = ReleaseOrder_DataArea.AddServiceOrder_v4(da, ro)
ReleaseOrder_ReleaseOrderRequest.SetDataArea(request, da)

ret = BDE.ExecuteMethod("ServiceOrder_v4",                              |# 16
                        "ReleaseOrder",
                        request,
                        response,
                        result,
                        exceptionmessage,
                        exceptionid)

if ret <> 0 then                                                        |# 13
    abort.transaction()
    message(exceptionmessage)
    Exception.Delete(exceptionid)
    return
endif

Exception.Delete(exceptionid)                                           |# 17

commit.transaction()                                                    |# 18
message(sprintf$("Service Order %s has been created and released.", service.order))
}
```

Explanation:

1 ottextextapi contains the functions for exception handling.

2 ottextbdeapi contains the function BDE.ExecuteMethod().

3 Transaction handling must always be in the calling program of the BDE method. Note that when you execute a BDE method in an extension hook (for example in the Before Save hook in a table extension, your hook must not start an own transaction, because it needs to be executed in the transaction of the program that calls the table extension.

4 This is the initialization of a new request that must be built up. Use always the functions from the proxy DLL with the method name followed by an "_" as prefix.

5 The example creates a Service Order. The root component is the Service Order itself and the component name is the same as the BDE name, ServiceOrder_v4 in this case. Because the

Service Order number is a mandatory attribute of the Service Order, it must be passed as an argument. In this case the series in which the Service Order must be created.

6 Other attributes of the Service Order can be set as well. This also applies to the attributes of other components of the Service Order (Activity and Assignment Engineer) which are added later.

7 The Service Order is not directly connected to the request, but within the DataArea of the request. Here the DataArea is created, the Service Order is linked to it and the DataArea is connected to the request.

8 An Activity is created and

9 linked to the Service Order.

10 An Assignment Engineer is created and

11 linked to the Activity.

12 This is the call of the Public Interface to execute the BDE method to create the Service Order including the Activity and the Assignment Engineer. The XML request that has been built up by calling the functions is the proxy DLL looks like:

```
<Create.CreateRequest>
  <DataArea>
    <ServiceOrder_v4>
      <serviceOrderCode>JJD</serviceOrderCode>
      <serviceCenter>100</serviceCenter>
      <ServiceOrderActivity>
        <serviceOrderActivityPlannedStartTime>2019-05-03T07:22:35+02:00
        </serviceOrderActivityPlannedStartTime>
        <serviceOrderActivityPlannedFinishTime>2019-05-03T08:22:35+02:00
        </serviceOrderActivityPlannedFinishTime>
        <ActivityEngineer>
          <assignmentEngineer>0099101</assignmentEngineer>
        </ActivityEngineer>
      </ServiceOrderActivity>
    </ServiceOrder_v4>
  </DataArea>
</Create.CreateRequest>
```

13 If the method fails, the transaction must be aborted. When the transaction is started outside your code (for example when your code is part of a table extension), you don’t need to abort, but return the error (DALHOOKERROR). The *exceptionmessage* field contains the most import message from the BDE method call. If you need more messages, you can use the Exception object (see public Interfaces for Extensibility) to retrieve the other messages.

14 Retrieve the generated Service Order Number from the response by getting the DataArea of the reponse XML document, getting the Service Order from the DataArea and then getting the Service Order Number attribute of the Service Order. The XML response of the Create method looks like (the actual response contains much more elements, but they are left out for readability):

```
<CreateResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <DataArea>
    <ServiceOrder_v4>
      <serviceOrderCode>JJD000235</serviceOrderCode>
      <customerOrderReference></customerOrderReference>
      ...
      <ServiceOrderActivity>
        <serviceOrderActivityLineNumber>10</serviceOrderActivityLineNumber>
        <activityStatus>free</activityStatus>
        <serviceOrderActivityPlannedStartTime>2019-05-03T05:22:35Z
        </serviceOrderActivityPlannedStartTime>
        <serviceOrderActivityPlannedFinishTime>2019-05-03T06:22:35Z
        </serviceOrderActivityPlannedFinishTime>

        ...
        <ActivityEngineer>
          <assignmentLineNumber>1</assignmentLineNumber>
          <assignmentStatus>assigned</assignmentStatus>
          <assignmentEngineer>0099101</assignmentEngineer>
          ...
        </ActivityEngineer>
      </ServiceOrderActivity>
    </ServiceOrder_v4>
  </DataArea>
</CreateResponse>
```

15 Use this Service Order Number to build up the ReleaseOrder request.

16 Execute the ReleaseOrder method. If this method fails, the transaction needs to be aborted. In this case the created Service Order will be reverted as well.

17 Free the Exception object, which may contain information messages.

18 Commit the transaction. See also note 3 and 11.

Process Extensions

Process Extensions are the opposite of Public Interfaces. Instead of calling standard LN methods (Public Interfaces) from the extensions, the standard LN application calls the Process Extensions, provided that they are implemented.

Process Extensions explained

With the extension types table extension, session extension, etc. the behavior and functionality of Infor LN can be changed in many ways (see the Infor LN Extensions Development Guide). But there may be additional requirements where intervening the standard functionality is required, which cannot be achieved with extending the table, session or any other component. Examples are processing and/or print sessions where additional selection ranges are required or core algorithms for composing invoices where the standard criteria are not sufficient, especially when customer defined fields need to be included in the selection and/or algorithm.

How to request a new Process Extension

The possibility to skip objects in processing and/or print sessions is a generic approach and can be used in every session where this Process Extension type has been implemented in the standard application. If you need to additional criteria whether or not objects may be processed by a session and that session has not been prepared for this Process Extension type, you can request this.

The other types of Process Extensions, where core algorithms can be extended, can be requested as well.

Apply the following process if you need a new Process Extension:

1. Evaluate if the request is generic. Make sure the required feature cannot be achieved with personalizing a standard session or process or by using one of the other extension points (for example a table extension).
2. In case of a skip Process Extension type, create an incident in the Infor Customer Portal. In case of another Process Extension type, create an enhancement request in Infor's ERS system.
3. Infor Development will review the requested Process Extension and will either include it in the standard product or reject the request. If the Process Extension is developed it will be released via the regular delivery process.
4. After the new solution has been installed the customer can implement the new Process Extension using the extension modeler.
5. If an incident was created, Infor Support will complete the incident.

Available Process Extensions

The last chapter of this document describes the available Process Extensions for Infor LN (10.6) and how they must be implemented.

If the Process Extensions described in this document are not shown in the Extension Modeler in your environment, it may be necessary to apply a Knowledge Base article (KB) that can be found in the Infor Customer Portal. The applicable KB number is mentioned in the Process Extension description.

Chapter 2 Public Interfaces for Extensibility

Public Interfaces for Exception

The following functions are available:

Exception.Delete

Exception.GetMessage

Exception.NumberOfMessages

Exception.Delete

DLL:	tcextextapi This function is available from KB1970778.

Syntax:	`Exception.Delete(`
	`        ref             long             ioExceptionID )`

Usage:	`        Expl:   This function deletes the exception to free memory.`
	`        Pre:    ioExceptionID should refer to an Exception.`
	`        Post:   None`
	`        Input:`
	`                ioExceptionID   - the exception id.`
	`        Output:`
	`                ioExceptionID   - the exception id.`
	`        Return: None`

Exception.GetMessage

DLL:	tcextextapi This function is available from KB1970778.
Syntax:	`Exception.GetMessage(`

	`long iExceptionID,`
	`long iMessageIndex,`
	`ref domain tcmcs.s999m oMessageDescription mb )`

Usage:	`Expl: This function reads message description from all messages in`
	`the XML identified by iExceptionID for a certain index.`
	`Pre: iExceptionID should refer to an Exception.`
	`Post: None`
	`Input:`
	`iExceptionID - the exception id.`
	`iMessageIndex - the index for the message to be returned.`
	`iMessageIndex should be greater than zero and`
	`less than the number of messages.`
	`Output:`
	`oMessageDescription - The found message.`
	`Return: None`

Exception.NumberOfMessages

DLL:	tcextextapi This function is available from KB1970778.

Syntax:	`long Exception.NumberOfMessages(`
	`long iExceptionID )`

Usage:	`Expl: This function determines the number if messages that have been`
	`stored in the XML where iExceptionID refers to.`
	`Pre: iExceptionID should refer to a valid XML with the structure`
	`as described above.`
	`Post: None`
	`Input:`
	`iExceptionID - the exception id that points to the XML.`
	`Output:`
	`None`
	`Return: The number of found messages. If iExceptionID does not refers`
	`to a valid XML the return value is 0.`

Chapter 3 Public Interfaces for Common

Public Interfaces for Common

The following functions are available:

Common.ConvertAmount

Common.ConvertQuantity

Common.GetFinancialCompanyOfEntity

Common.GetFormattedAddress

Common.GetParameters

Common.RoundAmount

Common.RoundQuantity

Common.ConvertAmount

DLL:	tcextemmapi This function is available from KB1970778.

Syntax:	
	```
	long Common.ConvertAmount(
	                domain  tcncmp          iFinancialCompany,
	                domain  tcamnt          iSourceAmount,
	                domain  tcccur          iSourceCurrency,
	                domain  tcrtyp          iExchangeRateType,
	                domain  tcdate          iRateDateUTC,
	                domain  tcccur          iTargetCurrency,
	        ref     domain  tcamnt          oTargetAmount,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID )
	```

Usage:	
	```
	        Expl:   This function converts an amount in one currency to
	                an amount in another currency.
	                The rates are read based on the given iRateDateUTC for the
	                given iExchangeRateType. If the iExchangeRateType is not passed,
	                the function will use the Internal Rate Type defined for the
	                company.
	```

```
        Pre:    Company must be a valid company.
        Post:   None
        Input:
                iFinancialCompany       - Financial Company: Mandatory
                iSourceAmount           - Source Amount
                iSourceCurrency         - Source Currency: Mandatory
                iExchangeRateType       - Source Rate Type; if not filled,
                                          the Internal Rate Type will be used.
                iRateDateUTC            - Rate Date (UTC)
                iTargetCurrency         - Target Currency: Mandatory
        Output:
                oTargetAmount           - Target Amount, result of the
                                          conversion.
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Retrun: 0                       - Amount is converted.
                DALHOOKERROR            - Otherwise.
```

Common.ConvertQuantity

DLL: tcextcomapi

This function is available from KB3627590.

Syntax:

```
long Common.ConvertQuantity(
                domain  tcitem          iItem,
                domain  tccuni          iFromUnit,
                domain  tcqst1          iFromQuantity,
                domain  tccuni          iToUnit,
        ref     domain  tcqst1          oToQuantity,
        ref     domain  tcmcs.s999m     oExceptionMessage mb,
        ref             long            oExceptionID )
```

Usage:

```
        Expl:   This function converts a quantity specified in a unit to
                a quantity specified in another unit.
                Rounding will not be done. Common.RoundQuantity can be used for
                rounding.
        Post:   None
        Input:  iItem                   - Item
                iFromUnit               - From Unit: Mandatory
                iFromQuantity           - From Quantity
                iToUnit                 - To Unit: Mandatory
        Output: oToQuantity             - To Quantity
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
```

```
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0               - Quantity is converted.
                <> 0            - Otherwise.
```

Common.GetFinancialCompanyOfEntity

DLL: tcextemmapi

This function is available from KB2052340.

Syntax:

```
long Common.GetFinancialCompanyOfEntity(
                domain  tcncmp           iLogisticCompany,
                domain  tcemm.enty       iEntityType,
                domain  tcemm.enio       iEntity,
        ref     domain  tcncmp           oFinancialCompany,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function gets the financial company of the logistic
                company linked to entity.
        Pre:
        Post:
        Input:  i.logistic.company      - Logistic Company: Mandatory
                i.entity.type           - Entity Type: Mandatory
                i.entity                - Entity: Mandatory
        Output:
                oFinancialCompany       - Financial company of the enterprise
                                          unit linked to the entity.
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0                       - Financial company is found.
                DALHOOKERROR            - Otherwise.
```

Common.GetFormattedAddress

DLL: tcextcomapi

This function is available from KB2013553.

Syntax:

```
long Common.GetFormattedAddress(
                domain  tccom.cadr       iAddress,
                        long             iNumberOfLines,
                domain  tccty            iFromCountry,
        ref             string           oFormattedAddress(,) fixed,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl.:  This functions formats an given addresscode into a
                formatted address array of a given number of lines.
        Pre:    iAddress must exists in tccom130
                Multibyte Array used for oFormattedAddress must be
                declared and initialized.
        Post:   NA
        Input:
                iAddress               - The address code for which the
                                         formatted address must be generated.
                                         (mandatory)
                iNumberOfLines         - The maximum number of lines of the
                                         formatted address.(mandatory)
                iFromCountry           - When iFromCountry is filled,
                                         the country information will
                                         only be used in the formated
                                         address when it differs from
                                         iFromCountry.
                                         When left empty always the country
                                         information is used in the formated
                                         address.
|*Example       if i.from.country = "NLD" the Dutch addresses will not have
                country information in the formatted address. This can be used
                when a company located in "NLD" sends invoices or mailings to
                addresses in "NLD".

                        Moonen Shipyards BV
                        Kade 35
                        1780 AA Den Helder

                        The addresses outside "NLD" will have country information in
                        the formatted address:

                        Moonen F›¼□ördertechnik GmbH
                        Balhorner Feld 1
                        33106 Paderborn
                        Deutschland

                if i.from.country = empty both addresses will have country
                information in the formatted address:

                        Moonen Shipyards BV
                        Kade 35
                        1780 AA Den Helder
                        Nederland

                        Moonen F›¼□ördertechnik GmbH
                        Balhorner Feld 1
                        33106 Paderborn
                        Deutschland
|
        Output: oFormattedAddress      - Array with formatted address lines.
                oExceptionMessage      - The last message if the return
                                         value is not equal to 0.
                                         If more than one  message is
                                         given, these are present in the
                                         oExceptionID
                oExceptionID           - An ID that refers to all error
                                         information. Use the functions in
```

```
                                          Exception to get all relevant
                                          information.
        Return values:
                0                       - Formatted address is returned
                DALHOOKERROR            - on errors
```

Common.GetParameters

DLL: tcextcomapi

This function is available from KB2210841.

Syntax:

```
long Common.GetParameters(
                domain  tcncmp          iCompany,
                domain  tctabl.c        iTableCode,
        ref     domain  tcmcs.s999m     oExceptionMessage mb,
        ref             long            oExceptionID,
                                        ... )
```

Usage:

```
        Expl:   This function reads one or more parameters from a standard LN
                parameter table.
                Note that this function uses variable arguments, for input and
                output. Data is retrieved via name value pairs: the field
                mnemonic and the field value.

                Several parameter-fields are stored as arrays. Individual elements
                of these parameter-fields can be retrieved by adding the element
                number to the field-mnemonic. Example: "qipo(28)", "mvst(2)".

                Example: When the 'Minimal Picks' and 'Quarantine Inventory
                Available for Planning by Origin' are needed from the
                Inventory Handling parameters, the call of
                this public interface is as follows:

                if Common.GetParameters(
                                |* Fixed arguments:
                                        company,                --> input
                                        "whinh000",             --> input
                                        exception.message,      --> output
                                        exception.id,           --> output
                                |* Variable arguments:
                                        "minp",                 --> input
                                        minimal.picks,          --> output
                                        "qipo(1)",              --> input
                                        l.qipo(1),              --> output
                                        "qipo(2)",              --> input
                                        l.qipo(2),              --> output
                                        "qipo(3)",              --> input
                                        l.qipo(3)) <> 0 then    --> output
                        |* Error, do something
                        Exception.Delete(exception.id)
                endif

        Pre:    None
        Post:   None
        Input:
                iCompany          - Company: Mandatory
```

```
                                    Note: for reading "tccom000" always the current
                                    company is taken.
                iTableCode        - Code of the parameter table: Mandatory.
                                    Example: "tpctm000"
                ...               - The field mnemonic of the required parameters.
        Output:
                oExceptionMessage - The last message if the return value is not
                                    equal to 0. If more than one  message is
                                    given, these are present in the oExceptionID.
                oExceptionID      - An ID that refers to all error information.
                                    Use the functions in Exception to get all
                                    relevant information.
                ...               - The value of the required parameters.
        Return: 0                 - Parameter(s) read.
                <> 0              - Error occurred.
```

Common.RoundAmount

DLL: tcextcomapi

This function is available from KB2210841.

Syntax:

```
long Common.RoundAmount(
                        double            iUnroundedAmount,
                domain  tcmcs.st14        iDomainName,
                domain  tcccur            iCurrency,
        ref             double            oRoundedAmount,
        ref     domain  tcmcs.s999m       oExceptionMessage mb,
        ref             long              oExceptionID )
```

Usage:

```
        Expl.:  This functions rounds an amount with a given domain and currency.
        Pre:    NA
        Post:   NA
        Input : iUnroundedAmount        - The amount that must be rounded.
                iDomainName             - The domain name. (mandatory)
                iCurrency               - The currency. (mandatory)
        Output: oRoundedAmount          - The rounded amount.
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return values:
                0                       - Rounded amount is returned
                <> 0                    - on errors
```

Common.RoundQuantity

DLL:	tcextcomapi This function is available from KB2054574.

Syntax:	
	```
	long Common.RoundQuantity(
	                        double           iUnroundedQuantity,
	                domain  tcmcs.st14       iDomainName,
	                domain  tccuni           iUnit,
	        ref             double           oRoundedQuantity,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl.:  This function rounds a quantity dependending on given
	                domain and unit.
	        Pre:    NA
	        Post:   NA
	        Input : iUnroundedQuantity      - The quantity that must be rounded.
	                iDomainName             - The domain name. (mandatory)
	                iUnit                   - The unit. (mandatory)
	        Output: oRoundedQuantity        - The rounded quantity.
	                oExceptionMessage       - The last message if the return
	                                          value is not equal to 0.
	                                          If more than one  message is
	                                          given, these are present in the
	                                          oExceptionID
	                oExceptionID            - An ID that refers to all error
	                                          information. Use the functions in
	                                          Exception to get all relevant
	                                          information.

	        Return values:
	                0                       - Rounded quantity is returned
	                DALHOOKERROR            - on errors
	```

Chapter 4 Public Interfaces for Item

Public Interfaces for Item

The following functions are available:

Item.GetCostingData

Item.GetData

Item.GetOrderingData

Item.GetCostingData

DLL:	tiextcprapi This function is available from KB2066826.

Syntax:	
	```
	long Item.GetCostingData(
	                domain  tcncmp          iLogisticCompany,
	                domain  tcitem          iItem,
	                domain  tcemm.grid      iEnterpriseUnit,
	                        boolean         iForceRead,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID,
	                                        ... )
	```

Usage:	
	```
	        Expl:   This function retrieves data from Item - Costing Data.
	                Individual fields are retrieved, multiple fields can be retrieved
	                in one call.

	                The enterprise.unit argument is only used when the parameter
	                Standard Cost by Enterprise Unit is set to Active in Implemented
	                Software Components. In that case it is necessary to supply an
	                Enterprise Unit when the item-type requires this. For item types
	                Cost or Service, no Enterprise Unit is expected.

	                The set of supported fields for retrieval is limited. The ones
	                marked with EU only have relevance when Standard Cost by
	                Enterprise Unit is Active.

	                Mnemonic        -    Description                 -   Scope
	                  type          - Item Costing Type              - EU
	```

```
                  base          - Standard Cost Base            - EU
                  cwar          - Warehouse                     - EU
                  cofc          - Supplying Purchase Office     - EU
                  scos          - Costing Source                - EU
                  sueu          - Supplying Enterprise Unit     - EU
                  spit          - Surcharges by Item            -
                  vpwh          - Surcharges by Warehouse       -
                  ccur          - Item Costing Currency         -
                  ltcp          - Last Calculation Date         -
                  chrt          - Standard Cost Component Scheme-
                  coyn          - Combined Ownership Allowed    -
                  inlc          - Included Landed Costs         -
                  lcst          - Landed Costs Set              -

                Note: this function uses variable arguments, for input and
                output. Data is retrieved via value pairs: specify the field
                and the field value.
                Example: when item costing warehouse and currency are needed,
                the function must be called as follows:

                if Item.GetCostingData(
                                        |* Fixed arguments:

                                        company,                  --> input
                                        item,                     --> input
                                        enterprise.unit,          --> input
                                        force.read,               --> input
                                        exception.message,        --> output
                                        exception.id,             --> output

                                        |* Variable arguments:

                                        "cwar",                   --> input
                                        warehouse,                --> output
                                        "ccur",                   --> input
                                        currency) <> 0 then       --> output
                        |* Error, do something
                        Exception.Delete(exception.id)
                endif

        Pre:    None
        Post:   None
        Input:
                iLogisticCompany        - Logistic Company: Mandatory
                iItem   -               - Item: Mandatory
                iEnterpriseUnit         - Enterprise Unit - not mandatory
                iForceRead              - Option to force new query instead of
                                          using cached information
                ...                     - The field mnemonic of the required
                                          field.
        Output:
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
                ...                     - The value of the required field.
        Return: 0                       - Data read
```

	`DALHOOKERROR            - Otherwise.`

Item.GetData

DLL:	tcextibdapi This function is available from KB2043720.

Syntax:	
	```
	long Item.GetData(
	                domain  tcncmp          iLogisticCompany,
	                domain  tcitem          iItem,
	                domain  tcsite          iSite,
	                        boolean         iForceRead,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID,
	                                        ... )
	```

Usage:	
	```
	        Expl:   Retrieve data from Items
	                The required field is specified by the field menomonic,
	                the output value is filled in the output argument depending on
	                the data type.
	                This function is Multi Site aware, data is read from general
	                level or site level, depending on the input arguments Site
	                and implementation phase of Multi Site.

	                Note: this function uses variable arguments, for input and
	                output. Data is retrieved via value pairs: specify the field
	                and the field value.
	                Example: when item project and customized are needed, the
	                function must be called as follows:

	                if Item.GetData(
	                                        |* Fixed arguments:

	                                        company,                --> input
	                                        item,                   --> input
	                                        site,                   --> input
	                                        force.read,             --> input
	                                        exception.message,      --> output
	                                        exception.id,           --> output

	                                        |* Variable arguments:

	                                        "cprj",                 --> input
	                                        project,                --> output
	                                        "cust",                 --> input
	                                        customized) <> 0 then --> output
	                        |* Error, do something
	                        Exception.Delete(exception.id)
	                endif
	        Pre:    None
	        Post:   None
	        Input:
	                iLogisticCompany        - Logistic Company: Mandatory.
	                iItem   -               - Item: Mandatory.
	                iSite   -               - Site: Not Mandatory.
	                iForceRead              - Option to force new query in stead of
	```

```
                                        using cached information.
                ...                   - The field mnemonic of the required
                                        field.
        Output:
                oExceptionMessage     - The last message if the return
                                        value is not equal to 0.
                                        If more than one  message is
                                        given, these are present in the
                                        oExceptionID.
                oExceptionID          - An ID that refers to all error
                                        information. Use the functions in
                                        Exception to get all relevant
                                        information.
                ...                   - The value of the required field.
        Return: 0                     - Data read.
                DALHOOKERROR          - Otherwise.
```

Item.GetOrderingData

DLL: tcextibdapi

This function is available from KB2043720.

Syntax:

```
long Item.GetOrderingData(
                domain  tcncmp           iLogisticCompany,
                domain  tcitem           iItem,
                domain  tcsite           iSite,
                        boolean          iForceRead,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID,
                                         ... )
```

Usage:

```
        Expl:   Retrieves data from Items - Ordering.
                The required field is specified by the field menomonic,
                the output value is filled in the output argument depending on
                the data type.
                This function is Multi Site aware, data is read from general
                level or site level, depending on the input arguments Site
                and implementation phase of Multi Site.

                Note: this function uses variable arguments, for input and
                output. Data is retrieved via value pairs: specify the field
                and the field value.
                Example: when item ordering warehouse and order interval are
                needed, the function must be called as follows:

                if Item.GetData(
                                      |* Fixed arguments:

                                      company,                  --> input
                                      item,                     --> input
                                      site,                     --> input
                                      force.read,               --> input
                                      exception.message,        --> output
                                      exception.id,             --> output

                                      |* Variable arguments:
```

```
                                   "cwar",                  --> input
                                   warehouse,               --> output
                                   "oint",                  --> input
                                   order.interval) <> 0 then --> output
                   |* Error, do something
                   Exception.Delete(exception.id)
           endif
   Pre:    None
   Post:   None
   Input:
           iLogisticCompany        - Logistic Company: Mandatory
           iItem   -               - Item: Mandatory
           iSite   -               - Site: Not Mandatory
           iForceRead              - Option to force new query in stead of
                                     using cached information.
           ...                     - The field mnemonic of the required
                                     field.
   Output:
           oExceptionMessage       - The last message if the return
                                     value is not equal to 0.
                                     If more than one  message is
                                     given, these are present in the
                                     oExceptionID.
           oExceptionID            - An ID that refers to all error
                                     information. Use the functions in
                                     Exception to get all relevant
                                     information.
           ...                     - The value of the required field.
   Return: 0                       - Data read.
           DALHOOKERROR            - Otherwise.
```

Chapter 5 Public Interfaces for Job Shop

Public Interfaces for ProductVariant

The following functions are available:

ProductVariant.GenerateProductStructureWithoutProject

ProductVariant.GenerateProductStructureWithoutProject

DLL: tiextpcfapi

This function is available from KB2295738.

Syntax:

```
long ProductVariant.GenerateProductStructureWithoutProject(
                domain  tccpva           iProductVariant,
                domain  tcdate           iReferenceDate,
        ref     domain  tcitem           oItem,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This Public Interface generates a Product Structure based on a
                configured Product Variant, without a Project. This can also be
                done using session tipcs2220m000.
                The configured Product Variant can be created using
                ProductVariant.StartConfigurator or via the Product Configurator
                (tipcf5120m000).

        Pre:    Db.retry point must be set
                Product Variant must be configured.
        Post:   Transaction must be aborted or committed.

        Input:  iProductVariant         - Product Variant (Mandatory). Cannot be
                                          for an Assembly Item
                iReferenceDate          - Reference Date (Optional). If empty,
                                          the current date is used.
        Output: oItem                   - The created custom Item.
                oExceptionMessage       - The last message if any message is
                                          found. If more than one message is
                                          given, these are present in the
                                          oExceptionID.
                oExceptionID            - An ID that refers to the exception
                                          information. Use the functions in
```

	```
	                                          Exception to get all relevant
	                                          information.
	        Return: 0                       - Structure generation completed.
	                <> 0                    - Otherwise.
	```

Chapter 6 Public Interfaces for Purchase

Public Interfaces for PurchaseRequisition

The following functions are available:

PurchaseRequisition.ApproveApprovalRecord

PurchaseRequisition.ApproveApprovalRecord

DLL: tdextpurapi

This function is available from KB2185794.

Syntax:

```
long PurchaseRequisition.ApproveApprovalRecord(
                domain  tcrqno           iPurchaseRequisition,
                domain  tcsern           iSequenceNumber,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl.:  This function can be used to approve the given Requisition Approval
                Progress record. Approval is only allowed if all previous records
                have already been approved. This function checks whether the
                current user is allowed to perform the approval. The purchase
                requisition parameter/setting 'Approval Authorizations'
                (tdpur000.apau.2/tdpur082.apau) is taken into account for this.
                This function cannot be used if approval of requisitions is
                done through Workflow.
        Pre:    Caller must set retry-point
        Post:   Caller must commit/abort transaction
        Input:  iPurchaseRequisition    - Purchase Requisition; Mandatory.
                iSequenceNumber         - Sequence number of the Approval Progress
                                          record; Mandatory
        Output: oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions
                                          in Exception to get all
                                          relevant information.
        Return: 0                       - The Approval Progress record is approved.
```

	`<> 0                    - An error occurred`

Public Interfaces for PurchaseRequisitionLine

The following functions are available:

PurchaseRequisitionLine.ConvertToPurchaseOrder

PurchaseRequisitionLine.ConvertToPurchaseRFQ

PurchaseRequisitionLine.ConvertToPurchaseOrder

DLL:	tdextpurapi This function is available from KB2185794.

Syntax:	
	```
	long PurchaseRequisitionLine.ConvertToPurchaseOrder(
	                domain  tcrqno          iPurchaseRequisition,
	                domain  tcpono          iRequisitionLine,
	                domain  tcseri          iOrderSeries,
	                domain  tccotp          iOrderType,
	                domain  tcseri          iSubcontractingOrderSeries,
	                domain  tccotp          iSubcontractingOrderType,
	                domain  tcseri          iServiceSubcontractingOrderSeries,
	                domain  tccotp          iServiceSubcontractingOrderType,
	                domain  tcyesno         iCalculateNewPriceAndDiscounts,
	                domain  tcyesno         iCurrencyFromBusinessPartner,
	                domain  tcorno          iPreviousPurchaseOrder,
	                domain  tcpono          iPreviousPurchaseOrderLine,
	        ref     domain  tcorno          oGeneratedOrder,
	        ref     domain  tcpono          oGeneratedOrderLine,
	        ref     domain  tcpono          oGeneratedOrderSequence,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID )
	```

Usage:	
	```
	        Expl:   This function converts the given requisition line to a purchase
	                order. Conversion is only allowed if the requisition header has
	                the proper status (i.e. Approved or In Process), and the
	                requisition line is not rejected. The Conversion Type of the
	                requisition line must be 'Purchase Order'.

	                When input argument iPreviousPurchaseOrder is filled, this
	                function tries to add a line to that order. If the requisition
	                line does not match with the given order, then a new purchase
	                order will be generated. If maximal commingling must be obtained,
	                then the order in which the requisition lines are converted is
	                important.
	                When input argument iPreviousPurchaseOrderLine is filled, the
	```

```
                function tries to add another sequence to that order line.
        Pre:    Caller must set a retry-point
        Post:   Caller must commit or abort the transaction.
        Input:  iPurchaseRequisition     - Requisition; Mandatory
                iRequisitionLine         - Requisition Line; Mandatory
                iOrderSeries             - Order Series that will be used to
                                           generate the purchase order. Depending
                                           on the setup, this can be empty.
                iOrderType               - Order Type that will be used to
                                           generate the purchase order. Depending
                                           on the setup, this can be empty.
                iSubcontractingOrderSeries       - Order Series for subcontracting
                                                   purchase orders.
                iSubcontractingOrderType         - Order Type for subcontracting
                                                   purchase orders.
                iServiceSubcontractingOrderSeries
                                                 - Order Series for service
                                                   subcontracting purchase orders.
                iServiceSubcontractingOrderType - Order Type for service
                                                   subcontracting purchase orders.
                iCalculateNewPriceAndDiscounts
                                         - Yes:  LN calculates new prices and
                                                 discounts when converting
                                                 requisitions to purchase orders.
                                           No:   The prices defined in the
                                                 requisition are transferred to
                                                 the purchase order.

                iCurrencyFromBusinessPartner
                                         - Yes:  LN uses the business partner's
                                                 currency on the purchase order.
                                                 The requisition's price and
                                                 amount are converted to this
                                                 currency.
                                           No:   LN uses the currency of the
                                                 requisition on the purchase order.
                iPreviousPurchaseOrder
                                         - If filled, the function tries to add
                                           a line to that order. Mandatory if
                                           iPreviousPurchaseOrderLine is filled.
                iPreviousPurchaseOrderLine
                                         - If filled, the function tries to add
                                           another sequence to that order line.
        Output: oGeneratedOrder - The generated purchase order
                oGeneratedOrderLine      - The generated purchase order line
                oGeneratedOrderSequence - The generated purchase order
                                                   line sequence
        Return: 0                        - The requisition line is converted
                <> 0                     - An error occurred
```

PurchaseRequisitionLine.ConvertToPurchaseRFQ

DLL:	tdextpurapi This function is available from KB2185794.

Syntax:

```
long PurchaseRequisitionLine.ConvertToPurchaseRFQ(
                domain  tcrqno           iPurchaseRequisition,
                domain  tcpono           iRequisitionLine,
                domain  tcyesno          iAddToExistingRFQ,
                domain  tcqono           iExistingRFQ,
                domain  tcseri           iRFQSeries,
                domain  tcrfq.type       iRFQType,
                domain  tcqono           iPreviousRFQ,
        ref     domain  tcqono           oGeneratedRFQ,
        ref     domain  tcpono           oGeneratedRFQLine,
        ref     domain  tcpono           oGeneratedRFQSequence,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function converts the given requisition line to a purchase
                RFQ. Conversion is only allowed if the requisition header has
                the proper status (i.e. Approved or In Process), and the
                requisition line is not rejected. The Conversion Type of the
                requisition line must be 'RFQ'.
        Pre:    Caller must set a retry-point
        Post:   Caller must commit or abort the transaction.
        Input:  iPurchaseRequisition    - Requisition; Mandatory
                iRequisitionLine        - Requisition Line; Mandatory
                iAddToExistingRFQ       - Yes: the requisition line is converted
                                               to an existing RFQ. Use iExistingRFQ
                                               to indicate to which RFQ the
                                               requisition line must be added.
                                               Notes:
                                                * this will override the value
                                                  that is passed in iPreviousRFQ.
                                                * this option is only allowed if
                                                  the concept of
                                                  'Enhanced Line Handling' is
                                                  used for the given RFQ.
                                          No: this option allows a more regular
                                              form of commingling. Several
                                              requisition lines can be converted
                                              to one (or more) RFQ's, but only
                                              if the passed RFQ matches the data
                                              on the requisition line.
                                              See explanation at 'iPreviousRFQ'.
                iExistingRFQ            - Indicates the RFQ to which the given
                                          requisition line must be converted.
                                          This field is mandatory if
                                          iAddToExistingRFQ is Yes. The field
                                          must be empty if iAddToExistingRFQ
                                          is No.
                iRFQSeries              - The RFQ series that will be used when
                                          a new RFQ is generated.
                iRFQType                - The RFQ Type that will be used when
                                          a new RFQ is generated.
                iPreviousRFQ            - If filled, the function tries to add
                                          a requisition line to that RFQ. If they
                                          do not match, then a new RFQ is
                                          generated. Filling this field can be done
                                          when calling this function multiple
                                          times in order to convert multiple
                                          requisition lines to one (or: as little
                                          as possible) RFQs.
                                          If maximal commingling must be obtained,
                                          then the order in which the requisition
                                          lines are converted is important.
        Output: oGeneratedRFQ           - The generated RFQ
                oGeneratedRFQLine        - The generated RFQ line
                oGeneratedRFQSequence    - The generated RFQ sequence
        Return: 0                        - The requisition line is converted
                <> 0                     - An error occurred
```

Public Interfaces for PurchaseContract

The following functions are available:

PurchaseContract.GetTotalAmounts

PurchaseContract.GetTotalAmounts

DLL: tdextpurapi

This function is available from KB2042557.

Syntax:

```
long PurchaseContract.GetTotalAmounts(
                domain  tccono           iPurchaseContract,
        ref     domain  tcccur           oContractCurrency,
        ref     domain  tcamnt           oContractOpenAmount,
        ref     domain  tcamnt           oContractTotalAmount,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl.:  Function calculates total contract net amounts in contract
                currency.
                Contract is selected in the current company.
        Pre:    N.A.
        Post:   N.A.
        Input:  iPurchaseContract        - Purchase Contract; Mandatory.
        Output: oContractCurrency        - Contract Currency.
                oContractOpenAmount      - Contract Open Amount: Sum of amounts
                                           of all non terminated contract lines.
                oContractTotalAmount - Contract Total Amount: Sum of amounts
                                           of all contract lines.
                oExceptionMessage        - The last message if the return
                                           value is not equal to 0.
                                           If more than one  message is
                                           given, these are present in the
                                           oExceptionID
                oExceptionID             - An ID that refers to all error
                                           information. Use the functions
                                           in Exception to get all
                                           relevant information.
        Return: 0                        - Amount could be determined.
                DALHOOKERROR             - Error occurred.
```


Chapter 7 Public Interfaces for Pricing

Public Interfaces for Pricing

The following functions are available:

Pricing.SimulatePurchasePrice

Pricing.SimulateSalesPrice

Pricing.SimulatePurchasePrice

DLL:	tdextpcgapi This function is available from KB2042557.

Syntax:	
	```
	long Pricing.SimulatePurchasePrice(
	                domain  tcncmp          iLogisticCompany,
	                domain  tcncmp          iFinancialCompany,
	                domain  tccom.bpid      iBuyFromBusinessPartner,
	                domain  tcccur          iCurrency,
	                domain  tcitem          iItem,
	                domain  tccom.bpid      iShipFromBusinessPartner,
	                domain  tccom.bpid      iInvoiceFromBusinessPartner,
	                domain  tccom.bpid      iPricingBusinessPartner,
	                domain  tccwoc          iPurchaseOffice,
	                domain  tcrtyp          iRateType,
	                domain  tcdate          iRateDate,
	        const   domain  tcratc          iRate(),
	        const   domain  tcratf          iRateFactor(),
	                domain  tccplt          iPriceList,
	                domain  tccotp          iOrderType,
	                domain  tccreg          iArea,
	                domain  tccdec          iDeliveryTerms,
	                domain  tcpaym          iPaymentMethod,
	                domain  tcolid          iOptionListID,
	                domain  tccpva          iProductVariant,
	                domain  tcuef.effn      iEffectivityUnit,
	                domain  tcguid          iSpecification,
	                domain  tcmpnr          iManufacturerPartNumber,
	                domain  tcmcs.cmnf      iManufacturer,
	                domain  tcyesno         iSubcontracted,
	```

```
        domain  tcsite          iSite,
        domain  tcefex.date     iPriceDate,
        domain  tdpcg.prit      iPriceType,
        domain  tcqsl1          iOrderQuantity,
        domain  tccuni          iOrderQuantityUnit,
        domain  tcconv          iOrderQuantityUnitConversionFactor,
        domain  tccono          iContract,
        domain  tcpono          iContractLine,
        domain  tccwoc          iContractOffice,
        domain  tcpono          iContractSequence,
ref     domain  tcpric          oPrice,
ref     domain  tccuni          oPriceUnit,
ref     domain  tcconv          oPriceUnitConversionFactor,
ref             boolean         oDerivedItemUsed,
ref     domain  tdpcg.prbk      oPriceBook,
ref     domain  tcprsg          oPriceStage,
ref     domain  tdgen.porg      oPriceOrigin,
ref     domain  tdpcg.made      oPriceMatrixDefinition,
ref     domain  tdpcg.prse      oPriceMatrixSequence,
ref     domain  tdpcg.maty      oDiscountMatrixType(),
ref     domain  tdgen.dorg      oDiscountOrigin(),
ref     domain  tdpcg.made      oDiscountMatrixDefinition() fixed,
ref     domain  tdpcg.prse      oDiscountMatrixSequence(),
ref     domain  tcdisc          oDiscountPercentage(),
ref     domain  tddiam          oDiscountAmount(),
ref     domain  tccdsc          oDiscountCode() fixed,
ref     domain  tddmth          oDiscountMethod(),
ref     domain  tdpcg.dssc      oDiscountSchedule() fixed,
ref     domain  tccono          oContract,
ref     domain  tcpono          oContractLine,
ref     domain  tccwoc          oContractOffice,
ref     domain  tcpono          oContractSequence,
ref     domain  tcmpr.pagr      oMaterialPriceAgreement,
ref     domain  tcprip          oTotalMaterialPriceSurcharges,
ref     domain  tcprip          oTotalMaterialPrice,
ref     domain  tcmcs.s999m     oExceptionMessage mb,
ref             long            oExceptionID )
```

Usage:

```
Expl:   This public interface simulates purchase price and discounts
        retrieval, including material price surcharge information.
        Price and discount retrieval is according to standard LN logic,
        searching through the hierarchical price structure. The output
        indicates from which level the information is retrieved.
Pre:    NA
Post:   NA
Input:
        |***************************************************************
        |* Mandatory input arguments.
        |***************************************************************
        iLogisticCompany                - Logictic Company
        iFinancialCompany               - Financial Company
        iBuyFromBusinessPartner         - Buy-from Business Partner
        iCurrency                       - Currency
        iItem                           - Item
        |***************************************************************
        |* Optional input arguments; if not filled, defaults are
        |* retrieved.
        |***************************************************************
        iShipFromBusinessPartner        - Ship-from Business Partner
        iInvoiceFromBusinessPartner     - Invoice-from Business Partner
        iPricingBusinessPartner         - Pricing Business Partner
        iPurchaseOffice                 - Purchase Office
        iRateType                       - Exchange Rate Type
```

```
                iRateDate                       - Rate Date; if zero, then Price
                                                  Date is used.
                iRate                           - Rate; if first element zero,
                                                  rate and rate factor are
                                                  defaulted.
                iRateFactor                     - Rate Factor; if first element
                                                  is zero, rate and rate factor
                                                  are defaulted.
                iPriceList                      - Price List
                iOrderType                      - Order Type
                iArea                           - Area
                iDeliveryTerms                  - Delivery Terms; not supported.
                iPaymentMethod                  - Payment Method
                iOptionListID                   - Option List ID
                iProductVariant                 - Product Variant
                iEffectivityUnit                - Effectivity Unit
                iSpecification                  - Specification
                iManufacturerPartNumber         - ManufacturerPartNumber
                iManufacturer                   - Manufacturer
                iSubcontracted                  - Subcontracted; if empty, 'no'
                                                  is used
                iSite                           - Site
                iPriceDate                      - Price Date; of zero, then
                                                  current date and time is used.
                iPriceType                      - Price Type; if empty, 'buying'
                                                  is used
                iOrderQuantity                  - Order Quantity
                iOrderQuantityUnit              - Order Quantity Unit
                iOrderQuantityUnitConversionFactor
                                                - Order Quantity Unit Conversion
                                                  Factor
                iContract                       - Contract; if not filled,
                                                  Contract, Contract Line and
                                                  Contract Office are defaulted
                                                  (if applicable) without user
                                                  interaction.
                iContractLine                   - Contract Line; see Contract
                iContractOffice                 - Contract Office; see Contract
                iContractSequence               - Contract Sequence
        Output:
                |****************************************************************
                |* Price Output.
                |****************************************************************
                oPrice                          - Price
                oPriceUnit                      - Price Unit
                oPriceUnitConversionFactor      - Price Unit Conversion Factor
                oDerivedItemUsed                - Derived Item Used
                oPriceBook                      - Price Book
                oPriceStage                     - Price Stage
                oPriceOrigin                    - Price Origin
                oPriceMatrixDefinition          - Price Matrix Definition
                oPriceMatrixSequence            - Price Matrix Sequence
                |****************************************************************
                |* Discount Output; memeory allocation (11 levels) for arrays
                |* needed.
                |****************************************************************
                oDiscountMatrixType             - Discount Matrix Type (array)
                oDiscountOrigin                 - Discount Origin (array)
                oDiscountMatrixDefinition       - Discount Matrix Definition
                                                                        (array)
                oDiscountMatrixSequence         - Discount Matrix Sequence
                                                                        (array)
                oDiscountPercentage             - Discount Percentage (array)
                oDiscountAmount                 - Discount Amount (array)
                oDiscountCode                   - Discount Code (array)
                oDiscountMethod                 - Discount Method (array)
                oDiscountSchedule               - Discount Schedule (array)
                |***************************************************************
                |* Contract Output.
                |***************************************************************
                oContract                       - Contract; filled if price /
                                                  discount origin is 'contract'
                oContractLine                   - Contract Line; see Contract
                oContractOffice                 - Contract Office; see Contract
                |***************************************************************
                |* Material Price Output.
                |***************************************************************
                oMaterialPriceAgreement         - Material Price Agreement
                oTotalMaterialPriceSurcharges   - Material Price Surcharges
                oTotalMaterialPrice             - Total Material Price
                |***************************************************************
                |* Technical Output.
                |***************************************************************
                oExceptionMessage               - The last message if the return
                                                  value is not equal to 0.
                                                  If more than one  message is
                                                  given, these are present in the
                                                  oExceptionID
                oExceptionID                    - An ID that refers to all error
                                                  information. Use the functions
                                                  in Exception to get all
                                                  relevant information.
        Return: 0                               - Price could be determined.
                DALHOOKERROR                    - Error occurred.
```

Pricing.SimulateSalesPrice

DLL:	tdextpcgapi This function is available from KB2042557.

Syntax:

```
long Pricing.SimulateSalesPrice(
                domain  tcncmp          iLogisticCompany,
                domain  tcncmp          iFinancialCompany,
                domain  tccom.bpid      iSoldToBusinessPartner,
                domain  tcccur          iCurrency,
                domain  tcitem          iItem,
                domain  tccom.bpid      iShipToBusinessPartner,
                domain  tccom.bpid      iInvoiceToBusinessPartner,
                domain  tccom.bpid      iPricingBusinessPartner,
                domain  tccwoc          iSalesOffice,
                domain  tcrtyp          iRateType,
                domain  tcdate          iRateDate,
        const   domain  tcratc          iRate(),
        const   domain  tcratf          iRateFactor(),
                domain  tccplt          iPriceList,
                domain  tccotp          iOrderType,
                domain  tccreg          iArea,
                domain  tccdec           iDeliveryTerms,
                domain  tdpcg.pror       iPriceBookOrigin,
                domain  tcpaym           iPaymentMethod,
                domain  tcmcs.chan       iChannel,
                domain  tccpva           iProductVariant,
                domain  tcuef.effn       iEffectivityUnit,
                domain  tcsite           iSite,
                domain  tcefex.date      iPriceDate,
                domain  tcqsl1           iOrderQuantity,
                domain  tccuni           iOrderQuantityUnit,
                domain  tcconv           iOrderQuantityUnitConversionFactor,
                domain  tccono           iContract,
                domain  tcpono           iContractLine,
                domain  tccwoc           iContractOffice,
        ref     domain  tcpric           oPrice,
        ref     domain  tccuni           oPriceUnit,
        ref     domain  tcconv           oPriceUnitConversionFactor,
        ref             boolean          oDerivedItemUsed,
        ref     domain  tdpcg.prbk       oPriceBook,
        ref     domain  tcprsg           oPriceStage,
        ref     domain  tdgen.porg       oPriceOrigin,
        ref     domain  tdpcg.made       oPriceMatrixDefinition,
        ref     domain  tdpcg.prse       oPriceMatrixSequence,
        ref     domain  tdpcg.maty       oDiscountMatrixType(),
        ref     domain  tdgen.dorg       oDiscountOrigin(),
        ref     domain  tdpcg.made       oDiscountMatrixDefinition() fixed,
        ref     domain  tdpcg.prse       oDiscountMatrixSequence(),
        ref     domain  tcdisc           oDiscountPercentage(),
        ref     domain  tddiam           oDiscountAmount(),
        ref     domain  tccdsc           oDiscountCode() fixed,
        ref     domain  tddmth           oDiscountMethod(),
        ref     domain  tdpcg.dssc       oDiscountSchedule() fixed,
        ref     domain  tccono           oContract,
        ref     domain  tcpono           oContractLine,
        ref     domain  tccwoc           oContractOffice,
        ref     domain  tcmpr.pagr       oMaterialPriceAgreement,
        ref     domain  tcprip           oTotalMaterialPriceSurcharges,
        ref     domain  tcprip           oTotalMaterialPrice,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
Expl:   This public interface simulates sales price and discounts
        retrieval, including material price surcharge information.
        Price and discount retrieval is according to standard LN logic,
        searching through the hierarchical price structure. The output
        indicates from which level the information is retrieved.
Pre:    NA
Post:   NA
Input:
        |***************************************************************
        |* Mandatory input arguments.
        |***************************************************************
        iLogisticCompany                - Logictic Company
        iFinancialCompany               - Financial Company
        iSoldToBusinessPartner          - Sold-to Business Partner
        iCurrency                       - Currency
        iItem                           - Item
        |***************************************************************
        |* Optional input arguments; if not filled, defaults are
        |* retrieved.
        |***************************************************************
        iShipToBusinessPartner          - Ship-to Business Partner
        iInvoiceToBusinessPartner       - Invoice-to Business Partner
        iPricingBusinessPartner         - Pricing Business Partner
        iSalesOffice                    - Sales Office
        iRateType                       - Exchange Rate Type
        iRateDate                       - Rate Date; if zero, then Price
                                          Date is used.
        iRate                           - Rate; if first element zero,
                                          rate and rate factor are
                                          defaulted.
        iRateFactor                     - Rate Factor; if first element
                                          is zero, rate and rate factor
                                          are defaulted.
        iPriceList                      - Price List
        iOrderType                      - Order Type
        iArea                           - Area
        iDeliveryTerms                  - Delivery Terms; not supported.
        iPriceBookOrigin                - Price Book Origin; not supported.
        iPaymentMethod                  - Payment Method
        iChannel                        - Channel; not supported.
        iProductVariant                 - Product Variant
        iEffectivityUnit                - Effectivity Unit
        iSite                           - Site
        iPriceDate                      - Price Date; of zero, then
                                          current date and time is used.
        iOrderQuantity                  - Order Quantity
        iOrderQuantityUnit              - Order Quantity Unit
        iOrderQuantityUnitConversionFactor
                                        - Order Quantity Unit Conversion
                                          Factor
        iContract                       - Contract; if not filled,
                                          Contract, Contract Line and
                                          Contract Office are defaulted
                                          (if applicable) without user
                                          interaction.
        iContractLine                   - Contract Line; see Contract
        iContractOffice                 - Contract Office; see Contract
Output:
        |****************************************************************
        |* Price Output.
        |****************************************************************
        oPrice                          - Price
        oPriceUnit                      - Price Unit
        oPriceUnitConversionFactor      - Price Unit Conversion Factor
        oDerivedItemUsed                - Derived Item Used
        oPriceBook                      - Price Book
        oPriceStage                     - Price Stage
        oPriceOrigin                    - Price Origin
        oPriceMatrixDefinition          - Price Matrix Definition
        oPriceMatrixSequence            - Price Matrix Sequence
        |****************************************************************
        |* Discount Output; memeory allocation (11 levels) for arrays
        |* needed.
        |****************************************************************
        oDiscountMatrixType             - Discount Matrix Type (array)
        oDiscountOrigin                 - Discount Origin (array)
        oDiscountMatrixDefinition       - Discount Matrix Definition
                                                                (array)
        oDiscountMatrixSequence         - Discount Matrix Sequence
                                                                (array)
        oDiscountPercentage             - Discount Percentage (array)
        oDiscountAmount                 - Discount Amount (array)
        oDiscountCode                   - Discount Code (array)
        oDiscountMethod                 - Discount Method (array)
```

```
                oDiscountSchedule              - Discount Schedule (array)
                |***************************************************************
                |* Contract Output.
                |***************************************************************
                oContract                      - Contract; filled if if price /
                                                 discount origin is 'contract'
                oContractLine                  - Contract Line; see Contract
                oContractOffice                - Contract Office; see Contract
                |***************************************************************
                |* Material Price Output.
                |***************************************************************
                oMaterialPriceAgreement        - Material Price Agreement
                oTotalMaterialPriceSurcharges  - Material Price Surcharges
                oTotalMaterialPrice            - Total Material Price
                |***************************************************************
                |* Technical Output.
                |***************************************************************
                oExceptionMessage              - The last message if the return
                                                 value is not equal to 0.
                                                 If more than one  message is
                                                 given, these are present in the
                                                 oExceptionID
                oExceptionID                   - An ID that refers to all error
                                                 information. Use the functions
                                                 in Exception to get all
                                                 relevant information.
        Return: 0                              - Price could be determined.
                DALHOOKERROR                   - Error occurred.
```

Chapter 8 Public Interfaces for Planning

Public Interfaces for ItemOrderPlan

The following functions are available:

ItemOrderPlan.StartPlan

ItemOrderPlan.StartPlan

DLL:	cpextrrpapi This function is available from KB3521627.

Syntax:	
	```
	long ItemOrderPlan.StartPlan(
	                        long            iStartMode,
	                domain  tcmcs.st30      iStartFilter,
	                        long            iSessionIndex,
	        const           string          iQueryExtend(),
	                domain  cpcom.plnc      iPlanningScenario,
	                domain  cpitem          iPlanItem,
	                domain  cprrp.perl      iPeriodLength,
	                domain  cpcom.date      iStartDate,
	                domain  tcyesno         iSkipEmptyDays,
	                domain  tcyesno         iShowDistribution,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID )
	```

Usage:	
	```
	        Expl:   This function starts the Item Order Plan (cprrp0520m000),
	                but only if Planning is implemented.
	                The Item Order Plan is started in the Item View.

	        Input:  iStartMode              Not used. Session is always
	                                        started MODELESS (Parent and child are
	                                        parallel sessions that can be manipulated
	                                        simultaneously).
	                iStartFilter            Not Used.
	                iSessionIndex           Not Used.
	                iQueryExtend            Not Used
	                iPlanningScenario       When empty, the Actual Scenario will
	                                        be used.
	                iPlanItem               Plan Item, Mandatory
	```

```
                iPeriodLength           When empty, then Detail is used
                iStartDate              When zero, then current date is used
                iSkipEmptyDays          When empty, then Yes is used
                iShowDistribution       When empty, then Yes is used

        Output: No information of selected records is returned.

                oExceptionMessage       The last message if any message is
                                        found. If more than one message is
                                        found, these are present in the
                                        oExceptionID
                oExceptionID            An ID that refers to the exception
                                        information. Use the functions in
                                        Exception to get all relevant
                                        information.

        Return: 0                       Session started
                <> 0                    Otherwise.
```

Chapter 9 Public Interfaces for Freight

Public Interfaces for FreightOrderCluster

The following functions are available:

FreightOrderCluster.ConfirmDelivery

FreightOrderCluster.ConfirmDelivery

DLL:	fmextlbdapi This function is available from KB2050777.

Syntax:	
	```
	long FreightOrderCluster.ConfirmDelivery(
	                domain  tcorno          iFreightOrderCluster,
	                domain  tcdate          iActualLoadDate,
	                domain  tcdate          iActualUnloadDate,
	                domain  fmlbd.conv      iShippedOrCompleted,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID )
	```

Usage:	
	```
	        Expl:   This public interface can be used to set the status of the
	                Freight Order Cluster and it's Cluster Lines to Shipped or
	                Completed, based on the iShippedOrCompleted indicator.
	                This public interface will have an internal transaction
	                handling, so calling this Public Interface inside a transaction
	                is not allowed, to prevent this from happening, this public
	                interface will not work when called inside a logical transaction
	        Pre:    Transaction handling is done within this function! Calling this
	                Public Interface from inside a transaction is forbidden and will
	                be blocked
	        Post:   Exception handling can be done, when needed.
	        Input:  iFreightOrderCluster    - Freight Order Cluster: Mandatory
	                iActualLoadDate         - Actual Load Date: Mandatory when
	                                          iShippedOrComplete is set to Shipped
	                iActualUnloadDate       - Actual Unload Date: Mandatory when
	                                          iShippedOrComplete is set to Complete
	                iShippedOrCompleted     - Confirm the delivery of the Freight
	                                          Order Cluster to either Shipped
	                                          (fmlbd.conv.shipped) or Completed
	                                          (fmlbd.conv.completed)
	        Output: oExceptionMessage       - The last message if the return
	```

	```
	                                         value is not equal to 0.
	                                         If more than one  message is
	                                         given, these are present in the
	                                         oExceptionID
	                oExceptionID           - An ID that refers to all error
	                                         information. Use the functions in
	                                         Exception to get all relevant
	                                         information.
	        Return: 0                      - Confirm Delivery succeeded
	                DALHOOKERROR           - No data found or error occured
	```

Public Interfaces for FreightPlanning

The following functions are available:

FreightPlanning.ActualizePlan

FreightPlanning.GenerateForSpecificOrderSelection

FreightPlanning.MoveShipmentToLoad

FreightPlanning.ActualizePlan

DLL:	fmextlbdapi This function is available from KB2070843.

Syntax:	
	```
	long FreightPlanning.ActualizePlan(
	                domain  tcorno           iPlan,
	                        boolean          iReplanAllowed,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl:  This Public Interface will actualize the given plan
	        Pre:   db.retry.point is set
	        Post:  transaction handling (abort/commit)
	               exception handling can be done when applicable
	        Input: iPlan                   - Plan which must be actualized:
	                                         Mandatory
	               iReplanAllowed          - Should potential freight order lines
	                                         in the plan which must be replanned
	                                         be replanned before actualization
	                                         takes place, true or false.
	        Output: oExceptionMessage      - The last message if the return
	                                         value is not equal to 0.
	                                         If more than one  message is
	                                         given, these are present in the
	                                         oExceptionID
	```

	```
	         oExceptionID           - An ID that refers to all error
	                                  information. Use the functions in
	                                  Exception to get all relevant
	                                  information.
	     Return: 0/DALHOOKERROR
	```

FreightPlanning.GenerateForSpecificOrderSelection

DLL:	fmextlbdapi This function is available from KB2070843.

Syntax:	
	```
	long FreightPlanning.GenerateForSpecificOrderSelection(
	                domain  tccwoc          iShippingOffice,
	                domain  fmfoc.cplg      iPlanningGroup,
	                domain  fmlbd.algo      iPlanningAlgorithm,
	                domain  fmlbd.dtdr      iShipmentDatesBasedOn,
	                domain  fmlbd.rsel      iCarrierSelectionCriterion,
	                domain  tcmcs.long      iNumberOfSpecificOrders,
	        ref     domain  tcorno          iSpecificOrders() fixed,
	                        boolean         iSpecificOrderLineSelection,
	        ref     domain  tcpono          iSpecificOrderLines(),
	                domain  tcdsca          iPlanningDescription mb,
	                domain  tcyesno         iPlanningAlgorithmBinding,
	                domain  tcyesno         iFreightManagementLeadingPlan,
	                domain  tcyesno         iAllowMeansOfTransportInMultiplePlans,
	                domain  tcyesno         iCalculateAdditionalCosts,
	                domain  tcyesno         iReplan,
	                domain  tcorno          iPlanForReplan,
	                domain  tcyesno         iOnlyReplanLines,
	                domain  fmfoc.lnst      iReplanAdditionThroughStatus,
	                domain  fmlbd.replan    iReplanningMethod,
	                domain  tcyesno         iCommittedInventoryOnly,
	                domain  tcyesno         iDetailedPlanningLog,
	                        boolean         iShowProgressIndicator,
	        ref     domain  tcorno          oPlan,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID )
	```

Usage:	
	```
	        Expl:   This public interface allows to generate a freight plan for
	                a selection of freight orders and lines. It is assumed the
	                freight orders that are present in the order arrays are having
	                the status 'Expected' or 'Planned'.
	                When planning is not possible due to unreachable addresses or
	                low capacity of transport means, the plan will still be
	                generated till the stage it fails to plan, this will lead to an
	                unusable plan, but will allow for user analysis via the planning
	                log.
	                This public interface will not print the planning reports which
	                are printed normaly in the session "Generate Plan"
	                (fmlbd0280m000). Errors / information messages will be available
	                in the freight planning log, session "Planning Log"
	                (fmlbd0530m000).
	        Pre:    This Public Interface has it's own transaction management, as an
	                exception to the standard Public Interfaces. As generate plan
	                can be a long running transaction, the Public Interface should
	                not lock records too long.
	```

```
                Exception handling can be done when applicable
        Post:   N.a.
        Input:  iShippingOffice         - Shipping Office: Mandatory
                iPlanningGroup          - Planning Group: Mandatory
                iPlanningAlgorithm      - Planning Algorithm: Mandatory
                                          Possible values:
                                          - Direct Shipping (fmlbd.algo.sing)
                                          - Consolidation (fmlbd.algo.cons)
                                          - Pooling (fmlbd.algo.pool)
                iShipmentDatesBasedOn   - Shipping Dates based On: Mandatory
                                          Possible values:
                                          - Earliest of Possible Dates
                                                        (fmlbd.dtdr.earliest)
                                          - Latest of Possible Dates
                                                        (fmlbd.dtdr.latest)
                                          - Minimum of Planned Unload Dates
                                                        (fmlbd.dtdr.minimum)
                                          - Average of Planned Unload Dates
                                                        (fmlbd.dtdr.average)
                iCarrierSelectionCriterion
                                        - Carrier Selection: Mandatory
                                          Possible values:
                                          - Cheapest (fmlbd.rsel.cheap)
                                          - Fastest (fmlbd.rsel.fast)
                                          - Shortest (fmlbd.rsel.short)
                iNumberOfSpecificOrders - Number of specific orders in the
                                          specific order array: Mandatory
                iSpecificOrders         - Array containing the freight order
                                          headers for which planning must be
                                          generated.
                iSpecificOrderLineSelection - Apply specific line selection
                                          This is to be used when specific
                                          freight order lines must be planned,
                                          when complete freight orders are to
                                          be planned it is not required to pass
                                          all the freight order lines.
                iSpecificOrderLines     - Array containing the freight order
                                          lines for which planning must be
                                          generated.
                iPlanningDescription    - Description of the freight plan
                iPlanningAlgorithmBinding
                                        - Planning Algorithm Binding, if
                                          planning is not possible for a certain
                                          algorithm, the logic will plan for
                                          other algorithms as well. This flag
                                          can disable that behavior.
                iFreightManagementLeadingPlan
                                        - Create a freight management
                                          leading load plan.
                iAllowMeansOfTransportInMultiplePlans
                                        - Allow the means of transport to be
                                          planned in multiple plans.
                iCalculateAdditionalCosts
                                        - Calculate Additional Costs for
                                          the generated freight plan
                iReplan                 - Replanning
                iPlanForReplan          - Plan for Replanning: Mandatory when
                                          iReplan is set to Yes
                iOnlyReplanLines        - Only Replan Freight Order Lines to
                                          be Replanned.
                iReplanAdditionThroughStatus
                                        - Replanning, allow updating of existing
                                          Loads/Shipments till this status.
                iReplanningMethod       - Replanning Method
                                          Possible Values
                                          - Replan Freight Order Lines Separatly
                                            (fmlbd.replan.separate)
                                          - Combine with Planned Freight Orders
                                            (fmlbd.replan.combine)
                iCommittedInventoryOnly - Plan for Committed Inventory Only
                iDetailedPlanningLog    - Plan with the detailed planning log
                iShowProgressIndicator  - Show the progress indicator to the
                                          user, this will give the user an
                                          indication on the process flow and
                                          the time required to complete the
                                          generation of the freight plan.
        Output: oPlan                   - The generated plan
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0/DALHOOKERROR
```

FreightPlanning.MoveShipmentToLoad

DLL: fmextlbdapi

This function is available from KB2070843.

Syntax:

```
long FreightPlanning.MoveShipmentToLoad(
                domain  tcorno           iShipment,
                domain  tcorno           iDestinationLoad,
                        boolean          iAllowMergeShipmentLines,
                        boolean          iDeleteEmptyLoad,
                        boolean          iCalculateFreightCostsIfInteractive,
                        boolean          iCalculateAdditionalCosts,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
Expl:   This Public Interface will move a Shipment to a Load.
Pre:    db.retry.point is set
Post:   transaction handling (abort/commit)
        exception handling can be done when applicable
Input:  iShipment               - Shipment which must be moved:
                                  Mandatory
        iDestinationLoad        - Load to which the shipment must be
                                  moved: Mandatory
        iAllowMergeShipmentLines- When iDestinationLoad already has a
                                  shipment line which has the same
                                  freight order line of any of the
                                  iSourceShipment lines, the lines must
                                  be merged. This flag determines if
                                  that should be allowed (true/false)
        iDeleteEmptyLoad        - If the load from which the
```

	```
	                                          iSourceShipment is moved becomes empty
	                                          then it will be possible to remove
	                                          the empty load with the setting of
	                                          this flag.
	                iCalculateFreightCostsIfInteractive
	                                        - If freight costs calculation is setup
	                                          to be recalculated interactively in
	                                          the parameters, in the standard a
	                                          question will be raised, with this
	                                          flag the calculation in such cases can
	                                          be set to true or false.
	                iCalculateAdditionalCosts
	                                        - If freight costs are to be calculated
	                                          should the additional costs also be
	                                          calculated, true or false.
	        Output: oExceptionMessage       - The last message if the return
	                                          value is not equal to 0.
	                                          If more than one  message is
	                                          given, these are present in the
	                                          oExceptionID
	                oExceptionID            - An ID that refers to all error
	                                          information. Use the functions in
	                                          Exception to get all relevant
	                                          information.
	        Return: 0/DALHOOKERROR
	```

Public Interfaces for Load

The following functions are available:

Load.ConfirmDelivery

Load.ConfirmDelivery

DLL:	fmextlbdapi This function is available from KB2050777.

Syntax:	
	```
	long Load.ConfirmDelivery(
	                domain  tcorno          iLoad,
	                domain  tcdate          iActualLoadDate,
	                domain  tcdate          iActualUnloadDate,
	                domain  fmlbd.conv      iShippedOrCompleted,
	        ref     domain  tcmcs.s999m     oExceptionMessage mb,
	        ref             long            oExceptionID )
	```

Usage:	
	```
	        Expl:   This public interface can be used to set the status of the load
	                and it's shipments and shipment lines to Shipped or Completed,
	```

```
                based on the iShippedOrCompleted indicator.
        Pre:    This Public Interface will update records and as such must be
                called within a transaction!
                When implementing this Public Interface, please be aware of the
                performance aspect. In standard, this functionality will have
                it's own transaction management, while the Public Interface does
                not!. Therefore it is strongly advisable to call this Public
                Interface inside a seperate transaction and after returning
                directly perform the abort is case of errors and the commit in
                case this function returns 0. Long logical transaction must be
                prevented as records will be locked and are not available for
                other transactions!
        Post:   Exception handling can be done, when needed.
        Input:  iLoad                   - Load: Mandatory
                iActualLoadDate         - Actual Load Date: Mandatory when
                                          iShippedOrComplete is set to Shipped
                iActualUnloadDate       - Actual Unload Date: Mandatory when
                                          iShippedOrComplete is set to Complete
                iShippedOrCompleted     - Confirm the delivery of the Load to
                                          either Shipped (fmlbd.conv.shipped) or
                                          Completed (fmlbd.conv.completed)
        Output: oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0                       - Confirm Delivery succeeded
                DALHOOKERROR            - No data found or error occured
```

Public Interfaces for Shipment

The following functions are available:

Shipment.ConfirmDelivery

Shipment.ConfirmDelivery

DLL:	fmextlbdapi This function is available from KB2050777.

Syntax:	`long Shipment.ConfirmDelivery(`
	`                domain   tcorno           iShipment,`

```
                domain  tcdate           iActualLoadDate,
                domain  tcdate           iActualUnloadDate,
                domain  fmlbd.conv       iShippedOrCompleted,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This public interface can be used to set the status of the
                shipment and it's shipment lines to Shipped or Completed, based
                on the iShippedOrCompleted indicator.
                This public interface will have an internal transaction
                handling, so calling this Public Interface inside a transaction
                is not allowed, to prevent this from happening, this public
                interface will not work when called inside a logical transaction
        Pre:    Transaction handling is done within this function! Calling this
                Public Interface from inside a transaction is forbidden and will
                be blocked
        Post:   Exception handling can be done, when needed.
        Input:  iShipment               - Shipment: Mandatory
                iActualLoadDate         - Actual Load Date: Mandatory when
                                          iShippedOrComplete is set to Shipped
                iActualUnloadDate       - Actual Unload Date: Mandatory when
                                          iShippedOrComplete is set to Complete
                iShippedOrCompleted     - Confirm the delivery of the Shipment
                                          to either Shipped (fmlbd.conv.shipped)
                                          or Completed (fmlbd.conv.completed)
        Output: oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0                       - Confirm Delivery succeeded
                DALHOOKERROR            - No data found or error occured
```

Chapter 10 Public Interfaces for BOD & BDE

Public Interfaces for BOD

The following functions are available:

BOD.ActionsAfterProcessingIncomingRequest

BOD.ActionsAfterProcessingIncomingRequestWithAutomaticProcessing

BOD.ActionsBeforeProcessingIncomingRequest

BOD.ConvertFromERPItem

BOD.ConvertToERPItem

BOD.ExecuteMethod

BOD.ExecuteOnAcknowledgeForChameleon

BOD.ExecuteOnLoadForChameleon

BOD.ExecuteOnProcessForChameleon

BOD.ExecuteOnSyncForChameleon

BOD.ExecuteOnUpdateForChameleon

BOD.ExecutePublishForChameleon

BOD.ExecuteShowForChameleon

BOD.GetIdAccountingEntity

BOD.GetIdLocation

BOD.GetIdLogicalID

BOD.GetPublishingAllowed

BOD.HandleStagingAfterProcessingIncomingRequest

BOD.HandleStagingBeforeProcessingIncomingRequest

BOD.InterruptPublishing

BOD.Publish

BOD.PublishLNMessage

BOD.ResumePublishing

BOD.ActionsAfterProcessingIncomingRequest

DLL:	tcextbodapi This function is available from KB2040021.

Syntax:	
	```
	long BOD.ActionsAfterProcessingIncomingRequest(
	                        long             iXMLRequest,
	        ref             long             oResult,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl:   This function publishes the staged BODs that were triggered
	                from the processing of the incoming request and resets the
	                parameter that enabled the staging of outgoing BODs in the
	                BOD.ActionsBeforeProcessingIncomingRequest() function.
	                Preferably, it should be called in the AfterExecuteHook
	                of every OnEvent section, e.g. the OnProcess, OnLoad, OnSync,
	                OnShow, OnAcknowledge etc.
	        Pre:    Before the incoming BOD is processed, function
	                BOD.ActionsBeforeProcessingIncomingRequest() must be called.
	        Post:   NA
	        Input:  iXMLRequest             - XML structure with request. Mandatory
	        Output: oResult                 - 0 if succes, otherwise <> 0
	                oExceptionMessage       - The last message if the return
	                                          value is not equal to 0.
	                                          If more than one message is
	                                          given, these are present in the
	                                          oExceptionID
	                oExceptionID            - An ID that refers to all error
	                                          information. Use the functions in
	                                          Exception to get all relevant
	                                          information.
	        Return: 0 / DALHOOKERROR
	```

BOD.ActionsAfterProcessingIncomingRequestWithAutomaticProcessing

DLL:	tcextbodapi This function is available from KB2040021.

Syntax:	
	```
	long BOD.ActionsAfterProcessingIncomingRequestWithAutomaticProcessing(
	                        long             iXMLRequest,
	                domain  tcmcs.str132     iObjectType,
	                domain  tcmcs.str132     iObject,
	        ref             long             oResult,
	```

	ref domain tcmcs.s999m oExceptionMessage mb, ref long oExceptionID)
Usage:	Expl: This function publishes the staged BODs that were triggered from the processing of the incoming request and resets the parameter that enabled the staging of outgoing BODs in the BOD.ActionsBeforeProcessingIncomingRequest() function. Furthermore, it starts the automatic processing (if any) for the object that was created/updated in LN from the incoming request. Preferably, this function should be called in the AfterExecuteHook of every OnEvent section, e.g. the OnProcess, OnLoad, OnSync, OnShow, OnAcknowledge etc. Pre: Before the incoming BOD is processed, function BOD.ActionsBeforeProcessingIncomingRequest() must be called. Post: NA Input: iXMLRequest - XML structure with request. Mandatory iObjectType - object type for which automatic processing must be started, e.g. "PurchaseOrder" iObject - The identifier of the given ObjectType e.g. the purchase order if ObjectType is "PurchaseOrder" Output: oResult - 0 if succes, otherwise <> 0 oExceptionMessage - The last message if the return value is not equal to 0. If more than one message is given, these are present in the oExceptionID oExceptionID - An ID that refers to all error information. Use the functions in Exception to get all relevant information. Return: 0 / DALHOOKERROR

BOD.ActionsBeforeProcessingIncomingRequest

DLL:	tcextbodapi This function is available from KB2040021.
Syntax:	long BOD.ActionsBeforeProcessingIncomingRequest(long iXMLRequest, ref boolean oCancel, ref domain tcmcs.s999m oExceptionMessage mb, ref long oExceptionID)
Usage:	Expl: This function validates the incoming request and sets parameters needed during the processing. It validates if the Accounting Entity and the Location (Location is not checked for master data BODs) of the incoming request matches the Accounting Entity and Location of the current company (in Tenant, Accounting Entity and Location setup or in BOD Parameters) and sets output argument oCancel to true in case of a mismatch. It extracts the LastModificationPerson/IDs/ID from the incoming request and determines the linked LN user. If the LN user can be determined, it switches user to the LN user.

```
                It sets a parameter that enables the staging of outgoing BODs
                that are triggered from the processing of the incoming request.
                Preferably, it should be called in the BeforeExecuteHook
                of every OnEvent section, e.g. the OnProcess, OnLoad, OnSync,
                OnShow, OnAcknowledge etc.
        Pre:    NA
        Post:   After the incoming request is processed,
                BOD.ActionsAfterProcessingIncomingRequest() or
                BOD.ActionsAfterProcessingIncomingRequestWithAutomaticProcessing()
                must be called.
        Input:  iXMLRequest             - XML structure with request. Mandatory
        Output: oCancel                 - true, if BOD must be cancelled
                                        - false, if BOD can be processed
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0                       - No error occurred while executing the
                                          actions.
                DALHOOKERROR            - An error occurred while executing the
                                          actions, or a mismatch was detected
                                          during accounting entity / location
                                          validation and BOD parameter Inbound
                                          Routing Error Handling is set to
                                          Return Error.
```

BOD.ConvertFromERPItem

DLL:

tcextbodapi

This function is available from KB2044306.

Syntax:

```
long BOD.ConvertFromERPItem(
                domain  tcitem           iERPItem,
        ref     domain  tcmcs.str50      oBODItem,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl.   : This function converts the ERP item
                  to the external item used in the BODs.
        Pre     : -
        Post    : -
        Input   : iERPItem                -  Mandatory.
        Output  : oBODItem
                  oExceptionMessage       - The last message if the return
                                            value is not equal to 0.
                                            If more than one  message is
                                            given, these are present in the
                                            oExceptionID
                  oExceptionID            - An ID that refers to all error
                                            information. Use the functions in
                                            Exception to get all relevant
```

```
                                  information.
        Return   : 0
```

BOD.ConvertToERPItem

DLL: tcextbodapi

This function is available from KB2044306.

Syntax:

```
long BOD.ConvertToERPItem(
                domain  tcmcs.str50      iBODItem,
        ref     domain  tcitem           oERPItem,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl.    : This function sets the ERP LN item, using the item as received
                   from other product.  (e.g. WMS)

        Process:   Dynamic translation by project lookup.
                   The first 9 characters of the item (or less dependent on item
                   segmentation) are taken, and looked up as project code in
                   the project table. If this code exists, the item code is
                   assumed to represent a customized item and not converted.
                   Otherwise it.s treated as standard item and leading spaces
                   are added.

                   EXAMPLES:
                         CBO item string         ERP item
                         -----------------------------------------
                         "PR0000001BIKE"         "PR0000001BIKE"
                                                 Project = PR0000001

                         "PR1      BIKE"         "PR1      BIKE"
                                                 Project = PR1

                         "         BIKE"         "         BIKE"
                         "BIKE"                  "         BIKE"
                                                 Project = ""
        Pre      : -
        Post     : -
        Input    : iBODItem               - CBO item (e.g. "BIKE"). Mandatory.
        Output   : oERPItem               - LN item
                                            (e.g. "         BIKE              ")
                   oExceptionMessage      - The last message if the return
                                            value is not equal to 0.
                                            If more than one  message is
                                            given, these are present in the
                                            oExceptionID
                   oExceptionID           - An ID that refers to all error
                                            information. Use the functions in
                                            Exception to get all relevant
                                            information.
        Return   : 0
```

BOD.ExecuteMethod

DLL: tcextbodapi

This function is available from KB2019035.

Syntax:

```
long BOD.ExecuteMethod(
                domain  tcbod.name       iNoun,
                domain  tcmcs.str30      iMethod,
                        long             iXMLRequest,
        ref             long             oXMLResponse,
        ref             long             oXMLResult,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function executes a non-batch method for a specified
                BOD.
        Pre:    NA
        Post:   NA
        Input:  iNoun           - The (protected) Noun for which the method
                                  must be executed, e.g.
                                  "ReceiveDeliveryWarehousingBOD". Mandatory
                iMethod         - The method to be executed,
                                  e.g. "Create" or "Change". Mandatory
                iXMLRequest     - XML structure with request. Mandatory
        Output: oXMLResponse    - XML structure with response (if method is
                                  executed successfully)
                oXMLResult      - XML structure with result (in case of error)
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return values:
                0                       - method is executed successfully
                DALHOOKERROR            - on errors
```

BOD.ExecuteOnAcknowledgeForChameleon

DLL: tcextbodapi

This function is available from KB2019035.

Syntax:

```
long BOD.ExecuteOnAcknowledgeForChameleon(
                domain  tcbod.name       iProtectedNoun,
                        long             iXMLRequest,
        ref             long             oXMLResponse,
        ref             long             oXMLResult,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function executes the OnAcknowledge method of a protected
                noun of the chameleon BOD and can only be called from the
                OnExecuteHook of the OnAcknowledge method of an incoming public
                noun. The function is used to route an incoming BOD with
                Acknowledge verb to a protected noun.
        Pre:    NA
        Post:   NA
        Input:  iProtectedNoun  - The (protected) Noun for which the
                                  OnAcknowledge method must be executed, e.g.
                                  "SalesOrderInBOD". Mandatory
                iXMLRequest     - XML structure with request. Mandatory
        Output: oXMLResponse    - XML structure with response (if method is
                                  executed successfully)
                oXMLResult      - XML structure with result (in case of error)
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0 / DALHOOKERROR
```

BOD.ExecuteOnLoadForChameleon

DLL: tcextbodapi

This function is available from KB2019035.

Syntax:

```
long BOD.ExecuteOnLoadForChameleon(
                domain  tcbod.name       iProtectedNoun,
                        long             iXMLRequest,
        ref             long             oXMLResponse,
        ref             long             oXMLResult,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function executes the OnLoad method of a protected noun
                of the chameleon BOD and can only be called from the OnExecuteHook
                of the OnLoad method of an incoming public noun. The function
                is used to route an incoming BOD with Load verb to a protected
                noun.
        Pre:    NA
        Post:   NA
        Input:  iProtectedNoun  - The (protected) Noun for which the OnLoad
                                  method must be executed, e.g.
                                  "SalesOrderInBOD". Mandatory
                iXMLRequest     - XML structure with request. Mandatory
        Output: oXMLResponse    - XML structure with response (if method is
                                  executed successfully)
                oXMLResult      - XML structure with result (in case of error)
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
```

```
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0 / DALHOOKERROR
```

BOD.ExecuteOnProcessForChameleon

DLL: tcextbodapi

This function is available from KB2019035.

Syntax:

```
long BOD.ExecuteOnProcessForChameleon(
                domain  tcbod.name       iProtectedNoun,
                        long             iXMLRequest,
        ref             long             oXMLResponse,
        ref             long             oXMLResult,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function executes the OnProcess method of a protected noun
                of the chameleon BOD and can only be called from the OnExecuteHook
                of the OnProcess method of an incoming public noun. The function
                is used to route an incoming BOD with Process verb to a
                protected noun.
        Pre:    NA
        Post:   NA
        Input:  iProtectedNoun  - The (protected) Noun for which the OnProcess
                                  method must be executed, e.g.
                                  "SalesOrderInBOD". Mandatory
                iXMLRequest     - XML structure with request. Mandatory
        Output: oXMLResponse    - XML structure with response (if method is
                                  executed successfully)
                oXMLResult      - XML structure with result (in case of error)
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0 / DALHOOKERROR
```

BOD.ExecuteOnSyncForChameleon

DLL: tcextbodapi

This function is available from KB2019035.

Syntax:

```
long BOD.ExecuteOnSyncForChameleon(
                domain  tcbod.name       iProtectedNoun,
                        long             iXMLRequest,
        ref             long             oXMLResponse,
        ref             long             oXMLResult,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function executes the OnSync method of a protected noun
                of the chameleon BOD and can only be called from the OnExecuteHook
                of the OnSync method of an incoming public noun. The function
                is used to route an incoming BOD with Sync verb to a protected
                noun.
        Pre:    NA
        Post:   NA
        Input:  iProtectedNoun  - The (protected) Noun for which the OnSync
                                  method must be executed, e.g.
                                  "SalesOrderInBOD". Mandatory
                iXMLRequest     - XML structure with request. Mandatory
        Output: oXMLResponse    - XML structure with response (if method is
                                  executed successfully)
                oXMLResult      - XML structure with result (in case of error)
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0 / DALHOOKERROR
```

BOD.ExecuteOnUpdateForChameleon

DLL: tcextbodapi

This function is available from KB2019035.

Syntax:

```
long BOD.ExecuteOnUpdateForChameleon(
                domain  tcbod.name       iProtectedNoun,
                        long             iXMLRequest,
        ref             long             oXMLResponse,
        ref             long             oXMLResult,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function executes the OnUpdate method of a protected noun
                of the chameleon BOD and can only be called from the OnExecuteHook
                of the OnUpdate method of an incoming public noun. The function
                is used to route an incoming BOD with Update verb to a protected
                noun.
        Pre:    NA
        Post:   NA
        Input:  iProtectedNoun  - The (protected) Noun for which the OnUpdate
                                  method must be executed, e.g.
                                  "SalesOrderInBOD". Mandatory
                iXMLRequest     - XML structure with request. Mandatory
        Output: oXMLResponse    - XML structure with response (if method is
```

	```
	                             executed successfully)
	         oXMLResult      - XML structure with result (in case of error)
	         oExceptionMessage       - The last message if the return
	                                   value is not equal to 0.
	                                   If more than one  message is
	                                   given, these are present in the
	                                   oExceptionID
	         oExceptionID            - An ID that refers to all error
	                                   information. Use the functions in
	                                   Exception to get all relevant
	                                   information.
	 Return: 0 / DALHOOKERROR
	```

BOD.ExecutePublishForChameleon

DLL:	tcextbodapi This function is available from KB2019035.

Syntax:	
	```
	long BOD.ExecutePublishForChameleon(
	                domain  tcbod.name       iProtectedNoun,
	                        long             iXMLRequest,
	        ref             long             oXMLResponse,
	        ref             long             oXMLResult,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl:   This function executes the publishing for a protected noun
	                of the chameleon BOD and can only be called from the OnExecuteHook
	                of the PublishEvent method of a protected noun.
	                It translates the protected noun to the public noun and
	                executes the PublishEvent method of the public noun
	        Pre:    NA
	        Post:   NA
	        Input:  iProtectedNoun  - The protected Noun, which is the BOD
	                                  that calls this function, e.g.
	                                  "ProductionOrderSFCBOD". Mandatory
	                iXMLRequest     - XML structure with request. Mandatory
	        Output: oXMLResponse    - XML structure with response (if PublishEvent
	                                  method is executed successfully)
	                oXMLResult      - XML structure with result (in case of error)
	                oExceptionMessage       - The last message if the return
	                                          value is not equal to 0.
	                                          If more than one  message is
	                                          given, these are present in the
	                                          oExceptionID
	                oExceptionID            - An ID that refers to all error
	                                          information. Use the functions in
	                                          Exception to get all relevant
	                                          information.
	        Return: 0 / DALHOOKERROR
	```

BOD.ExecuteShowForChameleon

DLL:	tcextbodapi This function is available from KB2019035.

Syntax:	
	```
	long BOD.ExecuteShowForChameleon(
	                domain  tcbod.name       iProtectedNoun,
	                        long             iXMLRequest,
	        ref             long             oXMLResponse,
	        ref             long             oXMLResult,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl:   This function executes the Show method of the public noun
	                of the chameleon BOD and can only be called from the OnExecuteHook
	                of the Show method of a protected noun.
	                It translates the protected noun to the public noun and
	                executes the Show method of the public noun. Then it renames
	                the response XML to the response XML of the protected noun
	        Pre:    NA
	        Post:   NA
	        Input:  iProtectedNoun  - The protected Noun, which is the BOD
	                                  that calls this function, e.g.
	                                  "ProductionOrderSFCBOD". Mandatory
	                iXMLRequest     - XML structure with request. Mandatory
	        Output: oXMLResponse    - XML structure with response (if method is
	                                  executed successfully)
	                oXMLResult      - XML structure with result (in case of error)
	                oExceptionMessage       - The last message if the return
	                                          value is not equal to 0.
	                                          If more than one  message is
	                                          given, these are present in the
	                                          oExceptionID
	                oExceptionID            - An ID that refers to all error
	                                          information. Use the functions in
	                                          Exception to get all relevant
	                                          information.
	        Return: 0 / DALHOOKERROR
	```

BOD.GetIdAccountingEntity

DLL:	tcextbodapi This function is available from KB1987704.

Syntax:	
	```
	long BOD.GetIdAccountingEntity(
	                domain  tcncmp           iCompany,
	                domain  tcbod.name       iNoun,
	                        long             iEntityType,
	        const           string           iEntityCode(),
	                domain  tcmcs.tabl       iRootTable,
	        ref     domain  tcbod.acen       oAccountingEntity,
	        ref             boolean          oAccountingEntityisSet,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:

```
        Expl:   This function determines the Accounting Entity.
        Pre:    NA
        Post:   NA
        Input:  iCompany        - Company for which the accounting
                                  entity is determined. Mandatory
                iNoun           - Noun: Mandatory
                iEntityType     - Entity Type: Mandatory for transactional data BODs
                                  Possible values: 1 (Warehouse), 2 (Department),
                                  3 (Project)
                iEntityCode     - Entity Code: Mandatory for transactional data BODs
                                  Possible values: The warehouse, department or
                                  project.
                                  The Entity Type and Entity Code are used to
                                  determine the Tenant, AccountingEntity and
                                  Location.
                iRootTable      - Root Table: Mandatory for master data BODs
        Output: oAccountingEntity       - Accounting Entity
                oAccountingEntityisSet  - Accounting Entity set (true or false).
                oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0                       - Accounting Entity is determined.
                DALHOOKERROR            - Otherwise.
```

BOD.GetIdLocation

DLL: tcextbodapi

This function is available from KB1987704.

Syntax:

```
long BOD.GetIdLocation(
                domain  tcncmp          iCompany,
                domain  tcbod.name      iNoun,
                        long            iEntityType,
        const           string          iEntityCode(),
                domain  tcmcs.tabl      iRootTable,
        ref     domain  tcbod.lctn      oLocation,
        ref             boolean         oLocationisSet,
        ref     domain  tcmcs.s999m     oExceptionMessage mb,
        ref             long            oExceptionID )
```

Usage:

```
        Expl:   This function determines the Location.
        Pre:    NA
        Post:   NA
        Input:  iCompany        - company for which the location
                                  is determined. Mandatory
                iNoun           - Noun: Mandatory
                iEntityType     - Entity Type: Mandatory for transactional data BODs
                                  Possible values: 1 (Warehouse), 2 (Department),
                                  3 (Project)
                iEntityCode     - Entity Code: Mandatory for transactional data BODs
```

```
                                 Possible values: The warehouse, department or
                                 project.
                                 The Entity Type and Entity Code are used to
                                 determine the Tenant, AccountingEntity and
                                 Location.
                iRootTable     - Root Table: Mandatory for master data BODs
        Output: oLocation              - Location
                oLocationisSet         - Location set (true or false).
                oExceptionMessage      - The last message if the return
                                         value is not equal to 0.
                                         If more than one  message is
                                         given, these are present in the
                                         oExceptionID
                oExceptionID           - An ID that refers to all error
                                         information. Use the functions in
                                         Exception to get all relevant
                                         information.
        Return: 0                      - Location is determined.
                DALHOOKERROR           - Otherwise.
```

BOD.GetIdLogicalID

DLL: tcextbodapi

This function is available from KB1987704.

Syntax:

```
long BOD.GetIdLogicalID(
                domain  tcncmp           iCompany,
                domain  tcbod.name       iNoun,
                        long             iEntityType,
        const           string           iEntityCode(),
                domain  tcmcs.tabl       iRootTable,
        ref     domain  tcbod.loid       oLogicalID,
        ref             boolean          oLogicalIDisSet,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function determines the Logical ID.
        Pre:    NA
        Post:   NA
        Input:  iCompany       - company for which the logical Id
                                 is determined. Mandatory
                iNoun          - Noun: Mandatory
                iEntityType    - Entity Type: Mandatory for transactional data BODs
                                 Possible values: 1 (Warehouse), 2 (Department),
                                 3 (Project)
                iEntityCode    - Entity Code: Mandatory for transactional data BODs
                                 Possible values: The warehouse, department or
                                 project.
                                 The Entity Type and Entity Code are used to
                                 determine the Tenant, AccountingEntity and
                                 Location.
                iRootTable     - Root Table: Mandatory for master data BODs
        Output: oLogicalId             - LogicalID
                oLogicalIdisSet        - LogicalID set (true or false).
                oExceptionMessage      - The last message if the return
                                         value is not equal to 0.
```

```
                                                If more than one  message is
                                                given, these are present in the
                                                oExceptionID
                oExceptionID                  - An ID that refers to all error
                                                information. Use the functions in
                                                Exception to get all relevant
                                                information.
        Return: 0                             - LogicalId is determined.
                DALHOOKERROR                  - Otherwise.
```

BOD.GetPublishingAllowed

DLL: tcextbodapi

This function is available from KB2267552.

Syntax:

```
long BOD.GetPublishingAllowed(
        ref             boolean          oAllowed,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function checks if BOD publishing is allowed in the
                current company.
        Pre:    NA
        Post:   NA
        Input:  NA
        Output: oAllowed                      - true: publishing is allowed
                                              - false: publishing is not allowed
                oExceptionMessage             - The last message if the return
                                                value is not equal to 0.
                                                If more than one  message is
                                                given, these are present in the
                                                oExceptionID
                oExceptionID                  - An ID that refers to all error
                                                information. Use the functions in
                                                Exception to get all relevant
                                                information.
        Return: 0                             - OK.
                <> 0                          - Error occurred.
```

BOD.HandleStagingAfterProcessingIncomingRequest

DLL: tcextbodapi

This function is available from KB2040021.

Syntax:

```
long BOD.HandleStagingAfterProcessingIncomingRequest(
                domain  tcbod.name       iNoun,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function publishes the staged BODs that were triggered
```

```
                from the processing of the incoming request and resets the
                parameter that enabled the staging of outgoing BODs in the
                BOD.HandleStagingBeforeProcessingIncomingRequest() function.
                Preferably, it should be called in the AfterExecuteHook
                of every OnEvent section, e.g. the OnProcess, OnLoad, OnSync,
                OnShow, OnAcknowledge etc.
        Pre:    Before the incoming BOD is processed,
                function BOD.HandleStagingBeforeProcessingIncomingRequest()
                must be called.
        Post:   NA
        Input:  iNoun                   - The noun of the incoming request
                                          e.g. "PurchaseOrderBOD"
        Output: oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0 / DALHOOKERROR
```

BOD.HandleStagingBeforeProcessingIncomingRequest

DLL: tcextbodapi

This function is available from KB2040021.

Syntax:

```
long BOD.HandleStagingBeforeProcessingIncomingRequest(
                domain  tcbod.name       iNoun,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID )
```

Usage:

```
        Expl:   This function sets a parameter that enables the staging of
                outgoing BODs that are triggered from the processing of the
                incoming request.
                Preferably, it should be called in the BeforeExecuteHook
                of every OnEvent section, e.g. the OnProcess, OnLoad, OnSync,
                OnShow, OnAcknowledge etc.
        Pre:    NA
        Post:   After the incoming BOD is processed, function
                BOD.HandleStagingAfterProcessingIncomingRequest()
                must be called
        Input:  iNoun                   - The noun of the incoming request
                                          e.g. "PurchaseOrderBOD"
        Output: oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0 / DALHOOKERROR
```


BOD.InterruptPublishing

DLL:	tcextbodapi This function is available from KB2267552.

Syntax:	
	```
	long BOD.InterruptPublishing(
	                        boolean          iAllBods,
	                domain  tcbod.name       iNoun,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl:   This function interrupts the publishing of all BODs or one
	                specified BOD in the current process. If parallel processing is
	                implemented, the BOD publishing will be interrupted in the parallel
	                processes as well.
	                For example: session Calculate Standard Cost publishes the
	                ItemMasterBOD for each item for which the cost price is updated.
	                If you don't need the ItemMasterBOD from this session, you can
	                call this function in the Before Command hook of the main
	                processing with iNoun = "ItemMasterCommonBOD". You can resume the
	                publishing by calling function BOD.ResumePublishing().
	                If a process can publish multiple BODs (e.g. SalesOrderBOD,
	                CustomerReturnBOD and ShipmentBOD) and you want to interrupt the
	                publishing of two BODs, you can call this function two times and
	                specify iAllBods as false and iNoun as the BOD for which the
	                publishing must be interrupted.
	        Pre:    N/A
	        Post:   Function BOD.ResumePublishing() must be called to resume the
	                publishing.
	        Output:
	                iAllBods                - true: The publishing of all BODs in the
	                                          current process is interrupted.
	                                          false: The publishing of the specified
	                                          BOD in the current process is interrupted.
	                iNoun                   - The (protected) Noun for which the
	                                          publishing must be interrupted.
	                                          Example: ItemMasterCommonBOD.
	                                          Mandatory if iAllBods is false. Not
	                                          needed / ignored if iAllBods is true.
	                oExceptionMessage       - A message if the return value is not equal
	                                          to 0. This message contains the root cause of
	                                          the request failure.
	                oExceptionID            - An ID that refers to all error information
	                                          if the return value is <> 0.
	                                          Use the functions in Exception to get
	                                          all relevant information.
	        Return:
	                0                       - The action to interrupt the publishing was
	                                          set successfully.
	                <> 0                    - The action to interrupt the publishing can
	                                          not be set.
	```

BOD.Publish

DLL: tcextbodapi

This function is available from KB1994284.

Syntax:

```
long BOD.Publish(
                domain  tcbod.name       iNoun,
                domain  tcmcs.tabl       iRootTable,
        const           string           iActionCode(),
                        long             iEntityType,
        const           string           iEntityCode(),
        const           string           iDocumentId(),
        const           string           iProcessingAction(),
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID,
                                         ... )
```

Usage:

```
        Expl:   This function publishes any BOD.
        Pre:    Transaction handling is needed.
        Post:   Transaction handling is needed: A commit or abort must be done.
        Input:  iNoun            - Noun: Mandatory
                iRootTable       - Root Table: Mandatory for master data BODs
                iActionCode      - Action Code: Mandatory
                                   Possible values: "Add", "Change", "Delete",
                                   "Replace", "Canceled".
                iEntityType      - Entity Type: Mandatory for transactional data BODs
                                   Possible values: 1 (Warehouse), 2 (Department),
                                   3 (Project)
                iEntityCode      - Entity Code: Mandatory for transactional data BODs
                                   Possible values: The warehouse, department or
                                   project.
                                   The Entity Type and Entity Code are used to
                                   determine the Tenant, AccountingEntity and
                                   Location.
                iDocumentId      - Document ID: Mandatory
                iProcessingAction - Processing Action: Mandatory
                                   Possible values: "OnlyStage", "StageOrPublish",
                                   "OnlyPublish".
                Variable arguments are Identifiers
                                 - The first Identifier is Mandatory
        Output: oExceptionMessage        - The last message if the return
                                           value is not equal to 0.
                                           If more than one  message is
                                           given, these are present in the
                                           oExceptionID
                oExceptionID             - An ID that refers to all error
                                           information. Use the functions in
                                           Exception to get all relevant
                                           information.
        Return: 0                        - BOD is published.
                DALHOOKERROR             - Otherwise.
```

BOD.PublishLNMessage

DLL: tcextbodapi

This function is available from KB2040021.

Syntax:

```
long BOD.PublishLNMessage(
                domain  tcmcs.str30m     iMessageType mb,
        ref     domain  tcmcs.s999m      oExceptionMessage mb,
        ref             long             oExceptionID,
                                         ... )
```

Usage:

```
        Expl:   This function publishes the LnMessageBOD with a specific MessageType.
                The first three arguments are mandatory, in the given order.
                iMessageType can f.i. have value "PurchaseScheduleLine".

                The other arguments should be 'xml_elementname - xml_elementvalue'
                pairs, e.g.:
                        "Description",          tfgld011.desc,
                The number of input arguments can vary.

                The DocumentID is mandatory as one of the xml_element names:
                        "DocumentID",           table.key1,
                        "DocumentID",           table.key2,

                The maximum composed DocumentID_ID string length is 132.
                If the "ActionCode" is not set, the default "Add" is used.
                If "DisplayID" is not set the "DocumentID" is used to create the
                DisplayID.

                To create a user area node a UserAreaName, UserAreaType and
                UserAreaValue are mandatory. UserAreaDescription,
                UserAreaStartDate and UserAreaEndDate can also be used.
                --> In the function call, the same order must be used!

                Allowed values for UserAreaType are:
                    -  "DateTimeType"
                    -  "IndicatorType"
                    -  "NumericType"
                    -  "StringType"

                If an xml_elementname is not in the list of possible
                xml_elementnames, the xml_element value is added to the
                "MessageTypeDetails" node, e.g.:
                input arguments:
                        "MessageType"           "TestType",
                        "Element.1"             field.1,
                        "Element.2"             field.2,
                result node:
                        <MessageTypeDetails>
                                <TestTypeGroup>
                                        <Element.1>field.1</Element.1>
                                        <Element.2>field.2</Element.2>
                                </TestTypeGroup>
                        </MessageTypeDetails>

                xml_elementvalues should apply to the following rules:
                utc dates should be converted to string representation in ISO.
                e.g.: "EffectiveDateTime",      utc.to.iso(datefield, UTC_ISO_Z)
                Use the english description associated with a specific value
                in an enumerated domain.
                e.g.:   "TransactionCategory",  enum.descr$("tfgld.catg",
                                                             tfgld011.catg,
                                                             "2")
                Attributes of an xml_element are reflected using an @ and
                must be placed directly under the xml_element on which they
                are related, e.g.:
                        "PurchaseInvoiceAmount",                "10.000",
                        "PurchaseInvoiceAmount@currencyID",     "EUR",

                        <PurchaseInvoiceAmount currencyID="EUR">10.000
                                                        </PurchaseInvoiceAmount>


                Possible xml_element names:
                    -  "ActionCode"
                    -  "Authorization"
                    -  "Description"
                    -  "DocumentReference"
                    -  "DocumentReference_ID"
                    -  "DocumentReference_LineNumber"
                    -  "DocumentReference_ScheduleLineNumber"
                    -  "DisplayID"
                    -  "DistributionGroup_Contact"
                    -  "DistributionGroup_ContactGroup"
                    -  "DistributionGroup_Email"
                    -  "DistributionGroup_Person"
                    -  "DistributionGroup_PersonGroup"
                    -  "DocumentID"
                    -  "Message_Description"
                    -  "Message_ID"
                    -  "MessageType"
                    -  "Status_Code"
                    -  "Status_EffectiveDateTime"
                    -  "Status_ArchiveIndicator"
                    -  "UserAreaDescription"
                    -  "UserAreaEndDate"
                    -  "UserAreaName"
                    -  "UserAreaStartDate"
                    -  "UserAreaType"
                    -  "UserAreaValue"
        Pre:    Transaction handling is needed.
        Post:   Transaction handling is needed: A commit or abort must be done.
        Input:  iMessageType: Mandatory
                Variable arguments are pairs of two arguments with
                name and value. At least one argument with name "DocumentID"
                is Mandatory.
        Output: oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0                       - BOD is published.
                DALHOOKERROR            - Error.
```

BOD.ResumePublishing

DLL:	tcextbodapi This function is available from KB2267552.

Syntax:	
	```
	long BOD.ResumePublishing(
	                        boolean          iAllBods,
	                domain  tcbod.name       iNoun,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl:   This function resumes the publishing of all BODs or one
	                specified BOD in the current process.
	        Pre:    Function BOD.InterruptPublishing() should have been called to
	                interrupt the publishing.
	        Post:   N/A
	        Output:
	                iAllBods                - true: The publishing of all BODs in the
	                                          current process is resumed.
	                                          false: The publishing of the specified
	                                          BOD in the current process is resumed.
	                                          Value false is not allowed if function
	                                          BOD.InterruptPublishing() was called
	                                          with iAllBods as true.
	                iNoun                   - The (protected) Noun for which the
	                                          publishing must be resumed.
	                                          Mandatory if iAllBods is false. Not
	                                          needed / ignored if iAllBods is true.
	                oExceptionMessage       - A message if the return value is not equal
	                                          to 0. This message contains the root cause of
	                                          the request failure.
	                oExceptionID            - An ID that refers to all error information
	                                          if the return value is <> 0.
	                                          Use the functions in Exception to get
	                                          all relevant information.
	        Return:
	                0                       - The action to resume the publishing was set
	successfully.
	                <> 0                    - The action to resume the publishing can not be
	set.
	```

Public Interfaces for BDE

The following functions are available:

BDE.ExecuteMethod

BDE.ExecuteMethod

DLL:	tcextbdeapi This function is available from KB2061133.

Syntax:	
	```
	long BDE.ExecuteMethod(
	                domain  tcbod.name       iBusinessObject,
	                domain  tcmcs.str30      iMethod,
	                        long             iXMLRequest,
	        ref             long             oXMLResponse,
	        ref             long             oXMLResult,
	        ref     domain  tcmcs.s999m      oExceptionMessage mb,
	        ref             long             oExceptionID )
	```

Usage:	
	```
	        Expl:   This function executes a method for a specified public BDE.
	        Pre:    If the method to be executed updates the database, a db.retry.point()
	                must have been set in the calling program.
	        Post:   If the method to be executed updates the database, a commit.transaction()
	                must be done by the calling program. However, in case of a non-zero
	                return value, an abort.transaction() must be done, otherwise updates may
	                have been done partially, which, once committed, may cause significant
	                data corruption.
	        Input:  iBusinessObject         - The Business Object for which the method
	                                          must be executed, e.g.
	                                          "Item_v3". Mandatory
	                iMethod                 - The method to be executed,
	                                          e.g. "Create" or "Change". Mandatory
	                iXMLRequest             - XML structure with request. Mandatory
	        Output: oXMLResponse            - XML structure with response (if method is
	                                          executed successfully)
	                oXMLResult              - XML structure with result (in case of error)
	                oExceptionMessage       - A message if the return value is not equal
	                                          to 0. This message contains the root cause of
	                                          the of the method failure.
	                oExceptionID            - An ID that refers to all error information. Use
	                                          the functions in Exception to get the error
	                                          messages. Note that more detailed error
	                                          information is in oXMLResult.
	        Return values:
	                0                       - method is executed successfully
	                DALHOOKERROR            - on errors
	```

Chapter 11 Public Interfaces for Project

Public Interfaces for ProjectEstimate

The following functions are available:

ProjectEstimate.GenerateStructuralElements

ProjectEstimate.GenerateStructuralElements

DLL:	tpextestapi This function is available from KB2220662.

Syntax:	`long ProjectEstimate.GenerateStructuralElements(`
	`                domain  tccprj          iProject,`
	`                domain  tpest.vers      iEstimateVersionFrom,`
	`                domain  tpest.vers      iEstimateVersionTo,`
	`                domain  tpest.esid      iEstimateStructureFrom,`
	`                domain  tpest.esid      iEstimateStructureTo,`
	`                domain  tccpcp          iCostComponentFrom,`
	`                domain  tccpcp          iCostComponentTo,`
	`                domain  tpptc.cstl      iExtensionFrom,`
	`                domain  tpptc.cstl      iExtensionTo,`
	`                domain  tppdm.cspa      iBudgetTopElement,`
	`                        boolean         iUpdateFreeVersions,`
	`                        boolean         iUpdateActualVersions,`
	`                        boolean         iUpdateOnlyUsedStructures,`
	`        ref     domain  tcmcs.s999m     oExceptionMessage mb,`
	`        ref             long            oExceptionID )`

Usage:	`        Expl:   This function updates the Estimate Structures for a given`
	`                Project and its Estimate Versions as in session "Generate`
	`                Structural Elements" (tpest1220m000).`
	`                The Structural Elements of the Estimate Structures are generated`
	`                from the applicable sources like Activities, Elements, Cost`
	`                Components, Extensions and User Defined Structures.`
	`        Pre:    This function has its own transaction management. There should`
	`                be no pending logical transaction before calling this function.`
	`        Post:   There is no need to execute an abort or commit after calling`
	`                this function, that is handled within the function.`
	`        Input:  iProject                - Project: Mandatory`
	`                iEstimateVersionFrom    - Estimate Version From`

```
        iEstimateVersionTo      - Estimate Version To
        iEstimateStructureFrom  - Estimate Structure From
        iEstimateStructureTo    - Estimate Structure To
        iCostComponentFrom      - Cost Component From
                                  Only relevant when Estimate Structures
                                  of type 'Cost Component' are selected.
        iCostComponentTo        - Cost Component To
                                  Only relevant when Estimate Structures
                                  of type 'Cost Component' are selected.
        iExtensionFrom          - Extension From
                                  Only relevant when Estimate Structures
                                  of type 'Extension' are selected.
        iExtensionTo            - Extension To
                                  Only relevant when Estimate Structures
                                  of type 'Extension' are selected.
        iBudgetTopElement       - Budget Top Element: Optional
                                  The Top Element for structures of
                                  type Element.
                                  Only relevant when only one Estimate
                                  Version is selected, and Estimate
                                  Structures of type 'Element' are
                                  selected.
        iUpdateFreeVersions     - Update Free Versions: Mandatory
                                  Update Estimate Structures of
                                  Estimate Versions with Status 'Free'.
                                  Possible values:
                                  True : Update Free Versions
                                  False: Do not update Free versions
        iUpdateActualVersions   - Update Actual Versions: Mandatory
                                  Update Estimate Structures of
                                  Estimate Versions with Status 'Actual'.
                                  Possible values:
                                  True : Update Actual Versions
                                  False: Do not update Actual Versions
        iUpdateOnlyUsedStructures
                                - Update Only Used Structures: Mandatory
                                  Only update Estimate Structures which
                                  are used in an Estimate Version.
                                  Possible values:
                                  True : Update only Estimate Structures
                                         which are used in an Estimate
                                         Version
                                  False: Update all Estimate Structures
Output:
        oExceptionMessage       - The last message if the return
                                  value is not equal to 0.
                                  If more than one  message is
                                  given, these are present in the
                                  oExceptionID
        oExceptionID            - An ID that refers to all error
                                  information. Use the functions in
                                  Exception to get all relevant
                                  information.
Return: 0                       - Successful
        <> 0                    - An error occurred
```

Chapter 12 Process Extensions

Process Extensions for HandlingUnitBuilding

The following process extension(s) is/are available:

HandlingUnitBuilding.CustomShipmentLine

HandlingUnitBuilding.CustomShipmentLine

Allows to perform additional structure updates after a handling unit is reused or filled up.

This process extension is available from KB2086985.

Technical information for this process extension:

Usage:	With this Process Extension, it is possible to implement modifications in the handling unit structure when a handling unit is linked to the shipment line as a result of picking inventory or (re)generation of handling units on a shipment line. Note: This process extension will be called after the handling unit has been linked to the shipment line. A handling unit can either be generated, filled-up (anonymous stock is put in an already existing handling unit) or reused (picked handling unit is reused, or part of a picked handling unit is reused).

To implement this process extension, you need to implement the following method(s):

whext.wmd0001.custom.shipment.line

Syntax:	`long whext.wmd0001.custom.shipment.line(`
	`                domain   whinh.shpm        i.shipment,`
	`                domain   tcpono            i.shipment.line,`
	`                domain   whhuid            i.handling.unit,`
	`                         boolean           i.handling.unit.reused,`
	`        ref     domain   tcmcs.str132m     o.error.message.array() fixed mb )`

Usage:	`Expl:   This function can be implemented to influence the handling unit`

	```
	building process during creation of handling units for the
	given shipment line.
	This function will be called from the standard process just
	after a handling unit is linked to the given shipment line.
	If a handling unit is re-used from inventory (during picking /
	release outbound advice when picking is not in the outbound
	process), the flag i.handling.unit.reused will be set to true.

	Do not change the record buffers with the additional logic, or
	if required, store and restore the record buffer prior to
	changing records, or inserting new records.

	Error messages can be logged in the error message array.
	The messages that are to be logged should be started with a @.
	Implementation note: it is wise to start the message with a
	dedicated hard coded string, so it is clear to the end user that
	the message logged is set in the Extension.

	Pre:    NA
	Post:   NA
	Input:  i.shipment      - shipment
	        i.shipment.line - shipment line
	        i.handling.unit - handling unit
	        i.handling.unit.reused - picked handling unit is reused
	Output: o.error.message.array()
	Return: 0/DALHOOKERROR
	```

Process Extensions for IntrastatTransaction

The following process extension(s) is/are available:

IntrastatTransaction.CheckBlockedForReporting

IntrastatTransaction.CheckBlockedForReporting

Check whether, at adding Intrastat Transaction, the status must be set to Blocked for Reporting.

This process extension is available from KB3549440.

Technical information for this process extension:

Usage:	
	```
	With this Process Extension, it is possible to have additional checks
	before an Intrastat Transaction record is written, to set the status
	'Blocked for Reporting' at creating an Intrastat Transaction.
	```

To implement this process extension, you need to implement the following method(s):

lpext.ita0001.check.intrastat.transaction.blocked.for.reporting

Syntax:

```
long lpext.ita0001.check.intrastat.transaction.blocked.for.reporting(
        ref             boolean         o.blocked.for.reporting )
```

Usage:

```
        Expl:
                Use this method to check whether, at adding Intrastat Transaction
                record, the status must be set to "Blocked for Reporting".

                At moment of adding an Intrastat Transaction, LN will do checks
                first. Only if the Intrastat Transaction status is not set to
                "Blocked for Reporting" according to the standard logic in LN,
                this method in the Process Extension is called.
                In this method, own checks can be implemented and the variable
                o.blocked.for.reporting can be set. If the variable is
                set to false, the Intrastat Transaction status will not be changed.
                Otherwise, the status will be set to "Blocked for Reporting".

                Fields that can be used in this process extension:
                Although the Intrastat Transaction is about to be added and not
                yet committed, the Intrastat Transaction fields from lpita360 table
                are already available.

                Note that dal messages set in this function will be ignored by
                the standard.

                ----------------------------------------------------------------
                Start of Example of Implementation
                Pseudocode:

                Hook: Declarations
                        table   tlpita360       |* Intrastat Transactions

                Hook: lpext.ita0001.get.intrastat.transaction.blocked.for.reporting

                        function extern long
                        lpext.ita0001.get.intrastat.transaction.blocked.for.reporting(
                                ref boolean o.blocked.for.reporting)
                        {
                                domain  <domain>        field.value

                                |* Default value is set to False.
                                o.blocked.for.reporting = false

                                select  cisli310.<field>:field.value
                                from    cisli310
                                where   cisli310._index1 = {
                                                        :lpita360.fcmp,
                                                        :lpita360.tran,
                                                        :lpita360.idoc,
                                                        :lpita360.line}
                                as set with 1 rows
                                selectdo
                                        |**************************************
                                        |* If customized criteria are met
                                        |* so the Intrastat transaction status
                                        |* is set to "Blocked for Reporting".
                                        |**************************************
                                        if field.value = <customized criteria> then
                                                o.blocked.for.reporting = true
                                        endif
                                endselect

                                return(0)
                        }
                End of Example of Implementation
                ----------------------------------------------------------------

        Pre:    N.A.
        Post:   N.A.
        Input:  N.A.
        Output: o.blocked.for.reporting - Indicates whether the Intrastat
                                          Transaction record status must be set
                                          to "Blocked for Reporting".
        Return: 0                       - Success
                <> 0                    - Returning a value <> 0 will abort the
                                          creation of Intrastat Transactions.
```

Process Extensions for Invoice

The following process extension(s) is/are available:

Invoice.CustomComposingCriteriaMet

Invoice.CustomComposingCriteriaMet

Determines whether billable line should be composed in the current invoice number or not.

This process extension is available from KB2305642.

Technical information for this process extension:

Usage:	`With this Process Extension, it is possible to have custom composing criteria defined by customer/extender to check whether the current billable line should be composed in the current invoice or not. If this Process Extenstion returns false then program will look for the next available invoice number and if available then this process extension is called again. If no invoice number is available then a new invoice number will be created and this process extenstion will not be called again.`

To implement this process extension, you need to implement the following method(s):

ciext.sli0003.invoice.custom.composing.criteria.met

Syntax:

```
long ciext.sli0003.invoice.custom.composing.criteria.met(
        ref             boolean          o.custom.composing.criteria.met )
```

Usage:

```
        Expl:
                Use this method to check your own defined composing criteria,
                whether the current billable line should be composed in the
                current invoice number or not.

                At the time of composing of billable lines together, LN performs
                its own check to determine whether the lines can be composed
                together based on the standard composing criteria logic.
                Only when standard logic in LN allows the current billable line
                to be composed together in the current invoice, this method in
                the Process Extension is called.

                Default value of "o.custom.composing.criteria.met" is True.

                In this method, own checks can be implemented and the variable
                o.custom.composing.criteria.met can be set. If the variable is
                set to false, then the current billable line will not be
                composed in the current invoice but then it will look for the
                next available invoice number and if available then this process
                extension is called again. If no invoice number is available
                then a new invoice number will be created and this process
                extenstion will not be called again.
                If the variable is set to true then the current billable
                line will be added in the current invoice number.

                Fields that are available to be used in this Process Extension:
                - All fields of table: Billable lines (cisli810)
                - All fields of table: Although the Invoice is not yet
                                       committed, the Invoice Header (cisli305)
                                       fields are already available. The Invoice
                                       header key fields can be used to read
                                       other tables like Invoice Lines
                                       (cisli310) or Invoice Lines - Additional
                                       Fields (cisli311).

                Note:
                1) Dal messages set in this function will be ignored by
                the standard.
                2) User must use bind variables when they read data,
                otherwise the standard flow may be affected.
                3) External variables that are available to be used in this
                Process.

                -----------------------------------------------------------------
                Start of Example of Implementation
                Requirement -   Each Packing slip should be composed separately
                                in a new invoice number.
                Solution    -   Packing slip is not available in the invoicing
                                method to make it one of the standard composing
                                criteria. So Customer can use this process
                                extension to make it one of the custom composing
                                criteria.

                                Packing slip is available on the Billale line
                                and is available in the Invoice line Additional
                                Fields (cisli311).
                Pseudocode  -
                Hook: Declarations
                        table   tcisli305       |* Invoice Header
                        table   tcisli810       |* Billable Lines

                Hook: ciext.sli0003.invoice.custom.composing.criteria.met

                        function extern long
                        ciext.sli0003.invoice.custom.composing.criteria.met(
                                ref boolean o.custom.composing.criteria.met)
                        {
                                domain  cisli.fldv2     field.value     fixed
                                domain  cisli.fldc      field.code      fixed

                                field.code = "PKSP"

                                |* Default value is set to True.
                                o.custom.composing.criteria.met = true

                                select  cisli311.fldv:field.value
                                from    cisli311
                                where   cisli311._index1 =
                                                        {:cisli305.sfcp,
                                                        :cisli305.tran,
                                                        :cisli305.idoc}
                                and     cisli311.fldc = :field.code
                                and     cisli311.fldv <> :cisli810.pksp
                                as set with 1 rows
                                selectdo
                                        |*************************************
                                        |* If different Packing Slip is
                                        |* available in  the invoice then the
                                        |* current billable line should not be
                                        |* inserted into given invoice number
                                        |* so the composing criteria does not
                                        |* meet here.
                                        |*************************************
                                        o.custom.composing.criteria.met = false
                                endselect

                                return (0)
                        }
                End of Example of Implementation
                ----------------------------------------------------------------

        Pre:    N.A.
        Post:   N.A.
        Input:  N.A.
        Output: o.custom.composing.criteria.met -
                                        Determines whether current billable line
                                        should be composed in the current
                                        invoice or not.
        Return: 0                       - Success
                DALHOOKERROR            - When an error occurs during checking;
                                          the invoice will not be composed
                                          together.
```

Process Extensions for OutboundAdvice

The following process extension(s) is/are available:

OutboundAdvice.SkipPrint

OutboundAdvice.SkipPrint

Skips Printing of Outbound Advice.

This process extension is available from KB2092284.

To implement this process extension, you can use the information below:

Usage:	`Process Extension OutboundAdvice.SkipPrint can be used`
	`to skip the printing of Outbound Advice.`
	
	`Sessions where this Process Extension can be implemented:`
	`- Print Outbound Advice (whinh4460m000)`
	
	`Fields that are available to be used in this Process Extension:`
	`- All fields of tables:`
	`        - Outbound Advice (whinh225)`
	`        - Outbound Order Lines (whinh220)`

Process Extensions for PaymentReceipt

The following process extension(s) is/are available:

PaymentReceipt.CustomXMLHandling

PaymentReceipt.CustomXMLHandling

Define custom elements for XML Payments/Receipts used at creating the XML file.

This process extension is available from KB2109354.

Technical information for this process extension:

Usage:	`With this Process Extension, it is possible to define extra custom`
	`elements to be used in the XML Payment/Receipt Layout.`

To implement this process extension, you need to implement the following method(s):

tfext.cmg0001.define.custom.payment.elements

Syntax:

```
long tfext.cmg0001.define.custom.payment.elements(
        ref             long            o.number.of.elements,
        ref     domain  tcmcs.st12      o.custom.element.codes() fixed,
        ref     domain  tcmcs.str132m   o.custom.element.descriptions() fixed mb,
        ref     domain  tfcmg.xttp      o.custom.element.data.types() )
```

Usage:

```
        Expl:
                Use this method to define extra custom defined elements to be
                used in the XML Payment layout and file. These custom element
                codes must start with "8" or "9"; other defined element codes
                will be ignored.

                Mapping a custom element to a Remittance Related XML Tag, only
                the custom elements starting with "8" can be used.
                The custom elements starting with "9" can only be used for
                mapping to non-remittance related XML Tags.

                The custom defined elements can then be used for mapping to
                an XML attribute in session XML Payment/Receipt Layout Lines
                (tfcmg0125m000).
                If a custom defined element is mapped, during building the XML
                file, a value must be retrieved for the element. For that, the
                following method must be used:
                        tfext.cmg0001.get.value.for.custom.payment.element

                ----------------------------------------------------------------
                Start of Example of Implementation

                long    length.element.code
                long    length.element.description
                long    elem.nr

                long    dummy.long

                dummy.long = rdi.domain.string(
                                        domainof(o.custom.element.codes),
                                        length.element.code,
                                        dummy.long)
                dummy.long = rdi.domain.string(
                                        domainof(o.custom.element.descriptions),
                                        length.element.description,
                                        dummy.long)

                elem.nr = 0

                |***************************************************************
                |* We will add 2 custom elements.
                |***************************************************************
                o.number.of.elements = 2

                if alloc.mem(   o.custom.element.codes,
                                length.element.code,
                                o.number.of.elements) +
                   alloc.mem(   o.custom.element.descriptions,
                                length.element.description,
                                o.number.of.elements) +
                   alloc.mem(   o.custom.element.data.types,
                                o.number.of.elements) <> 0 then
                        return(DALHOOKERROR)
                endif

                |***************************************************************
                |* Add the element Name for the Pay-to BP
                |***************************************************************
                elem.nr = elem.nr + 1
                o.custom.element.codes(1, elem.nr) = "910210000000"
                o.custom.element.descriptions(1, elem.nr) =
                        "Composed Payments/Pay-to Business Partner/Name"
                o.custom.element.data.types(elem.nr) = tfcmg.xttp.string

                |***************************************************************
                |* Add the element Transaction Amount Positive
                |***************************************************************
                elem.nr = elem.nr + 1
                o.custom.element.codes(1, elem.nr) = "911200000000"
                o.custom.element.descriptions(1, elem.nr) =
                        "Composed Payments/Transaction Amount Positive"
                o.custom.element.data.types(elem.nr) = tfcmg.xttp.decimal

                return(0)

                End of Example of Implementation
                ----------------------------------------------------------------

                In above example, a custom element is added to
                influence the transaction amount in the file (i.e. making the
                transaction amount positive); note that this impacts the control
                amount in the file and thus a custom element is needed as well
                to determine the control amount in the extension.

        Pre:    N.A.
        Post:   N.A.
        Input:  N.A.
        Output: o.number.of.elements    - Number of additional custom payment
                                          elements
                o.custom.element.codes  - Array with element codes. The element
                                          code must start with a "9".
                                          Maximum of 12 characters.
                                          E.g. "910100000000".
                o.custom.element.descriptions
                                        - Array with descriptions of the
                                          element codes.
                                          Maximum of 132 characters.
                o.custom.element.data.types
                                        - Array with data types of the
                                          element code.
                                          Data type should be one of the
                                          following values:
                                          tfcmg.xttp.string      (String)
                                          tfcmg.xttp.date        (Date)
                                          tfcmg.xttp.utc         (UTC Date)
                                          tfcmg.xttp.decimal     (Decimal Number)
                                          tfcmg.xttp.bool        (Boolean)

        Return: 0                       - Success
```

```
                DALHOOKERROR            - When an error occurs in defining
                                          the custom elements.
```

tfext.cmg0001.define.custom.receipt.elements

```
Syntax: long tfext.cmg0001.define.custom.receipt.elements(
                ref             long            o.number.of.elements,
                ref     domain  tcmcs.st12      o.custom.element.codes() fixed,
                ref     domain  tcmcs.str132m   o.custom.element.descriptions() fixed mb,
                ref     domain  tfcmg.xttp      o.custom.element.data.types() )
Usage:          Expl:
                        Use this method to define extra custom defined elements to be
                        used in the XML Receipt layout and file. These custom element
                        codes must start with "8" or "9"; other defined element codes
                        will be ignored.

                        Mapping a custom element to a Remittance Related XML Tag, only
                        the custom elements starting with "8" can be used.
                        The custom elements starting with "9" can only be used for
                        mapping to non-remittance related XML Tags.

                        The custom defined elements can then be used for mapping to
                        an XML attribute in session XML Payment/Receipt Layout Lines
                        (tfcmg0125m000).
                        If a custom defined element is mapped, during building the XML
                        file, a value must be retrieved for the element. For that, the
                        following method must be used:
                                tfext.cmg0001.get.value.for.custom.receipt.element

                        An example implementation is given in the method
                        tfext.cmg0001.define.custom.payment.elements, which is likely
                        same.

                Pre:    N.A.
                Post:   N.A.
                Input:  N.A.
                Output: o.number.of.elements    - Number of additional custom receipt
                                                  elements
                        o.custom.element.codes  - Array with element codes. The element
                                                  code must start with a "9".
                                                  Maximum of 12 characters.
                                                  E.g. "910900000000".
                        o.custom.element.descriptions
                                                - Array with descriptions of the
                                                  element codes.
                                                  Maximum of 132 characters.
                        o.custom.element.data.types
                                                - Array with data types of the
                                                  element code.
                                                  Data type should be one of the
                                                  following values:
                                                  tfcmg.xttp.string      (String)
                                                  tfcmg.xttp.date        (Date)
                                                  tfcmg.xttp.utc         (UTC Date)
                                                  tfcmg.xttp.decimal     (Decimal Number)
                                                  tfcmg.xttp.bool        (Boolean)
```

```
        Return: 0                       - Success
                DALHOOKERROR            - When an error occurs in defining
                                          the custom elements.
```

tfext.cmg0001.get.value.for.custom.payment.element

Syntax:

```
domain tcmcs.s999m tfext.cmg0001.get.value.for.custom.payment.element(
                domain  tcncmp           i.payment.company,
                domain  tcmcs.st12       i.custom.element.code,
                domain  tccwoc           i.accounting.office )
```

Usage:

```
        Expl:
                Use this method to get a value for the defined custom payment
                element code during the creation of the Payment XML File.

                The custom elements must be defined with the following method:
                        tfext.cmg0001.define.custom.payment.elements

                Following tables are current when this extension is triggered
                (during creation of the XML Bank File):

                tccom000        Implemented Software Components (Companies)
                tccom100        Business Partners
                tccom115        Bank Accounts by Pay-by Business Partner
                tccom125        Bank Accounts by Pay-to Business Partner
                tccom130        Addresses
                tccom139        Cities by Country
                tcmcs002        Currencies
                tcmcs010        Countries
                tcmcs029        Business Partner Types
                tcmcs046        Languages
                tcmcs143        States/Provinces

                tfcmg001        Bank Relations
                tfcmg003        Payment/Receipt Methods
                tfcmg009        Financial Institutions
                tfcmg011        Bank Branches

                tfcmg103        Composed Payments
                tfcmg002        Reasons for Payment
                tfcmg025        Payment/Receipt XML Layout Details

                The below table is current only for Remittance Related elements:
                tfcmg101        Payment/Trade Notes Payable Advice

                ----------------------------------------------------------------
                Start of Example of Implementation

                domain  tcnama          name.of.bp
                domain  tfgld.amnt      composed.amount

                name.of.bp      = ""
                composed.amount = 0.0

                on case i.custom.element.code
                case "910210000000":
                        |* Name of Composed Pay-to BP
                        get.var(pid, "tccom100.nama", name.of.bp)
                        return(name.of.bp)
                case "911200000000":
                        |* Transaction Amount Composed Payment (Positive)
                        |* Return always a POSITIVE value of the Amount field
                        get.var(pid, "tfcmg103.amnt", composed.amount)
                        return( trim$(

                                sprintf$("%@ZZZZZZZZZZZZZZ9V.99@" ,
                                           abs(composed.amount) )))
                default:
                        break
                endcase

                return("")

                End of Example of Implementation
                ----------------------------------------------------------------

        Pre:    N.A.
        Post:   N.A.
        Input:  i.payment.company       - Payment Company
                i.custome.element.code  - The custom element code mapped to
                                          an XML attribute.
                i.accounting.office     - Accounting Office.
        Output: N.A.
        Return: String with the value of the custom payment element.
```

tfext.cmg0001.get.value.for.custom.receipt.element

Syntax:

```
domain tcmcs.s999m tfext.cmg0001.get.value.for.custom.receipt.element(
                domain  tcncmp           i.receipt.company,
                domain  tcmcs.st12       i.custom.element.code,
                domain  tccwoc           i.accounting.office )
```

Usage:

```
        Expl:
                Use this method to get a value for the defined custom receipt
                element code during the creation of the Payment XML File.

                The custom elements must be defined with the following method:
                        tfext.cmg0001.define.custom.receipt.elements

                Following tables are current when this extension is triggered
                (during creation of the XML Bank File):

                tccom000        Implemented Software Components (Companies)
                tccom100        Business Partners
                tccom115        Bank Accounts by Pay-by Business Partner
                tccom125        Bank Accounts by Pay-to Business Partner
                tccom130        Addresses
                tccom139        Cities by Country
                tcmcs002        Currencies
                tcmcs010        Countries
                tcmcs029        Business Partner Types
                tcmcs046        Languages
                tcmcs143        States/Provinces

                tfcmg001        Bank Relations
```

	```
	tfcmg003        Payment/Receipt Methods
	tfcmg009        Financial Institutions
	tfcmg011        Bank Branches
	tfcmg027        Direct Debit Mandates

	tfcmg403        Composed Direct Debits
	tfcmg025        Payment/Receipt XML Layout Details

	The below table is current only for Remittance Related elements:
	tfcmg401        Direct Debit Advice

	An example implementation is given in the method
	tfext.cmg0001.get.value.for.custom.payment.element, which is
	likely same.

	Pre:    N.A.
	Post:   N.A.
	Input:  i.receipt.company       - Receipt Company
	        i.custome.element.code  - The custom element code mapped to
	                                  an XML attribute.
	        i.accounting.office     - Accounting Office.
	Output: N.A.
	Return: String with the value of the custom receipt element.
	```

Process Extensions for Procurement

The following process extension(s) is/are available:

Procurement.DefaultPurchaseOffice

Procurement.DefaultPurchaseOffice

Determine customer specific default Purchase Office for Procurement.

This process extension is available from KB2155811.

Technical information for this process extension:

Usage:	
	```
	With this Process Extension, it is possible to set a customer determined
	purchase office from an extension to a value other than the defaulted
	value based on LN logic .
	This purchase office (if valid) is then used in all flows where a
	purchase office is defaulted.
	```

To implement this process extension, you need to implement the following method(s):

tdext.pur0001.get.customer.determined.default.purchase.office

Syntax:	
	```
	long tdext.pur0001.get.customer.determined.default.purchase.office(
	                domain  tcmcs.str8       i.defaulting.origin,
	                domain  tccwoc           i.purchase.office.from.standard.logic,
	                domain  tcemm.grid       i.enterprise.unit,
	                domain  tccom.bpid       i.buy.from.business.partner,
	                domain  tccom.bpid       i.ship.from.business.partner,
	                domain  tcitem           i.item,
	                domain  tcncmp           i.logistic.company,
	                domain  tcsite           i.site,
	                domain  tccwar           i.warehouse,
	        ref     domain  tccwoc           o.purchase.office )
	```

Usage:	
	```
	        Expl:   Use this method to get a customer determined default purchase
	                office from an extension. This office will be used in
	                Procurement.

	                The standard logic in LN determines the purchase office from
	                master data like enterprise unit, item data or user profile.

	                Using this process extension, the defaulted value can be
	                changed to another value.

	                Note:
	                1. The input arguments can be used to determine the option.
	                2. LN will check if the provided default is valid.
	                3. The purchase office defaulted based on standard LN logic will
	                   overrule the selected office from the extension, otherwise
	                   standard logic may fail.
	        Pre:    NA
	        Post:   NA
	        Input:  i.defaulting.origin                    - Defaulting Origin,

	                                Possible Values:
	                                tdpur100 - RFQ
	                                tdpur200 - Purchase Requisition
	                                tdpur300 - Purchase Contract
	                                tdpur310 - Purchase Schedule
	                                tdpur400 - Purchase Order
	                                Empty    - Generation of a purchase order /
	                                           schedule via 'generate purchase
	                                           order/schedule flows' when e.g.,
	                                           calling from other packages (Project,
	                                           Service, Warehousing, etc.).
	                i.purchase.office.from.standard.logic   - As defaulted under the
	                                                          standard logic
	                i.enterprise.unit                       - Enterprise Unit
	                i.buy.from.business.partner             - Buy-from Business
	                                                          Partner
	                i.ship.from.business.partner            - Ship-from Business
	                                                          Partner
	                i.item                                  - Item
	                i.logistic.company                      - Logistic Company
	                i.site                                  - Site
	                i.warehouse                             - Warehouse
	        Output: o.purchase.office                       - The purchase office
	                                                          determined by the
	                                                          extension.
	        Return: 0                       -       Success
	                DALHOOKERROR            -       When an error occurs in the
	```

	determination of the purchase office.

Process Extensions for ProjectPCS

The following process extension(s) is/are available:

ProjectPCS.SkipClose

ProjectPCS.SkipClose

Project PCS Skip Close of Project.

This process extension is available from KB2220251.

Technical information for this process extension:

Usage:	With this process extension, the Project close functionality can be controlled for specific Main Projects, Sub Projects & Single Projects.

To implement this process extension, you need to implement the following method(s):

tiext.pcs0001.project.pcs.skip.close

<table>
<tr><td>Syntax:</td><td><pre>long tiext.pcs0001.project.pcs.skip.close(
        ref   boolean   o.skip.close,
        ref   string   o.message())</pre></td></tr>
<tr><td>Usage:</td><td><pre>   Expl: This function is called in the process of Closing Projects:
                        - Main Projects and Structures
                        - Sub Projects
                        - Single Projects
                To decide if the process must be skipped for a specific Closing
                Project Type.
                A message may be returned, to present information about the
                decision to the user.

                When this function is called all fields of tables:
                - General Project Data (tipcs020)
                - Project Details (tipcs030)
                are current.

        Pre: NA</pre></td></tr>
</table>

	```
	Post:   NA
	Input:  NA
	Output: o.skip.close            - decision result wherein
	        o.message               - message, multibyte - max 300 characters

	Return: 0                       - success
	        DALHOOKERROR            - error
	```

tiext.pcs0001.project.pcs.skip.close.implemented

Syntax:	`boolean tiext.pcs0001.project.pcs.skip.close.implemented( )`

Usage:	
	```
	Expl:   This function checks if the process extension
	        ProjectPCS.SkipClose is implemented.
	pre:    N.a.
	Post:   N.a.
	Input:  N.a.
	Output: N.a.
	Return: true/false
	```

Process Extensions for PurchaseOrderLine

The following process extension(s) is/are available:

PurchaseOrderLine.SkipPrintPurchaseOrderReminder

PurchaseOrderLine.SkipPrintPurchaseOrderReminder

Skips Purchase Order Line when Printing Purchase Order Reminders.

This process extension is available from KB2215353.

To implement this process extension, you can use the information below:

Usage:	
	```
	Process Extension PurchaseOrderLine.SkipPrintPurchaseOrderReminder can
	be used to skip specific Purchase Order Lines when printing Purchase
	Order Reminders.

	Sessions where this Process Extension can be implemented:
	- Print Purchase Order Reminders (tdpur4403m000)

	Fields that are available to be used in this Process Extension:
	- All fields of table Purchase Order Lines (tdpur401)
	```

	Pseudocode:
	Below you can find an example:
	
	Hook: Declarations
	
	        table ttdpur401
	
	Hook: ext.skip
	
	        function extern boolean ext.skip()
	        {
	                if <condition on tdpur401 = true> then
	                        return(true)
	                endif
	
	                return (false)
	        }

Process Extensions for PurchaseSelfBilledInvoice

The following process extension(s) is/are available:

PurchaseSelfBilledInvoice.CustomCompose

PurchaseSelfBilledInvoice.CustomCompose

Define a custom composing order for Purchase Self-Billing.

This process extension is available from KB2064131.

Technical information for this process extension:

Usage:	With this Process Extension, it is possible to set an own custom compose criteria in the Self-Billing Method. The Purchase Self-Billing (tfacp2290m000) process will compose the invoices based on the defined custom composing criteria.

To implement this process extension, you need to implement the following method(s):

tfext.acp0001.set.custom.composing.fields

Syntax:	long tfext.acp0001.set.custom.composing.fields(

	ref string o.composing.fields())
Usage:	Expl: Use this method in order to use a Custom Compose Criteria for the Self-Billing Method. If an implementation of this method is done, and the string with composing fields is filled, and this method is returning 0, then in the Self-Billing Method (tcmcs0157m000) session the Compose Criteria (tcmcs057.cmpc) field can be set as Custom Compose. Via this method, one or more fields can be defined in a (comma separated) string. Those defined fields will be used as composing criteria for creating the Purchase Self-Billed Invoice in the Purchase Self-Billing (tfacp2290m000) process. In the Custom Compose flow, the standard Self-Billing will use the below pre-defined fields already for composing the Purchase Self-Billed Invoice (as these are invoice header attributes): tfacp240.ifbp, tfacp240.otbp, tfacp240.ccur, tfacp240.vatc, tfacp240.ptyp, tfacp240.rtyp, tfacp240.mcfr So, via this method, EXTRA composing attributes can be defined. Note that only attributes of below tables may be used: - Order Data for Approval (tfacp240) - Purchase Consumptions (tfacp249) Also Customer Defined Fields (CDF), added to the above mentioned tables (tfacp240 and tfacp249), can be used as extra composing fields. EXAMPLE: If the composing must also be done on Receipt (tfacp249.rcno) and an own CDF field (tfacp249.cdf_0001), this method should return: o.composing.fields = "tfacp249.rcno, tfacp249.cdf_0001" return(0) Pre: N.A. Post: N.A. Input: N.A. Output: o.composing.fields - String with composing composing fields (comma separated) Maximum string length is 500 (sb). Return: 0 - Success DALHOOKERROR - When an error occurs in setting the composing fields.

Process Extensions for SalesCheckInventory

The following process extension(s) is/are available:

SalesCheckInventory.CustomSorting

SalesCheckInventory.DefaultInventoryShortageOption

SalesCheckInventory.SkipShortageLine

SalesCheckInventory.CustomSorting

Define a customer specific 'Sort Lines By' for Sales Check Inventory.

This process extension is available from KB2066317.

Technical information for this process extension:

Usage:	`With this Process Extension, it is possible to set a customer defined sorting from an extension. This sorting will be used in 'Check Inventory for Sales Orders' (tdsls4217m000) when the 'Sort Lines By' option 'Customer Defined' is selected.`

To implement this process extension, you need to implement the following method(s):

tdext.sls0001.get.customer.defined.order.by

Syntax:

```
long tdext.sls0001.get.customer.defined.order.by(
        ref             string          o.order.by.string.mb() )
```

Usage:

```
        Expl:   Use this method to get a customer defined order by clause from
                an extension. This order by clause will be used in 'Check
                Inventory for Sales Orders' (tdsls4217m000) when the 'Sort Lines
                By' option 'Customer Defined' is selected.

                The standard order by clauses used by the 'Sort Lines By' are:

                - Order Date:                 " order by tdsls417.odat asc "
                - Customer Requested Date:    " order by tdsls417.ddtc asc "
                - Planned Delivery Date:      " order by tdsls417.ddta asc "

                Using 'Customer Defined' option, an order by constructed by
                the extension could be, as an example:

                " order by tdsls417.cdf_0001 desc, tdsls417.odat asc "

                Note:
                1. Only attributes from table tdsls417 can be used,
                   including customer defined fields.
                2. Run time errors occur when specifying attributes from other
```

	tables or when wrong syntax is constructed. Pre: NA Post: NA Input: NA Output: o.order.by.string.mb - Maximum string length is 500. Multi byte attributes are allowed. Return: 0 - Success DALHOOKERROR - When an error occurs in determination of the order by clause.

SalesCheckInventory.DefaultInventoryShortageOption

Determine customer specific default 'Automatic Inventory Shortage Option' for Sales Check Inventory.

This process extension is available from KB2150032.

Technical information for this process extension:

Usage:	With this Process Extension, it is possible to set a customer determined automatic inventory option (enumerated value of domain tdsls.ssop) to a value other than the defaulted value based on LN Master Data (sales order type) from an extension. This option will be used in 'Check Inventory for Sales Orders' (tdsls4217m000) and any other process triggering the automatic inventory handling in Sales.

To implement this process extension, you need to implement the following method(s):

tdext.sls0002.get.customer.determined.default.automatic.inventory.shortage.option

Syntax:	`long tdext.sls0002.get.customer.determined.default.automatic.inventory.shortage.option(`
	`domain tdsls.koor i.document.type,`
	`domain tcorno i.document,`
	`domain tcpono i.line,`
	`domain tcpono i.sequence,`
	`domain tcpono i.component.sequence,`
	`domain tccotp i.order.type,`
	`domain tcitem i.item,`
	`boolean i.recheck.inventory.promised.line,`
	`domain tdsls.ssop`
	`i.automatic.inventory.shortage.option.from.master.data,`
	`ref domain tdsls.ssop o.automatic.inventory.shortage.option )`

Usage:	Expl: Use this method to get a customer determined default automatic inventory handling option from an extension. This option will be used in 'Check Inventory for Sales Orders' (tdsls4217m000) and any other process triggering the automatic inventory handling in

```
                Sales.

                The standard logic in LN determines the automatic inventory
                shortage handling option (enumerated value of domain
                tdsls.ssop) from the sales order type and is also
                determined by the item and if inventory is rechecked.

                Using this process extension, the defaulted value can be
                changed to another option of the existing options of the domain.

                As an example: the option defaulted by standard LN may be
                'ATP - When Available' (tdsls.ssop.atp.fixed.wh). Based on logic
                in the extension, this value may be overruled for specific
                order lines by 'ATP - When Available - Single Delivery'
                (tdsls.ssop.del.date).

                Note:
                1. The input arguments can be used to determine the option.
                2. Only values from domain tdsls.ssop can be used.
                3. Run time errors occur when non-existing enumumerated options
                   of domain tdsls.ssop are provided; an empty value is not
                   overruling the default value from the master data.
                4. Standard LN logic may overrule the selected option, otherwise
                   standard logic may fail.
        Pre:    NA
        Post:   NA
        Input:  i.document.type         -       Sales Quote Line or
                                                Sales Order Line or
                                                Sales Order Line Component
                i.document              -       Sales Quote or Sales Order
                i.line                  -       Sales Quote Line or
                                                Sales Order Line
                i.sequence              -       Sales Quote Alternative or
                                                Sales Order Sequence
                i.component.sequence    -       Sales Order Component Line
                i.order.type            -       Sales Order Type
                i.item                  -       Sales Quote Line Item or
                                                Sales Order Line Item or
                                                Sales Order Component Item
                i.recheck.inventory.promised.line
                                        -       If inventory is being rechecked
                                                (True/False)
                i.automatic.inventory.shortage.option.from.master.data
                                        -       The option determined by
                                                standard LN.
        Output: o.automatic.inventory.shortage.option
                                        -       The option determined by the
                                                extension.
        Return: 0                       -       Success
                DALHOOKERROR            -       When an error occurs in the
                                                determination of the inventory
                                                option.
```

SalesCheckInventory.SkipShortageLine

Skips Sales Order Inventory Shortage Lines during Sales Check Inventory.

This process extension is available from KB2150032.

To implement this process extension, you can use the information below:

Usage:	
	```
	Process Extension SalesCheckInventory.SkipShortageLine can be used
	to skip Sales Order Inventory Shortage Lines during Sales Check
	Inventory.

	Note: Shortage lines can be present for Sales Order Lines and Sales
	Order Line Components. Sales Order Line Components are used when
	'Advanced Kitting' is implemented and Component Handling of the
	sales order line is set to 'Component Lines'.

	Shortage Lines for Sales Quotes are not supported/handled in this flow.

	Session where this Process Extension can be implemented:
	- Check Inventory Sales Orders (tdsls4217m000)

	Fields that are available to be used in this Process Extension:
	- Only the primary key fields of the Sales Order Inventory Shortage
	  Lines (tdsls417) are current. These fields are:
	        - tdsls417.orno
	        - tdsls417.pono
	        - tdsls417.sqnb
	        - tdsls417.csqn
	        - tdsls417.koor - Applicable values in this process are
	                          'Sales Order Line' and
	                          'Sales Order Line Component'

	The Check Inventory Sales Orders process checks and rechecks
	(if applicable) the inventory for sales order lines and sales order
	component lines, based on shortage lines in table tdsls417. The primary
	key fields of this table are current and can e.g., be used to read data
	from the sales order (line) and also sales order component line in case
	of component handling to build skip conditions.

	External variables that are available to be used in this Process
	Extension:
	- proc_ext_skip_shortage_line_rechecking [ type: boolean ].
	  Supported values are:

	        - True  - The process is rechecking inventory for (promised)
	                  lines.
	        - False - The process is checking the inventory for lines.

	Note: tables and external variables must also be declared in the
	Process Extension.

	Pseudocode:
	Below you can find an example how to handle the conditions for sales
	order lines and component lines.

	Hook: Declarations

	        table ttdsls417
	        table ttdsls401
	        table ttdsls463

	        extern  boolean         proc_ext_skip_shortage_line_rechecking

	Hook: ext.skip

	        function extern boolean ext.skip()
	```

```
{
        on case tdsls417.koor
        case tdsls.koor.sls.line:       |* Sales Order Line

                read data using tdsls417.orno, tdsls417.pono,
                tdsls417.sqnb

                if <condition on the data read = true> then
                        return(true)
                endif

                break

       case tdsls.koor.sls.line.comp:   |* Sales Order Line Component

                read data using tdsls417.orno, tdsls417.pono,
                tdsls417.sqnb, tdsls417.csqn

                if <condition on the data read = true> then
                        return(true)
                endif

                So in case of a component line, component lines
                can also be skipped based on tdsls401 data
                if the data is read.

                break

        default:
                break
        endcase

        return (false)
}
```

Process Extensions for SalesOrder

The following process extension(s) is/are available:

SalesOrder.SkipReleaseToWarehousing

SalesOrder.SkipReleaseToWarehousing

Skips Sales Order Lines and Sales Order Line Components when Releasing to Warehousing.

This process extension is available from KB2070268.

To implement this process extension, you can use the information below:

Usage:

```
Process Extension SalesOrder.SkipReleaseToWarehousing can be used
to skip Sales Order Lines and Sales Order Line Components when
Releasing to Warehousing.

Note: Sales Order Line Components are used when 'Advanced Kitting'
is implemented and Component Handling of the sales order line is
set to 'Component Lines'.

Sessions where this Process Extension can be implemented:
- Release Sales Orders to Warehousing (tdsls4246m000)
- All sessions and processes that trigger the release of lines to
  Warehousing (like automatic processing logic).

Fields that are available to be used in this Process Extension:
- All fields of table Sales Order Lines (tdsls401) when releasing
  order lines or component lines.
- All fields of table Sales Order Line Components (tdsls463) when
  releasing component lines.

External variables that are available to be used in this Process
Extension:
- proc_ext_skip_rtw_so_line_type [ type: string(20) ].
  Supported values are:

        - order_line
        - component_line

Note: tables and external variables must also be declared in the
Process Extension.

The sales order release to warehousing process releases sales order
lines. In case of component handling (tdsls401.cphl =
tdcphl.component.lines), sales order line is selected first and then
the component lines are released.
The sales order line table is tdsls401.
The sales order line components table is tdsls463.
So in case of component lines, skip conditions can be built both on
current tdsls401 data and tdsls463 data as instructed below.

The process extension must be applied to these two tables.

Pseudocode:
Below you can find an example how to handle the conditions for sales
order lines and component lines.

Hook: Declarations

        table ttdsls401
        table ttdsls463

        extern          string  proc_ext_skip_rtw_so_line_type(20)

Hook: ext.skip

        function extern boolean ext.skip()
        {
                on case proc_ext_skip_rtw_so_line_type
                case "order_line":
                        if <condition on tdsls401 = true> then
                                return(true)
                        endif
                        So in case of order_line, component handling
                        sales order lines can also be skipped based on
                        tdsls401 data.

                        break
                case "component_line":
                        if <condition on tdsls463 = true> then
                                return(true)
                        endif
                        AND / OR
                        if <condition on tdsls401 = true> then
                                return(true)
                        endif

                        So in case of component_line, component lines
                        can also be skipped based on tdsls401 data.

                        break
                default:
                        break
                endcase

                return (false)
        }
```