EXPRESSION REFERENCE

Table of Contents

1 Introduction

1.1 General Information

TicketXpert.NET (TXP) and *WorkflowXpert.NET* (WXP) have their own expression language, which can be used for individual customization or substantial extensions at different locations. Expressions are particularly helpful, when it is not possible to solve a complex configuration problem by only using the user interface – the expression system thus offers extension options specially tailored to power users.

Expressions mainly can be used in three modules: in the *Messaging & Collaboration* module for the creation of custom message templates, in the *Service Level Management* for the computation of non-standard escalation dates, and in the *Workflow Management*. Because not all elements of the expression language can be used in every module, the following sections describe the application of the expression language in the respective module.

1.2 Access to Ticket Fields

You can access the content of ticket fields via miscellaneous expressions. At this, please make sure that the ticket fields you want to access via expression do not use any mathematical operators (e.g. "-" or "+") or similar special characters. These special characters can prevent the proper access to the ticket fields.

2 Messaging & Collaboration

For every message type, there are several placeholders available, which will be replaced by the respective text on sending. Additionally to the default placeholders that are located on the right side (placeholder bar) and faded in dynamically on mouseover, there is an option of adding individually defined placeholders.

2.1 Prefix and Suffix

Every placeholder defined individually begins with the prefix {$ and ends with the suffix $} (start and end token). Between the prefix and the suffix, an arbitrary expression can be entered, which can be interpreted in context to the message to be sent.

If an expression cannot be interpreted, NO text is inserted into the message in this spot.

2.2 Expressions

The following screenshot shows the use of expressions in a message template.
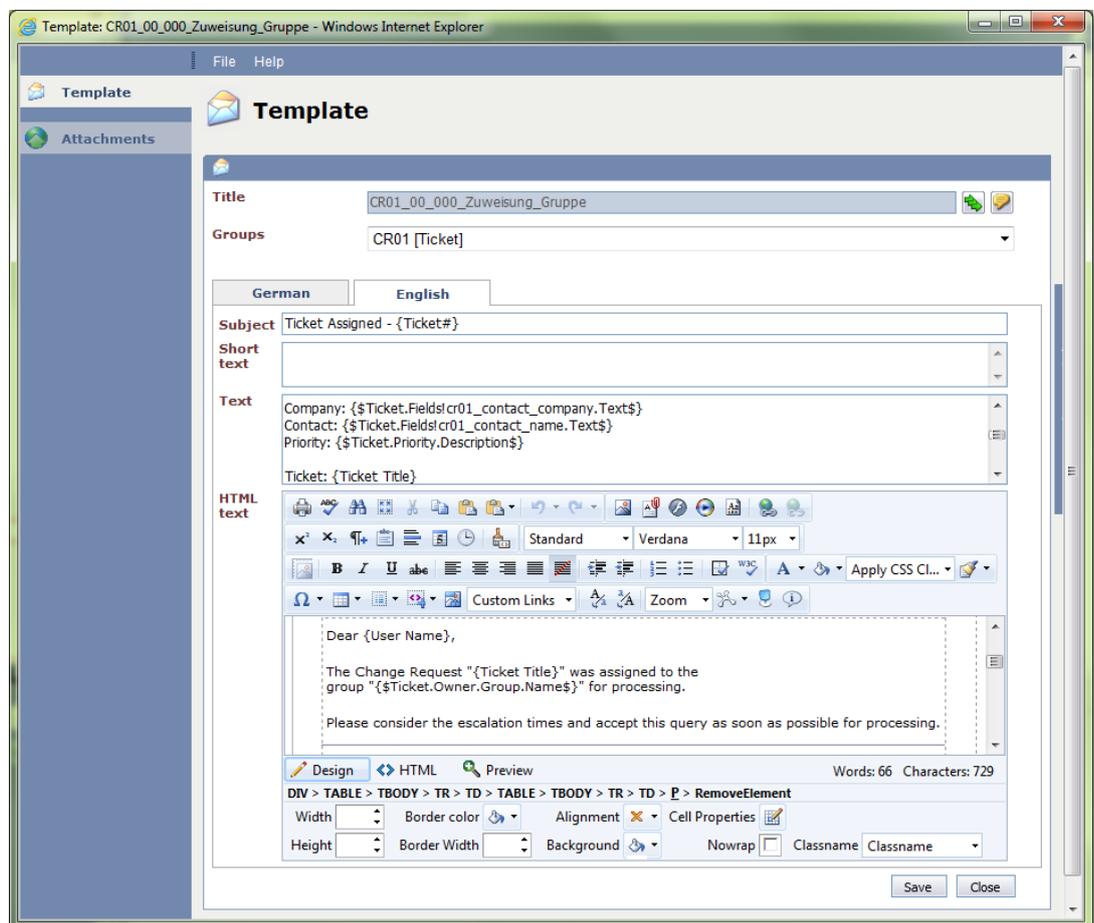
Figure 2-1

A listing of the available expressions can be found in Chapter 4.

3   Workflow Management

3.1   Using Expressions in Workflow Plug-Ins

The following example shows the use of an expression in the plug-in *Conditional Switch on Expression*. It has two exits and decides whether the ticket should be directed to the exit *Condition fulfilled* or not, depending on the underlying expression. When phrasing the expression, it is necessary to express a relation, which enables the plug-in to make a yes/no decision. In this very example, the condition is fulfilled if the ticket owns a field called *RequiresAnswer* and if it has the content *True*.
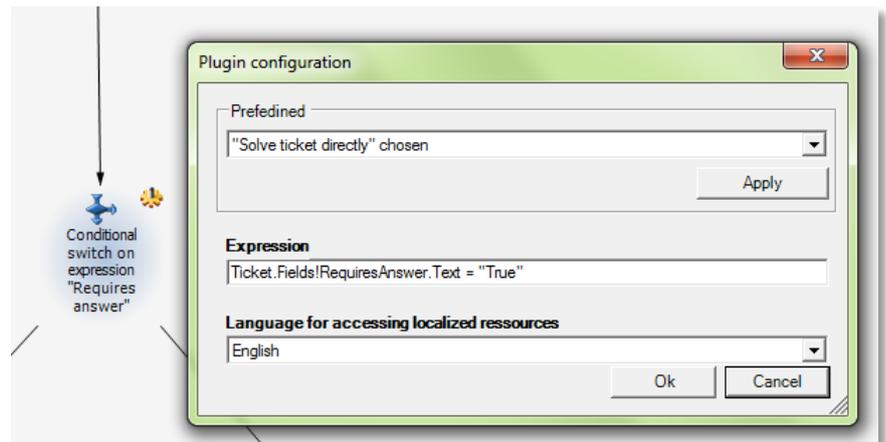
Figure 3-1

The next example illustrates another use of the *Conditional Switch on Expression* plug-in. In this case, it should decide whether a ticket is to be assigned to a user. The special key word *NULL* is used here. This condition is fulfilled if the ticket is assigned to no user.
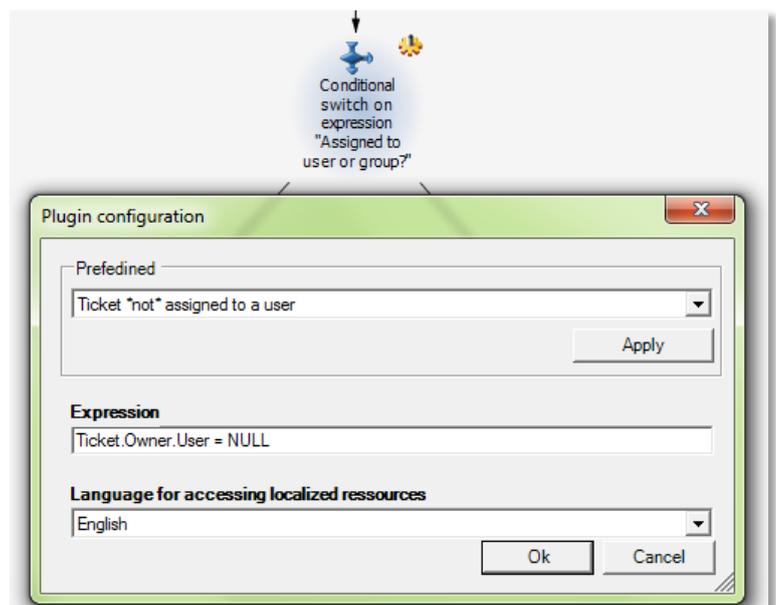
Figure 3-2

When defining workflow activities, expressions can be used for validation. If the validation fails (i.e. the condition phrased in the expression is not fulfilled), the workflow activity will not be processed. In the following example, pressing the *OK* button is only accepted, if the ticket contains a solution.
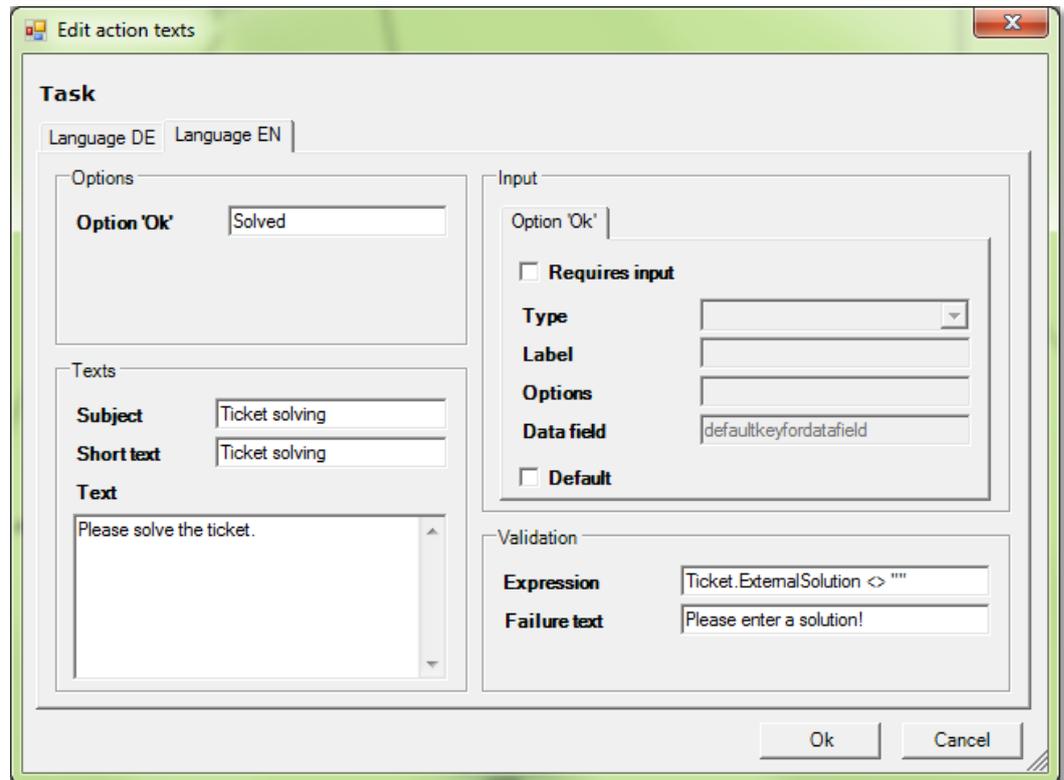
Figure 3-3

3.2 Using the Special Data Fields

Special data fields can be used in the workflow, in order to implement the solution of the last user/group action into the following action, for example. These data fields have to be distinct and there should not be multiple data fields of the same name, as the content of a field could be be overwritten by another field otherwise.

The data fields are stated during the configuration without any additional characters, as seen in the first figure. If the content of the field is to be displayed, the name of the data field has to be set in double curly braces:

```
{{DataFieldName}}
```

An action of approval is used as an example. For this purpose, two user actions are configured consecutively in the workflow. The first action is to be processed by an approving authority, which is to approve a request.

If the approval is not given, the user has to enter a reason. This reason is displayed as a selection box with three alternatives.

In the following figure, the configuration of this action is displayed:

Figure 3-4 Configuration of the ticket action for selecting a reason

After performing this action, the ticket would return from the approving authority to the requester awaiting the result of the approval in order to be able to make a decision concerning the further process. In order not to have to browse several different tabs needlessly to find the solution, the second action is configured in a way to make the solution visible directly in the user task.

Figure 3-5 Configuration of the output

The fields framed in red demonstrate the configuration of the data field. The data field *ApprovalReason* is configured to save the user's selection and to display it later on (or use it in another way).

In the following figure, the data field has been configured for display in the activities list. Thus, the user can see the result of the last action immediately – without having to open the ticket first.

Figure 3-6 Display of the selection of the last action in the activities list

It is furthermore possible to configure the entry of the reason for approval as a free text field, in which the user can enter the reasons.

Displaying these entries is then possible using the same measures as described in the example of the selection box before.
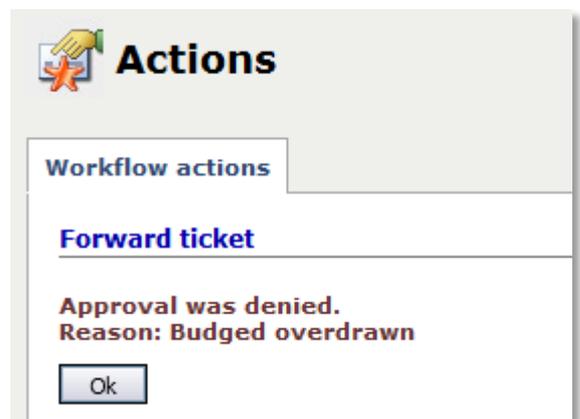
Figure 3-7 Display of a data field, which has been filled via a free text field

4 Expressions

The following expressions are available in Xpert.NET:

4.1 General Expressions

EXPRESSION	DESCRIPTION
General.<Attribute>	Returns general information. The following attributes are available:
CurrentDate	Current date (Format: YYYY-MM-dd)
CurrentTime	Current time (Format: hh:mm)
CurrentDate['<format>']	Current date (formatted); formatting options can be found on the following web page: http://msdn.microsoft.com/de-de/library/8kb3ddd4.aspx
CurrentDate.UTC['<format>']	Current date in UTC (formatted)
CurrentTime['<format>']	Current time (formatted)
CurrentTime.UTC['<format>']	Current time in UTC (formatted)
helpdesklink	URL to *TicketXpert /WorkflowXpert*
CurrentUser	Access to the current user
CurrentUser.<UserField>	See expressions for the affected user of a ticket.
NewLine	Inserts a line break.

<div align="right">

Table 4-1

</div>

4.2 General Expressions for Users, Groups and Process Roles

The following expressions can be used for user, group, and role queries. However, they cannot be used on their own. Whenever user, group, or role information can be accessed via an expression, a reference to these general expressions will be made at the respective location.

EXPRESSION	DESCRIPTION
	Expressions for users
<Attribute>	Returns information on a user's attributes. Further attributes for users can be added as well; they can be queried by using the attribute's name. *Note*: In order to add further attributes, a tool is necessary. For this purpose, please contact your project leader or the support. If no additional attributes have been defined for the user, the following attributes can be used:
Email	Returns the user's e-mail address.
FirstName	Returns the user's first name.
LastName	Returns the user's last name.
LoginName	Returns the user's login name.
FullName	Returns the user's full name.
MainGroup	Returns the user's main group.
ObjectGuid	Returns the user's ObjectGUID.
MainGroup.<GroupField>	Returns information on a group field of the user's main group. For *<GroupField>,* the same expressions as for groups can be used.
TextBuildingBlock!<Name>.Plaintext	Returns a group-dependent text building block in plain text format (replace *<Name>* with the name of the text building block).

TextBuildingBlock!<Name>.HTMLText	Returns a group-dependent text building block in HTML format (replace *<Name>* with the name of the text building block).
TextBuildingBlock!<Name>.Plaintext	Returns a user-dependent text building block in plain text format (replace *<Name>* with the name of the text building block).
TextBuildingBlock!<Name>.HTMLText	Returns a user-dependent text building block in HTML format (replace *<Name>* with the name of the text building block).
Expressions for groups	
<Attribute>	Returns the group's attributes. Further attributes for groups can be added as well; they can be queried by using the attribute's name. *Note*: In order to add further attributes, a tool is necessary. For this purpose, please contact your project leader or the support.
Description	Returns the group's description.
Name	Returns the group's name.
Email	Returns the group's email address.
MainGroup	Returns the group's main group.
TextBuildingBlock!<Name>.Plaintext	Returns a group-dependent text building block in plain text format (replace *<Name>* with the name of the text building block).
TextBuildingBlock!<Name>.HTMLText	Returns a group-dependent text building block in HTML format (replace *<Name>* with the name of the text building block).
Expressions for process roles	
ProcessRole.<Attribute>	Returns information on process roles in connection with the attribute.
DataId	Returns the unique database ID of the process role.
Description	Returns the description of the process role.
FriendlyName	Returns the language-independent name.
IsDeleted	Returns *true* if the process role has been deleted; otherwise *false*.
Name	Returns the language-dependent name.
ObjectGuid	Returns the unique database ID of the process role as a GUID.
ProcessSchemaDataId	Returns the unique database ID of the process schema this process role is assigned to.

<div align="right">

Table 4-2

</div>

4.3   Static Expressions

In *Xpert.NET*, it is possible to access static objects like previously created tickets, tasks, KB articles, expenses, and CIs via expressions.

AUSDRUCK	DESCRIPTION
Ci["Configuration Item"].<CIExpression>	Returns information on a certain configuration item in connection with the CI expression. *Example*: Ci["Main Server"].Ci.ObjectGuid
Expense["Expense"].<ExpenseExpression>	Returns information on a certain expense in connection with the expense expression. *Example*: Expense["23840"].Expense.AllocationType.Title

EXPRESSION	DESCRIPTION
KB["KB-Artikel"].<KBExpression>	Returns information on a certain KB article in connection with the KB expression. *Examples*: Kb["KB-0751"].Article.CreatedDate Individual versions of an article can be accessed as well. *Example*: Kb["KB-0751.v2"].Article.CreatedDate Individual revisions of a KB article can be accessed as well. *Example*: Kb["KB-0751.03"].Article.CreatedDate
Task["Task"].<TaskExpression>	Returns information on a certain task in connection with the task expression. *Example*: Task["Tsk-0004"].Start
Ticket["Ticket"].<TicketExpression>	Returns information on a certain ticket in connection with the ticket expression. *Example*: Ticket["IR_02251"].DirectUrl.Activities

Table 4-3

4.4   Expressions for Configuration Items

EXPRESSION	DESCRIPTION
Ci.Affected.<UserExpression>	Returns information on the affected user of a CI via a user expression.
Ci.Creator.<UserExpression>	Returns information on the creator of the CI via a user expression.
Ci.DataId	Internal ID of the CI (primary key)
Ci.Fields!<Field name>.<Attribute>	Returns information on CI fields under specification of an additional attribute. The following attributes are available:
DataId	Internal field ID (primary key)
ObjectGuid	Unique field ID for all databases
Text	Text of the specified field
Value	Value of the specified field
Ci.Fields.Count	Number of CI fields
Ci.NumberOfChilds	Number of contained CIs
Ci.ObjectGuid	Unique ID of a CI for all databases
Ci.Owner.<UserExpression>	Returns information on the owner of a CI via a user expression.
Ci.Supplier.<UserExpression>	Returns information on the supplier of a CI via a user expression.
Ci.ParentCi.<Attribute>	Returns information on parent CIs in connection with the attribute. If there are more parents than one, the first one is used.
Schema	Returns the schema of the parent CI.
State	Returns the status of the parent CI.
Ci.Relations.Count	Number of relations of the CI
Ci.Schema.<Attribute>	Returns information on a CI schema in connection with an attribute.
DataId	Internal ID of the CI key (primary key)
FriendlyName	Name of the schema (language-independent)
IsVirtual	Denotes whether the schema is a virtual schema (*true* or *false*).
NumberOfChilds	Denotes the number of subschemas.
ObjectGuid	Unique ID of the schema in all databases.
Ci.Status.<Attribute>	Returns information on the status of a CI in connection with an attribute.
DataId	Internal ID of the status (primary key)
FriendlyName	Name of the CI status (language-independent)
ObjectGuid	Unique ID of the status in the entire database

Ci.UniqueId	Unique ID of the CI visible in the interface
Ci.VisibleId	Visible ID of the CI
Ci.Zone.<Attribute>	Returns information on CMDB zones in connection with an attribute.
DataId	Returns the internal ID of the zone (primary key).
Fields!<FieldName>.DataId	Returns the internal ID of the specified field (primary key). (*<Field name>* denotes the respective field of a CI zone.)
Fields!<FieldName>.ObjectGuid	Returns the unique ID of the zone field in all databases. (*<Field name>* denotes the respective field of a CI zone.)
Fields!<FieldName>.Text	Returns the text of the specified field. (*<Field name>* denotes the respective field of a CI zone.)
Fields!<FieldName>.Value	Returns the value of the specified field. (*<Field name>* denotes the respective field of a CI zone.)
FriendlyName	Returns the name of the CI zone (language-independent).
NumberOfChilds	Returns the number of direct subzones.
ObjectGuid	Returns the unique ID of the zone in all databases.
ParentZone.<ZoneExpression>	Returns the "parent zone". For *<ZoneExpression>*, all expressions of *Ci.Zone* can be used.
Schema.DataId	Returns the internal ID of the zone schema (primary key).
Schema.FriendlyName	Returns the name of the zone schema (language-independent).
Schema.ObjectGuid	Returns the unique ID of the zone schema in all databases.
ZoneType	Returns the zone type (either *TopLevelZone* or *CiZone*)

Table 4-4

4.5 Expressions for Dynamic Placeholders

EXPRESSION	DESCRIPTION
Placeholder.<Name>	Returns the text of the dynamic placeholder (replace *<Name>* with the placeholder's name).
Placeholder.<Name>.HTML	Returns the text of the dynamic placeholder (replace *<Name>* with the placeholder's name). Line breaks are replaced with * *.

Table 4-5

4.6 Expressions for the Early Warning System

EXPRESSION	DESCRIPTION
EarlyWarnAlert.<Attribute>	Returns information on early warning rules in connection with the attribute.
Rule!Name	Returns the name of the early warning rule.
Subscription!Name	Returns the name of the early warning subscription.
Tickets!Url	Returns the URLs of the tickets affected by this rule as plain text.
Tickets!Url.Html	Returns the URLs of the tickets affected by this rule as HTML.
Tickets[<Index>]	Returns the respective ticket (*<Index>* starts with 0).
Title	Returns the title of the respective ticket.
Ticket#	Returns the ticket number of the respective ticket.
CreatedDateTime	Returns the date and time of creation of the respective ticket.
Status.Name	Returns the name of the status of the respective ticket.
Priority.Description	Returns the description of the priority of the respective ticket.

EXPRESSION	DESCRIPTION
Fields!<Feldname>.Text	Returns the text of the specified field of the respective ticket.
TriggerTicket	Returns the ticket that has triggered the early warning system.
TriggerTicket.Ticket#	Returns the number of the ticket that has triggered the early warning system.

4.7 Expressions for Expenses

EXPRESSION	DESCRIPTION
Expense.<Attribute>	Returns information about an expense in connection with an attribute.
DataId	Returns the unique ID of an expense in the database.
ExecutionDate	Returns the execution time of an expense in dd.mm.yyyy hh:mm:ss format.
Expense.User.<UserExpression>	Returns information on the expense creator in connection with the user expression.
Expense.Ticket.<TicketExpression>	Returns information on the ticket linked to the expense in connection with the ticket expression.
Expense.Task.<TaskExpression>	Returns information on the task linked to the expense in connection with the task expression.
Expense.External.<Attribute>	Returns information on external details of an expense in connection with the attribute.
Expense.Internal.<Attribute>	Returns information on internal details of an expense in connection with the attribute.
Title	Returns the (external or internal) title of the expense.
Description	Returns the (external or internal) description of the expense.
Total	Returns the (external or internal) total of the expense.
AllocationType.<Attribute>	Returns information on the allocation type in connection with the attribute.
ObjectGuid	Returns the ObjectGuid of the allocation type.
Title	Returns the name of the allocation type.
Description	Returns the description of the allocation type.
Charge	Returns the charges of the allocation type.
Factor	Returns the factor of the allocation type.
Expense.Project.<Attribute>	Returns information on the project of an expense in connection with the attribute.
ObjectGuid	Returns the ObjectGuid of the project.
Title	Returns the name of the project.
Start	Returns the start date of the project.
End	Returns the end date of the project.
Expense.CostCenter.<Attribute>	Returns information on the cost center used in the expense in connection with the attribute.
ObjectGuid	Returns the ObjectGuid of the cost center.
Number	Returns the number of the cost center assigned in the project.
Title	Returns the title of the cost center.
Start	Returns the start date of the cost center.
End	Returns the end date of the cost center.
EnableTimeAccount	Returns true if the time account is activated for cost center, otherwise false.

EXPRESSION	DESCRIPTION
Expense.Type.<Attribute>	Returns *true* if the time account has been activated for the cost center, otherwise *false*.
Title	Returns the name of the expense type.
DefaultAllocationType.<AllocationTypeExpression>	Returns information on the default allocation type for the used expense type in connection with the *AllocationTypeExpression*.

<div align="right">

Table 4-7

</div>

4.8   Expressions for the Knowledge Base

EXPRESSION	DESCRIPTION
	Language-independent
KB.Article.<Attribute>	Returns information on a KB article in connection with the attribute.
ID	Returns the ID of the KB article.
CreatedDate	Returns the creation date of the article.
ChangedDate	Returns the date of change of the article.
RevisedDate	Returns the revision date of the article.
Creator.<UserExpression>	Returns the full name of the creator of the article. Additionally, all user expressions can be used.
Modifier.<UserExpression>	Returns the full name of the user who has modified the article last. Additionally, all user expressions can be used.
Revisor.<UserExpression>	Returns the full name of the user who revised the article.
KnowledgeOwner.<UserExpression>	Returns the knowledge owner of the article. Additionally, all user expressions can be used.
KnowledgeOwnerGroup.<GroupExpression>	Returns the group the knowledge owner is a member of. Additionally, all group expressions can be used.
Owner	Returns the owner (knowledge owner) of a KB article as an object
RatingCount	Returns the number of ratings for this KB article.
RatingAverage	Returns the average rating of the KB article.
VisitsCount	Returns the number of persons who already viewed the article.
ValidStartDate	Returns the start date of the period of validity.
ValidEndDate	Returns the end date of the period of validity.
Faq	Returns *true/false*, depending on whether the article has been marked as a FAQ article.
Inherit	Returns whether the article is visible in sub-categories as well.
ReviewDate	Returns the date of resubmission of the article.
Url	Returns the direct URL of the article.
KB.Article.Categories.ObjectGuids	Returns a comma-separated list of all ObjectGuids of KB categories an article is in.
KB.Stage.Url	Returns the direct URL to the knowledge base.
	Language-dependent
KB.Article.<Attribute>	Returns language-dependent information on a KB article in connection with the attributes below.

	Returns the title of the article.
Details.Title!<Language>	(<*Language*> = the language code of the desired language (EN = English) or *All* for all published languages) *Examples*: - KB.Article.Details.Title!EN - KB.Article.Details.Title!EN,FR - KB.Article.Details.Title!All
Details.Keywords!< Language >	Returns keywords of the article. (<Language>: See *Title*)
Details.ContentHTML!< Language>	Returns the content of the article. (<Language>: See *Title*)
Details.Boilerplate!< Language>	Returns the text boilerplate of the article. (<Language>: See *Title*)
KB.Article.Categories.Names	Returns all localized names of KB categories containing the article. The categories are separated as follows: " | " (space pipe space).

<div align="right">**Table 4-8**</div>

4.9   Expressions for Messages

EXPRESSION	DESCRIPTION
Message.Attachments.<Attribute>	Returns information on file attachments of messages in connection with the attribute.
Count	Returns the number of file attachments.
Names	Returns a comma-separated list of the names of the file attachments.
Size	Returns the total size of file attachments in Byte.
Message.Body.Plain (*)	Returns the message text of a previous message in plaintext format.
Message.Body.Html (*)	Returns the message text of a previous message in HTML format.
Message.CreatedDateTime	Returns the localized date (and time), on which the e-mail has been processed by the system.
Message.Priority.Name	Returns the priority the initial message has been sent with.
Message.SendDateTime	Returns the localized date (and time) the message initially has been sent at.
Message.Sender	Returns the sender of the initial message.
Message.Subject (*)	Returns the subject of the previous message.

Tabelle 4-9 Note: (*) – These expressions are only available for the ticket actions *Response e-mail* and *Forward e-mail*.

4.10   Expressions for Tasks

EXPRESSION	DESCRIPTION
Task.Changed	Returns the date and time of the last change.
Task.ClientObjectGuid	Returns the unique ID of the client the task has been created in.
Task.Closed	Returns the date and time of closing the task (NULL, if it has not been closed yet).
Task.Comments.<Attribute>	Returns information on task comments in connection with the attribute. Date/time information on comments is specified in the local time.
All!Text	Returns all comments of the task including user and date/time.
Count	Returns the total number of comments of the task.
Last!CreatedDateTime	Returns the creation time of the last comment.

Last!Text	Returns the text of the last comment.
Last!User.<UserExpression>	Returns the user who has created the last comment via a user expression.
Task.CommentsHtml.All!Text	Returns all comments in HTML format including user and date/time. Date/time information on comments is specified in the local time.
Task.CommentsHtml.Last!Text	Returns the text of the last comment in HTML format (without user and date/time).
Task.CommentsPlainText.All!Text	Returns all comments in plain text format including user and date/time. Date/time information on comments is specified in the local time.
Task.CommentsPlainText.Last!Text	Returns the text of the last comment in plain text format (without user and date/time).
Task.Created	Returns the date and time of the task's creation.
Task.Creator.<UserExpression>	Returns the creator of the task via a user expression.
Task.DataId	Returns the unique ID of the task in the database.
Task.Description	Returns the description of the task.
Task.DueDate	Returns the end date of the task.
Task.Executor.<UserExpression>	Returns information on the current task executor via a user expression.
Task.ExecutorGroup.<GroupExpression>	Returns information on the current task executor group via a user expression (NULL, if the task has been assigned to a user).
Task.ExecutorRole.<RoleExpression>	Returns information on the current task executor process role via a role expression (only set, if the task has been created within a process schema).
Task.FullId	Returns the ID of the task as displayed on the interface (e.g. TSK 5).
Task.HasNonClosedSubTasks	Returns *true,* if there are no open subtasks; otherwise *false.*
Task.HasNonClosedSuccesorTickets	Returns *true,* if there are no open successor tickets for the task; otherwise *false.*
Task.HasSubTasks	Returns *true,* if the task contains further subtasks; otherwise *false.*
Task.HasSuccessorTickets	Returns *true,* if the task contains further successor tickets; otherwise *false.*
Task.LverDefaultCostCenterObjectGuid	Returns the internal ID (database) of the default cost center of the task (can be NULL).
Task.LverDefaultProjectObjectGuid	Returns the internal ID (database) of the default project of the task (can be NULL).
Task.LverDefaultsAreReadOnly	Returns *true,* if the Lver default values are read-only; otherwise *false.*
Task.ObjectGuid	Returns the internal ID (database) of the task (GUID).
Task.OlaTimeSpan	Returns the timespan of the task's OLA (can be NULL).
Task.ParentTask.<TaskExpression>	Returns the parent task, if there is one (for *<TaskExpression>,* all of the subexpressions in this list can be used, e.g. *Task.ParentTask.Description*).
Task.ParentTicket.<TicketExpression>	Returns the parent ticket of the task and solves the ticket expressions accordingly (for *<TicketExpression>*, see 4.14 *Expressions for Tickets*) *Examples*: - Task.ParentTicket.Status.FriendlyName - Task.ParentTicket.Fields!*<FieldName>*.Text
Task.Start	Returns the starting time of the task
Task.TaskAux.<Attribute>	Returns information on the elements linked to the task in connection with the following attributes.

AttachmentsCount	Returns the number of the task's file attachments.
CommentsCount	Returns the number of the task's comments.
ExpenseCount	Returns the number of the task's booked expenses.
SubTasksCount	Returns the number of the subtasks.
TicketsCount	Returns the number of the task's successor tickets.
Task.TaskFolder.<Attribute>	Returns information on task folders in connection with the attribute.
DataId	Returns the unique ID (database) of the task folder.
Description	Returns the description of the task folder.
ObjectGuid	Returns the unique ID (database) of the task folder (GUID).
ParentTaskFolder.<TaskFolderexpression>	Returns the parent task folder (for *<TaskFolderexpression>,* all of the listed TaskFolder subexpressions can be used, e.g. *Task.TaskFolder.ParentTaskFolder.Description*).
Title	Returns the name of the task folder.
Task.TaskLifeCycle.<LifecycleExpression>	Returns the used life cycle (for *<LifeCycleExpression>* see 4.11 *Expressions for Task Life Cycles*).
Task.TaskSchema.<SchemaExpression>	Returns the used schema of a task (for *<SchemaExpression>*, see 4.12 *Expressions for Task Schemas*).
Task.TaskStatus.<StatusExpression>	Returns the current status of the task (for *<StatusExpression>*, see 4.13 *Expressions for Task Statuses*).
Task.Title	Returns the title of a task.

<div align="right">**Table 4-10**</div>

4.11 Expressions for Task Life Cycles

EXPRESSION	DESCRIPTION
TaskLifeCycle.<Attribute>	Returns information on task life cycles in connection with the attribute.
DataId	Returns the unique ID (database) of the life cycle used by the task.
Description	Returns the description of the life cycle.
FriendlyName	Returns the language-independent name of the life cycle.
StartTaskStatus.<StatusExpression>	Returns the configured start status of the life cycle (for *<StatusExpression>*, see 4.13 *Expressions for Task Statuses*).
StartWorkflow	Returns the GUID (ID in the database) of the workflow configured as a start workflow for the life cycle.

<div align="right">**Table 4-11**</div>

4.12 Expressions for Task Schemas

EXPRESSION	DESCRIPTION
TaskSchema.<Attribute>	Returns information on task schemas in connection with the attribute.
DataId	Returns the unique ID (database) of the task schema.
Description	Returns the description of the task schema.
FriendlyName	Returns the language-independent name of the schema.
LifeCycle.<LifecycleExpression>	Returns the life cycle used in the task schema (for *<LifeCycleExpression>* see 4.11 *Expressions for Task Life Cycles*).
Name	Returns the language-dependent name of the task schema.
ObjectGuid	Returns the unique ID (database) of the task schema (GUID).

<div align="right">**Table 4-12**</div>

4.13   Expressions for Task Statuses

EXPRESSION	DESCRIPTION
TaskStatus.<Attribute>	Returns information on task statuses in connection with the attribute.
DataId	Returns the unique ID (database) of the status.
FriendlyName	Returns the language-independent name of the status.
StatusTypeDataId	Returns the ID of the status type.
Visible	Returns *true*, if the status has been marked as visible; otherwise *false.*

Table 4-13

4.14 Expressions for Tickets

EXPRESSION	DESCRIPTION
Ticket.AffectedGroup.<GroupExpression>	Returns information on the affected group of a ticket via a group expression.
Ticket.AffectedUser.<UserExpression>	Returns information on the affected user of a ticket via a user expression.
Ticket.Attachments.<Attribute>	Returns information on file attachments in tickets.
Names.HTML	Returns a list of the attachment names in HTML format.
Names.PlainText	Returns a list of the attachment names in plain text.
Count	Returns the number of the attachments of the ticket.
Ticket.CIs.<Attribute>	Returns information on CIs.
Count	Returns the number of the CIs linked to the ticket.
[<Index>].<CiExpression>	Access to the linked CI in the specified index. Value of the respective CI expression (replace *<CiExpression>* with one of the CIs' expressions, e.g. *Ticket.Cis[0].Affected.FirstName*).
Ticket.Clones.Count	Returns the number of the ticket's clones.
Ticket.Clones[#]["<Schema>"].<Attribute>	Returns information on the tickets cloned from the origin ticket in connection with the attribute. The specification [#] enables access to individual locations of an array, if such an array is returned; [0] for the first entry. A ticket schema's friendly name can be entered instead of *<Schema>* in order to return only successor tickets based on it. Every ticket expression is permitted as an attribute as well.
ObjectGuid	Returns the ObjectGuid of a parent ticket. In doing so, an array is returned.
Ticket#	Returns the ticket number of the parent ticket.
Ticket.Comments.<Attribute>	Returns information on ticket comments in connection with the attributes below. Date and time information of the comments is specified in the local time.
All!Text	Returns all comments formatted as saved.
Count	Returns the number of comments to the ticket.
Last!CreatedDateTime	Returns the creation date of the last comment.
Last!Text	Returns the last comment formatted as saved.
Last!User.<UserExpression>	Returns information on the last user that has entered a comment. User expression can be used for retrieving information on the user.
Ticket.CommentsHtml.All!Text	Returns all comments formatted as HTML (line breaks are replaced with * *). Date and time information of the comments is specified in the local time.
Ticket.CommentsHtml.Last!Text	Returns the last comment formatted as HTML (line breaks are replaced with *).*
Ticket.CommentsPlainText.All!Text	Returns all comments formatted as plain text.
Ticket.CommentsPlainText.Last!Text	Returns the last comment formatted as plain text.
Ticket.CompletionDateTime	Returns the completion date of the ticket.
Ticket.Correspondence.Count	Returns the number of correspondences linked to this ticket.
Ticket.CreatedDateTime.<Attribute>	Returns information on the creation date of the ticket in connection with the attributes below.
['<format>']	Returns the creation time (formatted).

UTC['<format>']	Returns the creation time in UTC (formatted), e.g. ['yyyy/MM/dd HH:mm:ss']. Additionally, all date formats supported by the .NET framework are supported here as well, e.g.: ['d']: 6/15/2008 ['D']: Sunday, June 15, 2008 ['f']: Sunday, June 15, 2008 9:15 PM ['F']: Sunday, June 15, 2008 9:15:07 PM ['g']: 6/15/2008 9:15 PM ['G']: 6/15/2008 9:15:07 PM ['m']: June 15 ['o']: 2008-06-15T21:15:07.0000000 ['R']: Sun, 15 Jun 2008 21:15:07 GMT ['s']: 2008-06-15T21:15:07 ['t']: 9:15 PM ['T']: 9:15:07 PM ['u']: 2008-06-15 21:15:07Z ['U']: Monday, June 16, 2008 4:15:07 AM ['y']: June, 2008 Some of these expressions depend on the regional settings. If they are used without relation to the user (e.g. in the workflow), the server settings are relevant.
DateDDMMYYYY	Returns the creation time in the format DD.MM.YYYY (e.g. 25.11.2011).
Ticket.CreationMethod	Creation method of a ticket. Possible values: • Mail2Ticket • Manual • ContainerTicketWizard
Mail2TicketAccount.Name	Returns the name of the Mail2Ticket account the ticket has been created with. If the ticket has not been created via Mail2Ticket, the expression returns "".
Ticket.Creator.<UserExpression>	Returns information on the ticket creator by using a user expression.
Ticket.CustomAttributes!IsDirectSolve. Value	Possible return values: *true* and *false.* If *true* is returned, the ticket wizard has been closed by clicking the button *Solve ticket directly.*
Ticket.DefaultCostCenter.<Attribute>	Returns information on the default cost center specified in a ticket in connection with the attribute.
Title	Returns the name of a cost center.
Number	Returns the number of a cost center.
Ticket.DataId	Returns the ticket ID from the database (unique ID).
Ticket.DirectUrl.Activities	This expression returns a link as soon as there are pending workflow activities for a ticket. This link will then directly open the tab with the workflow actions of a ticket, so they can be performed.
Ticket.Escalations.<Attribute>	Returns information on escalations in connection with the attribute. In contrast to *Ticket.PendingEscalations*, information on already achieved or escalated escalations is returned as well.
Count	Returns the number of escalations.
Text	Returns information on the current escalation.
!<EscalationName>.Deadline	Returns the date and the time (localized) of the specified escalation.
Ticket.EstimatedTimeExpense	Returns the estimated time expense for a ticket (if set, otherwise NULL).
Ticket.Expenses.Count	Returns the number of expenses booked in a ticket.
Ticket.ExternalSolution.<Attribute>	Returns the external solution of the ticket in HTML format.
PlainText	Returns the external solution of the ticket as plain text.

PlainText.Length	Returns the number of characters contained in the external solution.
Ticket.Fields!<Name>.<Attribute>	Returns information on ticket fields using the attributes below. (*<Name>* has to be replaced by the name of the ticket field; ticket fields containing umlauts, spaces, or hyphens are not supported here).
Text	Returns the text value of the respective ticket field formatted as saved.
Ci.*<CiExpression>*	Returns the value of the CI expression (replace *<CiExpression>* with an expression for the CIs, e.g. *Ticket.Fields!CiList1. Ci.Affected.FirstName*). This expression is only valid, if the ticket field has been configured as a CI list in the Ticket Wizard.
TextHTML	Text value of the respective ticket field formatted as HTML (line breaks are replaced with * ;* replace *<Name>* with the name of the ticket field)
TextPlain	Returns the text value of the respective ticket field formatted as plain text (replace *<Name>* with the name of the ticket field).
Value	Returns the value of the respective ticket field. This value is ONLY set for selection fields and for *Multiple choice fields.*
ValuePlain	Returns the value of the respective ticket field as plain text. This is valid for the *Multiple choice field* as well.
ZonedDateTimeDDMMYY ZonedDateTimeDDMMYYHHMM ZonedDateTimeDDMMYYHHMMSS ZonedDateTimeDDMMYYYY ZonedDateTimeDDMMYYYYHHMM ZonedDateTimeDDMMYYYYHHMMSS	Returns ticket fields with date content in user-defined format. The requirement for using these expressions is the ticket field having the type *DateTime*. Via the expression *ZonedDateTime,* the date is converted from the universal to the correct time zone.
Ticket.Fields!<FieldName>.User. <UserExpression>	This expression allows for accessing user information via a ticket field, if the fields contains data like the ObjectGuid, UserBrowser path, login name, or an e-mail address. Via the user expression, this allows for accessing user information.
Ticket.Fields!<FieldName>. User.NewPassword	Via this expression, a new user password can be returned. This expression only returns a result, if the user has requested a new password.
Ticket.Fields!<FieldName>.Group. <GroupExpression>	This expression allows for accessing group information via a ticket field, if the fields contains data like the ObjectGuid, UserBrowser path, group name, or an e-mail address. Via the group expression, this allows for accessing group information.
Ticket.HasNonClosedSubTasks	Returns the value *true,* if the ticket contains further open subtasks; otherwise *false.*
Ticket.HasParent	Checks whether the current ticket has a parent ticket. *True* if yes, *false* if no.
Ticket.HasSubTasks	Returns the value *true,* if the ticket contains subtasks; otherwise *false.*
Ticket.IdNumber	Returns the ticket's number without the static prefix (e.g. 0009 instead of CR_0009)
Ticket.InternalSolution.<Attribute>	Returns the internal solution of the ticket formatted as HTML whenever no attribute is specified.
PlainText	Returns the internal solution of the ticket formatted as plain text.
Ticket.KB.Count	Returns the number of KB articles linked to this ticket.
Ticket.Linked[#]["<Schema>"]. <Attribute>	Returns information on linked tickets in connection with the attribute. The specification [#] enables access to individual locations of an array if such an array is returned; [0] for the first entry. A ticket schema's friendly name can be entered instead of *<Schema>* in order to return only successor tickets based on it. Every ticket expression is permitted as an attribute as well.
ObjectGuid	Returns the ObjectGuid of a parent ticket. In doing so, an array is returned.

Ticket#	Returns the ticket number of the parent ticket.
Ticket.Mails.Count	Returns the number of messages contained in this ticket.
Ticket.MessageTag	Returns the specific expression to identify the ticket distinctly.
Ticket.ObjectGuid	Returns the Guid of the ticket from the database (unique Guid of the ticket).
Ticket.Owner.Group.<GroupExpression>	Returns the owner group of the ticket. It is additionally possible to retrieve further information on the group via group expressions.
Ticket.OriginalMessage.<Attribute>	Returns information on the original message (the first incoming e-mail to a ticket), including ticket images, in connection with the attribute. All parts available for message expressions can be used here as well, e.g.: • Ticket.OriginalMessage.Subject • Ticket.OriginalMessage.Body.Html • Ticket.OriginalMessage.Body.Plain Warning: These expressions can only be used with tickets generated via Mail2Ticket. It is advisable to check in advance whether the ticket actually is such a ticket.
Ticket.Owner.User.<UserExpression>	Returns information on the owner (user) of a ticket. If this value equals NULL, the owner of the ticket is currently a group; if not, it is a user. Additionally, user expressions can be entered in order to retrieve further information on the owner.
Ticket.Parent.<Attribute>	Returns a possible parent ticket and allows access to it. Otherwise, NULL is returned. Additionally, the following attributes are available.
Clones.Count	Returns the number of clones of the parent ticket.
Ticket.Parents[#]["<Schema>"].<Attribute>	Returns information on parent ticket(s) in connection with the attribute. The specification [#] enables access to individual locations of an array, if such an array is returned; [0] for the first entry. A ticket schema's friendly name can be entered instead of *<Schema>* in order to return only successor tickets based on it. Every ticket expression is permitted as attribute as well.
ObjectGuid	Returns the ObjectGuid of a parent ticket. In doing so, an array is returned.
Ticket#	Returns the ticket number of the parent ticket.
Ticket.PendingEscalations.<Attribute>	Returns information on pending escalations in connection with the attribute.
Count	Returns the number of escalations.
Text	Returns information on the current escalation.
!<EscalationName>.Deadline	Returns the date and the time (localized) of the specified escalation.
Ticket.Priority.<Attribute>	Returns information on ticket priorities using the following attributes.
Description	Returns the description of the priority.
EscalationTimes.Count	Returns the number of escalation times for the current ticket.
EscalationTimes.Text	Returns the escalation times of the current ticket, including the escalation time name and the configured escalation time.
FriendlyName	Returns the name of the priority of the ticket (language-independent).
IsCritical	Denotes whether the priority of the ticket is marked as critical.
Name	Returns the name of the priority.
Ticket.Responsible.User.<UserExpression>	Returns information on the responsible user. General user expressions can be used here.
Ticket.Responsible.Group.<GroupExpression>	Returns information on the responsible group. General group expressions can be used here.
Ticket.Schema.FriendlyName	For one activated language: name of the ticket schema. For multiple activated languages: unique name of the ticket schema without localization.

Ticket.Service.<Attribute>	Returns information on the services used in the service portfolio in connection with an attribute. For these expressions, it is required to check whether a ticket has been created via the Service Portfolio.
Manager.<Attribute>	Returns information on the Service Manager group in connection with the attributes.
Manager.Group.<GroupExpression>	Returns the Service Manager group. All general group expressions can be used in order to retrieve further information.
HasServiceCi	Returns a Boolean value. *True* if the superordinate service of the service transaction the ticket has been created in owns a service CI, otherwise f*alse.*
ServiceCi.<Attribute>	Returns information on the service CI of the superordinate service of the service transaction the ticket has been created in. All expressions for CIs can be used as attributes.
Ticket.Schema.ProcessRoles! *<FriendlyName>*.**DefaultGroup.** **<GroupExpression>**	Returns the desired default process role from a process schema along with its properties. *<FriendlyName>* is to be replaced with the name of the process role. The *<GroupExpression>* can be replaced with the available attributes of the group.
Ticket.ServiceTransaction. **ProcessRoles!***<FriendlyName>*.**Group.** **<GroupExpression>**	Returns the desired process role along with its properties. *<FriendlyName>* is to be replaced with the name of the process role. The *<GroupExpression>* can be replaced with the available attributes of the group.
Ticket.StartDateTime	This expression returns the start time of a ticket.
Ticket.StateObjectGuid	Returns a unique ID (in all databases) of the current ticket status.
Ticket.Status.<Attribute>	Returns information on the current ticket status only in connection with the attributes listed below.
FriendlyName	Returns the friendly name of the current ticket status (language-independent).
Name	Returns the language-dependent name of the current ticket status
Ticket.Successors.<Attribute>	Returns information on successor tickets in connection with the attributes listed below.
Count	Returns the number of successor tickets
[<Index>]	Access to the successor ticket at the specified index (the index starts at 0). Example: Ticket.Successors1
[<Index>].ObjectGuid	Access to the ObjectGuid of the successor ticket at the specified index (the index starts with 0). Example: Ticket.Successors1.ObjectGuid
Ticket.Successors[#]["<Schema>"]. **<Attribute>**	Returns information on successor tickets in connection with the attribute. The specification [#] enables access to individual locations of an array, if such an array is returned; [0] for the first entry. A ticket schema's friendly name can be entered instead of *<Schema>* in order to return only successor tickets based on it. Every ticket expression is permitted as an attribute as well.
ObjectGuid	Returns the ObjectGuid of a parent ticket. In doing so, an array is returned.
Ticket#	Returns the ticket number of the parent ticket.
Ticket.Ticket#	Returns the ticket number.
Ticket.Title	Returns the title field of the ticket.
Ticket.Url.<Attribute>	Returns the direct URL to the ticket. Additionally, an attribute can be entered here.

Details Fields UploadFiles Actions History Comments Solution Expenses Overview Todos	Returns the direct URL to the ticket, focusing on the specified tab.
Ticket.UrlQueryString	Returns the query string for the URL access of the respective ticket (i.e. everything after *?* without the actual address to the site).

<div align="right">**Table 4-14**</div>

4.15   Expressions for Workflow Items

The following expressions are available for workflow items. However, they can only be used in workflows. These expressions are not case sensitive.

EXPRESSION	DESCRIPTION
WorkflowItem.AffectedGroup.<GroupExpression>	Returns information on the affected group. All general group expressions can be used here.
WorkflowItem.AffectedGroupObjectGuid	Returns the ObjectGuid of the affected group of the workflow item.
WorkflowItem.AffectedUser.<UserExpressions>	Returns information on the affected user. All general user expressions can be used here.
WorkflowItem.AffectedUserObjectGuid	Returns the ObjectGuid of the affected user of the workflow item.
WorkflowItem.ClientObjectGuid	Returns the unique ID of the client this workflow item has been created in.
WorkflowItem.ConnectorObjectGuid	Returns the ObjectGuid of the connector.
WorkflowItem.CreatorUser.<UserExpression>	Returns information on the creator. All general user expressions can be used here.
WorkflowItem.CreatorUserObjectGuid	Returns the ObjectGuid of the creator of the workflow item.
WorkflowItem.DataId	Returns the data ID of the workflow item.
WorkflowItem.IdDescriptive	Returns the descriptive ID of a workflow item. For a ticket: CR_00367, for example.
WorkflowItem.IdUniqueForType	Returns the unique ID for the type of the workflow item.
WorkflowItem.IsCompleted	Returns whether a workflow item has been completed (Boolean).
WorkflowItem.IsErroneous	Returns whether the item has an error (Boolean).
WorkflowItem.IsSuspended	Returns whether the item has been suspended (Boolean).
WorkflowItem.IsWaitingForJoin	Returns whether the item is waiting for a join (Boolean).
WorkflowItem.LastActionInput	Returns the input of the last action.
WorkflowItem.LastActionResult	Returns the result of the last action.
WorkflowItem.LastException	Returns the last error of the workflow item.
WorkflowItem.ObjectType	Returns the ObjectType of the workflow item (ticket, KB article, etc.)
WorkflowItem.Owner.Group.<Attribute>	Returns information on the owner group. All expressions for the owner group can be used as an attribute here.
WorkflowItem.OwnerGroupObjectGuid	Returns the ObjectGuid of the owner group of the workflow item.
WorkflowItem.Owner.User.<Attribute>	Returns information on the owner. All expressions for the owner can be used as an attribute here.

WorkflowItem.OwnerUserObjectGuid	Returns the ObjectGuid of the owner of the workflow item.
WorkflowItem.PluginObjectGuid	Returns the ObjectGuid of the plug-in the workflow item is currently in.
WorkflowItem.PreviousConnectorObjectGuid	Returns the ObjectGuid of the previous connector.
WorkflowItem.PreviousPluginObjectGuid	Returns the ObjectGuid of the plug-in the workflow item has been in previously.
WorkflowItem.PreviousWorkflowObjectGuid	Returns the ObjectGuid of the workflow the workflow item has been in previously.
WorkflowItem.ProcessSchemaDataId	Returns the data ID of the process schema the workflow item has been created with.
WorkflowItem.SuspendedUntil	Returns the date, until which a workflow item has been suspended.
WorkflowItem.WorkflowObjectGuid	Returns the ObjectGuid of the current workflow of the workflow item.

Table 4-15

5 Examples

5.1 Delimitation of Date/Time

In order to query whether a ticket has been created between 07:00 and 18:00, for example, the following expression can be used:

```
Ticket.CreatedDateTime['HH.MM'] > "07.00"&&

Ticket.CreatedDateTime['HH.MM'] <"18.00"
```

This expression queries, whether the ticket has been created *after* 07:00 and *before* 18:00. Thus, for example, a switch in the workflow can be controlled.

5.2 Solve Ticket Directly

The workflow provides an option to query whether the Ticket Wizard has been completed via the button *Solve directly*:

```
Ticket.CustomAttributes!IsDirectSolve.Value = "true"
```

The expression above checks whether the Ticket Wizard has been completed with the button *Solve directly*. In this case, the whole expression returns *true*.

6 Operators

6.1 Introduction

In conjunction with expressions, operators can be used in order to perform comparisons, apply mathematical operations or connect strings.

An operator is a sign symbolizing a certain function, mainly in the form of an operation. Among these operators are e.g. basic arithmetic operations, relations, and logical operators.

Examples:

- < is less than

- <> does not equal

- + add/combine

On both sides of the operator, the so-called operands can be found. There is one on the left side of the operator (*left hand*) and one on the right side (*right hand*).

In order to process operator functions, it is necessary that both sides have the same data type in most cases. *5 > Hallo*, for example, does generally not make any sense. The result of such an operation is of a certain data type as well. Relations like *5 > 4*, for example, result in a logical value (*True* or *False*). The result data type of an arithmetic operation can, however, consist of several different data types. It depends on the input data types and the arithmetic operation performed.

The tables starting from Chapter 6.3 display an overview of the possible operations with matching examples. Please remember that, in order to have easily displayable characters, some operators in TXP/WXP can differ from the generally known ones. Moreover, there are operators mostly originating in the field of IT.

6.2 Using the Operators

Operators can be used for the display of data, for example. This option is available in message templates, workflow actions for users and groups, and for workflow switches depending on such an expression as well.

A short example will demonstrate how operators can be used in a workflow action:
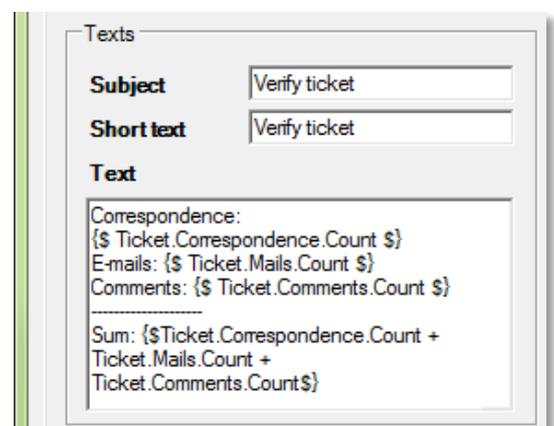
Figure 6-1 Configuring the workflow action

The following data has been entered into the text field in the figure above:

```
Correspondence: {$ Ticket.Correspondence.Count $}

E-mails: {$ Ticket.Mails.Count $}

Comments: {$ Ticket.Comments.Count $}

--------------------

Sum: {$Ticket.Correspondence.Count + Ticket.Mails.Count +
Ticket.Comments.Count$}
```

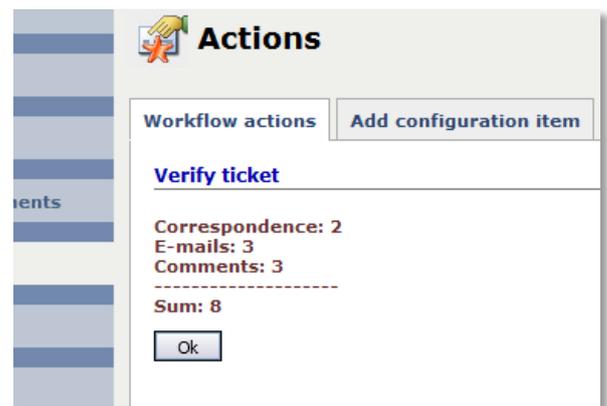In the ticket, this configuration is displayed as follows:

The example shows that when using operators, the start token *{$* and the end token *$}* can only be used at the beginning and at the end of the entire operation. If the expression token is closed after every expression, each expression is processed separately, which will result in no operation.

6.3   Relational Operators

Relational operators generally have the result type *Boolean* (logical value), which can have the following three values:

-        True
-        False
-        Null

These relational operators are often used in switch plug-ins in the workflow. There, tickets can take different directions in the workflow – depending on the expression.

OPERATOR	RESULT TYPE	RESULT
<=	Boolean	[Left] is less than or equals [right]
	Example: Ticket.CreatedDateTime <= Ticket.CompletionDateTime	
<>	Boolean	[Left] does not equal [right]

	Example: Ticket.CreatedDateTime <> Ticket.CompletionDateTime	
>=	Boolean	[Left] is greater than or equals [right]
	Example: Ticket.CreatedDateTime >= Ticket.CompletionDateTime	
=	Boolean	[Left] equals [right]
>	Boolean	[Left] is greater than [right]
<	Boolean	[Left] is less than [right]

<div align="right">**Table 6-1**</div>

6.4 Operators for Strings

One part of the operators for strings is their combination. All data types that can be displayed as strings in any way can be combined.

➜ In one expression chain, only similar operators can be used – thus either && or ||. It is not possible to use && and || simultaneously.

OPERATOR	RESULT TYPE	RESULT
&	String	[Left] is combined with [right]
	Example: `Ticket.AffectedUser.firstname & „ „ & Ticket.AffectedUser.lastname` *Result example*: `Max Muster`	
+	String	[Left] is combined with [right]
	Note: Unnecessary characters (e.g. space characters, line breaks etc.) are removed at the beginning and at the end of every of string before combining the strings. *Example*: `Ticket.AffectedUser.lastname + „ , „ + Ticket.AffectedUser.firstname` *Result example*: `Muster,Max`	
~+	String	[Left] is combined with [right]
	Note: Unnecessary characters are not removed when using this operator. *Example*: `Ticket.AffectedUser.lastname ~+ „ , „ ~+ Ticket.AffectedUser.firstname` *Result example*: `Muster , Max`	

<div align="right">**Table 6-2**</div>

6.5   Arithmetic Operators

OPERATOR	LEFT HAND	RIGHT HAND	RESULT TYPE	RESULT
/	TimeSpan	Double Integer	TimeSpan	TimeSpan([Left].Ticks/[Right])
	Note: TimeSpan is a special data type. It denotes the time span between two dates. The result type is converted into the type TimeSpan again here.			
/	Double Integer	Double Integer	Double Integer	[Left] divided by [Right]
	Note: The result type is determined by the more precise data type in this constellation. If the most precise data type is an integer, the result type is an integer as well. If the most precise data type is a double (and thus more precise than an integer), the result type is a double as well.			
*	TimeSpan	Double Integer	TimeSpan	TimeSpan([Left].Ticks * [Right])
	Note: TimeSpan is a special data type. It denotes the time span between two dates. The result type is converted into the type TimeSpan again here.			
*	Double Integer	Double Integer	Double Integer	[Left] multiplied by [Right]
	Note: The result type is determined by the more precise data type in this constellation. If the most precise data type is an integer, the result type is an integer as well. If the most precise data type is a double (and thus more precise than an integer), the result type is a double as well.			
-	TimeSpan	TimeSpan	TimeSpan	[Right] is subtracted from [left]
-	DateTime	TimeSpan	DateTime	[Right] is subtracted from [left]
-	IsoDateTime	TimeSpan	DateTime	[Left].UtcDateTime – [Right]
-	Double Integer	Double Integer	Double Integer	[Right] is subtracted from [left]
+	Double Integer	Double Integer	Double Integer	[Left] and [Right] are added
	Note: If the data type of one operand is a string type, the combination is used. *Example*: Ticket.CustomAttributes.Count + 1			
~+	Double Integer	Double Integer	Double Integer	[Left] and [Right] are added
	Note: If the data type of one operand is a string type, the combination is used.			

Table 6-3

6.6   Logical Operators

OPERATOR	MEANING	RESULT
&&	And	With this operator, two expressions for relations can be combined.
	Example: Ticket.AffectedUser.lastname = "Muster" && Ticket.AffectedUser.firstname = "Max" Here, both expressions have to be fulfilled (*true*) in order to fulfill the combined expression with *true*.	

||	Or	With this operator, two expressions for relations can be combined.

Example:

Ticket.Owner.User.lastname = "Muster" || Ticket.Owner.User.lastname = "Tester"

Here, only one of the two sub-expressions has to be fulfilled in order to fulfill the combined expression.

Table 6-4

6.7   Brackets

When working with expressions, brackets can be used as well. Thus arithmetic operations and complex expressions containing multiple expression conditions can be compiled. The brackets will be interpreted in the same way as is known from mathematics or computer languages.

Examples

The expression examples are printed in bold and a short explanation can be found below them.

((2+3)*4)+1

The result after the evaluation of the expression is *21*.

(Ticket.Fields!MyField1.Text = "1" || Ticket.Fields!MyField2.Text = "2") && Ticket.Fields!MyField3.Text = "3"

The ticket field *MyField3* has to have the value *3,* here. Moreover, one of the conditions within the brackets has to be fulfilled (thus either the field *MyField1* has to have the value *1* or the field *MyField2* has to have the value *2*).

7 Statistics and Change Management

7.1 Statistics

Created by	Marco Mehl
Creation date	29.12.2005
Doc-ID	DOC-041212-010
Version	3.9-4
Status	Approved
Replaces version	3.9-3
Release date	27.03.2014
Valid from	Immediately
Valid until	Cancellation
Document name	Expression Reference 3.9-4_EN

7.2 Change Management

VERSION	DATE	EXECUTED BY	COMMENT
1.0	29.12.2005	Thomas Luck	
1.1	11.01.2006	Alexander Schmidt	Adapted to new layout
1.2	22.02.2006	Alexander Schmidt	
1.3	28.02.2006	Thomas Luck	
1.4	15.06.2006	Alexander Schmidt	
1.4a	15.06.2006	Alexander Schmidt	
1.5	23.07.2006	Thomas Luck	
1.6	07.09.2006	Ralph Pfister	
1.7	29.11.2006	Thomas Luck	
1.8	08.12.2006	Thomas Luck	
1.9	21.12.2006	Ralph Pfister	Knowledgebase and Tasktool
1.10	11.01.2007	Michael Schuler	Tasktool
1.11	11.07.2007	Luca Marinucci	
1.12	26.10.2007	Thomas Luck	
1.13	02.05.2008	Marco Mehl	Layout and Additionsn (General, Ticket- and Ci-Expressions, Operators)
2.0	06.05.2008	Marco Mehl	Additional documentation (Operators and data fields)
2.1	08.05.2008	Marco Mehl	Minor non-conformities corrected
2.2	08.08.2008	Marco Mehl	Expressions for UTC Date/Timeadded
2.3	19.08.2008	Marco Mehl	Logicaloperatorsadded, diversification
2.4	08.01.2009	Marco Mehl	Two new CI-expressions added, CI-expressions corrected
2.5	20.03.2009	Marco Mehl	Message-expressions and example added
2.6	23.03.2009	Luca Marinucci	Expression added, to access logged in users

2.7	24.06.2009	Marco Mehl	UrlQueryString-Expression added; Ticket.Status.FriendlyName modified and Ticket.Status. Name added
2.8	14.09.2009	Maik Wisatzke	Layout changed
2.9	03.11.2009	Marco Mehl	General.NewLine, Ticket.DataId, Ticket.ObjectGuid, Ticket.Successors.Countadded; removed 2 deprecated expressions
2.10	20.04.2010	Marco Mehl	Advice for accessing ticket fieldsadded; Example added; Expression for Escalation-deadline added; expression for ExpenseCount added
3.0	18.08.2010	Maik Wisatzke	Adjustments for version 3.4
3.1	03.11.2010	Marco Mehl/Maik Wisatzke	4 new expressions for comments; obsolete CI-expressions deleted; new CI-expressions added; 2 early warning expressions added
3.2	29.04.2011	Marco Mehl	Hint for expression chains, task expressions revised
3.3	30.06.2011	Marco Mehl	Task.Title added, new messaging expressions added, EarlyWarnAlert.TriggerTicket.Ticket# corrected
3.4	18.08.2011	Marco Mehl	Ticket.Attachments.Count added; Beschreibung für added the use of brackets; Expressions for clones added; Expressions for process roles and Service Manager added
3.5	19.09.2011	Marco Mehl/Maik Wisatzke	Added ParentTicket Expression to Task Expressions; Added new figures
3.6	05.10.2011	Marco Mehl	Added MainGroup-Expression; Added EarlyWarnAlert.Subscription!Name
3.7	04.11.2011	Marco Mehl	Added expressions for task comments
3.8	29.12.2011	Maik Wisatzke	Added the Ticket.OriginalMessage expression, minor non-conformities corrected
3.9	09.05.2012	Maik Wisatzke/Steffi Kurnot	Restructuring of the expressions, Added expense expressions, and several ticket and service portfolio expressions
3.10	11.07.2012	Maik Wisatzke/ Stephanie Henze	Added Workflowitem Expressions; upgraded ticket escalation expressions; upgraded CiParentCI expressions
3.11	04.12.2012	Maik Wisatzke/ Steffi Kurnot	Added date formats
3.9-1	13.02.2013	Maik Wisatzke/ Steffi Kurnot/ Sephanie Henze	Corrections, restructuring
3.9-2	10.04.2013	Maik Wisatzke/ Stephanie Henze	Added the expression groups Ticket.Parents["Schema"], Ticket.Linked["schema"], Ticket.Successors["schema"] und Ticket.Clones["schema"].Count
3.9-3	28.01.2014	Maik Wisatzke/Anna Hajduk	Information added for the expression groups Ticket.DefaultCostCenter; KB.Article.Owner and Ticket.DirectUrl.Activities. CIZoneDateTime expressions deleted
3.9-4	27.03.2014	Maik Wisatzke/Anna Hajduk	Early warning and static expressions added, examples for the ObjectType added. General revision.