



OTRS

OTRS Features Manual

Release 2024.6.1

OTRS AG

Sep 06, 2024

CONTENTS

1	Introduction	3
1.1	Categories	3
2	Advanced Escalations	5
2.1	Administrator Interface	6
2.2	Agent Interface	28
2.3	External Interface	34
3	Advanced Ticket Overview	35
3.1	Administrator Interface	36
3.2	Agent Interface	36
3.3	External Interface	36
4	Automated FAQ Ticket Creator	37
4.1	Administrator Interface	38
4.2	Agent Interface	44
4.3	External Interface	46
5	Baramundi Inventory Connector	47
5.1	Administrator Interface	47
5.2	Agent Interface	52
5.3	External Interface	52
6	CI Assignment Attribute Dynamic Field Map	53
6.1	Administrator Interface	54
6.2	Agent Interface	56
6.3	External Interface	56
7	CI References	57
7.1	Administrator Interface	58
7.2	Agent Interface	62
7.3	External Interface	62
8	CIs in Customer Frontend	63
8.1	Administrator Interface	64
8.2	Agent Interface	66
8.3	External Interface	67
9	Configuration Management Connector	71
9.1	Administrator Interface	71
9.2	Agent Interface	77

9.3	External Interface	77
10	Customer-specific Services	81
10.1	Administrator Interface	82
10.2	Agent Interface	84
10.3	External Interface	86
11	Data Privacy Protection	87
11.1	Administrator Interface	88
11.2	Agent Interface	101
11.3	External Interface	102
12	Dynamic Field CI	103
12.1	Administrator Interface	104
12.2	Agent Interface	107
12.3	External Interface	108
13	Dynamic Field Value Import	109
13.1	Administrator Interface	110
13.2	Agent Interface	114
13.3	External Interface	114
14	Dynamic Sender Addresses	115
14.1	Administrator Interface	116
14.2	Agent Interface	119
14.3	External Interface	121
15	Escalation Suspend	123
15.1	Administrator Interface	124
15.2	Agent Interface	125
15.3	External Interface	125
16	Primary / Secondary	127
16.1	Administrator Interface	127
16.2	Agent Interface	130
16.3	External Interface	133
17	Ready2Adopt ITSM Processes	135
17.1	Administrator Interface	136
17.2	Agent Interface	137
17.3	External Interface	137
18	Ready2Adopt Web Services	139
18.1	Administrator Interface	140
18.2	Agent Interface	140
18.3	External Interface	140
19	Restrict Customer Data View	141
19.1	Administrator Interface	142
19.2	Agent Interface	144
19.3	External Interface	144
20	SAP Solution Manager Connector	145
20.1	Administrator Interface	145
20.2	Agent Interface	150
20.3	External Interface	150

21 Specific Ticket Notifications	151
21.1 Administrator Interface	151
21.2 Agent Interface	154
21.3 External Interface	156
22 Ticket Allocation	157
22.1 Administrator Interface	158
22.2 Agent Interface	164
22.3 External Interface	164
23 Ticket Invoker	167
23.1 Administrator Interface	167
23.2 Agent Interface	175
23.3 External Interface	175
24 Ticket Time Unit Dropdown	177
24.1 Administrator Interface	178
24.2 Agent Interface	178
24.3 External Interface	178
25 Time and Quota Management	179
25.1 Administrator Interface	180
25.2 Agent Interface	181
25.3 External Interface	184

This work is copyrighted by OTRS AG (<https://otrs.com>), Zimmersmühlenweg 11, 61440 Oberursel, Germany.

INTRODUCTION

This manual lists all features.

Note: The features are installed into **OTRS** by the *Customer Solution Team*. In case of *On-Premise* systems, the customer can install the features from the package manager, when the *Customer Solution Team* added the selected features to the repository. To install a feature, please contact the *Customer Solution Team* via support@otrs.com or in the *OTRS Portal*.

1.1 Categories

Each feature can be categorized based on its purpose.

Administration

- *Dynamic Field Value Import*

Automation & Processes

- *Automated FAQ Ticket Creator*
- *CI Assignment Attribute Dynamic Field Map*
- *Dynamic Sender Addresses*
- *Ready2Adopt ITSM Processes*
- *Ticket Allocation*

Customer Management

- *Customer-specific Services*
- *Time and Quota Management*

Individualization

- *Advanced Ticket Overview*
- *Specific Ticket Notifications*

Integration

- *Baramundi Inventory Connector*
- *Configuration Management Connector*
- *Ready2Adopt Web Services*

- *SAP Solution Manager Connector*
- *Ticket Invoker*

Knowledge Management & Self Service

- *CI References*
- *CIs in Customer Frontend*

Security & Permissions

- *Data Privacy Protection*
- *Restrict Customer Data View*

Ticket Management

- *Dynamic Field CI*
- *Primary / Secondary*

Time Management

- *Advanced Escalations*
- *Escalation Suspend*
- *Ticket Time Unit Dropdown*

ADVANCED ESCALATIONS

This feature makes your escalation management more flexible and adjusts it according to your customers or to different service level agreements. Escalation types defined in **OTRS**, such as *First Response Time*, *Update Time*, and *Solution Time*, can be enhanced by creating new types and defining your own names and properties.

The *Ticket Escalation Types* option in the administrator interface enables you to define when escalations should:

- start,
- stop,
- be suspended,
- be resumed
- and be restarted.

Ticket attributes, like its status or certain events like creating or answering a ticket, can be used as a trigger. For example, an escalation can start when a ticket is created, but the escalation can stop when the ticket is answered by the agent. If the status of a ticket is changed to *Pending Reminder*, the escalation is suspended, but if the status is changed back to *Open*, the escalation resumes. An up-to-date display of the escalation time left makes accurate service time management possible.

In the ticket detail view, a new widget *Advanced Escalations* appears. This shows through the use of different color bars and numerical values whether or not:

- the escalation time is still within the original time frame (green),
- the escalation time will run out soon (orange),
- the escalation is suspended (grey),
- the escalation time has been reached, i.e. the ticket has escalated (red) or
- escalation has been suspended or the ticket has been stopped (the widget is no longer visible).

The *Ticket Escalation Type Bundle* option enables you to assign newly created escalation types to different customers or service level agreements.

The following scenarios can now be handled more flexibly with *Advanced Escalations*:

- A customer requests a rework of a solution –the escalation must be adjusted.
- To present a solution, more information is required from the customer –the escalation must be suspended.
- A service technician can't get into the building or has no free access to the machine that needs to be fixed –the escalation must be suspended.

- And many more

Benefits

- Even more flexible escalation management –individually adoptable to customer or SLA.
- More precise service time management by detailed indication of remaining time.

Target Groups

- Customer service organizations having many partners or suppliers
- External IT service providers
- Call centers
- Sales departments and sales companies
- Advertising or communications agencies

Available in Service Package

- PLATINUM

Package Name in OTRS Package Manager

- OTRSAdvancedEscalations

Note: Not compatible with the following features:

- *Advanced Ticket Overview*
 - *Escalation Suspend*
-

2.1 Administrator Interface

With this package you are able to define your own escalations. You can define fully customized escalation types which contain information about under what circumstances a ticket escalation will start, restart, be suspended, be resumed or stopped.

After installation of the package two new modules will be available in the *Ticket Settings* group of the administrator interface.

2.1.1 Ticket Settings

After installation of the package two new modules will be available in the *Ticket Settings* group of the administrator interface and a new field is added to the *Service Level Agreements* module.

Note: The configuration of the standard escalations in OTRS are not related to advanced escalations. You should decide if you want to use the standard escalation types of OTRS or individual escalation types of advanced escalations. If you configure both then you will have parallel escalations.

Escalation Type Bundles

The *Escalation Types* can be grouped to so-called bundles. Bundles have to be connected to SLAs, and it is possible to filter them by customers, priorities and services in order to define special escalations on a per-customer basis and depending on certain calendar settings.

After installation of the package a module *Escalation Type Bundles* will be available in the *Ticket Settings* group of the administrator interface.

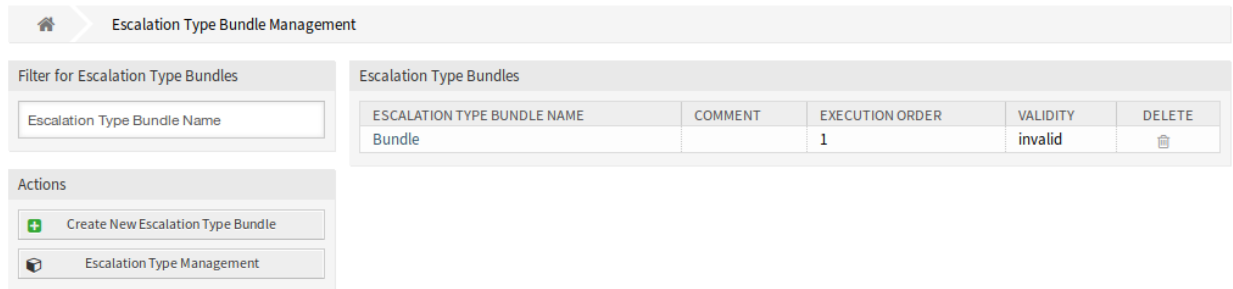


Fig. 1: Escalation Type Bundle Management Screen

The advanced escalations package works with the relationships chain *Escalation type* → *Escalation type bundle* → *SLA*.

One or more escalation type should be created and assigned to an escalation type bundle, and one or more escalation bundles should be related to one or more SLAs. The relationship between *Customer* → *Service* → *SLA* is the normal one on the system.

Manage Escalation Type Bundles

To create a new escalation type bundle:

1. Click on the *Create New Escalation Type Bundle* button in the left sidebar.
2. Fill in the required fields.
3. Click on the *Create* button.
4. You will be redirected to *Edit Escalation Type Bundle* screen to edit the escalation type bundle details.

To edit an escalation type bundle:

1. Click on an escalation type bundle in the list of escalation types bundles or you are already redirected here from *Create New Escalation Type Bundle* screen.
2. Modify the fields and the escalation type bundle details.
3. Click on the *Save* or *Save and finish* button.

To delete an escalation type bundle:

1. Click on the trash icon in the *Delete* column.
2. Click on the *OK* button in the confirmation dialog.

The screenshot shows a web form titled "Create New Escalation Type Bundle". The form contains the following fields and controls:

- Name:** A text input field with an asterisk (*) indicating it is mandatory.
- Comment:** A text input field.
- Description:** A larger text input area.
- Customers:** A text input field.
- Priorities:** A text input field.
- Services:** A text input field.
- Execution Order:** A text input field containing the value "1", with an asterisk (*) indicating it is mandatory.
- Validity:** A text input field containing the value "invalid", with an asterisk (*) indicating it is mandatory.
- Buttons:** A "Create" button and the text "or Cancel" are located at the bottom of the form.

Fig. 2: Create New Escalation Type Bundle Screen

Escalation Type Bundle Settings

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Name *

The name of this resource. Any type of characters can be entered to this field including uppercase letters and spaces. The name will be displayed in the overview table.

Comment

Add additional information to this resource. It is recommended to always fill this field as a description of the resource with a full sentence for better clarity, because the comment will be also displayed in the overview table.

Description

Longer text can be added here to describe the escalation type bundle.

Customers

Select a customer from the drop-down list. If one or more customers are selected, the bundle will only take effect on tickets which are assigned to one of the selected customers.

Priorities

Select a priority from the drop-down list. If one or more priorities are selected, the bundle will only take effect on tickets which matches one of the selected priorities.

Services

Select a service from the drop-down list. If one or more services are selected, the bundle will only take effect on tickets which matches one of the selected services.

Execution order *

The execution order takes place, when at least two bundles could start based on the types and the

Edit Escalation Type Bundle Bundle

★ Name:

Comment:

Description:

Customers:

Priorities:

Services:

★ Execution Order:

★ Validity:

Add escalation type:

Escalation type "First Response" ✕

Escalate after: Round up times

Use calendar: Notify on: %

or or [Cancel](#)

Fig. 3: Edit Escalation Type Bundle Screen

Escalation Type Bundles				
ESCALATION TYPE BUNDLE NAME	COMMENT	EXECUTION ORDER	VALIDITY	DELETE
Bundle		1	invalid	

Fig. 4: Delete Escalation Type Bundle Screen

attributes of the ticket. Only one bundle can be used though. The execution order will change the priority of the bundle, so that the bundle with higher priority will be used.

Validity *

Set the validity of this resource. Each resource can be used in OTRS only, if this field is set to *valid*. Setting this field to *invalid* or *invalid-temporarily* will disable the use of the resource.

Add escalation type

Bundles can contain several escalation types. On a bundle, all information regarding the time for the escalation is being stored.

Escalate after

Define the escalation time span and unit.

Round up times

Check this box to round up the times, so the start of the escalation will take place at the next full time unit.

For example if *1 hour* is set as escalation time and a ticket is being created at 10:23, the escalation will start at 11:00, rounding up to the next full hour. If *1 day* is set, the escalation will start on the next working day (based on the working time).

Use calendar

Select a calendar to used by the escalation. If no calendar is being used, a 24/7/365 calendar without weekends or other free days will be assumed.

Notify on

Define the notification time and the time when the color of the escalation will change to orange.

Escalation Types

Escalation types contain generic information about the conditions that need to be matched in order to start, restart, suspend, resume or stop an escalation on a ticket. The type itself does not contain any time settings (e.g. how many hours need to pass until the escalation starts).

Use this screen to add escalation types to the system. After installation of the package some escalation types are already added. The escalation type management screen is available in the *Escalation Types* module of the *Ticket Settings* group.

Manage Escalation Types

To create a new escalation type:

1. Click on the *Create New Escalation Type* button in the left sidebar.
2. Fill in the required fields.
3. Click on the *Create* button.
4. You will be redirected to *Edit Escalation Type* screen to edit the escalation type details.

To edit an escalation type:

1. Click on an escalation type in the list of escalation types or you are already redirected here from *Create New Escalation Type* screen.
2. Modify the fields and the escalation type details.
3. Click on the *Save* or *Save and finish* button.

Escalation Type Management

Filter for Escalation Types

Escalation Type Name

Actions

Create New Escalation Type

Escalation Type Bundle Management

Configuration import

Here you can upload a configuration file to import an escalation type to your system. The file needs to be in .yaml format as exported by escalation type module.

Browse... No file selected.

Import escalation type configuration

Escalation Types







ESCALATION TYPE NAME	COMMENT	VALIDITY	EXPORT	DELETE
First Response		valid		
Solution Time		valid		
Update Time		valid		

Fig. 5: Escalation Type Management Screen

Create New Escalation Type

★ Name:

Comments:

Description:

Allow resume from stop:

★ Validity:

Create or Cancel

Fig. 6: Create New Escalation Type Screen

Edit Escalation Type First Response

★ Name:

Comments:

Description:

Allow resume from stop:

★ Validity:

Start Restart Suspend Resume Stop

Hint: Within this tab you can set up the conditions that need to be matched to Start the escalation. Please use the checkbox in the first column of each block to enable the condition and set it up afterwards. Please keep in mind that the different blocks and the conditions within the blocks are logically connected by 'and', while the sets of ticket attributes are logically connected by 'or'.

General

Use this dynamic field as base time for this event:

Ticket attributes + Add attribute set

ACTIVE	CONDITION
<input checked="" type="checkbox"/>	A message from an/a <input type="text" value="Agent"/> is not present <input type="text" value="."/> .
and <input checked="" type="checkbox"/>	The ticket attribute <input type="text" value="StateType"/> did never equal <input type="text" value=""/> one of these values: <input type="text" value="closed"/> <input type="text" value="removed"/> <input type="text" value="merged"/> <input type="text" value="pending auto"/> <input type="text" value="pending reminder"/> <input type="text" value=""/>
and <input checked="" type="checkbox"/>	The ticket attribute <input type="text" value="StateType"/> equals now <input type="text" value=""/> one of these values: <input type="text" value="new"/> <input type="text" value="open"/> <input type="text" value=""/>

or or

Fig. 7: Edit Escalation Type Screen

To delete an escalation type:

1. Click on the trash icon in the *Delete* column.
2. Click on the *OK* button in the confirmation dialog.







Escalation Types				
ESCALATION TYPE NAME	COMMENT	VALIDITY	EXPORT	DELETE
First Response		valid		
Solution Time		valid		
Update Time		valid		

Fig. 8: Delete Escalation Type Screen

To export an escalation type:

1. Click on the export icon in the *Export* column.
2. Choose a location in your computer to save the `Export_EscalationTypeID_X.yml` file.

To import an escalation type:

1. Click on the *Browse...* button in the left sidebar.
2. Select a previously exported `.yml` file.
3. Click on the *Import escalation type configuration* button.

Escalation Type Settings

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Name *

The name of this resource. Any type of characters can be entered to this field including uppercase letters and spaces. The name will be displayed in the overview table.

Comment

Add additional information to this resource. It is recommended to always fill this field as a description of the resource with a full sentence for better clarity, because the comment will be also displayed in the overview table.

Description

Like comment, but longer text can be added here.

Allow resume from stop

If this is checked, it allows this escalation to be resumed although it had already been stopped at some point in the past (of course only if the related resume conditions match).

If the resume conditions match on an escalation type which has *Allow resume from stop* enabled, the escalation will be resumed with the time that was still left on this escalation when it was stopped (which means *was fulfilled*).

Validity *

Set the validity of this resource. Each resource can be used in OTRS only, if this field is set to *valid*. Setting this field to *invalid* or *invalid-temporarily* will disable the use of the resource.

Condition types

This widget has five tabs for *Start*, *Restart*, *Suspend*, *Resume* and *Stop* events. The following options are available for the tabs.

General

This setting is only available for the *Start* and *Restart* event of the escalation. This setting controls if the ticket creation time or the time of the current action should be taken as base time for the start or the restart of the escalation.

Example: If you set up *current time* as base time for the restart event, the full escalation time will be available (e.g. 1 hour) if the restart condition matches. If you set up *ticket creation time*, the escalation will restart based on the creation time of the ticket. This could be useful if you close a ticket as successful (which stops your escalation) but it turns out that the problem has not been solved.

You are also able to set a date or a date-time dynamic field as base time. If you create a ticket with this dynamic field filled and the escalation start then the value of the dynamic field will be used as base time. If the configured dynamic field is not filled then the fallback for the base time is used (current time or ticket creation time or no time value - escalation will only start if the dynamic field is filled).

Ticket Attributes

Several attribute sets can be set up. Each of these sets can contain several attribute conditions. The sets are logically connected by *OR*, which means that one of the sets has to be matched in order to trigger the event. The conditions within a set are logically connected by *AND*, which means that all of the conditions have to match in order to verify the whole set.

After adding a new set, you can add conditions to this set by using the *Add attribute set* button on the top right of the widget. After enabling the added condition in the widget, you can choose what ticket attribute you want to check.

Note: All configured conditions are used to evaluate values present on each action, which is defined as a snapshot of the ticket values taken from the ticket history. The time window for this snapshot is defined by the `TicketHistorySplitTimeThreshold` setting. All actions for a ticket are evaluated one by one each time the escalation is recalculated.

Ticket created by a customer

This condition type means that the ticket has to be created by customer user (by sending an email, using the external interface or by an agent using the *New Phone Ticket* screen).

Ticket Attributes

In this drop-down menu you can choose from a list of match types:

equals now

(Is) –One of the configured ticket attributes should match the particular attribute value from the action which is currently being evaluated. The condition evaluates as true if there is at least one match.

doesn't equal now

(Is not) –None of the configured ticket attributes should match the particular attribute value from the action which is currently being evaluated. The condition evaluates as true if there is no match.

equaled

(Was) –One of the configured ticket attributes should match the particular attribute value from the action which is currently being evaluated. All actions from the history of the particular ticket are being evaluated. The condition evaluates as true if there is at least one match in one of the historical ticket attribute values.

did never equal

(Was never) –None of the configured ticket attributes should match the particular attribute value from the action which is currently being evaluated. All actions from the history of the particular ticket are being evaluated. The condition evaluates as true if there is no match in one of the historical ticket attribute values.

changed to

(Changed to) –The configured ticket attribute changed to one of the given values. This implies that the value was different before (= on the previous action).

equals previous

(Was previously) –At least one of the configured ticket attributes should match the particular attribute value on the previous action, which means going back exactly one step in the ticket history. The condition evaluates as true if there is a match in one of the historical ticket attribute values of the previous action.

doesn' t equal previous

(Was not previously) –None of the configured ticket attributes should match the particular attribute value on the previous action, which means going back exactly one step in the ticket history. The condition evaluates as true if there is no match in the historical ticket attribute values of the previous action.

Message presence

This condition type defines if a customer user or agent message must be/must not be present.

Message delivery

If enabled, this will define that the customer user or agent has to send a message to match the condition.

To remove a condition, uncheck the checkbox in the *Active* column, and click on the *Save* button.

Service Level Agreements

It is possible to assign escalation type bundles to an SLA. Therefore, a multiselect field has been added to the *Add Service Level Agreement* screen.

New Field**Escalation Bundles**

Here you can choose from a list of all available bundles which should be assigned to this SLA. Assigning a bundle to a SLA will cause the escalation types, which are assigned to this bundle, to be considered for a ticket which has this SLA set.

Note: The calendar setting and the times in the SLA has no effect for the advanced escalations feature.

Add SLA

★ SLA:

Service:

Calendar:

Escalation - first response time (Notify by)
(minutes): 0 = no escalation - 24 hours = 1440 minutes - Only business hours are counted.

Escalation - update time (minutes): (Notify by)
0 = no escalation - 24 hours = 1440 minutes - Only business hours are counted.

Escalation - solution time (minutes): (Notify by)
0 = no escalation - 24 hours = 1440 minutes - Only business hours are counted.

★ Validity:

Comment:

Escalation Bundles :
Define the escalation bundles.

Dialog message:

Is being displayed if a customer chooses this SLA on ticket creation.

or Cancel

Fig. 9: Add Service Level Agreement Screen

2.1.2 Communication & Notifications

After installation of the package there will be some new events and smart tags available in ticket notifications.

Ticket Notifications

This package adds some new features to the ticket notification methods.

Notification Smart Tags

The notification smart tags only work in notifications related to the escalation. It does not work for `TicketCreate` but it will work for *Escalation: First Response (NotifyBefore)*. The following smart tags with information about each escalation are available:

<OTRS_TICKET_CustomEscalation_EscalationTime>

If the escalation is running, then this tag will return the escalation date-time when the escalation type will escalate (e.g. *2019-01-01 10:00:00*).

<OTRS_TICKET_CustomEscalation_NotifyTime>

If the escalation is running, then this tag will return the escalation notify date-time of the escalation type (e.g. *2019-01-01 10:00:00*).

<OTRS_TICKET_CustomEscalation_EscalationTimeIn>

If the escalation is running, then this tag will return the difference the current time and escalation time in format *3h 30m*.

<OTRS_TICKET_CustomEscalation_TypeName>

This tag will return the name of the escalation type (e.g. *solution time*).

Note: It is not needed to adjust the `CustomEscalation` in the notification tag, but it always takes the information of the used escalation event. If *First Response (NotifyBefore)* is used it will show the information of the first response.

Escalation Events

Escalation events of the advanced escalation always start with *Escalation: Escalation Type Name (Event)*. The following escalation events are available:

Escalation: Escalation Type Name (Start) (EscalationStart_[EscalationTypeID])

This event will be executed if the escalation has been started.

Escalation: Escalation Type Name (NotifyBefore) (EscalationNotifyBefore_[EscalationTypeID])

This event will be executed if the time reached where the agent gets notified for the escalation.

Escalation: Escalation Type Name (Breached) (EscalationBreached_[EscalationTypeID])

This event will be executed if the escalation time has been reached.

Escalation: Escalation Type Name (Restart) (EscalationRestart_[EscalationTypeID])

This event will be executed if the escalation has been restarted.

Escalation: Escalation Type Name (Suspend) (EscalationSuspend_[EscalationTypeID])

This event will be executed if the escalation has been suspended.

Escalation: Escalation Type Name (ResumeSuspend) (EscalationResumeSuspend_[EscalationTypeID])

This event will be executed if the escalation has been resumed from a suspend state.

Escalation: Escalation Type Name (ResumeStop) (EscalationResumeStop_[EscalationTypeID])

This event will be executed if the escalation has been resumed from a stop state.

Escalation: Escalation Type Name (Stop) (EscalationStop_[EscalationTypeID])

This event will be executed if the escalation has been stopped.

2.1.3 Administration

Two new tables are added to the database after installation of the package. The new tables can be used for reporting via SQL box.

Additionally there are several new system configuration options available. You will find these new options in the group `OTRSAdvancedEscalations`.

SQL Box

Escalation history is available for SQL reporting. The following chapters explain the structure of the database tables.

`escalation_history` Table

All escalation events will create new entries in the `escalation_history` table, which is the basis to calculate statistics about the completed escalation cycles.

To activate the additional reporting of the escalation events, you need to enable the following system configuration option:

- `Ticket::EventModulePost###EscalationHistory` (group: `OTRSAdvancedEscalations`, navigation: *Core* → *Event* → *EscalationHistory*).

Make sure OTRS daemon is running.

```
shell> /opt/otrs/bin/otrs.Daemon.pl status
```

This event module will track the following escalation events:

- `EscalationStart`
- `EscalationStop`
- `EscalationSuspend`
- `EscalationRestart`
- `EscalationResumeSuspend`
- `EscalationResumeStop`

The `escalation_history` table has the following columns:

id

This column contains the ID of the escalation history (auto increment).

event_trigger

This column contains the event of the escalation (e.g. EscalationStart).

object_id

This column contains object ID of the escalation (e.g. the ticket ID).

object_type

This column contains object type of the escalation (e.g. Ticket).

object_history_id

This column contains the related ID of the `ticket_history` table to the escalation event.

escalation_type_id

This column contains the escalation type ID of the escalation.

escalation_reached

This column contains whether the escalation time is already reached (possible values: 0/1).

escalation_datetime

This column contains the timestamp of the escalation date.

escalation_time

This column contains the rest of time (seconds) until the ticket will escalate.

escalation_wt

This column contains the rest of time (seconds) until the ticket will escalate (calculated with working calendars).

notify_datetime

This column contains the date-time timestamp of the notify start.

notify_time

This column contains the seconds until the notify start.

escalation_remaining_time

This column contains the rest of time until the ticket will escalate after a suspend of an escalation type.

Note: This column is only filled on suspend state.

escalation_remaining_wt

This column contains the rest of time until the ticket will escalate after a suspend of an escalation type (calculated with working calendars).

Note: This column is only filled on suspend state.

notify_remaining_time

This column contains the seconds until the notify start after a suspend of an escalation type.

Note: This column is only filled on suspend state.

notify_remaining_wt

This column contains the seconds until the notify start after a suspend of an escalation type (calculated with working calendars).

Note: This column is only filled on suspend state.

running_total_time

This column contains the total amount of seconds the timer was running based on the Timer (Start | Restart | Suspend | Resume | Stop) events.

running_total_wt

This column contains the total amount of seconds the timer was running based on the Timer (Start | Restart | Suspend | Resume | Stop) events (calculated with working calendars).

running_total_virtual_time

This column contains the total amount of seconds the timer was running based history entries.

running_total_virtual_wt

This column contains the total amount of seconds the timer was running based history entries (calculated with working calendars).

suspend_total_time

This column contains the total amount of suspended seconds of the escalation type based on the Timer (Start | Restart | Suspend | Resume | Stop) events.

suspend_total_wt

This column contains the total amount of suspended seconds of the escalation type based on the Timer (Start | Restart | Suspend | Resume | Stop) events (calculated with working calendars).

running_last_time

This column contains the seconds between a start or resuming event and a stop or suspending event (e.g. EscalationStart to EscalationSuspend or EscalationResume to EscalationStop).

running_last_wt

This column contains the seconds between a start or resuming event and a stop or suspending event (e.g. EscalationStart to EscalationSuspend or EscalationResume to EscalationStop) (calculated with working calendars).

running_last_virtual_time

This column contains the seconds between a start or resuming event and a stop or suspending event (e.g. EscalationStart to EscalationSuspend or EscalationResume to EscalationStop) based on the history entries of the ticket.

running_last_virtual_wt

This column contains the seconds between a start or resuming event and a stop or suspending event (e.g. EscalationStart to EscalationSuspend or EscalationResume to EscalationStop) based on the history entries of the ticket (calculated with working calendars).

suspend_last_time

This column contains the amount of seconds the ticket escalation was suspended last time based on the history entries of the ticket.

suspend_last_wt

This column contains the amount of seconds the ticket escalation was suspended last time based on the history entries of the ticket (calculated with working calendars).

create_time

This column contains create time of the escalation history entry.

create_by

This column contains ID of the user who triggered the history data set.

change_time

This column contains date and time when the escalation history data set was changed.

change_by

This column contains ID of the user who triggered the data set change.

escalation_history_data Table

All escalation events will create new entries in the `escalation_history` table. For each escalation event it is possible to save ticket and dynamic field data in a separate data table. Make sure that `TriggerEscalationStartEvents` is enabled. The attributes which will be saved can be configured in the following system configuration options:

- `EscalationHistoryData###Ticket` (group: `OTRSAdvancedEscalations`, navigation: `Core` → `EscalationHistoryData`).

Example configuration: `Queue` → `1`

- `EscalationHistoryData###DynamicField` (group: `OTRSAdvancedEscalations`, navigation: `Core` → `EscalationHistoryData`).

Example configuration: `DynamicField_Test` → `1`

To activate the additional reporting of the escalation events, you need to enable the following system configuration option:

- `Ticket::EventModulePost###EscalationHistory` (group: `OTRSAdvancedEscalations`, navigation: `Core` → `Event` → `EscalationHistory`).

Make sure OTRS daemon is running.

```
shell> /opt/otrs/bin/otrs.Daemon.pl status
```

The data of the ticket and dynamic fields will be saved in a separate table `escalation_history_data` with the following columns:

`id`

This column contains the ID of the escalation history (auto increment).

`escalation_history_id`

This column contains ID of the related `escalation_history` entry.

`field_key`

This column contains key of the related data (e.g. `DynamicField_Test` or `Queue`).

`field_value`

This column contains value of the related data (e.g. a dynamic field value or the values of ticket attributes).

`create_time`

This column contains create time of the escalation history data entry.

`create_by`

This column contains ID of the user who triggered the escalation history data set.

`change_time`

This column contains date and time when the escalation history data set was changed.

`change_by`

This column contains ID of the user who triggered the data set change.

System Configuration

In the ticket lists and also in the ticket list widgets, information regarding the escalation data is displayed in one column for each escalation type. These general columns will be made available to all users in all ticket lists by default.

For the general escalation columns, following traffic light colors for the escalation states are used:

- Green: No escalation times are reached or exceeds the limit.
- Orange: The warning time was reached or exceeds the limit, but no escalation time is reached or exceed the limit yet.
- Red: Escalation time is reached or exceeds the limit.
- Grey: Escalation time is currently paused.

In addition to this, you have the possibility to specify which advanced escalation columns you want to use in the ticket lists.

These possible advanced escalation columns are not active by default. If you want to use them, you have to switch them on explicitly. Please follow instructions as outlined in the sections below.

Advanced escalation columns are shown like this: *Escalation type (Column)*, e.g. *First Response Time (Escalation reached, yes/no)*. They are translatable.

Display Advanced Escalation Columns

It is possible to display the escalation times in the ticket lists widget and in the ticket lists.

Dashboard Ticket List Widgets

In the following example, we will add an advanced escalation column named `EscalationDatetime` to the *Escalations* widget on the dashboard, for all available escalation types.

1. Go to the *System Configuration* screen.
2. Search for the setting `AgentFrontend::Dashboard::Widget###EscalatedTickets`.
3. Under the `Config` key, add the following keys to the existing YAML configuration:

```
...
Config:
  ...
  Columns:
    ...
    EscalationType_EscalationDatetime:
      IsVisible: 2
```

4. Make sure to just **add** the new column name to the existing structure, taking care to follow the rules of the YAML syntax.

Note: To make the column available to users, but not visible by default, switch `IsVisible` key to 1.

5. Click on the tick button on the right to save the setting.
6. Deploy the modified system configuration.

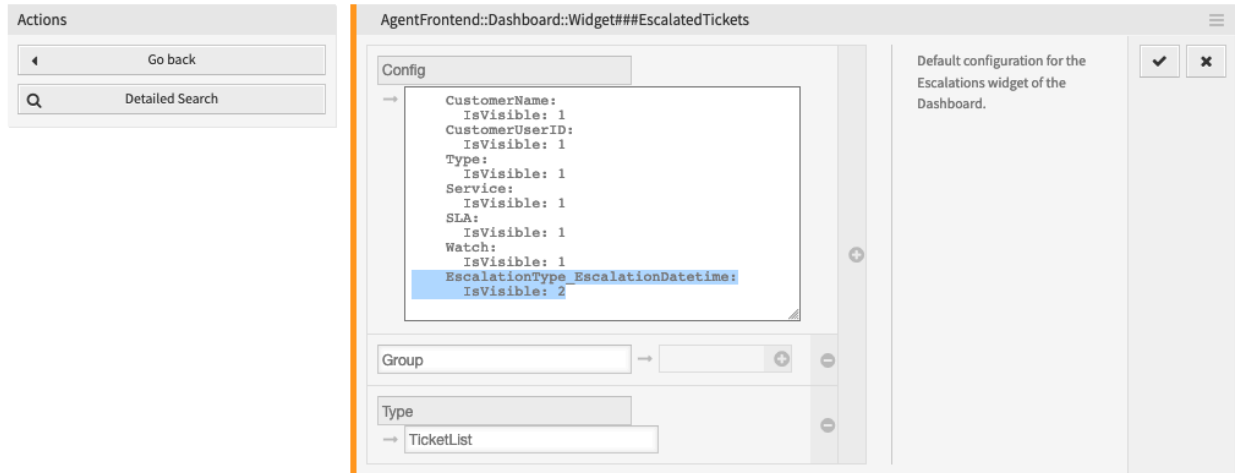


Fig. 10: Add Advanced Escalation Column to *Escalations* Widget Configuration

The referenced advanced escalation column will now be visible by default in the *Escalations* widget, for all available escalation types.

Priority	Created	First Response (Escalation reach date and time)	Update Time (Escalation reach date and time)	Solution Time (Escalation reach date and time)
3 normal	5 days ago	5 days ago	2 hours ago	5 days ago

Fig. 11: Advanced Escalation Columns in the *Escalations* Widget on Dashboard

Ticket List Screens

In the following example, we will add an advanced escalation column named `EscalationReached` to the *Escalated Tickets* screen, for all available escalation types.

1. Go to the *System Configuration* screen.
2. Search for the setting `AgentFrontend::TicketList::Escalations###DefaultConfig`.
3. Under the `Config` key, add the following keys to the existing YAML configuration:

```

...
Columns:
  ...
  EscalationType_EscalationReached:
    isVisible: 2
  
```

4. Make sure to just **add** the new column name to the existing structure, taking care to follow the rules of the YAML syntax.

Note: To make the column available to users, but not visible by default, switch `isVisible` key to 1.

5. Click on the tick button on the right to save the setting.

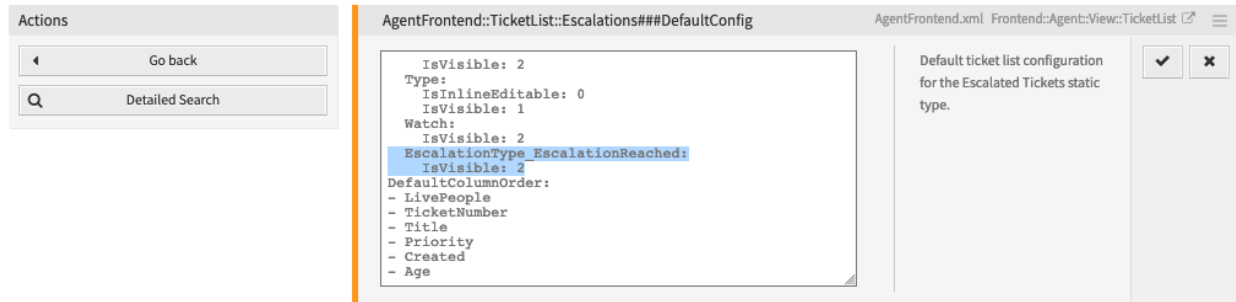


Fig. 12: Add Advanced Escalation Column to *Escalated Tickets* Screen Configuration

6. Deploy the modified system configuration.

The referenced advanced escalation column will now be visible by default in the *Escalated Tickets* screen, for all available escalation types.

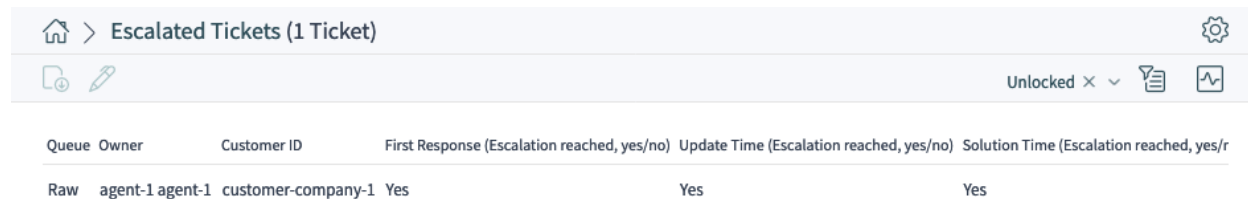


Fig. 13: Advanced Escalation Columns in the *Escalated Tickets* Screen

To switch on any other supported advanced escalation column, change the column name to the following format:

```
EscalationType_ColumnIdentifier
```

Where:

ColumnIdentifier

Any of supported column identifiers as listed in the *Possible Advanced Escalation Columns* section below.

For example, to switch on the `EscalationRemainingTime` column, use the following name:

```
EscalationType_EscalationRemainingTime
```

It is also possible to switch on the advanced escalation column for a particular escalation type only. To do this, change the column name to the following format:

```
Escalation_ID_ColumnIdentifier
```

Where:

ID

Specific escalation type ID.

You can find this out by visiting the administrator interface, and editing the type in the *Escalation Types* module. It will be visible in the URL address field:

```
.../otrs/index.pl?Action=AdminEscalationType;Subaction=Edit;ID=1;
```

In the example above, the ID is 1.

ColumnIdentifier

Any of supported column identifiers as listed in the *Possible Advanced Escalation Columns* section below.

For example, to switch on the `EscalationWorkingTime` column, but only for the escalation type with ID of 1, use the following name:

```
Escalation_1_EscalationWorkingTime
```

To change the default behavior of the general escalation columns which are available, but not visible by default, modify the column configuration in the following way:

```
Escalation_ID:
  isVisible: 0|1|2
```

Where:

ID

Specific escalation type ID.

You can find this out by visiting the administrator interface, and editing the type in the *Escalation Types* module. It will be visible in the URL address field:

```
.../otrs/index.pl?Action=AdminEscalationType;Subaction=Edit;ID=1;
```

In the example above, the ID is 1.

isVisible

Defines whether the column is not visible (0), not visible by default but the agent can make it visible (1) or visible by default (2).

For example, to make the general column for the escalation type with ID of 1 visible by default, provide the following configuration:

```
Escalation_1:
  isVisible: 2
```

Possible Advanced Escalation Columns

For each configured escalation type the following 22 advanced escalation columns are calculated and available. You will find a complete list with explanations below.

EscalationDatetime

Timestamp of the escalation date.

EscalationReached

Yes/no value whether escalation time is reached.

EscalationTime

Seconds until the escalation type is breached.

EscalationWorkingTime

Seconds until the escalation type is breached with working calendar (only if calendar is defined).

EscalationRemainingTime

Seconds until the escalation type is breached if suspended/stopped (only set if given).

EscalationRemainingWorkingTime

Seconds until the escalation type is breached if suspended/stopped with working calendar (only if calendar is defined and if given).

NotifyDatetime

Timestamp of notify start (only set if given).

NotifyTime

Seconds until the notify start (only if given).

NotifyRemainingTime

Seconds remaining until notify start if suspended/stopped (only set if given).

NotifyRemainingWorkingTime

Seconds remaining until notify start if suspended/stopped with working calendar (only if calendar is defined).

SuspendLastTime

Total amount of seconds that the timer was suspended last time.

SuspendLastWorkingTime

Total amount of seconds that the timer was suspended last time with working calendars (only if calendar is defined).

SuspendTotalTime

Total amount of seconds that the timer was suspended.

SuspendTotalWorkingTime

Total amount of seconds that the timer was suspended with working calendars (only if calendar is defined).

RunningTotalTime

Total amount of seconds that the timer was running (excluding suspend times).

RunningTotalWorkingTime

Total amount of seconds that the timer was running (excluding suspend times) with working calendars.

RunningTotalVirtualTime

Total amount of seconds that the timer was running with `BaseTime` as start date (excluding suspend times).

RunningTotalVirtualWorkingTime

Total amount of seconds that the timer was running with `BaseTime` as start date (excluding suspend times) with working calendars (only if calendar is defined).

RunningLastTime

Total amount of seconds that the timer was running last time (excluding suspend times).

RunningLastWorkingTime

Total amount of seconds that the timer was running last time (excluding suspend times) with working calendars.

RunningLastVirtualTime

Total amount of seconds that the timer was running last time with `BaseTime` as start date (excluding suspend times).

RunningLastVirtualWorkingTime

Total amount of seconds that the timer was running last time with `BaseTime` as start date (excluding suspend times) with working calendars (only if calendar is defined).

2.1.4 Example Usage

This chapter describes how to add a first response time escalation.

Creating the Escalation Type

Go to the administration interface and choose *Escalation Type*. Click on the *Create New Escalation Type* button.

Fill in the needed fields. After submitting the form, you will be redirected to the edit screen for the new escalation type. You will see the *Start* tab for the new escalation type. Choose from the available conditions which one you want to be needed in order to start the escalation. Switch to another tab to set up more conditions.

In this example, we would enable the *The ticket has been created by a customer* setting and the *An agent message is not present*. Use the drop-down menu in order to choose whether or not an agent message has to be present.

Switch to the *Stop* tab in order to set up the escalation stop conditions. In this case, the escalation should be stopped if an agent sends a message, so we enable the *An agent sent a message* setting.

Creating the Escalation Type Bundle

Go to the administration interface and choose *Escalation Type Bundle*. Click on the *Create new escalation type bundle* button.

Fill in the needed fields. See the description on the previous pages in order to fill in *Execution order* correctly. After submitting the form, you will be redirected to the edit screen for the new bundle. You will now be able to add your previously created escalation type by choosing it from the *Add escalation type* drop-down menu. After you have added the escalation type, you can set up its parameters. Set up a time span (like *1*) and a time unit (like *Hour(s)*). Save the screen.

Assigning the New Bundle to an SLA

In order to get the new escalation type to work, you need to assign it to an existing SLA. Go to the SLA management screen for an existing SLA and select your bundle from the list of available bundles. Save the screen.

Conclusion

The new escalation type in connection with the bundle and SLA assignment will cause new tickets which are being created by customers to escalate after one hour if no agent has responded.

2.2 Agent Interface

For assigning an advanced escalation to a ticket, this one should contain a valid customer ID and valid SLA, so the advanced escalation for the ticket will be assigned based on previous configuration. Advanced escalations currently does not work for unknown customers.

It is possible to show advanced escalations columns in ticket lists, ticket list widgets and statistics. The advanced escalations data is calculated on-the-fly and could have a huge negative performance impact on the whole system.

Note: Escalation automatism is not supported. Use the generic agent or generic interface to trigger escalation events.

2.2.1 Tickets

After installation of the package a new *Advanced Escalations* widget is displayed in the ticket detail view.

Display Escalations in a Widget

All advanced escalations that are associated to a ticket will be shown in *Advanced Escalations* widget in the ticket detail view.

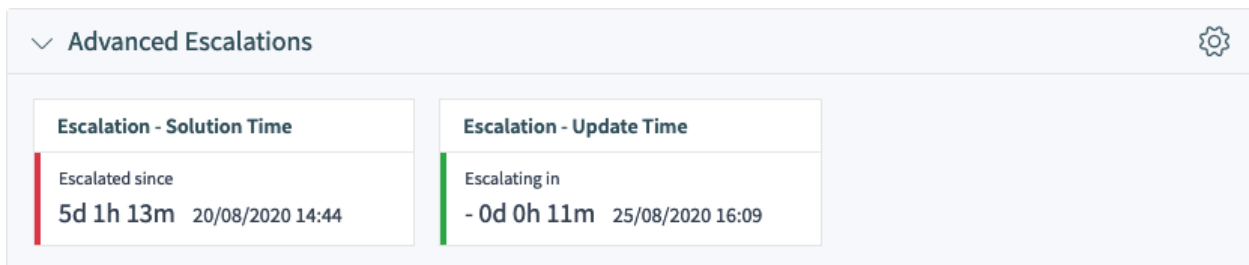


Fig. 14: Advanced Escalations Widget in Ticket Detail View

The escalations are displayed as cards with a colored bar, indicating the current escalation state for better visibility.

- Green bar: the escalation time has not been reached.
- Orange bar: the escalation notification time has been reached.
- Red bar: the escalation time has been reached.
- Grey bar: the escalation is suspended.

Debug Escalations

Advanced escalations might be very complex and it needs to be investigated if something does not work as expected. It cuts the ticket history entries into snapshots to calculate the escalation. Each snapshot will contain the current state of the ticket with all related data of the current time of the snapshot. A snapshot is 5 seconds of the ticket history by default, but it can be adjusted in the system configuration.

For more information about a single escalation and how it is calculated you can take a look at the ticket history. If your system has a customer ID and a SLA with related escalation bundles and types then you will be able to debug these, based on the ticket history of the current ticket.

To get access to the debug information, click on *View History* ticket action in the ticket detail view.

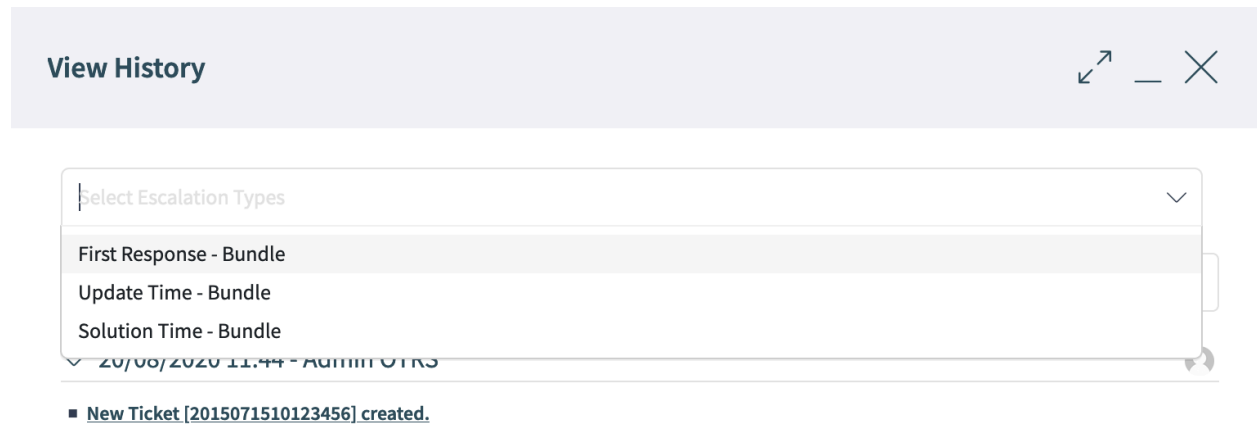


Fig. 15: Escalation Debug Mode –Select Escalation Type

It is usually in *Escalation Type Name –Escalation Bundle Name* format. In the example above the escalation type name is *First Response* and the escalation bundle name is *Bundle*.

After choosing an escalation type from the list, you will get a detailed information about the parameters, attribute values and conditions, right below relevant ticket history entry.

Escalation State

The first block contains the state of the current escalation type. In the example it will calculate the start condition of the escalation type and it was successful (success = green, red = no change of the state). Basically, for each snapshot it is possible that the escalation will calculate multiple state if it does not match to the first.

The order of the calculated states is based on the running, suspended or not running state of the escalation. Here is a short technical overview:

```
my %EscalationStatesMachine = (
  NotRunning => [ 'Start', 'Restart' ],
  Running    => [ 'Suspend', 'Restart', 'Stop' ],
  Suspended  => [ 'Resume', 'Stop' ],
);
```

So if the escalation is not running then it will only try to move to the *Start* or *Restart* state of the escalation. If it is suspended then only *Resume* or *Stop* are possible as next states.

21/08/2020 16:10 - agent-1 agent-1 👤

- Changed customer to "CustomerID=customer-company-1;CustomerUser=customer-1;"

Debug - Start - First Response - Bundle

Attribute Sets

Attribute	Set	Success	Fail
●	1	Agent Message Present,Is_StateType,WasNever_StateType	

Attributes

Attribute	Currently	Previous	Was
Priority	3 normal	3 normal	
Queue	Raw	Raw	
StateType	new	new	
State	new	new	
TicketType	Unclassified	Unclassified	

Fixed Filters

Attribute	Filters
●	Agent Message Present
●	Agent Message Sent
●	Created By Customer
●	Customer Message Present
●	Customer Message Sent

Fig. 16: Escalation Debug Mode –Debug Snapshot of Escalation Type

> Debug - Start - Solution Time - Bundle

Fig. 17: Escalation Debug Mode –Escalation State

Escalation Attribute Sets

The block with the attribute sets displays if the attribute sets of the escalation type for the given escalation state is matching for the current snapshot.

Attribute Sets			
Attribute	Set	Success	Fail
●	1		Is_StateType
●	2	Agent Message Sent	

Fig. 18: Escalation Debug Mode –Escalation Attribute Sets

There are 4 columns:

- **Attribute:** The circle will be displayed green if it matches or red if not.
- **Set:** This column contains the number of the attribute set in the escalation type. The number will be displayed green if it matches or red if not.
- **Success and Fail:** These columns show the matched conditions of an attribute set. At this point it will use internal names for matching the attributes. Here are some examples:
 - `Agent Message Present`: A message from an agent is present.
 - `Agent Message Sent`: An agent sent a message.
 - `Customer Message Present`: A message from a customer is present.
 - `Customer Message Sent`: A customer sent a message.
 - For ticket attributes it will show the internal condition:

```
Is      => "equals now",
IsNot   => "doesn't equal now",
Was     => "equaled",
WasNever => "did never equal",
ChangedTo => "changed to",
IsPrev  => "equals previous",
IsNotPrev => "doesn't equal previous",
```

- `Is_StateType`: The ticket attribute condition `StateType` of the attribute set of the current snapshot matched if green or not matched if red.

Escalation Attributes

The block with the attributes of the snapshot will contain all attributes which will be used for checking the condition of the escalation ticket attributes.

There are 4 columns:

- **Attribute:** This column contains the ticket attribute.
- **Currently:** This column contains the current value of the snapshot. This value will be used for the calculation of *equals now*, *doesn't equal now* and *changed to* conditions.

Attributes			
Attribute	Currently	Previous	Was
Priority	3 normal	3 normal	
Queue	Raw	Raw	
StateType	open	new	new
State	open	new	new
TicketType	Unclassified	Unclassified	

Fig. 19: Escalation Debug Mode –Escalation Attributes

- **Previous:** This column contains the value of the previous snapshot. This value will be used for the calculation of *equals previous*, *doesn't equal previous* and *changed to* conditions.
- **Was:** This column contains the value of all previous snapshot. This value will be used for the calculation of *did never equal* and *equaled* conditions.

Escalation Filters

The filter values are relating to the agent and customer message send or present attributes which are set in the conditions. If a filter is green then the filter is matched and red if a filter is not matched.





Fixed Filters	
Attribute	Filters
	Agent Message Present
	Agent Message Sent
	Customer Message Present
	Customer Message Sent

Fig. 20: Escalation Debug Mode –Escalation Filters

- **Agent Message Present:** A message from an agent is present.
- **Agent Message Sent:** An agent sent a message.
- **Customer Message Present:** A message from a customer is present.
- **Customer Message Sent:** A customer sent a message.

2.2.2 Statistics and Reports

The escalation information can be displayed in reports and statistics using the new statistics object modules.

Advanced Escalations Data in Statistics

By installing the advanced escalations package, you have the possibility to use the new statistics object modules:

TicketAccountedTimeEscalation

A new matrix statistic which only contains tickets with accounted time. It is a copy of the `TicketAccountedTime` statistic out of the **OTRS** framework, but you are able to show advanced escalations data and to filter the data by ticket attributes as well as by advanced escalations.

TicketEscalation

A new matrix statistic which shows configured columns on X- and Y-axis. It is a copy of the ticket matrix statistic out of the **OTRS** framework, but you are able to show advanced escalations data and to filter the data by ticket attributes as well as by advanced escalations.

TicketListEscalation

A new statistic which shows configured columns (advanced escalations data, too) on X-axis. You can specify a column which is used in an order by clause on Y-axis. You are able to filter the data by ticket attributes as well as by advanced escalations columns.

TicketSolutionResponseTimeEscalation

A new matrix statistic which only contains closed tickets. It is a copy of the default `TicketSolutionResponseTimeEscalation` statistic out of the **OTRS** framework, but you are able to select advanced escalations columns in the *Elevation by* field. As already mentioned in the other new statistics, you can filter the data by advanced escalations columns.

Filtering

In general, there are the following types of advanced escalations columns:

Yes/no columns

For this type of columns (e.g. `EscalationReached`), a drop-down field is displayed to select the wanted values. This type is used on X- and Y-axis and as filter.

Datetime columns

For this type of columns (e.g. `EscalationDatetime`), a complex field with the selection of an absolute or a relative time range is shown. This type is used on X- and Y-axis and as filter.

Note: Please note that in the `TicketListEscalation` statistic these columns are not available on the Y-axis.

Time columns (seconds)

For this type of columns (e.g. `RunningTotalTime`), a normal input field is shown. You can use these columns only as filter as *greater or equal* and/or *smaller or equal*.

Entered values are treated as minutes. For example if you enter 30 it is converted to $30 \times 60 = 1800$ seconds automatically.

2.3 External Interface

This package has no external interface.

ADVANCED TICKET OVERVIEW

With this feature you receive an additional way to view linked items. To see linked objects and tickets, you just have to move the mouse over the chain icon in the *Related* column of your ticket list, which is visible if the ticket contains linked items of any sort. Hovering the mouse cursor over the chain icon will open a pop-up showing all linked items grouped by type. To name a few: appointments, configuration items, knowledge items, or linked tickets.

In the pop-up you can directly open the drop-down of each linked item group by clicking on them. You can set the columns for each object and of course, it is possible to directly open the linked ticket or object from the pop-up. So with the new extended ticket overview, agents get a much easier overview of linked items, saving them a lot of time and effort.

Benefits

- Time saving.
- Less effort.
- Better overview.

Target Groups

- Customer service
- IT service
- Sales departments
- Facility management
- Human resources
- Logistics

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSAdvancedTicketOverview

3.1 Administrator Interface

This package has no administrator interface.

3.2 Agent Interface

This feature adds a popover view for linked items inside ticket lists.

The screenshot shows a ticket list with a popover window titled "Related Items" open over a ticket. The ticket list has columns for Number, Title, and Related. The popover window displays a list of related items grouped by type: Appointments (3), Knowledge Base Articles (4), and Tickets (2). The Tickets group is expanded to show a table of related tickets with columns for Title, Priority, Created, State, Sender, Lock, Queue, and Owner.

Number	Title	Related
2021102510000028	Test notific	🔗
2021102510000019	Test notific	🔗
2021051710000017	Calendar a	
2021032910000016	Stolen crec	🔗
2021032410000016	Credit card	
2020032410000411	Re: Some f	
2020032410000376	Re: subject alalal 1234	

Title	Priority	Created	State	Sender	Lock	Queue	Owner
Test notification	3 normal	2 months ago	open	Lacey Green	locked	Raw	John Smit
Test notification	3 normal	2 months ago	open	Lacey Green	unlocked	Raw	Admin OT

Fig. 1: Related Items Inside Ticket List

To see the related items, hover the mouse over the chain symbol in the column *Related*. The chain symbol is displayed only if the ticket has linked items. A popover window opens and displays the linked items grouped by types.

Note: This feature does not work on mobile devices.

3.3 External Interface

This package has no external interface.

AUTOMATED FAQ TICKET CREATOR

With the help of this feature, it is possible to create time-controlled tickets via knowledge base articles. This can refer to the knowledge base article itself or to a specific task that is related to the knowledge base article topic. This is particularly useful in situations, such as maintenance work, where routine activities take place at large intervals. The automatically generated ticket becomes a *To do* for the employee. And, through the knowledge base article documentation, everyone knows the steps of the task.

Example

An employee creates a knowledge base article for a specific maintenance task. The feature will now automatically create an associated ticket after a specified period of time, reminding an employee to either renew the knowledge base article or to perform the activity described in the knowledge base article. For new employees who do not have experience, the knowledge base article makes it possible for them to perform the activity too, because the steps are clearly described in the documentation.

Benefits

- Keep knowledge base articles up-to-date with reminders about necessary update.
- Less frequently opened *resubmission tickets*.
- Functionality can be used for one-time activities that include an end date.
- Even complex tasks can be done by new or untrained employees.

Target Groups

- IT service management
- Facility management
- Service providers
- Companies with recurring activities, such as maintenance work
- Companies with a large number of employees

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSAutomatedFAQTicketCreator

4.1 Administrator Interface

This chapter describes the new features that are available in the administrator interface after installation of the package.

4.1.1 Processes & Automation

After installation of the package some new dynamic fields are added to the system.

Dynamic Fields

After installation of the package some new dynamic fields are added to the system. The dynamic field management screen is available in the *Dynamic Fields* module of the *Processes & Automation* group.

New Dynamic Fields

This packages provides new dynamic fields and an OTRS daemon cron job to create new tickets with configured values based on knowledge base articles.

OTRSAutomatedFAQTicketCreatorStartTime

This dynamic field defines the start time for the ticket creation. This is a date/time field.

OTRSAutomatedFAQTicketCreatorEndTime

This dynamic field defines the end time for the ticket creation. This is a date/time field.

OTRSAutomatedFAQTicketCreatorFrequency

This dynamic field defines the frequency for the ticket creation. Possible values are: *Daily, Weekly, Monthly, Quarterly, Yearly*.

To create a ticket for each month choose the frequency *Monthly*.

OTRSAutomatedFAQTicketCreatorRepeatTimes

This dynamic field defines the repeat times for the ticket creation. Possible values are: 1-20.

If you choose the frequency *Monthly* and a repeat time of 2 then the ticket will be created each second month: January, March, May, July, etc.

OTRSAutomatedFAQTicketCreatorRepeatOnDays

This dynamic field defines the repeat on days for the ticket creation. Possible values are: *Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday*.

If you choose the frequency *Daily*, a repeat time of 2 and *Monday to Friday* as repeat on days then the ticket will be created each second day in the range between Monday to Friday: Monday, Wednesday Friday, Monday, etc.

OTRSAutomatedFAQTicketCreatorLastExecution

This dynamic field defines the last execution time for the ticket creation. This is a date/time field.

OTRSAutomatedFAQTicketCreatorSubject

This dynamic field defines the article subject for the ticket creation. Possible value is: *[Article title]*, for example: *This is the subject of the article*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:



















Dynamic Fields List ⚙️						
26-43 of 43 - Page: 12						
NAME	LABEL	ORDER	TYPE	OBJECT	VALIDITY	DELETE
OTRSAutomatedFAQTicketCreatorStartTime	Ticket create start time	26	Date / Time	FAQ	valid	
OTRSAutomatedFAQTicketCreatorEndTime	Ticket create end time	27	Date / Time	FAQ	valid	
OTRSAutomatedFAQTicketCreatorFrequency	Ticket create frequency	28	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorRepeatTimes	Ticket create repeat times	29	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorRepeatOnDays	Ticket create repeat on days	30	Multiselect	FAQ	valid	
OTRSAutomatedFAQTicketCreatorLastExecution	Ticket create last execution	31	Date / Time	FAQ	valid	
OTRSAutomatedFAQTicketCreatorSubject	Ticket create subject	32	Text	FAQ	valid	
OTRSAutomatedFAQTicketCreatorTitle	Ticket create title	33	Text	FAQ	valid	
OTRSAutomatedFAQTicketCreatorCustomerUser	Ticket create customer user	34	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorOwner	Ticket create owner	35	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorResponsible	Ticket create responsible	36	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorPriority	Ticket create priority	37	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorState	Ticket create state	38	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorService	Ticket create service	39	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorBody	Ticket create body	40	Multiselect	FAQ	valid	
OTRSAutomatedFAQTicketCreatorSLA	Ticket create SLA	41	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorType	Ticket create ticket type	42	Dropdown	FAQ	valid	
OTRSAutomatedFAQTicketCreatorQueue	Ticket create queue	43	Dropdown	FAQ	valid	

Fig. 1: Dynamic Field Management Screen

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Subject`

OTRSAutomatedFAQTicketCreatorTitle

This dynamic field defines the ticket title for the ticket creation. Possible value is: *[Ticket title]*, for example: *This is the title of the ticket*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Title`

OTRSAutomatedFAQTicketCreatorCustomerUser

This dynamic field defines the ticket customer user for the ticket creation. Possible values are: *[Customer user login]* → *[Customer user full name]*, for example: *customer-1* → *John Doe*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###CustomerUser`

OTRSAutomatedFAQTicketCreatorOwner

This dynamic field defines the ticket owner for the ticket creation. Possible values are: *[Owner user login]* → *[Owner full name]*, for example: *root@localhost* → *Admin OTRS*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Owner`

OTRSAutomatedFAQTicketCreatorResponsible

This dynamic field defines the ticket responsible for the ticket creation. Possible values are: *[Responsible user login]* → *[Responsible full name]*, for example: *root@localhost* → *Admin OTRS*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Responsible`

OTRSAutomatedFAQTicketCreatorPriority

This dynamic field defines the priority for the ticket creation. Possible values are: *[Priority Name]* → *[Priority Name]*, for example: *very low* → *1 very low*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Priority`

OTRSAutomatedFAQTicketCreatorState

This dynamic field defines the ticket state for the ticket creation. Possible values are: *[State name]* → *[State name]*, for example: *pending reminder* → *pending reminder*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###State`

OTRSAutomatedFAQTicketCreatorService

This dynamic field defines the ticket service for the ticket creation. Possible values are: *[Service name]* → *[Service name]*, for example: *1st Level Service* → *1st Level Service*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Service`

OTRSAutomatedFAQTicketCreatorBody

This dynamic field defines the article body for the ticket creation. Possible values are: *[Knowledge base article field index]* → *[Knowledge base article field caption]*, for example: *1* → *Symptom*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Body`

OTRSAutomatedFAQTicketCreatorSLA

This dynamic field defines the ticket SLA for the ticket creation. Possible values are: *[SLA name]* → *[SLA name]*, for example: *1st Level SLA* → *1st Level SLA*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###SLA`

OTRSAutomatedFAQTicketCreatorType

This dynamic field defines the ticket type for the ticket creation. Possible values are: *[Ticket type name]* → *[Ticket type name]*, for example: *Unclassified* → *Unclassified*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Type`

OTRSAutomatedFAQTicketCreatorQueue

This dynamic field defines the queue for the ticket creation. Possible values are: *[Queue name]* → *[Queue name]*, for example: *Raw* → *Raw*.

See also:

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###Queue`

To add new values:

1. Choose the dynamic field you would like to change the values for.
2. Add a new value in the *Field Settings* widget.
3. Click on the *Save* button to save the dynamic field.

System Configuration

Use the following system configuration option to copy dynamic field values of the knowledge base article to the ticket:

- `OTRSAutomatedFAQTicketCreator::Core::DynamicFieldMapping###DynamicField`

To define a default value for this dynamic field, you can also set a value to the following system configuration option:

- `OTRSAutomatedFAQTicketCreator::DefaultValues###DynamicField`

Usage

The following use case example will show how to edit and add a customer user.

To edit field values of a dynamic field:

1. Go to the *Dynamic Fields* module of the administrator interface.
2. Choose the dynamic field you like to change the values for.
3. Add a new value in the *Field Settings* widget.
4. Click on the *Save* or *Save and finish* button to save the dynamic field.

For example copy the login name, first name and last name of the customer user to the field.

Dropdown Field Settings

Possible values: ★ Key: ★ Value:

Add value:

Default value:

This is the default value for this field.

Fig. 2: Edit Dynamic Field Value

To edit default field values of a dynamic field:

1. Go to the *System Configuration* screen.
2. Select `OTRSAutomatedFAQTicketCreator` in the *Navigation* widget.
3. Navigate to `Core` → `OTRSAutomatedFAQTicketCreator` → `DefaultValues` in the navigation tree.
4. Add the default value for customer user to setting `OTRSAutomatedFAQTicketCreator::DefaultValues###CustomerUser`.

To copy a field value of a knowledge base article dynamic field into the new ticket dynamic field:

1. Go to the *System Configuration* screen.
2. Select `OTRSAutomatedFAQTicketCreator` in the *Navigation* widget.
3. Navigate to `Core` → `OTRSAutomatedFAQTicketCreator` in the navigation tree.
4. Search for setting `OTRSAutomatedFAQTicketCreator::Core::DynamicFieldMapping###DynamicField`.

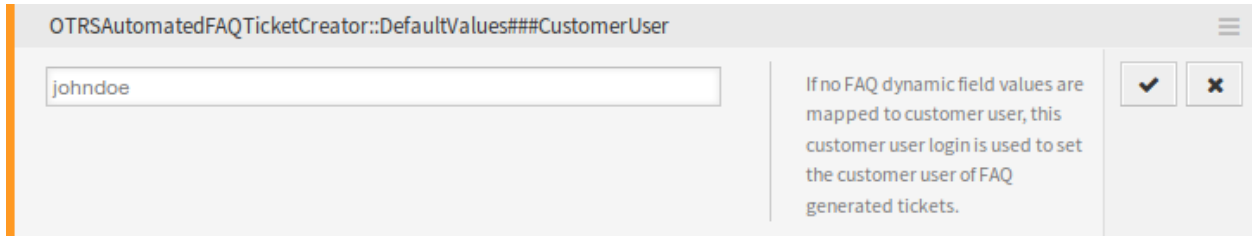


Fig. 3: System Configuration –Default Value

5. Set mapping for a dynamic field. You need to use the ticket field as key and the knowledge base article field as value.

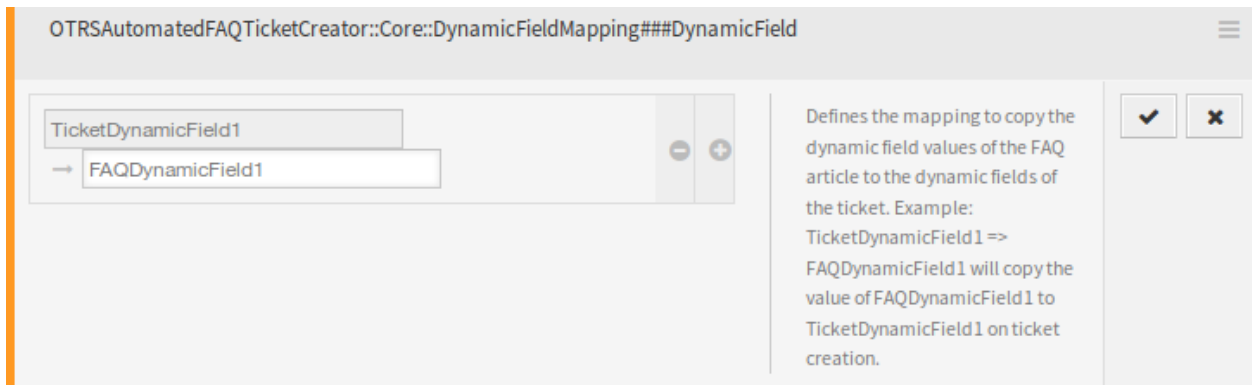


Fig. 4: System Configuration –Copy Value

Note: It is only possible to map dynamic fields of the same type.

To set default field values of a dynamic field for the ticket created dynamic fields:

1. Go to the *System Configuration* screen.
2. Select *OTRSAutomatedFAQTicketCreator* in the *Navigation* widget.
3. Navigate to *Core* → *OTRSAutomatedFAQTicketCreator* → *DefaultValues* in the navigation tree.
4. Search for setting `OTRSAutomatedFAQTicketCreator::DefaultValues###DynamicField`.
5. Set some default values, for example:
 - **Text:** `ExampleTicketDynamicField1Text` → *hallo*
 - **Multiselect:** `ExampleTicketDynamicField1Multiselect` → *Value1;Value2;Value3*
 - **Date:** `ExampleTicketDynamicField1Date` → *2014-03-03*
 - **Date/Time:** `ExampleTicketDynamicField1DateTime` → *2014-03-03 10:00:00*
 - **Checkbox:** `ExampleTicketDynamicField1Checkbox` → *1*

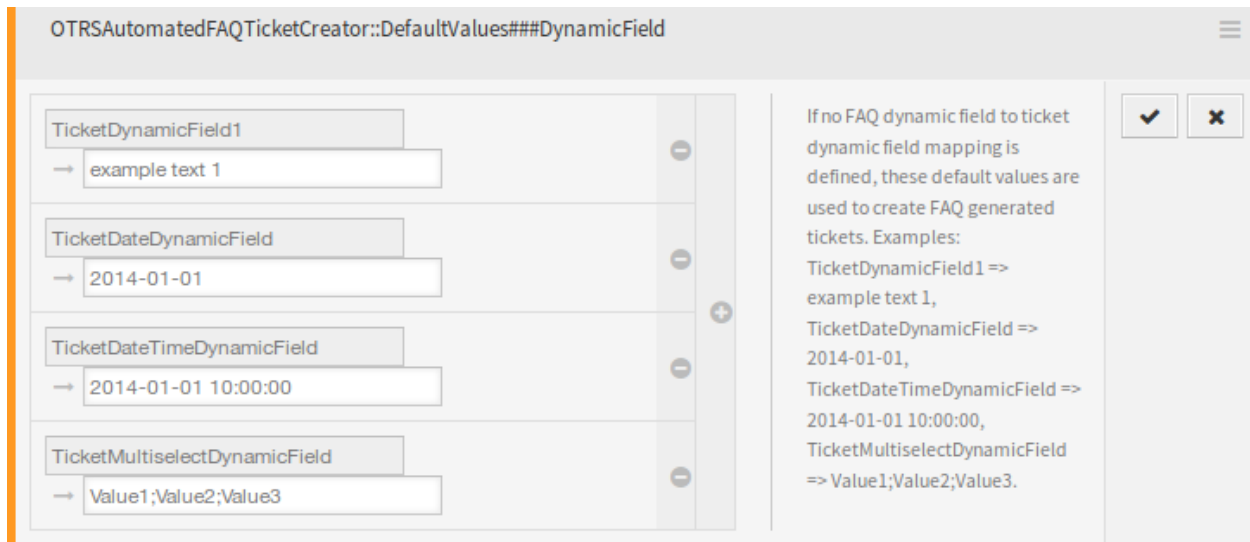


Fig. 5: System Configuration –Default Field Value

4.2 Agent Interface

This package has no dedicated agent interface, but the *Create Knowledge Base Article* and the *Update Knowledge Base Article* screens are updated with new dynamic fields.

4.2.1 Knowledge Base Articles

After installation of the package some new dynamic field are added to the *Create Knowledge Base Article* and the *Update Knowledge Base Article* screens.

It is possible to define the ticket fields to be auto-populated and the ones that has to be filled manually.

By default only certain dynamic fields are shown in the knowledge base article creation and edit screens, those that controls the frequency of the ticket creation but not ones that set values for the ticket, instead the ticket uses the default values provided in the system configuration settings described in *Dynamic Fields* chapter of the administrator interface.

See also:

To show the dynamic fields that set ticket values:

1. Go to the *System Configuration* screen.
2. Search for settings `Forms###AgentFrontend::KnowledgeBaseArticleCreate::Properties` and/or `Forms###AgentFrontend::KnowledgeBaseArticleUpdate::Properties`.
3. Update their `Hidden` values from `1` to `0` as any other dynamic field.

Use Case Example

The following example assumes that all knowledge base article dynamic fields that set ticket values are activated for the *Create Knowledge Base Article* screen and the configuration of each dynamic field is set with valid values for the system.

To create a knowledge base article which will be triggered for the ticket creation:

1. Create a new knowledge base article.
2. Set *2020-09-15 10:00:00* for *Ticket create start time*.
3. Set *2021-09-15 10:00:00* for *Ticket create end time*.
4. Set *Daily* for *Ticket create frequency*.
5. Set *01* for *Ticket create repeat times*.
6. Select *Monday* and *Friday* for *Ticket create repeat on days*.
7. Fill the following fields to define the values for the ticket which will be created:
 - Ticket create customer user
 - Ticket create owner
 - Ticket create responsible
 - Ticket create ticket type
 - Ticket create queue
 - Ticket create priority
 - Ticket create service
 - Ticket create SLA
 - Ticket create state

The screenshot shows the configuration interface for the 'Automated Knowledge Base Article Ticket Creator Settings'. It includes the following fields and values:

- Ticket create start time:** A toggle switch is turned on, and the value is '09/15/2020 - 10:00:00'.
- Ticket create end time:** A toggle switch is turned on, and the value is '09/15/2021 - 10:00:00'.
- * Ticket create frequency:** A dropdown menu is set to 'Daily'.
- * Ticket create repeat times:** A dropdown menu is set to '01'.
- Ticket create repeat on days:** A multi-select field contains 'Friday' and 'Monday'.

Fig. 6: Create Knowledge Base Article

Now every Monday and Friday in the range of 2020-2021 years the OTRS daemon cron job will create a ticket for this knowledge base article.

Warning: Times scheduled during daylight saving time start and daylight saving time end can have unexpected effects. At daylight saving time start it can be skipped and at daylight saving time end it can be executed twice. It is highly not recommended to schedule times when the daylight saving time starts or ends according to the server timezone.

4.3 External Interface

This package has no external interface.

BARAMUNDI INVENTORY CONNECTOR

This package contains functionality to synchronize configuration items from a baramundi Inventory instance with the OTRS CMDB.

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSBaramundiInventoryConnector

Note: This feature requires the *Configuration Management Connector* feature.

5.1 Administrator Interface

This package enables OTRS to perform requests toward a remote *baramundi Inventory* installation. Only unidirectional communication with the controllers `Endpoints`, `SoftwareScanRules` and `EndpointInventorySoftware` is supported. This allows to synchronize computers, software and their links into the OTRS CMDB.

The package provides a console command to manually or automatically trigger the supported controllers in order to keep the systems synchronized.

5.1.1 Processes & Automation

This package adds a new web service and a network transport to the system.

Web Services

A new web service has been created automatically on package installation.

Configuration

To configure the *baramundi Inventory* remote web service:

1. Go to the *Web Service Management* screen and look for a web service named `Baramundi`.
2. Click on the *Baramundi* web service to open for configuration.
3. Click on the *Configure* button to edit the network transport in the section *OTRS as requester* next to the selected network transport `HTTP::RESTBarmundi`.
4. Change the endpoint to match your *baramundi Inventory* server and set correct values for *BasicAuth* user and password.
5. Click on the *Save* or *Save and finish* button.

If you are satisfied with the functionality at a later time it is recommended to change the debug threshold on the web service to *Error*. This considerably reduces logging and potentially improves the web service performance.

Triggering the Remote Web Service Manually

The package provides a console command to trigger the three supported controllers by specifying the requested controller as an argument.

Note: This feature is only available to *On-Premise* customers. If you are a *Managed* customer, this feature is taken care of by the *Customer Solutions Team* in **OTRS**. Please contact us via support@otrs.com or in the [OTRS Portal](#).

The controller `Endpoints` retrieves devices/clients managed by *baramundi* and synchronizes them into the CMDB class `BaramundiCI`.

```
bin/otrs.Console.pl Maint::BaramundiInventory::Trigger Endpoints
```

The controller `SoftwareScanRules` retrieves software managed by *baramundi* and synchronizes them into the CMDB class `BaramundiCISoftware`.

```
bin/otrs.Console.pl Maint::BaramundiInventory::Trigger SoftwareScanRules
```

The controller `EndpointInvSoftware` retrieves associations between devices and software managed by *baramundi* and synchronizes them into OTRS via *DependsOn/RequiredFor* links between the configuration items.

```
bin/otrs.Console.pl Maint::BaramundiInventory::Trigger EndpointInvSoftware
```

Please note that execution is asynchronous and requests might take several minutes to be completed (especially the `EndpointInvSoftware` controller), depending on the amount of data provided by *baramundi Inventory*.

Correct execution can be checked via the generic interface debugger. By default response data exceeding 200 kB is suppressed. If you desire to raise or lower this limit, please change the configuration value for `GenericInterface::Operation::ResponseLoggingMaxSize` in the system configuration.

OTRS prevents simultaneous execution of more than one controller of each type automatically.

General

★ Name: Debug threshold:

Description: Validity:

Remote system:

▶ OTRS as provider

▼ OTRS as requester

In requester mode, OTRS uses web services of remote systems.

Settings

Network transport:

Error Handling Modules

Error handling modules are used to react in case of errors during the communication. Those modules are executed in a specific order, which can be changed by drag and drop.

#	NAME	DESCRIPTION	BACKEND
1	No data found.		

Add error handling module:

Invokers

Invokers prepare data for a request to a remote web service, and process its response data.

NAME	DESCRIPTION	CONTROLLER	INBOUND MAPPING	OUTBOUND MAPPING
BaramundiEndpointInvSoftware	-	Generic::LinkObject	ITSMConfigItem	-
BaramundiEndpoints	-	ITSM::ConfigItem	ITSMConfigItem	-
BaramundiSoftwareScanRules	-	ITSM::ConfigItem	ITSMConfigItem	-

Add Invoker:

Save

or or

Fig. 1: Baramundi Web Service

Network Transport

Properties

Type: HTTP::REST

★ Endpoint:
 URI to indicate specific location for accessing a web service.
 e.g https://www.otrs.com:10745/api/v1.0 (without trailing backslash)

★ Timeout:
 Timeout value for requests.

Authentication:
 An optional authentication mechanism to access the remote system.

★ BasicAuth User:
 The user name to be used to access the remote system.

★ BasicAuth Password:
 The password for the privileged user.

★ Use Proxy Options:
 Show or hide Proxy options to connect to the remote system.

★ Use SSL Options:
 Show or hide SSL options to connect to the remote system.

★ Controller mapping for Invoker
 'BaramundiEndpointInvSoftware': The controller that the invoker should send requests to. Variables marked by a ':' will get replaced by the data value and passed along with the request. (e.g. /Ticket /:TicketID?UserLogin=:UserLogin&Password=:Password).

Valid request command for Invoker
 'BaramundiEndpointInvSoftware': A specific HTTP command to use for the requests with this Invoker (optional).

★ Controller mapping for Invoker
 'BaramundiEndpoints': The controller that the invoker should send requests to. Variables marked by a ':' will get replaced by the data value and passed along with the request. (e.g. /Ticket /:TicketID?UserLogin=:UserLogin&Password=:Password).

Valid request command for Invoker
 'BaramundiEndpoints': A specific HTTP command to use for the requests with this Invoker (optional).

★ Controller mapping for Invoker
 'BaramundiSoftwareScanRules': The controller that the invoker should send requests to. Variables marked by a ':' will get replaced by the data value and passed along with the request. (e.g. /Ticket /:TicketID?UserLogin=:UserLogin&Password=:Password).

Valid request command for Invoker
 'BaramundiSoftwareScanRules': A specific HTTP command to use for the requests with this Invoker (optional).

Default command:
 The default HTTP command to use for the requests.

or or

Triggering the Remote Web Service Automatically

If you would like the controllers to be run periodically in order to keep OTRS synchronized with *baramundi Inventory*, the daemon tasks provided by this package can be enabled in the `Daemon::SchedulerCronTaskManager::Task###Baramundi` system configuration. Please adjust the default frequency of 20 minutes to your requirements.

Warning: High frequencies might affect the system performance.

5.1.2 CMDB Settings

This package adds two new configuration item classes to the CMDB.

Configuration Items

Use this screen to manage class definition of configuration item classes. The configuration item class management screen is available in the *Configuration Items* module of the *CMDB Settings* group.

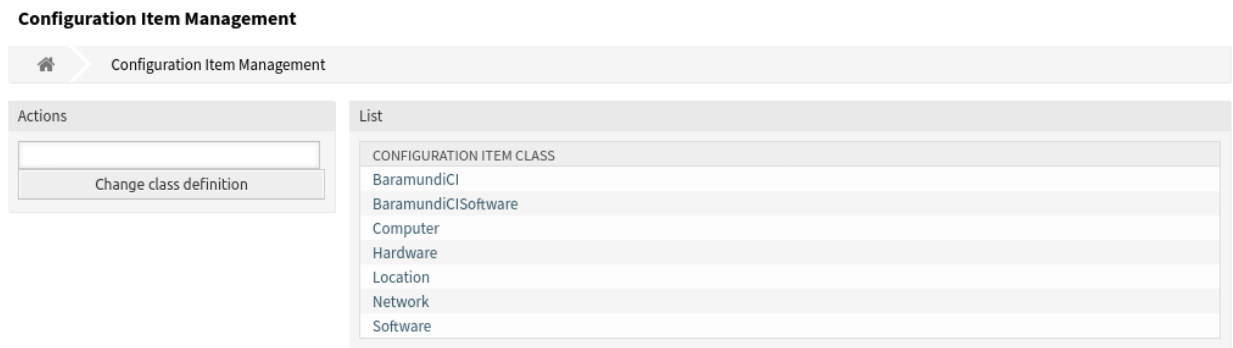


Fig. 3: Configuration Item Management Screen

New Classes

BaramundiCI

Class for the *baramundi Inventory* computers.

BaramundiCISoftware

Class for the *baramundi Inventory* software.

5.2 Agent Interface

This package has no agent interface.

5.3 External Interface

This package has no external interface.

CI ASSIGNMENT ATTRIBUTE DYNAMIC FIELD MAP

This feature makes it possible to link configuration items (CIs) from the OTRS CMDB with services and SLAs, thereby both providing service agents with a better overview and making it possible to offer purely configuration item oriented services. This is especially helpful for companies that provide services for a wide range of different devices or products and that must conform to strict SLAs.

In addition, the feature also makes it possible to map and link configuration item attributes to previously created dynamic fields. If the fields are merely text fields, it is possible to carry out simple mapping in the system configuration. If the configuration item has multiple values that must be displayed in a drop-down dynamic field in the ticket, an extended mapping must be configured.

Benefits

- Better overview for agents thanks to the assignment of configuration items to services and SLAs.
- Enables the provision of configuration item oriented services with numerous configuration item attributes.
- Greater flexibility and transparent linking of configuration item attributes to ticket dynamic fields.
- Optimized visualization of configuration item attributes with multiple values in a drop-down in the ticket.
- Configuration item attributes are readable and searchable.

Target Groups

- Companies that offer services for different devices or products
- Agents
- Internal and external IT
- Facility management

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSCIAssignmentAttributeDynamicFieldMap

Note: This feature requires the *Configuration Management* feature.

6.1 Administrator Interface

This package has no administrator interface, but it allows to set dynamic field values as well as service and SLA of tickets based on linked configuration items.

6.1.1 Assign Service and SLA

It is possible to add definition of two configurable fields in configuration items which contain service and SLA information.

Whenever a configuration item is linked to a ticket or unlinked from a ticket, these fields are used to update or remove service and SLA of the linked ticket.

To add the fields for configuration items:

1. Open the *Configuration Items* module of the *CMDB Settings* group in the administrator interface.
2. Select a configuration item class and click on the *Change class definition* button.
3. Add the new fields to the class definition.

```
- Key: TicketServiceName
Name: Service
Searchable: 1
Input:
  Type: Text
  Size: 50
  MaxLength: 50

- Key: TicketSLAName
Name: SLA
Searchable: 1
Input:
  Type: Text
  Size: 50
  MaxLength: 50
```

Note: The **Key** values must be the same as set in `ITSMConfigItem::ServiceField` and `ITSMConfigItem::SLAField` settings.

After this is configured correctly, just create a configuration item with both fields filled out. If you are going to link this newly created configuration item to a ticket, service and SLA from configuration item attributes are taken into the linked ticket.

Requirements and limitations for service and SLA linking:

- Service functionality must be enabled.
- Configuration item fields for service and SLA must be configured.
- Linked configuration item must contain valid service and SLA names.
- Service and SLA names of configuration item must be allowed for the linked tickets (e.g. not restricted via ACL).
- Linking a configuration item when another configuration item is already linked will update service and SLA again.

Requirements and limitations for service and SLA unlinking:

- Service functionality must be enabled.
- Configuration item fields for service and SLA must be configured.
- Service and SLA of ticket and configuration item must match.
- After unlinking, service and SLA of ticket will be removed, not changed to values before linking.

6.1.2 Assign Dynamic Field Values

It is possible to add definition of a configuration item attribute to dynamic field mapping.

Whenever a configuration item is linked to a ticket, it is checked if the configuration item has attributes defined in this mapping and if so, assigns these values to the dynamic fields of the corresponding ticket. Whenever a configuration item is unlinked from a ticket, the dynamic field value remains unchanged.

To configure the mapping:

1. Go to the *System Configuration* screen.
2. Select *OTRSCIAssignmentAttributeDynamicFieldMap* in the *Navigation* widget.
3. Navigate to *Core* → *OTRSCIAssignmentAttributeDynamicFieldMap* in the navigation tree.
4. Search for setting `ITSMConfigItem::TicketDynamicFieldMapping` and set the mapping between configuration item attributes and ticket dynamic fields.

For example:

```
NIC::IPAddress → IPAddress
NIC::IPoverDHCP → IPoverDHCP
SerialNumber → SerialNumber
WarrantyExpirationDate → WarrantyExpirationDate
```

Note: If you want to use configuration item attributes nested into a deeper structure, add the chain of attribute keys separated by `::`.

5. Search for setting `ITSMConfigItem::TicketDynamicFieldValueMapping` and set the mapping between configuration item attributes and the values of ticket dynamic fields. We have to specify mappings for this to solve a mismatch between possible values of configuration item attributes and possible values of ticket dynamic fields.

For example:

```
NIC::IPoverDHCP → No → 2
                  Yes → 1
```

The numbers define the order in which the values are shown in the drop-down dynamic field. If you would store the values *Yes* and *No* into a dynamic field with the configuration shown above, the system would try to store the value *Yes* or *No* whereas the dynamic field would expect key *1* for *Yes* or key *2* for *No*.

Note: This mapping can also be used for other fields, if the configuration item attribute value does not match a dynamic field key. In this case write the configuration item attribute value into the key part of the hash and the dynamic field key inside the item tag.

After this is configured correctly, just create a configuration item with the mapped fields filled out. If you are going to link this newly created configuration item to a ticket, the ticket dynamic fields are populated with configuration item attributes. The configuration item attributes can be shown in the *Properties* widget of the ticket detail view. The dynamic fields will be automatically updated when the CMDB is modified.

Limitations of the dynamic field assignment functionality:

- Dynamic fields are just filled if they do not have content yet.
- If a configuration item is unlinked from a ticket, the dynamic field values do not get deleted.
- If a configuration item attribute contains a list of values (for example multiple IP addresses of a computer configuration item) only the first value of this list is assigned to a dynamic field.

6.2 Agent Interface

This package has no agent interface.

6.3 External Interface

This package has no external interface.

CI REFERENCES

With this feature it is possible to enhance the configuration items in the configuration management database (CMDB) through additional fields and to create links and references between these or other data in **OTRS**. This can be useful for linking configuration items with each other when there is a dependency relationship between them or to store information regarding linked services and agents directly in the configuration item, as well as find this information with a quicker autocomplete search.

The following input fields with autocomplete search can be added:

- `ReferenceCI` field: a further configuration item.
- `ReferenceService` field: a further service.
- `ReferenceUser` field: a further agent.

In the administrator interface, it is possible to create new reference fields under *Configuration Items*. Then, it is necessary to add the visualization configuration as code for the desired configuration item class. If, for example, a new configuration item is being created in the said class, then the newly added reference field will appear in the form to be filled out. A search for the referenced configuration items using autocomplete can then be carried out, and these can be added. If the new configuration item is updated, an automatic linking to the referenced configuration item is carried out.

Benefits

- Enhancement of configuration items through additional input fields and additional, system-referenced data.
- Linking of configuration items with each other and with other data in **OTRS**.
- Quicker autocomplete search for values referenced in configuration item fields.
- Clear visualization of complex dependencies between configuration items and other configuration items, services, and agents.

Target Groups

- Configuration item managers
- Internal and external IT
- Facility management
- Sales

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- `OTRSCIRferences`

Note: This feature requires the *Configuration Management* feature.

7.1 Administrator Interface

This package has no administrator interface, but it allows to add additional input fields for configuration item classes.

7.1.1 Additional Configuration Item Fields

It is possible to add additional reference fields to refer to different data in **OTRS**, like to other configuration items, services and agents.

To add the fields for configuration items:

1. Open the *Configuration Items* module of the *CMDB Settings* group in the administrator interface.
2. Select a configuration item class and click on the *Change class definition* button.
3. Add the new fields to the class definition.

The following sections describe the possible input fields.

ReferenceCI Field

This field adds an input field with auto-completion feature to search for other configuration items in the configuration item dialog. The following example configuration is needed to insert this kind of field:

```
- Key: testci
Name: Test CI
Searchable: 1
Input:
  Type: ReferenceCI
  Required: 0
  Reference:
    Name: Computer
    LinkType: AlternativeTo
    LinkDirection: Source
    ImportExportKey: Name
```

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Key *

Must be unique and only accept alphabetic and numeric characters. If this is changed, data will not be readable from old definitions.

Name *

The label of the field in the form. Any type of characters can be entered to this field including uppercase letters and spaces.

Note: It is recommended to always use English words for names.

See also:

Names can be translated into other languages with custom language files. For more information, see the [Custom Language File](#) chapter in the administration manual.

Searchable

Defines whether the field is searchable or not. Possible values are *0* or *1*.

Input *

Initiates the definition of the input field.

Type *

Defines the type of the element. Must be placed indented as a logical block. The value is `ReferenceCI` in this case.

Required

Defines whether the field is mandatory or not. Possible values are *0* or *1*.

Reference

Initiates the definition of the reference field.

Name

Defines the class of the configuration item which should be possible to search for.

LinkType

Defines the type of the link which will be created if the value will be saved. Possible values are:

- `DependsOn`
- `AlternativeTo`
- `RelevantTo`
- `Includes`
- `ConnectedTo`

LinkDirection

Defines the direction of the link. Possible values are `Source` and `Target`.

ImportExportKey

Defines the value for the identification of the referenced configuration item. Possible values are `Name`, `Number` or a configured field key.

After a value is set for the input field the value will be used to set a link to the given configuration item. If there is already a value, then the old value will be unlinked. If the reference field in the class definition has been extended with the setting `CountDefault`, several configuration items can also be linked.

For the export and import of this field the name and the configuration item number are used. If a configuration item is not found for the import, then it will be imported 2 times to verify that the linked configuration item which is needed for the link is already imported.

Example export value: `ConfigItemName1`.

ReferenceService Field

This field adds an input field with auto-completion feature to search for services in the configuration item dialog. The following example configuration is needed to insert this kind of field:

```
- Key: testservice
Name: Test Service
Searchable: 1
Input:
  Type: ReferenceService
  Required: 0
  Reference:
    LinkType: AlternativeTo
    LinkDirection: Source
```

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Key *

Must be unique and only accept alphabetic and numeric characters. If this is changed, data will not be readable from old definitions.

Name *

The label of the field in the form. Any type of characters can be entered to this field including uppercase letters and spaces.

Note: It is recommended to always use English words for names.

See also:

Names can be translated into other languages with custom language files. For more information, see the [Custom Language File](#) chapter in the administration manual.

Searchable

Defines whether the field is searchable or not. Possible values are *0* or *1*.

Input *

Initiates the definition of the input field.

Type *

Defines the type of the element. Must be placed indented as a logical block. The value is `ReferenceService` in this case.

Required

Defines whether the field is mandatory or not. Possible values are *0* or *1*.

Reference

Initiates the definition of the reference field.

LinkType

Defines the type of the link which will be created if the value will be saved. Possible values are:

- DependsOn
- AlternativeTo
- RelevantTo
- Includes

- `ConnectedTo`

Additional link types can be defined in the system configuration.

LinkDirection

Defines the direction of the link. Possible values are `Source` and `Target`.

After a value is set for the input field the value will be used to set a link to the given configuration item. If there is already a value, then the old value will be unlinked. If the reference field in the class definition has been extended with the setting `CountDefault`, several configuration items can also be linked.

For the export and import of this field the name of the service is used.

Example export value: *Service 1*.

ReferenceUser Field

This field adds an input field with auto-completion feature to search for agents in the configuration item dialog. The following example configuration is needed to insert this kind of field:

```
- Key: testuser
  Name: Test User
  Searchable: 1
  Input:
    Type: ReferenceUser
    Required: 0
```

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Key *

Must be unique and only accept alphabetic and numeric characters. If this is changed, data will not be readable from old definitions.

Name *

The label of the field in the form. Any type of characters can be entered to this field including uppercase letters and spaces.

Note: It is recommended to always use English words for names.

See also:

Names can be translated into other languages with custom language files. For more information, see the [Custom Language File](#) chapter in the administration manual.

Searchable

Defines whether the field is searchable or not. Possible values are *0* or *1*.

Input *

Initiates the definition of the input field.

Type *

Defines the type of the element. Must be placed indented as a logical block. The value is `ReferenceUser` in this case.

Required

Defines whether the field is mandatory or not. Possible values are *0* or *1*.

For the export and import of this field the login of the agent is used.

Example export value: *root@localhost*.

7.2 Agent Interface

This package has no agent interface, but the configuration item class definitions might be extended with new fields.

Note: In order to grant users access to the *Asset Management* menu, you need to add them as member to the group *itsm-configitem*.

Use the *Add Configuration Item* menu item in the main menu to add new configuration items to the configuration management database.

To add a configuration item:

1. Select a class from the list of classes.
2. Fill in the required fields.
3. Search for a different configuration item, service or agent to set the value for it.
4. Click on the *Save* button.

Now the value is automatically linked to the created configuration item.

See also:

The fields in the *Create Configuration Item* screen can be very different on each classes. The new fields added by this feature is described in the [Administrator Interface](#) chapter.

7.3 External Interface

This package has no external interface.

CIS IN CUSTOMER FRONTEND

This feature uses the customer ID attribute of your configuration item classes and makes configuration items visible in the external interface. The customer user has read-only access to the following attributes:

- ID of their configuration item
- Name of the configuration item
- Class of the configuration item
- Deployment state
- Current incident state
- Date and time of the last update

On mail-in this feature automatically links affected configuration items based on the given configuration item ID found in the body of the email. It is also helpful for your service desk as, on ticket creation, they only have access to the configuration items of the requester's organization or department. This eases the selection of affected configuration items a lot, especially if you have many configuration items in your CMDB.

Benefits

- Automatic assignment of configuration items saves time.
- Customers can assign configuration items to new tickets.

Target Groups

- Customer service organizations
- External IT service providers
- Logistics
- Technical field service

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSCIsInCustomerFrontend

Note: This feature requires the *Configuration Management* feature.

8.1 Administrator Interface

This package has no administrator interface.

8.1.1 Extend Class Definition

The definition of a configuration item class has to be altered to make it able to be displayed in the external interface.

By default, the `CustomerID` field is configured in the system configuration to store the customer ID information in the configuration item definition.

You can use more than one `CustomerID` per configuration item, to make one configuration item accessible by more than just one customer. To do so, repeat the following steps for the maximum amount of customers one configuration item may grant access to (for example: `PartnerA`, `PartnerB`, `PartnerC`, etc.).

See also:

To use more than one `CustomerID` or use another name than `CustomerID`, it is necessary to change the setting `ITSMConfigItem::CustomerIDField` in system configuration or add additional entries for each entry field.

If your class definition does not contain the `CustomerID` attribute, then you have to add it manually.

To add the fields for configuration items:

1. Open the *Configuration Items* module of the *CMDB Settings* group in the administrator interface.
2. Select a configuration item class and click on the *Change class definition* button.
3. Add the new fields to the class definition. The input type could be either `Text` or `CustomerCompany`.

Text Field

Text based field gives the flexibility to use any string as value to match the `CustomerID` for one or more customers. The value must be entered manually by editing each configuration item, but it needs to be done carefully, because any mismatch will prevent the configuration item to be displayed in the external interface.

```
- Key: CustomerID
Name: Customer Company
Searchable: 1
Input:
  Type: Text
  Size: 50
  MaxLength: 100
```

CustomerCompany Field

A customer company field needs to have correctly configured customer companies in the system, as it will be presented as a drop-down list in the configuration item add and edit screens. The source of the drop-down will be the list of companies. `CustomerID` field in all customer users must point to the correct customer ID in customer companies administration.

```
- Key: CustomerID
Name: Customer Company
Searchable: 1
Input:
  Type: CustomerCompany
```

4. Save the new definition.
5. Edit a configuration item from the modified class. Find the *Customer Company* field (or another field you have added) and fill it with the customer ID of an existing customer.
6. Login to the external interface with any customer user that has the customer ID described above.
7. Add the endpoint to *External Interface*.
8. Go to *Company Configuration Items*. The edited configuration item must be listed.

8.1.2 Define Strictness of Customer ID Restriction

The setting `ITSMConfigItem::CustomerCIPermissionByLink` is set to 0 by default, so configuration items are only accessible in the external interface if the company (customer ID) of the customer user matches the value of a configured field. If this behavior is more strict than desired or not all configuration items can/should be configured in such a way, the configuration can be changed to consider links between configuration items (permission inheritance by links). When linking configuration items to new tickets in the external interface and in the agent interface (if enabled), only direct permission is considered though (i. e. `CustomerID` field in configuration item matches).

For example:

- There are multiple computer configuration items assigned to the customer company. All computer configuration items are linked to at least one network configuration item (switches). These devices do not belong to the customer and are therefore not visible. Also the network configuration items are linked to other network configuration items (router).
- By default, only the companies computers are visible under *Company Configuration Items*, are shown in linked tickets and (depending on the configuration) can be viewed in detail and used for new tickets.
- If the setting `ITSMConfigItem::CustomerCIPermissionByLink` is set to 1, all switches connected to a computer will be visible under *Company Configuration Items*, are shown in linked tickets and can be viewed in detail, but not used for to be linked to new tickets.
- If the setting `ITSMConfigItem::CustomerCIPermissionByLink` is set to 2, the routers will be included as well.

Warning: The link type and direction is not relevant for determining the permission. Therefore please carefully consider which value to use for `ITSMConfigItem::CustomerCIPermissionByLink` in order to prevent unwanted disclosure of configuration items.

8.1.3 Hide Configuration Item Fields in External Interface

It could be possible that configuration items has fields that customer does not need to view, or that contains sensitive information that customers must not know. For these cases an administrator can restrict a field by placing a simple new attribute `NotForCustomer` to the field definition on a particular class.

To hide a field in external interface:

1. Open the *Configuration Items* module of the *CMDB Settings* group in the administrator interface.
2. Select a configuration item class and click on the *Change class definition* button.
3. Add the attribute `NotForCustomer: 1` to the field definition.

For example:

```
- Key: Vendor
Name: Vendor
Searchable: 1
Input:
  Type: Text
  Size: 50
  MaxLength: 50
  NotForCustomer: 1
```

4. Edit the configuration items of this class in order to create a new version, so the new version will take the new definition.

To avoid the need to create new versions for configuration items if the definition of a class is updated to hide a field from external interface, it will be applied to all configuration items of that class. On the other hand, if the class definition is updated to show a field that was previously hidden and the configuration item was already updated to the class definition where the field was hidden, the field will not be shown until the configuration item is updated to last class definition where the field is set to be shown again.

The intention of this behavior is to enforce the privacy of the data that should not be displayed in the external interface.

8.2 Agent Interface

This package has no agent interface.

8.2.1 Restriction for Linking

This feature restricts the link interface for the agent when linking a ticket with a configuration item. Only configuration items and the tickets which belong to the same company of the customer user can be selected. The feature can be disabled in the system configuration via the setting `ITSMConfigItem::RestrictAgentLinking`.

8.2.2 Postmaster Filter

There is a postmaster filter added to search incoming emails for a configuration item identifier (usually the configuration item number) and links all found configuration items with the new ticket that was created from this email. The email body and the subject are searched, all found configuration item numbers will be linked with the ticket. This feature is deactivated by default and needs to be enabled in the system configuration via the setting `PostMaster::PostFilterModule###100-ITSMConfigItemLink`.

8.3 External Interface

This feature makes it possible for a customer user to see and select the configuration items of his company in the external interface to link them to new tickets.

8.3.1 Link Configuration Item With Ticket

In the *New Ticket* screen the customer user can search and select the configuration items that are assigned to his company. When the ticket is created, the selected configuration items will be automatically linked to the new ticket.

To link a configuration item to a new ticket:

1. Create a new ticket in the external interface.
2. Fill in the required fields.
3. Click on the *Add affected configuration items* button at the end of the ticket form. A new set of fields will appear below.
4. In the new section fill any search criteria for class, name or number.
 - *Class* drop-down list only shows classes that are already prepared and contain configuration items assigned to a `CustomerID` (for more information take a look at the [Administrator Interface](#)).
 - *Name* field matches any configuration item which name starts with the content of the field. For example, if *ca* is typed in the field, it will match *cat*, *car*, *card*, etc.
 - *ConfigItem#* field matches any configuration item which number starts with the content of the field. For example, if *12* is typed in the field, it will match *123*, *1211*, *1298*, etc.

Note: These search fields will only reduce the search results in order to have smaller lists to be able to select faster. Leaving them in blank will bring the full list of configuration items assigned to the company of the current customer user.

5. Select the configuration items from the list.
6. Click on the *Create Ticket* button.

8.3.2 Company Configuration Items

A new screen *Company Configuration Items* shows a list of all configuration items that belong to the same company as the customer user. Belonging to the same company means, that the configuration items must have a `CustomerID` field that has the same value as the customer user. In the external ticket detail view all linked configuration items that belong to the company of the customer user are shown as well.

If desired, the required association of configuration items to the company of the customer user can be decreased by setting a maximum level to check linked configuration items. If this feature is enabled, configuration items are also accessible to customer users if they do not belong to the company directly, but a linked configuration item is (up to the configured number of links away).

The configuration item details can be access from the configuration items list. This details screen shows the last version of the configuration item, links to other configuration items (if they belong to the same company).

See also:

This screen is not added to any menu of the external interface by default. To create a link to the *Company Configuration Items* screen, an administrator needs to add `/itsmconfigitem/overview` to the *Link* input field and add a name in the *Name* field in one of the following settings of the system configuration:

- `ExternalFrontend::Menu###Top`
- `ExternalFrontend::Menu###Main`
- `ExternalFrontend::Menu###Bottom`

Home > Company Configuration Items

Company Configuration Items

Computer

Incident State	Deployment State	ConfigItem#	Name	Class	Last changed
Operational	Production	1022000001	Desktop PC	Computer	11/02/2020 12:48:55

Fig. 1: Company Configuration Items Screen

See also:

The visible columns can be defined in the following setting:

- `ExternalFrontend::ITSMConfigItemOverview###ShowColumns`

If multiple classes of configuration items are available in the external interface, an *All* filter is displayed to see all classes. The *Class* column is always visible, when the filter *All* is selected.

Clicking on a configuration item will show the detail view.

Home > Company Configuration Items

Configuration Item: Desktop PC

Configuration Item Details

Customer Company	Super Support Inc.
Owner	"Lacey Green" <lacey@example.org>
CPU	
Ram	
Hard Disk	
↳ Capacity	
Network Adapter	NIC
↳ IP over DHCP	Yes

Configuration Item Information

Class:
Computer

Number:
1022000001

Name:
Desktop PC

Current Deployment State:
[Production](#)

Current Incident State:
[Operational](#)

Created:
11/02/2020 12:48:55

Created:
11/02/2020 12:48:55

Fig. 2: Configuration Item Details Screen

CONFIGURATION MANAGEMENT CONNECTOR

This package contains the feature that provides ITSM functionality for the generic interface. It allows to add invokers to request configuration items from remote systems and sync them to the OTRS database.

Additionally this package contains an invoker to sync links between objects from a remote system to the local link database. It is also possible to synchronize configuration items between two OTRS instances using a special synchronization invoker.

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSConfigurationManagementConnector

Note: This feature requires the *Configuration Management* feature.

9.1 Administrator Interface

This package adds a new settings screen to the *Web Service Management* screen to configure the invokers.

9.1.1 Processes & Automation

This package adds three new invokers to *Invokers* section of the *Web Service Management* screen.

Web Services

After installation of the package, three new invokers will be available in the *Invokers* section. Selecting an invoker from the drop-down list will open a new settings window.

Configuration Item Invoker

This package provides the functionality to request a list of configuration items from a remote system with the generic interface. A complete mapping between the name of remote value keys to the local configuration item class definitions is possible. In order to give the possibility to define all class attributes in the mapping, an advanced mapping is provided.

To use this invoker:

1. Go to the *Web Service management* screen.
2. Add a new web service or select an existing one.
3. Select the `ITSM::ConfigItem` invoker in the *Invokers* section.

For handling remote configuration items and links, an advanced mapping is provided with this package. To use this functionality select the `ITSMConfigItem` in the mapping select box.

Invoker Details

★ Name:
 The name is typically used to call up an operation of a remote web service.

Description:

Invoker backend:
 This OTRS invoker backend module will be called to prepare the data to be sent to the remote system, and to process its response data.

Mapping for outgoing request data:
 The data from the invoker of OTRS will be processed by this mapping, to transform it to the kind of data the remote system expects.

Mapping for incoming response data:
 The response data will be processed by this mapping, to transform it to the kind of data the invoker of OTRS expects.

or

Fig. 1: Invoker Details for `ITSM::ConfigItem`

One of those advanced mapping features are static values. These make it possible to define static values for defined keys, for example setting the configuration item class for each item. These entries can overwrite data send from the remote system.

Another feature provided by the advanced mapping are transforming strings to list structures. For creating a list out of a string, a list separator can be defined, for example `;`. The separator field takes a regular expressions which make a more complex separators like `;(?:\s+)?` (a `;` optionally followed by multiple white spaces) possible.

It is possible to define where the index of the elements should take place in the configuration item structure. For this the placeholder `###INDEX###` has to be placed in the key mapping. The configuration in the screenshot would store the separated IP address list into a new interface each. If no index is defined in the mapping, the index will increase the main attribute counter, like a suffix.

The configuration item `Number` attribute is used to make the logical connection between the remote data

Mapping ITSMConfigItem

Default rule for unmapped keys:
 This rule will apply for all keys with no mapping rule.

Default rule for unmapped values:
 This rule will apply for all values with no mapping rule.

New static value:

★ Key: ★ Value:

New key map:

or or

Fig. 2: Static Mapping for ITSM::ConfigItem

▼ Mapping for Key Interface::###INDEX###::IPAddress

Key mapping: ★ Map key: matching the: ★ to new
 key:

List separator:

Value mapping:

New value map:

Fig. 3: Key Mapping for ITSM::ConfigItem

and the data stored in the OTRS database. To create and update configuration items with this invoker values for the following keys are required:

```
- Name
- Class
- Number
- DeploymentState
- IncidentState
```

The values for these keys (except `Number`) can be static or provided by the remote system.

Generic Link Object Invoker

Additionally to the configuration item functionality, this package provides the functionality to link objects, like ticket to configuration items or configuration items to configuration items together. The invoker can get used to pull new links from a remote system or keep the provided links in sync and delete removed links from the OTRS database. There are no limits in linking different objects.

To use this invoker:

1. Go to the *Web Service management* screen.
2. Add a new web service or select an existing one.
3. Select the `Generic::LinkObject` invoker in the *Invokers* section.

For linking different objects together the following mapping keys need values:

```
- SourceClass
- SourceObject
- TargetClass
- TargetObject
- Type
```

The OTRS internal object ID can be provided directly with the key `SourceKey` and `TargetKey` or can get looked up by providing the OTRS object number with the keys `SourceNumber` and `TargetNumber`.

To keep the OTRS in synchronized with the remote system it is possible to select different link combinations, e.g. parent-child ticket to ticket, to get synchronized. This means if a remote link combination was removed the local link will get removed, too.

The synchronization of configuration item classes can be limited by selection the classes that should get synchronized from the list in the invoker administration view.

Configuration Item Synchronization Invoker

This package provides the functionality to synchronize configuration items between two OTRS instances using a special synchronization invoker. That invoker allows to define invokers for create, update and delete synchronizations, that gets called automatically if needed.

To synchronize configuration items between two systems, it is needed to add a web service with an invoker of type `ITSM::ConfigItemSync`. This invoker is the basic (search) invoker that is used to perform ITSM object searches on the remote system to determine configuration items to be modified.

To use this invoker:

1. Go to the *Web Service management* screen.

Mapping ITSMConfigItem

Default rule for unmapped keys:
 This rule will apply for all keys with no mapping rule.

Default rule for unmapped values:
 This rule will apply for all values with no mapping rule.

New static value:

- ★ Key: ★ Value:
- ★ Key: ★ Value:
- ★ Key: ★ Value:
- ★ Key: ★ Value:
- ★ Key: ★ Value:

New key map:

▼ Mapping for Key SourceNumber

Key mapping: ★ Map key: matching the: ★ to new
 key:

List separator:

Value mapping:

New value map:

▼ Mapping for Key TargetNumber

Key mapping: ★ Map key: matching the: ★ to new
 key:

List separator:

Value mapping:

New value map:

or or

Invoker Details

★ Name:
 The name is typically used to call up an operation of a remote web service.

Description:

Invoker backend:
 This OTRS invoker backend module will be called to prepare the data to be sent to the remote system, and to process its response data.

Synchronize links:

Limit CI class sync to:

Mapping for outgoing request data:
 The data from the invoker of OTRS will be processed by this mapping, to transform it to the kind of data the remote system expects.

Fig. 5: Mapping Synchronization

2. Add a new web service or select an existing one.
3. Select the `ITSM::ConfigItemSync` invoker in the *Invokers* section.

It works with two steps on every sync action:

1. At first depending on the change that was made in the local system (add, update or delete a configuration item) the invoker performs a related search on the remote system to determine, if it is needed to create, update or delete a remote configuration item.
2. On the second step, a related invoker for create actions, update actions or delete actions is called, which executes the main changes on the remote system. Those invokers have to be created as well and needs to be of invoker type `ITSM::ConfigItem` to make sure the communication works properly.

See the enclosed overview of example invokers:

Within the search invoker configuration, the worker invokers to perform the changes needs to be configured separately for every action in the related drop-down menus.

If an invoker is not configured for the related action, it will be omitted. The configured invoker will be listed within every drop-down and can be selected easily for the different actions.

The different invokers needs to map their data. For this to work properly, it is recommend to use the mapping module `ITSMConfigItem`, which will be described through this documentation, but in fact it is also possible to use other types of mappings like XSLT if configured properly.

Since this structure calls remote systems, it is needed to configure at least the connection credentials to access the remote CMDB.

All other configurations can be set optional, but they are not needed for the basic feature to work.

To be sure the search invoker just listens on specific events, this package extends the possibilities of the Invoker event filter feature from OTRS, that can be used to add additional conditions to the event triggers, which are configured for the search invoker. One example could be, that just configuration items of ITSM class `Computer` gets synchronized to the remote system.

The configuration of the remote system, that gets synchronized does not need to be special but for it needs to provide at least an operation for the search executions and a separate operation for each sync action that

Invokers

Invokers prepare data for a request to a remote web service, and process its response data.

NAME	DESCRIPTION	CONTROLLER	INBOUND MAPPING	OUTBOUND MAPPING
ConfigItemCreate	Invoker to create remote configuration items (called automatically).	ITSM::ConfigItem	-	-
ConfigItemDelete	Invoker to remove remote configuration items (called automatically).	ITSM::ConfigItem	-	-
ConfigItemSearch	Invoker to search remote configuration items based on local configuration item numbers and perform remote actions.	ITSM::ConfigItemSync	-	-
ConfigItemUpdate	Invoker to update remote configuration items (called automatically).	ITSM::ConfigItem	-	-

Fig. 6: Invoker Overview

needs to be performed.

9.2 Agent Interface

This package has no agent interface.

9.3 External Interface

This package has no external interface.

Invoker Details

★ Name:

The name is typically used to call up an operation of a remote web service.

Description:

Invoker backend:

This OTRS invoker backend module will be called to prepare the data to be sent to the remote system, and to process its response data.

Mapping for outgoing request data:

The data from the invoker of OTRS will be processed by this mapping, to transform it to the kind of data the remote system expects.

Mapping for incoming response data:

The response data will be processed by this mapping, to transform it to the kind of data the invoker of OTRS expects.

Invoker Create:

This invoker backend module will be called on create synchronizations

Invoker Update:

This invoker backend module will be called on update synchronizations

Invoker Delete:

This invoker backend module will be called on delete synchronizations

Event Triggers:

EVENT	ASYNCHRONOUS	CONDITION	EDIT	DELETE
No data found.				

This invoker will be triggered by the configured events.

Add Event Trigger:

Asynchronous

To add a new event select the event object and event name and click on the "+" button.
 Asynchronous event triggers are handled by the OTRS Scheduler Daemon in background (recommended).
 Synchronous event triggers would be processed directly during the web request.

or or

Fig. 7: Search Invoker Details

Mapping ITSMConfigItem

Default rule for unmapped keys:
 This rule will apply for all keys with no mapping rule.

Default rule for unmapped values:
 This rule will apply for all values with no mapping rule.

New static value:

★ Key: ★ Value:

★ Key: ★ Value:

New key map:

or or

Fig. 8: Search Invoker Mapping

Operations

Operations are individual system functions which remote systems can request.

NAME	DESCRIPTION	CONTROLLER	INBOUND MAPPING	OUTBOUND MAPPING
ConfigItemCreate	-	ConfigItem::ConfigItemCreate	ITSMConfigItem	ITSMConfigItem
ConfigItemDelete	-	ConfigItem::ConfigItemDelete	ITSMConfigItem	ITSMConfigItem
ConfigItemSearch	-	ConfigItem::ConfigItemSearch	ITSMConfigItem	ITSMConfigItem
ConfigItemUpdate	-	ConfigItem::ConfigItemUpdate	ITSMConfigItem	ITSMConfigItem

Fig. 9: Operation Overview

CUSTOMER-SPECIFIC SERVICES

This feature makes it possible to assign services to customers so that only the assigned service is displayed when a ticket is created and only the appropriate SLAs are valid.

After installation of the package, a new module *Customers Service* appears in the *Users, Groups & Roles* group in the administrator interface. Here you can define which services should be assigned to customers.

Furthermore, you can define in the system configuration which of these options should be the leading one.

Benefits

- Efficient and well-structured administration of multiple services and SLAs for many customers.
- Avoid the creation of unwarranted tickets that stress out your service team.
- Optimal guidance for the customer regarding for which services he or she can open a ticket.
- Speed-up of ticket handling process.

Target Groups

- Internal IT departments
- External IT service providers
- Call centers
- Agencies
- Consulting businesses
- Companies offering a wide range of services or products

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSCustomerSpecificServices

10.1 Administrator Interface

After installation of the package a new module will be available in the *Users, Groups & Roles* group of the administrator interface.

10.1.1 Users, Groups & Roles

After installation of the package, a new module *Customers Service* appears in the *Users, Groups & Roles* group in the administrator interface. Here you can define which services should be assigned to customers.

Customers Services

Use this screen to add one or more customers to one or more services. To use this function, at least one customer and one service need to have been added to the system. The management screen is available in the *Customers Services* module of the *Users, Groups & Roles* group.

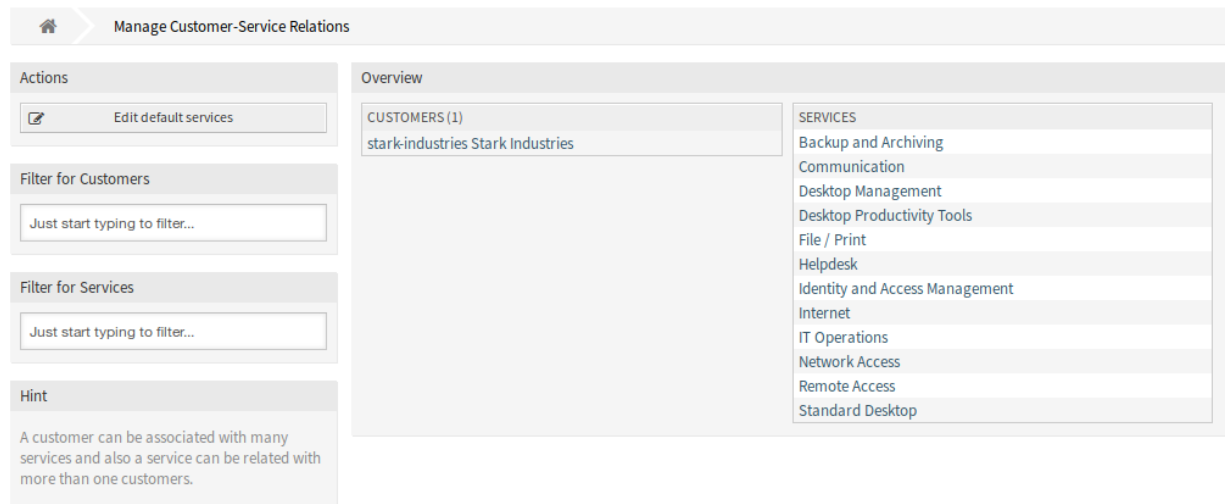


Fig. 1: Manage Customer-Service Relations

Manage Customers Services Relations

Note: Before your first use, please make sure that you activate the *Services* option in the administrator interface.

Note: The determination of the customers is based on a customer company search. So it is very important to adjust the `CustomerCompanySearchListLimit` setting in the `CustomerCompany` section in the `Kernel/Config.pm`. Otherwise not all customers can be found and displayed.

Note: This feature can be affected if the setting `CustomerUserExcludePrimaryCustomerID` is set to 1 in the `Kernel/Config.pm`. By excluding the primary customer ID and not setting up customer IDs manually, no services will be shown.

To allocate some services to a customer:

1. Click on a customer in the *Customers* column.
2. Select the services you would like to allocate to the customer.
3. Click on the *Save* or *Save and finish* button.

Allocate Services to Customer stark-industries

SERVICE	<input type="checkbox"/> ACTIVE
Backup and Archiving	<input checked="" type="checkbox"/>
Communication	<input type="checkbox"/>
Desktop Management	<input type="checkbox"/>
Desktop Productivity Tools	<input type="checkbox"/>
File / Print	<input checked="" type="checkbox"/>
Helpdesk	<input type="checkbox"/>
Identity and Access Management	<input type="checkbox"/>
Internet	<input checked="" type="checkbox"/>
IT Operations	<input type="checkbox"/>
Network Access	<input type="checkbox"/>
Remote Access	<input type="checkbox"/>
Standard Desktop	<input type="checkbox"/>

or or

Fig. 2: Change Service Relations for Customer

To allocate some customer to a service:

1. Click on a service in the *Services* column.
2. Select the customers you would like to allocate to the service.
3. Click on the *Save* or *Save and finish* button.

Note: If several customers or services are added to the system, use the search box to find a particular customer or use the filter box to find a particular service by just typing the name to filter.

Multiple customers or services can be assigned in both screens at the same time. Additionally clicking on a customer or clicking on a service in the relations screen will open the *Edit Customer* screen or the *Edit Service* screen accordingly.

Allocate Customers to Service Backup and Archiving

CUSTOMER ID (1)	<input checked="" type="checkbox"/> ACTIVE
stark-industries Stark Industries	<input checked="" type="checkbox"/>

or or

Fig. 3: Change Customer Relations for Service

Warning: Accessing a customer or a service provides no back link to the relations screen.

Note: By setting a checkbox in the header of a column will set all the checkboxes in the selected column.

Note: The allocations of customer users and services should be done with this screen. It is not recommended to use the screen *Customer Users Services*.

Manage Default Services

It is possible to add default services, so that all customer users of the customer may access them. This prevents having to add each service to each customer.

To edit the default services:

1. Click on the *Edit default services* button in the left sidebar.
2. Select the services which should be selectable for all customer.
3. Click on the *Save* or *Save and finish* button.

Warning: Mixing default services and customer specific services can be confusing. If a customer has specific services assigned, then the default services will be not applied.

10.2 Agent Interface

After installation of the package a new widget will be available in the customer detail view.

When selecting a customer user in any new ticket screens, only the services related to the customer of that customer user will be available.

Manage Customer-Service Relations Allocate Services to Customer ^-DEFAULT>

Actions
Go to overview

Filter for Services
Just start typing to filter...

Hint
A customer can be associated with many services and also a service can be related with more than one customers.

Allocate Services to Customer

SERVICE	ACTIVE
Backup and Archiving	<input type="checkbox"/>
Communication	<input type="checkbox"/>
Desktop Management	<input type="checkbox"/>
Desktop Productivity Tools	<input type="checkbox"/>
File / Print	<input type="checkbox"/>
Helpdesk	<input type="checkbox"/>
Identity and Access Management	<input type="checkbox"/>
Internet	<input type="checkbox"/>
IT Operations	<input type="checkbox"/>
Network Access	<input type="checkbox"/>
Remote Access	<input type="checkbox"/>
Standard Desktop	<input type="checkbox"/>

Save or Save and finish or Cancel

Fig. 4: Allocate Services to Customer Screen

10.2.1 Customers

After installation of the package, a new widget *Services* appears in the customer detail view. In this widget all assigned services and the amount of open and closed tickets are displayed.


Services (3 Services) 			
Service ^	Open	Closed	Total
Backups and Archiving	1	0	1
File / Print	0	0	0
Internet	0	0	0

Fig. 5: Services Widget

10.3 External Interface

This package has no external interface, but the service selection can be restricted to the configured ones.

10.3.1 Restrict Service Selection

See also:

The `ExternalFrontend::TicketCreate###Service` setting needs to be enabled to use this feature.

In the *New Ticket* screen only the services which are related to the customer of the customer user will be available. After the service is chosen by the customer user, the associated service level agreements (SLA) will become available, if they are configured.

DATA PRIVACY PROTECTION

This feature takes care about the data privacy of specific information within **OTRS**. Based on customizable rules, different actions can be performed on different types of data.

Warning: The use of this feature can result in a complete loss of data! We recommend a backup of the database **before** execution and if possible a test on a test system.

Warning: The use of this feature is at your own risk. *OTRS AG* does not take any responsibility for lost data.

Benefits

- Compliance with data protection guidelines.

Target Groups

- IT service management
- Human resources
- Internal IT
- Government
- IT service
- Security management

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSDataPrivacyProtection

11.1 Administrator Interface

This package has no administrator interface. All data privacy protection rules must be added via the system configuration and executed by a console command.

Note: If you are a *Managed* customer, this feature is taken care of by the *Customer Solutions Team* in OTRS. Please contact us via support@otrs.com or in the [OTRS Portal](#).

11.1.1 System Configuration Settings

For security reasons, this package does not come with any pre-configured rules. Therefore the functionality does not work out-of-the-box and the rules must be configured by an administrator first.

To add a rule configuration:

1. Go to the *System Configuration* screen of the administrator interface.
2. Select *OTRSDataPrivacyProtection* in the *Navigation* widget.
3. Navigate to *Core* → *OTRSDataPrivacyProtection* in the navigation tree.
4. Add a rule configuration in YAML format to setting `OTRSDataPrivacyProtection::RuleConfiguration`.

The configuration of the individual rule sets are stored in YAML format and consists of the five options. The options marked with an asterisk are mandatory.

RuleName *

This string option has to be unique for all rules. It is a string, that is used to identify every different rule. The rule name appears in all related outputs and history entries, but does not affect any functionality.

RuleSource

This string option is just for informational usage. Different countries and regions have different types and kinds of data protection regulations, that might be part of laws and/or written papers. To identify the related original sources, the different configured rules are based on. `RuleSource` can be used to add names and/or descriptions, that will be added to related history entries (if available).

RuleType *

This string option describes the action, that will be executed on all objects, that were found during a search. The following actions are supported (in short format or in long format):

Anonymization Of PrivacyByAnonymization

The anonymization rule type is used to anonymize data sets, as identified in the data classification fields. Anonymization means, that the different fields will be replaced by a string *Anonymized*.

Warning: During anonymizations, the original data sets are replaced by the mentioned string in the database. The original data is not saved and is therefore irretrievably lost!

Pseudonymization Of PrivacyByPseudonymization

Like the anonymization, the pseudonymization rule type is used to remove the related data from their original fields. There is a major difference to the `Anonymization` rule type, as the data will be stored in a separated database table, called `data_pseudonymization`.

This table is not used by any other sub-system, and it is not usable via the GUI. It acts as a backup table, that can be (manually) searched by any administrator, that has access to the database or using the *SQL Box* module of the administrator interface.

During pseudonymizations, a universal unique identifier (UUID) will be created for the related data field, which is used to identify the original data later. The original data will then be copied to the backup table, using the UUID as the field identifier. Afterwards the original data will be replaced by just the UUID, which is effectively the same as an anonymization, but including a pointer to the stored original source data.

Deletion Or PrivacyByDeletion

The deletion rule type is used to delete data sets, as identified in the data classification fields. Deletion means, that the different fields will be replaced by a string *Deleted*. Technically the original data will be deleted, as it will be replaced with a non-sensitive string, it therefore works equal to the *Anonymization* rule type.

Warning: During deletions, the original data sets are replaced by the mentioned string in the database. The original data is not saved and is therefore irretrievably lost!

DataClassification *

This list option is used to identify the data types for which the associated actions are to be applied. It contains the different fields of any data object as an array. Every object drivers provides a list of possible data classification fields, that might be used.

See also:

The specific fields are described in the *Drivers* section below.

ObjectFilter *

This list option implements the search and filter criteria for every used driver. Every driver provides a list of possible search and filter options, that are allowed to be used.

See also:

Please see the *Drivers* section below for more detailed information.

The different types of information represented by objects (e.g. ticket, customer user, etc.) are called *object types* within this package. Therefore we are talking about *object types*.

The modules implemented for the specific data processing, search functions and verification of such specific object types are called *drivers* or *driver objects*.

11.1.2 Rule Execution

Once the rules have been defined, they can be applied to the existing data sets. There is a console command `Maint::DataPrivacy::Execute` for this.

Execute the console command with the `--help` option for more information:

```
otrs> /opt/otrs/bin/otrs.Console.pl Maint::DataPrivacy::Execute --help
```

This command essentially offers three different options:

- Checking the integrity and validity of the existing rules.
- Test execution of existing rules without changing data records.
- Execution of the existing rules, whereby the matching data sets are permanently changed.

The validation checks all available rules in the context of the affected drivers and object types. If certain options are missing or incorrect, the rule is declared invalid and execution is skipped for all drivers.

For security reasons, the validity of the corresponding rules is implicitly checked both before each dry run and before each execution and is either completely stopped or skipped in case of errors.

Warning: We recommend, that new rules or significant changes should be executed on test systems first to ensure that no data is accidentally changed or deleted.

Warning: We recommend that you back up the database first to ensure that untested data is not lost after rules or rule changes are executed.

Warning: Since rules are designed to change or completely delete data, it is very important to carefully check all rules in advance and execute the test runs for each rule change.

The console output of rule executions can be redirected into files to preserve the modified objects. Please see the following example:

```
otrs> /opt/otrs/bin/otrs.Console.pl Maint::DataPrivacy::Execute --execute-detail >␣  
↪rule-execution.txt
```

11.1.3 Drivers

This section describe the configuration and usage of the various drivers. In addition, this section contains sample configurations that can be copied and customized to your personal needs.

Customer Company Driver

The customer company driver provides the functionality to search and modify the information for customer companies.

Possible data classifications:

```
- CustomerID  
- CustomerCompanyName  
- CustomerCompanyCountry  
- CustomerCompanyStreet  
- CustomerCompanyZIP  
- CustomerCompanyCity  
- CustomerCompanyURL  
- CustomerCompanyComment  
- DynamicField_NameX
```

The driver supports dynamic fields for data classification. Dynamic fields will be identified by the prefix `DynamicField_` and the related field name.

Possible object filters:


```

- ValidID
- CustomerID
- CustomerCompanyStreet
- CustomerCompanyURL
- CustomerCompanyComment
- WildcardSearch
- CustomerCompanyZIP
- CustomerCompanyCountry
- CustomerCompanyName
- CustomerCompanyCity

```

Object filter descriptions:

- **Limit:** Limits the number of search results.
- **CreateTime:** Searches for dates *greater than or equal to* (>=) the given time.
- **WildcardSearch:** Affects all object filters, except `ValidID`. It will wrap the search term in wildcards, so they match all objects that contain the supplied value. For example, if this option is set to 1, filtering on `CustomerID` with `company` will in fact search for `*company*`.

Rule Configuration Examples

Here are some examples for rule configurations. These examples are valid YAML codes. You can copy these examples and modify them according to your needs.

Delete customer company name and customer company country by customer company name without wildcard search:

```

---
RuleName: Delete customer company name and customer company country by customer_
↳company name without wildcard search.
RuleType: PrivacyByDeletion
RuleSource: GDPR
DataClassification:
  CustomerCompany:
    - CustomerCompanyName
    - CustomerCompanyCountry
ObjectFilter:
  CustomerCompany:
    CustomerCompanyName: someCompanyName
    WildcardSearch: 0

```

Delete customer company name and customer company country by customer company name with wildcard search:

```

---
RuleName: Delete customer company name and customer company country by customer_
↳company name with wildcard search.
RuleSource: someRuleSource
RuleType: PrivacyByDeletion
DataClassification:
  CustomerCompany:
    - CustomerCompanyName
    - CustomerCompanyCountry
ObjectFilter:

```

(continues on next page)

(continued from previous page)

```
CustomerCompany:  
  CustomerCompanyName: someCompanyName  
  WildcardSearch: 1
```

Customer User Driver

The customer user driver provides the functionality to search and modify the information for customer users.

Possible data classifications:

```
- UserTitle  
- UserFirstname  
- UserLastname  
- UserEmail  
- UserLogin  
- UserComment  
- UserCountry  
- UserFax  
- UserMobile  
- UserCity  
- UserPhone  
- UserTitle  
- UserStreet  
- UserZip  
- DynamicField_NameX
```

The driver supports dynamic fields for data classification. Dynamic fields will be identified by the prefix `DynamicField_` and the related field name.

Possible object filters:

```
- UserCity  
- UserTitle  
- UserFirstname  
- UserPhone  
- ValidID  
- UserCountry  
- UserLogin  
- UserCustomerID  
- UserLastname  
- UserZip  
- UserMobile  
- UserEmail  
- UserFax  
- WildcardSearch  
- UserStreet  
- UserComment
```

Object filter descriptions:

- **Limit:** Limits the number of search results.
- **CreateTime:** Searches for dates *greater than or equal to* (\geq) the given time.
- **Valid:** Searches for valid or invalid users. Possible values are 0 or 1.

- **WildcardSearch:** Affects all object filters, except `ValidID`. It will wrap the search term in wildcards, so they match all objects that contain the supplied value. For example, if this option is set to `1`, filtering on `UserCustomerID` with `company` will in fact search for `*company*`.

Rule Configuration Examples

Here are some examples for rule configurations. These examples are valid YAML codes. You can copy these examples and modify them according to your needs.

Delete user first names and user last names by user first name with wildcard search:

```
---
RuleName: Delete user first names and user last names by user first name with
↳ wildcard search.
RuleType: PrivacyByDeletion
RuleSource: GDPR
DataClassification:
  CustomerUser:
    - UserFirstname
    - UserLastname
ObjectFilter:
  CustomerUser:
    UserFirstname: someFirstname
    WildcardSearch: 1
```

Anonymize user first names and user last names by user first name and without wildcard search:

```
---
RuleName: Anonymize user first names and user last names by user first name and
↳ without wildcard search.
RuleSource: someRuleSource
RuleType: PrivacyByAnonymization
DataClassification:
  CustomerUser:
    - UserFirstname
    - UserLastname
ObjectFilter:
  CustomerUser:
    UserFirstname: someFirstname
    WildcardSearch: 0
```

Delete user first names and user last names by user first name and user last name with wildcard search:

```
---
RuleName: Delete user first names and user last names by user first name and user
↳ last name with wildcard search.
RuleSource: someRuleSource
RuleType: PrivacyByDeletion
DataClassification:
  CustomerUser:
    - UserFirstname
    - UserLastname
ObjectFilter:
  CustomerUser:
    UserFirstname: someFirstname
    UserLastname: someLastname
    WildcardSearch: 1
```

Ticket Driver

The ticket driver provides the functionality to search and modify the information for tickets and related articles.

Possible data classifications for tickets:

- Title
- CustomerUserID
- CustomerID
- DynamicField_NameX

Possible data classifications for articles:

- From
- To
- Cc
- Subject
- Body
- Attachments
- DynamicField_NameX

The driver supports dynamic fields for data classification. Dynamic fields will be identified by the prefix `DynamicField_` and the related field name.

The data classification supports history types. Since history types may vary (framework versions, framework updates, installed packages etc.), the driver will dynamically determine those types by name.

History types have to be prefixed with the term `History`. Please see the examples in the following listing:

- HistoryAddNote
- HistoryAddSMS
- HistoryArchiveFlagUpdate
- HistoryBounce
- HistoryCustomerUpdate
- HistoryEmailAgent
- HistoryEmailCustomer
- HistoryEmailResend
- HistoryEscalationResponseTimeNotifyBefore
- HistoryEscalationResponseTimeStart
- HistoryEscalationResponseTimeStop
- HistoryEscalationSolutionTimeNotifyBefore
- HistoryEscalationSolutionTimeStart
- HistoryEscalationSolutionTimeStop
- HistoryEscalationUpdateTimeNotifyBefore
- HistoryEscalationUpdateTimeStart
- HistoryEscalationUpdateTimeStop
- HistoryFollowUp
- HistoryForward
- HistoryLock
- HistoryLoopProtection
- HistoryMerged
- HistoryMisc
- HistoryMove
- HistoryNewTicket
- HistoryOwnerUpdate
- HistoryPhoneCallAgent
- HistoryPhoneCallCustomer
- HistoryPriorityUpdate

(continues on next page)

(continued from previous page)

- HistoryRemove
- HistoryResponsibleUpdate
- HistorySendAgentNotification
- HistorySendAnswer
- HistorySendAutoFollowUp
- HistorySendAutoReject
- HistorySendAutoReply
- HistorySendCustomerNotification
- HistoryServiceUpdate
- HistorySetPendingTime
- HistorySLAUpdate
- HistoryStateUpdate
- HistorySubscribe
- HistorySystemRequest
- HistoryTicketDynamicFieldUpdate
- HistoryTicketLinkAdd
- HistoryTicketLinkDelete
- HistoryTimeAccounting
- HistoryTitleUpdate
- HistoryTypeUpdate
- HistoryUnlock
- HistoryUnsubscribe
- HistoryWebRequestCustomer

All contents of the classified history types will be affected during executions.

If attachments are classified, every attachment of all matching articles or tickets will be affected during executions.

Warning: The ticket driver is used to search for tickets, even if the rule contains filters for article fields. If article fields are part of the data classification, all articles of the related, matching ticket will be processed!

The following fields can be used as search terms or filters for tickets and articles. Possible object filters:

- Limit
- TicketID
- TicketNumber
- Title
- Queues
- QueueIDs
- UseSubQueues
- Types
- TypeIDs
- States
- StateIDs
- StateType
- StateTypeIDs
- Priorities
- PriorityIDs
- Services
- ServiceIDs
- SLAs
- SLAIDs
- Locks
- LockIDs

(continues on next page)

(continued from previous page)

- OwnerIDs
- ResponsibleIDs
- WatchUserIDs
- CustomerID
- CustomerUserLogin
- CreatedUserIDs
- CreatedTypes
- CreatedTypeIDs
- CreatedPriorities
- CreatedPriorityIDs
- CreatedStates
- CreatedStateIDs
- CreatedQueues
- CreatedQueueIDs
- TicketFlag
- ArticleFlag
- MIMEBase_From
- MIMEBase_To
- MIMEBase_Cc
- MIMEBase_Subject
- MIMEBase_Body
- AttachmentName
- FullTextIndex
- ContentSearch
- ContentSearchPrefix
- ContentSearchSuffix
- ConditionInline
- ArticleCreateTimeOlderMinutes
- ArticleCreateTimeNewerMinutes
- ArticleCreateTimeNewerDate
- ArticleCreateTimeOlderDate
- TicketCreateTimeOlderMinutes
- TicketCreateTimeNewerMinutes
- TicketCreateTimeNewerDate
- TicketCreateTimeOlderDate
- TicketChangeTimeOlderMinutes
- TicketChangeTimeNewerMinutes
- TicketLastChangeTimeOlderMinutes
- TicketLastChangeTimeNewerMinutes
- TicketLastChangeTimeNewerDate
- TicketLastChangeTimeOlderDate
- TicketChangeTimeNewerDate
- TicketChangeTimeOlderDate
- TicketCloseTimeOlderMinutes
- TicketCloseTimeNewerMinutes
- TicketCloseTimeNewerDate
- TicketCloseTimeOlderDate
- TicketPendingTimeOlderMinutes
- TicketPendingTimeNewerMinutes
- TicketPendingTimeNewerDate
- TicketPendingTimeOlderDate
- TicketEscalationTimeOlderMinutes
- TicketEscalationTimeNewerMinutes
- TicketEscalationTimeNewerDate
- TicketEscalationTimeOlderDate
- TicketEscalationUpdateTimeOlderMinutes

(continues on next page)

(continued from previous page)

```

- TicketEscalationUpdateTimeNewerMinutes
- TicketEscalationUpdateTimeNewerDate
- TicketEscalationUpdateTimeOlderDate
- TicketEscalationResponseTimeOlderMinutes
- TicketEscalationResponseTimeNewerMinutes
- TicketEscalationResponseTimeNewerDate
- TicketEscalationResponseTimeOlderDate
- TicketEscalationSolutionTimeOlderMinutes
- TicketEscalationSolutionTimeNewerMinutes
- TicketEscalationSolutionTimeNewerDate
- TicketEscalationSolutionTimeOlderDate
- ArchiveFlags

```

All possible object filter parameters can be used to filter tickets and articles. Most of the attributes can be single strings or array references, like:

```

TicketNumber: 123546
TicketNumber:
- 123546
- 123666

```

```

Title: SomeText
Title:
- SomeTest1
- SomeTest2

```

```

States:
- new
- open
StateIDs:
- 3
- 4

```

The corresponding YAML code could look as follows:

```

RuleName: My Explanation Rule
RuleType: PrivacyByDeletion
RuleSource: GDPR
DataClassification:
  Ticket:
    - CustomerUserID
    - CustomerID
ObjectFilter:
  Ticket:
    Queue:
      - Junk
      - Raw
    Services:
      - Service A
      - Service B

```

This rule would find all tickets, that are located in the queue *Junk* or *Raw* and which have the service *Service A* or *Service B* assigned. The fields `CustomerUserID` and `CustomerID` would be deleted.

There are several possible filter parameters, regarding relative times and dates, like:

```
- ArticleCreateTimeOlderMinutes
- ArticleCreateTimeNewerMinutes
- ArticleCreateTimeNewerDate
- ArticleCreateTimeOlderDate
```

A filter like `**TimeOlderMinutes*` means *older than X minutes*.

The following statement would mean: all tickets, that have a `CreateTime` older than one day (1440 minutes).

```
TicketCreateTimeOlderMinutes: 1440
```

The following statement would mean: all tickets, that have a `CreateTime` newer than one day (1440 minutes).

```
TicketCreateTimeNewerMinutes: 1440
```

This is principal valid for all filter parameters with this syntax.

For more descriptions about the single search parameters, check the [TicketSearch\(\)](#) in API reference.

Rule Configuration Examples

Here are some examples for rule configurations. These examples are valid YAML codes. You can copy these examples and modify them according to your needs.

Delete ticket titles by state names, that are older than one month:

```
---
RuleName: Delete ticket titles by state names, that are older than one month.
RuleSource: GDPR
RuleType: deletion
DataClassification:
  Ticket:
    - Title
ObjectFilter:
  Ticket:
    States:
      - new
      - open
    TicketCreateTimeOlderMinutes: 43200
```

Delete article subject and body by state names, that are located in specific queues:

```
---
RuleName: Delete article subject and body by state names, that are located in
↳specific queues.
RuleSource: GDPR
RuleType: deletion
DataClassification:
  Ticket:
    - Subject
    - Body
ObjectFilter:
  Ticket:
    States:
      - new
```

(continues on next page)

(continued from previous page)

```

- open
Queues:
- Postmaster
- Misc

```

Pseudonymize customer user IDs for tickets, that are closed and archived:

```

----
RuleName: Pseudonymize customer user IDs for tickets, that are closed and archived.
RuleSource: GDPR
RuleType: PrivacyByPseudonymization
DataClassification:
  Ticket:
    - CustomerUserID
ObjectFilter:
  Ticket:
    StateType:
      - Closed
    ArchiveFlags:
      - y

```

Anonymize customer IDs and some dynamic fields, that are closed, have certain services and are located in specific queues:

```

----
RuleName: Anonymize Customer IDs and some dynamic fields, that are closed, have ↵
↳ certain services and are located in specific queues.
RuleSource: GDPR
RuleType: PrivacyByAnonymization
DataClassification:
  Ticket:
    - CustomerID
    - DynamicField_SensitiveNames
    - DynamicField_SensitiveLocations
ObjectFilter:
  Ticket:
    StateType:
      - Closed
    Queue:
      - Special Queue A
      - Junk
    Services:
      - Sensitive Customer Service
      - VIP Customer Service

```

User Driver

The user driver provides the functionality to search and modify the information for users.

Possible data classifications:

```

- UserTitle
- UserFirstname
- UserLastname

```

(continues on next page)

(continued from previous page)

```
- userEmail
- UserMobile
```

Possible object filters:

```
- UserFirstname
- UserLastname
- UserLogin
- UserTitle
- CreateTime
- Valid
- Limit
- UserPreferences
- WildcardSearch
```

Object filter descriptions:

- **Limit:** Limits the number of search results.
- **CreateTime:** Searches for dates *greater than or equal to* (\geq) the given time.
- **Valid:** Searches for valid or invalid users. Possible values are 0 or 1.
- **WildcardSearch:** Affects the object filters `UserFirstname`, `UserLastname`, `UserLogin` and `UserTitle`. It will wrap the search term in wildcards, so they match all objects that contain the supplied value. For example, if this option is set to 1, filtering on `UserLogin` with `agent` will in fact search for `*agent*`.
- **UserPreferences:** Array containing the user preferences like user email address as keys with certain search criteria as values (see YAML configuration examples).

Rule Configuration Examples

Here are some examples for rule configurations. These examples are valid YAML codes. You can copy these examples and modify them according to your needs.

Delete user first names by user first name:

```
---
RuleName: Delete user first names by user first name.
RuleSource: GDPR
RuleType: PrivacyByDeletion
DataClassification:
  User:
    - UserFirstname
ObjectFilter:
  User:
    UserFirstname: someFirstname
```

Delete user first names and user last names by user email:

```
---
RuleName: Delete user first names and user last names by user email.
RuleSource: GDPR
RuleType: PrivacyByDeletion
DataClassification:
```

(continues on next page)

(continued from previous page)

```

User:
- UserFirstname
- UserLastname
ObjectFilter:
User:
UserPreferences:
UserEmail: someMail@example.com

```

Delete user first names and user last names with wildcard search:

```

---
RuleName: Delete user first names and user last names with wildcard search.
RuleSource: GDPR
RuleType: PrivacyByDeletion
DataClassification:
User:
- UserFirstname
- UserLastname
ObjectFilter:
User:
UserFirstname: someFirstname
WildcardSearch: 1

```

Delete user first names by user first name and create time, which are greater than or equal with the specified date:

```

---
RuleName: Delete user first names by user first name and create time, which are ↳
↳ greater than or equal with the specified date.
RuleSource: GDPR
RuleType: PrivacyByDeletion
DataClassification:
User:
- UserFirstname
ObjectFilter:
User:
CreateTime: 2019-01-01
UserFirstname: someFirstname

```

11.2 Agent Interface

This package has no agent interface.

11.3 External Interface

This package has no external interface.

DYNAMIC FIELD CI

This feature makes it possible to show configuration items from the OTRS CMDB in dynamic fields in ticket screens in the agent and external interface. This speeds up the ticket creation process for agents and customer users by enabling the use of filters so that only those configuration items that are relevant for the customer company are shown and can be selected in the ticket screens. In addition, the configuration item search function can also be used directly in the ticket. Filters can be defined, e.g. based on:

- Configuration item class
- Usage status
- Incident status
- Customer ID
- Customer user ID
- Attribute

In addition, the following configurations are possible:

- Automatic linking and type of linking between configuration item and ticket.
- How and in which ticket screens the configuration item should be shown (e.g. as a drop-down list or a tree view).
- Visualization in the external or agent interface.

Benefits

- Customer-based visualization and selection of configuration items when creating a ticket in the external interface.
- Faster ticket creation thanks to the use of filters to choose which configuration items should be shown, e.g. according to class, usage status, incident status, etc.
- Maximum flexibility through use in different ticket screens for customer users and/or agents.
- Rapid search for configuration items when creating a ticket.
- Useful for statistics and automatic ticket notifications.

Target Groups

- Customers of companies that offer services for different devices or products
- Customer service
- Internal and external IT
- Facility management
- Sales

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSDynamicFieldCI

Note: This feature requires the *Configuration Management* feature.

12.1 Administrator Interface

This package contains a configuration interface to create dynamic fields of the type *Configuration item*. These fields can store configuration items in a dynamic field and display them in the different views.

12.1.1 Processes & Automation

This package makes it possible to create dynamic fields of the type *Configuration item*, which can store configuration items in tickets.

Dynamic Fields

After installation of the package a new dynamic field type *Configuration item* will be available for tickets.

This dynamic field can be created the same way as default dynamic fields are created. For this navigate to the *Dynamic Fields* module of the *Processes & Automation* group in the administrator interface. In this screen you can select the *Configuration Item* field from the drop-down lists on the left side.

See also:

The usage of dynamic fields and the general dynamic field settings are described in the [Dynamic Fields](#) chapter of the administration manual.

Dynamic Field Settings

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Configuration Item Dynamic Field Settings

Dynamic field of type configuration item is used to store configuration items for tickets.

Configuration Item Field Settings

Filter Options

Class:
Filter for configuration item classes.

Deployment State:
Filter for configuration item deployment states.

Incident State:
Filter for configuration item incident states.

Restriction Options

▼ **Restriction by Customer ID**

Restriction by Interface:
Select if the available configuration items should be restricted by customer ID in agent and/or external interface.

Configuration Item Field:
The name of the configuration item field containing the customer ID.

Value Lookup for Customer ID:
Attribute: Select if the readable value for the customer ID field should be used for restriction.

AND

▼ **Restriction by Customer User ID**

Restriction by Interface:
Select if the available configuration items should be restricted by customer user ID in agent and/or external interface.

Configuration Item Field:
The name of the configuration item field containing the customer user ID.

Value Lookup for Customer User ID:
Attribute: Select if the readable value for the customer user ID field should be used for restriction.

AND

▼ **Restriction by Attribute**

Restriction by Other Attribute:
Select if the available configuration items should be restricted by another attribute.

Restriction by Interface:
Select the interface where the configuration item should be restricted by the selected attribute.

Configuration Item Field:
The name of the configuration item field containing the attribute value.

Value Lookup for Restriction:
Attribute: Select if the readable value for given attribute field should be used for restriction.
Note: if the attribute you want to restrict on is of type general catalog.

Link Options

Link Type:
Type of automatically created links for configuration items. "None" disables the linking.

Link Direction:
Please select a possible direction for selected link type.

Display Options

Multiselect:
Activate this option to allow multiple selection of values.

Add Empty Value:
Activate this option to create an empty selectable value.

Tree View:
Activate this option to display values as a tree.

Translatable Values:
If you activate this option the values will be translated to the user defined language.
Note: You need to add the translations manually into the language translation files.

Autocompletion:
Activate this option to enable autocompletion for the values.

Advanced Options

Follow CMDB Links:
If you activate this option CMDB items linked to the items set in the "Follow dynamic field" will be displayed as selectable options for this field.

CMDB Link Settings:
Define elements to show for this field if no items are selected in the "Follow dynamic field".

Follow Dynamic Field:
Dynamic field to follow CMDB links from.

Follow Dynamic Field Attribute:
The name of the configuration item field containing the dynamic field value.

Fig. 1: Configuration Item Dynamic Field Settings

Filter Options

Class

Filter for configuration item classes to narrow down the list of possible values.

Deployment State

Filter for configuration item deployment states to narrow down the list of possible values.

Incident State

Filter for configuration item incident states to narrow down the list of possible values.

Restriction Options

Restriction by Interface

Select if the available configuration items should be restricted by customer ID, by customer user ID or by any other attribute in agent and/or external interface.

Configuration Item Field

The name of the configuration item field containing the customer ID, the customer user ID or any other attribute. Usually this is the owner field.

Value Lookup for Customer ID, Customer User ID or Any Attribute

Select if the readable value for the customer ID, the customer user ID or any other attribute field should be used for restriction.

Note: If the filters are changed after the configuration item and the ticket has been linked, the link will not be automatically updated.

Link Options

Link Type

Type of automatically created links for configuration items. *None* disables the automatic linking.

Link Direction

If a *Link Type* is selected, choose the link direction from the point of ticket.

Display Options

Multiselect

Activate this option to allow multiple selection of values.

Add Empty Value

Activate this option to create an empty selectable value.

Tree View

Activate this option to display values as a tree.

Translatable Values

If you activate this option the values will be translated to the user defined language.

Note: You need to add the translations manually into the language translation files.

Autocompletion

Activate this option to enable autocompletion for the values.

Advanced Options**Follow CMDB Links**

If you activate this option CMDB items linked to the items set in the *Follow Dynamic Field* will be displayed as selectable options for this field.

CMDB Link Settings

Define elements to show for this field if no items are selected in the *Follow Dynamic Field*. Possible values:

- Does not show any items
- Show regular items

Follow Dynamic Field

Dynamic field to follow CMDB links from. This option only works if the setting *Restriction by Customer ID* or *Restriction by Customer User ID* is deactivated.

Follow Dynamic Field Attribute

The name of the configuration item field containing the dynamic field value.

Note: Do not forget to add the new dynamic field to ticket view screens.

12.2 Agent Interface

This package has no agent interface, but the dynamic field of type configuration item can be added to many screens.

12.2.1 Automatic Linking and Unlinking

When configuration items are added to or removed from tickets via the new dynamic field back end, links for the respective configuration items can be added or removed automatically. If this is configured, links will be compared and links will be added and/or removed, if necessary.

Warning: In order to prevent inconsistencies between the dynamic field content and links please do not add or remove links of the configured type and direction manually via the link mechanism in the ticket detail view. Those changes are not synchronized back to the dynamic field content.

See also:

The dynamic field configuration decides if linking happens and what link type and direction will be used. See the *Dynamic Fields* in the administrator interface.

12.3 External Interface

This package has no external interface, but the dynamic field of type configuration item can be added to many screens.

DYNAMIC FIELD VALUE IMPORT

This feature offers additional options for uploading values from CSV files for dynamic fields. The data from individual columns of the CSV file are assigned to a specific dynamic field that you can flexibly select. These files can additionally contain headers into which you can insert the desired name of the target fields. Before the value can be imported, the target fields have to be created manually.

After a successful upload, the *Change Overview* widget shows all new or deleted values for the dynamic fields affected. These values are marked in colors.

In addition, the feature enables the automatic creation of ACLs based on the dynamic fields chosen and their imported values. This significantly reduces the manual workload for ACL creation.

Dependencies between the dynamic fields are created from left to right. If values for more than two dynamic fields are imported, the allocation of the ACL relations starts in the column furthest to the left. Multi-level dependencies are therefore possible.

Benefits

- Fast import of modified product portfolios.
- Customer service and sales are more quickly up-to-date.
- Easier integration of partners from outsourced corporate areas.

Target Groups

- Call centers
- Customer service
- Wholesalers
- Sales
- Order management

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSDynamicFieldValueImport

13.1 Administrator Interface

This package provides functionality to import dynamic field possible values from CSV files. Furthermore it is possible to create left → right associated access control lists (ACLs) between these dynamic fields and their imported values.

13.1.1 Processes & Automation

This package makes it possible to import values for dynamic fields and adds a new widget into the left sidebar of the *Dynamic Fields Management* screen.

Dynamic Fields

After installation of the package a new widget *Value Import* will be available in the left sidebar of the *Dynamic Fields Management* screen.

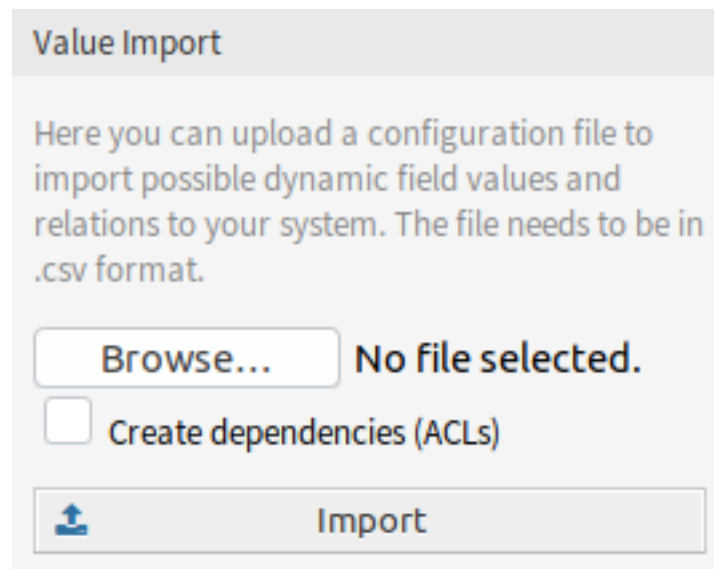


Fig. 1: Value Import Widget

To import values for a dynamic field:

1. Make sure that the target dynamic fields are already created.
2. Click on the *Browse...* button of the *Value Import* widget in the left sidebar.
3. Select a proper formatted `.csv` file.
4. Select the *Create dependencies (ACLs)* checkbox.
5. Click on the *Import* button.
6. Select the target dynamic fields and see the change difference.
7. Modify the ACL settings, if needed.
8. Click on the *Submit* button.

Example Usage

Target dynamic fields have to be created manually before they can be used for value import. Only select-type fields are allowed (drop-down and multiselect).

Create the following dynamic fields:

Object	Type	Name	Label	Possible values
Ticket	Multiselect	Manufacturer	Manufacturer	Leave it empty
Ticket	Multiselect	Car	Car	Leave it empty
Ticket	Multiselect	ModelYear	Model Year	Leave it empty

To guarantee a proper working of the feature provided by this package, please format your CSV files according to the example. The first line containing dynamic field names is optional, but the dynamic fields will be auto-selected in the next screen, if they are provided.

By default the separator defined in your language will be used (usually , or ;). Alternatively you can override this by using the CSV separator user preference.

```
Manufacturer,Car,ModelYear
BMW,X3,2016
BMW,318d,2016
BMW,318d,2017
BMW,320d,2016
Audi,A4,2015
Audi,A4,2016
Audi,A5,2015
Audi,A5,2016
Audi,A6,2016
```

The CSV file contains information about possible values for dynamic fields. The first header line contains the target dynamic field names.

Browse for the CSV file and check the *Create dependencies (ACLs)* box if you want to generate ACLs. Click on the *Import* button, a new screen will be opened.

In this overview you will see a table indicating CSV column, the selected target dynamic field, a colorized change difference between old and new possible values and (if ACL setting is enabled) a preview of the relations between the fields. This helps comparing the data before the values are written.

Data from every CSV column will be assigned to a specific dynamic field. The target field can be chosen. If a header line is used, the field names are pre-selected.

Warning: The existing possible values for all affected fields will be completely overwritten with the uploaded values. Therefore you should carefully check if the indication of removed and new values for each dynamic field matches your expectation. In order to retain existing values they need to be contained in the uploaded CSV file.

It is possible to automatically create left → right associated ACLs based on the selected dynamic fields and their imported values. This will reduce the effort to creating ACLs manually. The dependencies between dynamic fields and values are created from the left to right side.

The following settings are available when using this functionality.

Create ACLs?

If you want to generate ACLs, check this box. Left → right depending ACLs between affected fields

▼ Change Overview

COLUMN #	TARGET DYNAMIC FIELD	DIFF OF POSSIBLE VALUES	ACL RELATIONS BETWEEN FIELDS															
1	Manufacturer x	+Audi +BMW																
2	Car x	+318d +320d +A4 +A5 +A6 +X3	<table border="1"> <thead> <tr> <th>MANUFACTURER</th> <th>CAR</th> </tr> </thead> <tbody> <tr> <td>▪ Audi</td> <td>▪ A4 ▪ A5 ▪ A6</td> </tr> <tr> <td>▪ BMW</td> <td>▪ 318d ▪ 320d ▪ X3</td> </tr> </tbody> </table>	MANUFACTURER	CAR	▪ Audi	▪ A4 ▪ A5 ▪ A6	▪ BMW	▪ 318d ▪ 320d ▪ X3									
MANUFACTURER	CAR																	
▪ Audi	▪ A4 ▪ A5 ▪ A6																	
▪ BMW	▪ 318d ▪ 320d ▪ X3																	
3	ModelYear x	+2015 +2016 +2017	<table border="1"> <thead> <tr> <th>MANUFACTURER</th> <th>CAR</th> <th>MODELYEAR</th> </tr> </thead> <tbody> <tr> <td>▪ Audi</td> <td>▪ A4 ▪ A5</td> <td>▪ 2015 ▪ 2016</td> </tr> <tr> <td>▪ Audi</td> <td>▪ A6</td> <td>▪ 2016</td> </tr> <tr> <td>▪ BMW</td> <td>▪ 318d</td> <td>▪ 2016 ▪ 2017</td> </tr> <tr> <td>▪ BMW</td> <td>▪ 320d ▪ X3</td> <td>▪ 2016</td> </tr> </tbody> </table>	MANUFACTURER	CAR	MODELYEAR	▪ Audi	▪ A4 ▪ A5	▪ 2015 ▪ 2016	▪ Audi	▪ A6	▪ 2016	▪ BMW	▪ 318d	▪ 2016 ▪ 2017	▪ BMW	▪ 320d ▪ X3	▪ 2016
MANUFACTURER	CAR	MODELYEAR																
▪ Audi	▪ A4 ▪ A5	▪ 2015 ▪ 2016																
▪ Audi	▪ A6	▪ 2016																
▪ BMW	▪ 318d	▪ 2016 ▪ 2017																
▪ BMW	▪ 320d ▪ X3	▪ 2016																

Fig. 2: Change Overview Widget

▼ ACL settings

Create ACLs?

Left → right depending ACLs between affected fields will be created automatically on import.

ACL Prefix:

Given prefix is used for all created ACLs. Change prefix to affect execution order (interoperability with manually created ACLs).

Remove ACLs with other prefixes:

All previously auto-generated ACLs will be removed.
Note: If a different prefix was used previously and this option is not used, old ACLs will remain and might cause undesired behavior.

Auto deploy:

Automatically deploy ACLs after creation.
Note: This will also affect other ACLs that have not been deployed yet.

Fig. 3: ACL Settings Widget

will be created automatically on import.

ACL Prefix

Given prefix is used for all created ACLs. Change prefix to affect execution order (interoperability with manually created ACLs).

Remove ACLs with other prefixes

Select this checkbox, if all previously auto-generated ACLs should be removed.

Note: If a different prefix was used previously and this option is not used, old ACLs will remain and might cause undesired behavior.

Auto deploy

Select this checkbox, if you would like to automatically deploy ACLs after creation.

Note: This will also affect other ACLs that have not been deployed yet.

The ACL creation is implemented in the following way:

- If the dynamic field with the value(s) from the left side is matched, it is (only) possible to select the dynamic field with the value(s) from the right side.
- If values for more than two dynamic fields are imported (3 or more columns exist in the CSV file), ACL relations are always created starting from the leftmost column. This allows for multi-level dependencies.

After you have finished all checks, just submit the form. The values of the affected dynamic fields will be updated in the background and, if activated, the ACLs will be generated. Afterwards you will be redirected to the dynamic field overview again.

Now go to the *ACL Management* screen, and check the generated ACLs.

















ACL NAME	COMMENT	VALIDITY	EXPORT	COPY
9999-DynamicFieldValueImport-Manufacturer-Car-000	Created automatically by OTRSDynamicFieldValueImport.	valid		
9999-DynamicFieldValueImport-Manufacturer-Car-001	Created automatically by OTRSDynamicFieldValueImport.	valid		
9999-DynamicFieldValueImport-Manufacturer-Car-002	Created automatically by OTRSDynamicFieldValueImport.	valid		
9999-DynamicFieldValueImport-Manufacturer-Car-ModelYear-000	Created automatically by OTRSDynamicFieldValueImport.	valid		
9999-DynamicFieldValueImport-Manufacturer-Car-ModelYear-001	Created automatically by OTRSDynamicFieldValueImport.	valid		
9999-DynamicFieldValueImport-Manufacturer-Car-ModelYear-002	Created automatically by OTRSDynamicFieldValueImport.	valid		
9999-DynamicFieldValueImport-Manufacturer-Car-ModelYear-003	Created automatically by OTRSDynamicFieldValueImport.	valid		
9999-DynamicFieldValueImport-Manufacturer-Car-ModelYear-004	Created automatically by OTRSDynamicFieldValueImport.	valid		

Fig. 4: Generated ACLs for Dynamic Field Restrictions

Limitations

However this package makes it easy to import values for dynamic fields and generate the appropriate ACLs, there are some limitations.

- Dynamic field creation and configuration has to be done manually (e.g. to define if a field should be a drop-down or multiselect and if empty values are allowed).
- Dynamic field values have a length limit of 200 characters per value.
- ACLs are created in a way that secondary fields require a selection on their primary field before any selection is possible.
- ACLs will always be created for all dynamic fields from the CSV file. Unrelated dynamic fields have to be imported separately.

13.2 Agent Interface

This package has no agent interface.

13.3 External Interface

This package has no external interface.

DYNAMIC SENDER ADDRESSES

The sender address of your tickets in the standard version of **OTRS** is by default a queue-associated email address. In order to configure the sender address this feature uses a special dynamic field. When replying or forwarding tickets and event-based notifications, **OTRS** automatically sets the new sender address. When the dynamic field is blank, the queue-associated email address is used.

In addition, you can also use this feature to edit tickets from all over the world and use country-specific sender addresses. By grouping the countries and email addresses **OTRS** can automatically add the correct return address for each country.

Benefits

- Worldwide service with regional sender addresses.
- Easier identification of incoming emails for your customers.
- Improved customer loyalty.
- Simplified teamwork with partners or outsourced divisions.

Target Groups

- Call centers
- Customer service
- Sales
- Marketing
- Procurement

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSDynamicSenderAddresses

14.1 Administrator Interface

This package allows to base the sender address of outgoing emails on a dynamic field. This can be used for replies, forwards and event based notifications.

14.1.1 Communication & Notifications

After installation of the package, a new administrator module allows to add and modify values to be mapped from the configured dynamic field values into system email addresses. For each added value a valid system address can be selected. This mapping will be checked against dynamic field values in order to determine the sender address.

Sender Address Mapping

Use this screen to map the values contained in the configured dynamic field into system email addresses. The sender address mapping screen is available in the *Sender Address Mapping* module of the *Communication & Notifications* group.

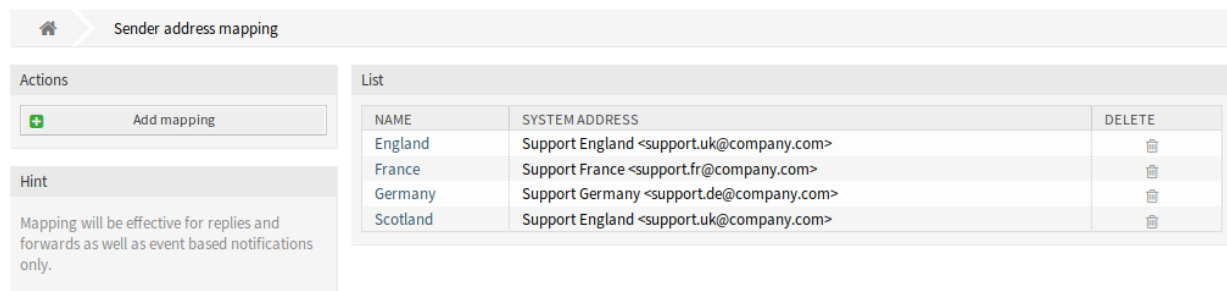


Fig. 1: Sender Address Mapping Management Screen

Map Email Addresses

Note: The mapping is based on the dynamic field set in setting `DynamicSenderAddresses###FieldName`. The dynamic field has to be created manually.

To add a mapping:

1. Click on the *Add Mapping* button in the left sidebar.
2. Fill in the required fields.
3. Click on the *Save* button.

To edit a mapping:

1. Click on a mapping in the list of mappings.
2. Modify the fields.
3. Click on the *Save* or *Save and finish* button.

Add mapping

★ Name:

★ System address:

or

Fig. 2: Add Mapping Screen

Edit mapping

★ Name:

★ System address:

or or

Fig. 3: Edit Mapping Screen

To delete a mapping:

1. Click on the trash icon in the list of mappings.
2. Click on the *Confirm* button.

List

NAME	SYSTEM ADDRESS	DELETE
England	Support England <support.uk@company.com>	
France	Support France <support.fr@company.com>	
Germany	Support Germany <support.de@company.com>	
Scotland	Support England <support.uk@company.com>	

Fig. 4: Delete Mapping Screen

See also:

This administration screen reads and writes the system configuration setting `DynamicSenderAddresses###Mapping`, where the key is the name of the mapping and the value is the internal ID of the system email address.

Warning: Do not change the setting manually. Use the administrator interface module instead.

Country	Value
England	3
France	4
Germany	2
Scotland	3

Mapping from dynamic field value to sender address. Do not change manually - use admin frontend module!

Fig. 5: DynamicSenderAddresses###Mapping Setting

Mapping Settings

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Name *

The name of this resource. Any type of characters can be entered to this field including uppercase letters and spaces. The name will be displayed in the overview table.

System address *

Select a system email address configured in *Email Addresses* module in the administrator interface.

14.1.2 Processes & Automation

A generic agent module is provided to have ticket values copied to a dynamic field to use this field as base for the sender address.

Generic Agent

The ticket service should be the base for the sender address, but only dynamic field values are supported by this feature.

In order to allow using the service, create a new dynamic field and use a generic agent job. The helper module will be added in order to automatically fill the new dynamic field with the service name.

To fill the dynamic field value with the service name:

1. Create the dynamic field *Service Copy*, but do not configure for display on any ticket screen.
 - Object type: Ticket
 - Field type: Text
 - Name: *ServiceCopy*
 - Label: *Service Copy*
2. Create a generic agent job.
 - *Event Based Execution* section: Set the events `TicketCreate` and `TicketServiceUpdate` as triggers.
 - *Select Tickets* section: Set filters if the field should not be set globally (e. g. restricting this to specific queues).
 - *Execute Custom Module* section:

- **Module:** `Kernel::System::GenericAgent::CopyTicketValueToDynamicField`
- **Param key:** `Ticket_Service`
- **Param value:** `ServiceCopy`

Module parameters control which fields should be copied. The key determines the ticket attribute (service, queue, etc.) and the value determines the name of the destination dynamic field.

Note: `Ticket_` prefix is needed if `Services` is used as parameter key. It will be included on the values for setting new values on ticket and it is not possible to get a real value for it, then as consequence the value for this ticket field will be empty.

3. Mapping from dynamic field values can now be used to link service names to actual sender addresses.

14.2 Agent Interface

This package has no agent interface.

14.2.1 Mapped Values as Sender Address

When a ticket reply or forward is created or when an event based notification is sent, the configured dynamic field of the ticket is checked. If the dynamic field contains a value, a mapping configuration is checked. If there is a mapped value in *Sender Address Mapping*, this value will be used as sender address.

If the dynamic field is not set or no mapping value exists, the default queue-based sender address will be used.

This feature should be used if your tickets have dynamic fields that contain information which sender address should be used, but are not email addresses. For this purpose the dynamic field values can be mapped to email addresses.

Example Usage

A dynamic field *Customer location* is configured to be used as sender address base. The configured queue sender address is `support@company.com`.

In the mapping module the following is configured:

- England → `support.uk@company.com`
- France → `support.fr@company.com`
- Germany → `support.de@company.com`
- Scotland → `support.uk@company.com`

When an agent replies to a ticket, the *Customer location* field is looked up and depending on the location the sender address is determined.

- Dynamic field value: `<empty field>`, sender address: `support@company.com` (address from queue).
- Dynamic field value: `England`, sender address: `support.uk@company.com` (mapped value).

Job Settings

★ Job name:

Validity:

▶ Automatic Execution (Multiple Tickets)

▼ Event Based Execution (Single Ticket)

Event Triggers:

TYPE	EVENT	DELETE
Ticket	TicketCreate	🗑
Ticket	TicketServiceUpdate	🗑

Additionally or alternatively to a periodic execution, you can define ticket events that will trigger this job. If a ticket event is fired, the ticket filter will be applied to check if the ticket matches. Only then the job is run on that ticket.

Add Event Trigger:

To add a new event select the event object and event name.

▶ Select Tickets

▶ Update/Add Ticket Attributes

▶ Add Note

▶ Execute Ticket Commands

▼ Execute Custom Module

Module:

Param 1 key: Param 1 value:

Param 2 key: Param 2 value:

Param 3 key: Param 3 value:

Param 4 key: Param 4 value:

Param 5 key: Param 5 value:

Param 6 key: Param 6 value:

Save Changes

or Cancel

- Dynamic field value: Ireland, sender address: `support@company.com` (no mapping found, therefore address from queue).
- Dynamic field value: France, sender address: `support.fr@company.com` (mapped value).

14.2.2 Direct Values as Sender Address

When a ticket reply or forward is created or when an event based notification is sent, the configured dynamic field of the ticket is checked. If the dynamic field contains a value, this value will be used as sender address.

If the dynamic field is not set, the default queue-based sender address will be used.

This feature should be used if your tickets have dynamic fields that contain email addresses which should be used as sender address.

Note: With this feature dynamic field values are not checked in advance, therefore all values for the configured field need to be valid email addresses to prevent delivery errors.

Example Usage

A dynamic field *Customer location address* is configured to be used as sender address. The configured queue sender address is `support@company.com`. When an agent replies to a ticket, the *Customer location address* field is looked up and the sender address is determined.

- Dynamic field value: <empty field>, sender address: `support@company.com` (address from queue).
- Dynamic field value: France, sender address: France (dynamic field value is an invalid email address, the email will not send).
- Dynamic field value: `support.uk@company.com`, sender address: `support.uk@company.com` (dynamic field value).

14.3 External Interface

This package has no external interface.

ESCALATION SUSPEND

This feature not only makes **OTRS** an even better tool for solving problems quickly and shortening response times, it also provides greater precision when it comes to time accounting. If you need to pause an escalation, this feature will do the job perfectly. The feature allows you to configure one or more statuses that will put one or more escalations (SLAs) automatically on hold. For example, if you are waiting for a third party answer or for a process step to be carried out, this feature will give you more freedom by allowing you to stop the escalation countdown. Removing the state will then reactivate the escalation(s) automatically, and the time left to process the ticket will be displayed once again.

This configurable status is basically a pause function for escalations. The time that passes when the feature is active is not counted in the ticket's resolution time. This provides greater time accounting precision.

Benefits

- Minimizes unnecessary escalations.
- Enables precise time accounting.
- Provides greater flexibility through its configurable status.

Target Groups

- IT service management
- Internal IT
- Customer service

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSEscalationSuspend

Note: Not compatible with the following feature:

- *Advanced Escalations*
-

15.1 Administrator Interface

This package has no administrator interface, but includes the functionality to stop or suspend escalations. If a ticket is assigned to one of several configurable states, the escalation is stopped (disabled).

15.1.1 Configuration

To configure the states where the escalation is stopped:

1. Go to the *System Configuration* screen.
2. Select *OTRSEscalationSuspend* in the *Navigation* widget.
3. Navigate to *Core* → *Ticket* in the navigation tree.
4. Add new entries to setting `EscalationSuspendStates`.
5. If suspend escalation is needed for already escalated tickets, enable the `SuspendEscalatedTickets` setting.

Note: Invalid states will not be considered as suspend states for the calculation.

When changing the state of the ticket to normal, the escalation will continue. In this case, it starts with the date when the status was changed, displaying the remaining time.

Therefore, the entire period when the ticket was not on a normal state is not counted for solution time. However, only periods when the ticket was in suspended states before first response is not counted for first response time.

15.1.2 Console Command

Note: This feature is only available to *On-Premise* customers. If you are a *Managed* customer, this feature is taken care of by the *Customer Solutions Team* in **OTRS**. Please contact us via support@otrs.com or in the [OTRS Portal](#).

This package contains a console command `Maint::Ticket::RebuildEscalationIndexOnline` handled by the OTRS daemon used for resetting the escalation times to the point it was suspended. This script is the one in charge to reset the escalation time to the point it was suspended.

To check if the script is running, execute the following command and find the `RebuildEscalationIndexOnline` task in the *Recurrent cron tasks* section.

```
otrs> /opt/otrs/bin/otrs.Console.pl Maint::Daemon::Summary
```

15.2 Agent Interface

This package has no dedicated agent interface.

Note: For this example escalations should be configured for at least one queue.

Example usage:

1. Go to one of the new ticket screens, and create a new ticket for one of the queues configured with escalation (for this example *Escalation - first response time* is OK).
2. Open the ticket detail view for the newly created ticket. Notice that the escalation is running.
3. Once the escalation is running, change the ticket state to one that has been configured in `EscalationSuspendStates` setting.
4. Go back to the ticket detail view for the suspended escalation ticket and verify that the escalation is not running.
5. Change the ticket state to one different from those configured on `EscalationSuspendStates` setting. The escalation continues.

There is a new filter *Suspended Escalations* for the ticket lists. Select the *Remove tickets with suspended escalations* checkbox to remove all tickets from the list that are in an escalation suspending state.

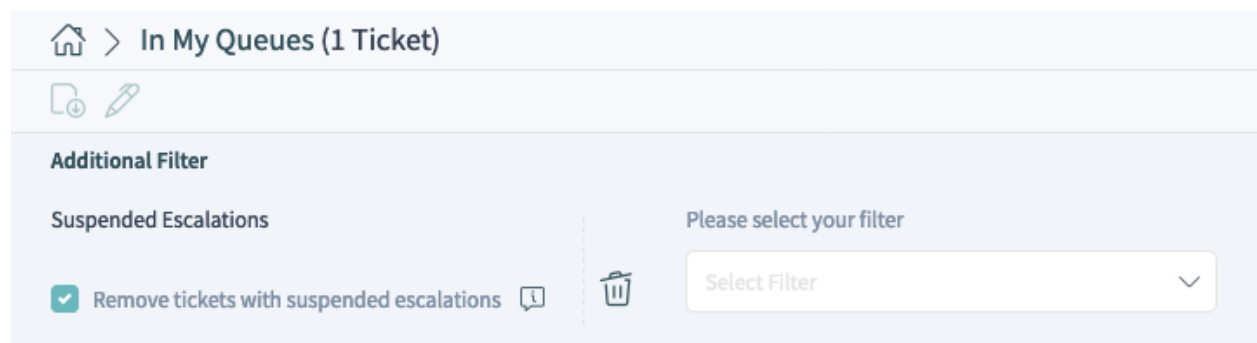


Fig. 1: Suspended Escalations Filter in Ticket List

Several ticket list screens also provide a default filter preset called *Escalation is not suspended*, which applies the aforementioned filter.

A suspended escalation can be seen in the ticket list, if hovering on an entry within the column *Escalation Time*. In the ticket detail view of a suspended ticket, the suspension status is visible in a property card.

15.3 External Interface

This package has no external interface.

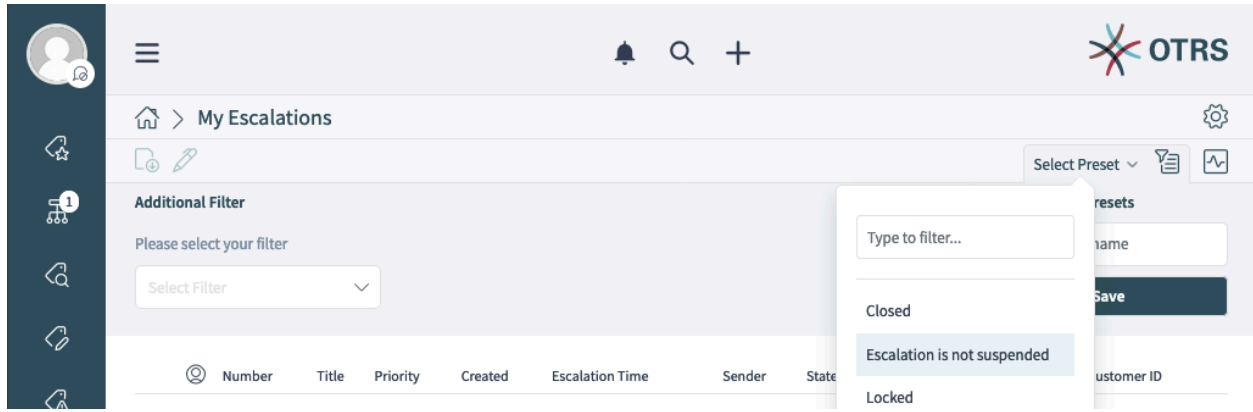


Fig. 2: Filter Preset in *My Escalations* Organizer Item

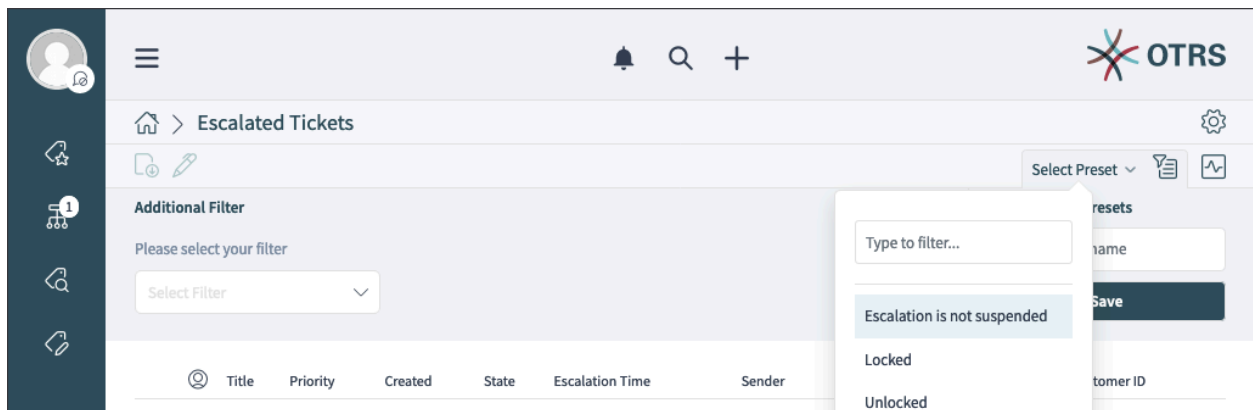


Fig. 3: Filter Preset in *Show Escalations / Escalated Tickets* Screen

PRIMARY / SECONDARY

This package adds support to define a primary ticket and some secondary tickets that follow their primary ticket updates:

- Replies (sent new articles as email)
- Notes (any new article)
- State changes (including pending time set)
- Priority changes
- Owner changes
- Responsible changes
- Lock status changes

The primary/secondary relations are very flexible. It is possible to create new primary ticket, remove the primary or the secondary status or assign a ticket as the secondary ticket of another ticket.

Note: This package has been renamed in this release. The former name was *Master / Slave* until OTRS 7.

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSPrimarySecondary

16.1 Administrator Interface

This package has no administrator interface.

16.1.1 New Dynamic Field

After installation of the package a new dynamic field of type *Primary / Secondary* will be added to the system where the primary/secondary information is stored. The creation of more fields of this type is not possible.

The *Primary / Secondary* dynamic field is added to the ticket create screens. This makes it possible to set the primary or the secondary status of a ticket at ticket creation time.

To remove the *Primary / Secondary* dynamic field from the ticket create screens:

1. Go to the *System Configuration* screen.
2. Search for the setting `Forms###AgentFrontend::TicketCreate::Phone::CreateProperties`.
3. Change the related form setting to remove the dynamic field from the screen. Since this dynamic field is a special dynamic field, the name is not `DynamicField_PrimarySecondary`, but it is just `PrimarySecondary`. Example:

```
- Name: PrimarySecondary
```

Same applies for the process management. The field which needs to be added to a user task activity dialog is called `PrimarySecondary`, not `DynamicField_PrimarySecondary`.

16.1.2 Exception for Closing Parent Ticket

The `Ticket::Acl::Module###1-Ticket::Acl::Module` is not compatible to use with this package by default. To allow closing parent tickets only if all its children are already closed, an additional ACL have to be added which makes an exception for primary/secondary tickets.

Here is an example ACL for this exception:

```
---
- ChangeBy: root@localhost
  ChangeTime: 2021-09-07 13:45:58
  Comment: Exception for primary/secondary tickets.
  ConfigChange:
    PossibleAdd:
      Endpoint:
        - AgentFrontend::Ticket::Action::Close
      Ticket:
        State:
          - '[regex]close'
  ConfigMatch:
    PropertiesDatabase:
      DynamicField:
        DynamicField_PrimarySecondary:
          - Primary
  CreateBy: root@localhost
  CreateTime: 2019-05-24 11:45:29
  Description: 'This ACL allows using the Primary/Secondary feature together with the
  "Ticket::Acl::Module###1-Ticket::Acl::Module" setting.'
  ID: 1
  Name: Primary/Secondary Exception
  StopAfterMatch: 0
  ValidID: 1
```

16.1.3 ACL Reference

This package adds new endpoints that can be used in ACLs. The following reference lists all endpoints added by this package and extends the existing ACL reference from the *Administrator Manual* with the new entries only.

```

----
- ChangeBy: root@localhost
ChangeTime: 2021-09-13 13:51:29
Comment: ACL Reference for Primary/Secondary.
ConfigMatch:
  Properties:
    DynamicField:
      DynamicField_PrimarySecondary:
        - Primary
        - SecondaryOf:TICKET_NUMBER (where TICKET_NUMBER is a primary ticket number)
    Frontend:
      Endpoint:
        - AgentFrontend::Ticket::Action::AddSecondaries
        - AgentFrontend::Ticket::Action::EmailToSecondaries
        - AgentFrontend::Ticket::Action::MoveToPrimary
        - AgentFrontend::Ticket::Action::NoteToSecondaries
        - AgentFrontend::Ticket::Action::SetOnlyToPrimary
        - AgentFrontend::Ticket::Action::SetToPrimary
        - AgentFrontend::Ticket::Action::SetToSecondary
        - AgentFrontend::Ticket::Action::SmsToSecondaries
        - AgentFrontend::Ticket::Action::UnsetPrimary
        - AgentFrontend::Ticket::Action::UnsetSecondary
        - AgentFrontend::Ticket::Action::UpdateSecondaries
    PropertiesDatabase:
      # Match properties (existing values from the database).
      # Please note that Frontend is not in the database, but in the framework.
      # See section "Properties", the same configuration can be used here.
ConfigChange:
  Possible:
    # Reset possible options (white list).
    Endpoint:
      # Limit the functions on agent interface.
      - AgentFrontend::Ticket::Action::AddSecondaries
      - AgentFrontend::Ticket::Action::EmailToSecondaries
      - AgentFrontend::Ticket::Action::MoveToPrimary
      - AgentFrontend::Ticket::Action::NoteToSecondaries
      - AgentFrontend::Ticket::Action::SetOnlyToPrimary
      - AgentFrontend::Ticket::Action::SetToPrimary
      - AgentFrontend::Ticket::Action::SetToSecondary
      - AgentFrontend::Ticket::Action::SmsToSecondaries
      - AgentFrontend::Ticket::Action::UnsetPrimary
      - AgentFrontend::Ticket::Action::UnsetSecondary
      - AgentFrontend::Ticket::Action::UpdateSecondaries
    Ticket:
      # Possible ticket options (white list).
      DynamicField_PrimarySecondary:
        - Primary
        - SecondaryOf:TICKET_NUMBER (where TICKET_NUMBER is a primary ticket number)
  PossibleAdd:
    # Add options (white list).
    # See section "Possible", the same configuration can be used here.

```

(continues on next page)

(continued from previous page)

```
PossibleNot:
  # Remove options (black list).
  # See section "Possible", the same configuration can be used here.
CreateBy: root@localhost
CreateTime: 2021-09-13 13:51:29
Description: This reference lists the possible endpoints added by Primary/Secondary.
ID: 1
Name: 201-ACL-Reference
StopAfterMatch: 0
ValidID: 3
```

16.2 Agent Interface

This package adds new actions to the ticket detail view to set and change the primary/secondary status and to update the secondary tickets. Bulk actions are available to do the same in ticket lists.

Note: The ticket handling with *Primary / Secondary* has significantly changed in comparison to *Master / Slave*. The major changes are:

- Changed: Normal ticket updates will not be applied to secondary tickets. The agent have to use the primary/secondary ticket actions to replicate the actions done in a primary ticket.
- New: Beside replications now the agent can add information and/or messages to the secondary tickets only without adding the information or messages to the primary ticket.
- New: Added bulk actions.
- New: Filter for primary/secondary tickets in any ticket list.

16.2.1 Create Primary / Secondary Relation

To create a primary ticket:

1. Go to the ticket detail view.
2. Select the *Set to Primary* or the *Set Only to Primary* action in the *Actions* menu.
3. Click on the *Set* button.

The *Set to Primary* action makes it possible to set secondary tickets at the same time as well as to add a message if needed. If the *Properties* section is configured, additional information can be added.

Using the *Set Only to Primary* action sets the ticket as primary immediately without confirmation. In this case, secondary tickets can be added later.

To create a secondary ticket:

1. Go to the ticket detail view.
2. Select the *Set to Secondary* action in the *Actions* menu.
3. Select a primary ticket from the list and add a message if needed.
4. Click on the *Set* button.

It is possible to move a primary ticket as secondary ticket to a different primary ticket. In this case the old secondary tickets of the primary ticket will be unset.

To move a secondary ticket to a different primary ticket:

1. Go to the ticket detail view of the secondary ticket.
2. Select the *Move to Different Primary* action in the *Actions* menu.
3. Select another primary ticket and add a message if needed.
4. Click on the *Move* button.

To unset a primary ticket:

1. Go to the ticket detail view of the primary ticket.
2. Select the *Unset Primary* action in the *Actions* menu.
3. Click on the *Unset* button.

With this ticket action the primary ticket can be set to a normal ticket which is no primary or secondary any more.

With the checkbox *Keep secondary tickets as linked tickets* it can be controlled that the link between the unset primary ticket and its secondary ticket shall be kept or not.

If message has been added or if some properties are changed via the configured *Properties* section it can be controlled whether the message shall be replicated to the old primary ticket and/or to the secondary tickets. The first behavior can be set with the checkbox *Update old primary* while the latter can be set with the checkbox *Update secondaries*.

There is a *Replace Primary and Move Secondaries* button which makes possible to move the secondary tickets to an other primary ticket. Without this, the secondary status of the secondary tickets will be also unset.

To unset a secondary ticket:

1. Go to the ticket detail view of the secondary ticket.
2. Select the *Unset Secondary* action in the *Actions* menu.
3. Click on the *Unset* button.

If the *Keep primary ticket as linked ticket* checkbox is checked, only the secondary status of the ticket will be removed and the existing link to the primary ticket will be kept.

The primary/secondary relations are based on *Primary / Secondary* dynamic field using the *ParentChild* link type.

16.2.2 Update or Replicate Updates to Secondaries

Normal ticket updates will not be applied to secondary tickets. The agent have to use the primary/secondary ticket actions to replicate the actions done in a primary ticket.

For updating secondary tickets or replicating information from the primary ticket to the secondary tickets the following ticket actions are available:

- Update Secondaries
- Add Note to Secondaries
- Send Email to Secondaries
- Send SMS to Secondaries (if SMS is available in the system)

To update secondary tickets from a primary ticket:

1. Go to the ticket detail view of the primary ticket.
2. Select the *Update Secondaries* action in the *Actions* menu.
3. Change the priority to *1 very low* and add a message.
4. Check the *Update primary ticket* checkbox.
5. Go to the ticket detail view of the secondary ticket.
6. The priority of the secondary ticket has been also changed to *1 very low* and the same article has been added to this ticket as the primary ticket has.

For this example the priority of the primary ticket has been changed, but there are other activities that can be done instead to test this behavior, like change the state, the owner, the responsible, the lock status, reply the ticket, add a note or close the ticket.

When a new article is created on the primary ticket, the secondary tickets replicate the article using the own customer user.

When updating the secondary tickets, the agents can decide if they want to update the primary ticket as well. There is a checkbox in each update action of the secondary ticket to control this behavior.

16.2.3 Primary / Secondary Bulk Actions

A new primary/secondary icon has been added to the breadcrumb bar of each ticket lists to enable the bulk action. The bulk action screen enables the following actions:

Add to New Primary

To add the selected tickets to a newly created primary ticket.

Unset Primary

To unset the current primary ticket.

Set to Secondary

To assign the selected tickets as secondary ticket of a primary ticket.

Unset Secondary

To unset secondary tickets.

Using this bulk action screen is similar to other bulk actions. The agents have to have proper permission to tickets that are affected by the bulk action.

There are two options that may appear while using bulk actions. Displaying the options depends on the chosen primary/secondary bulk action and on the primary/secondary status.

Move secondary tickets to new primary

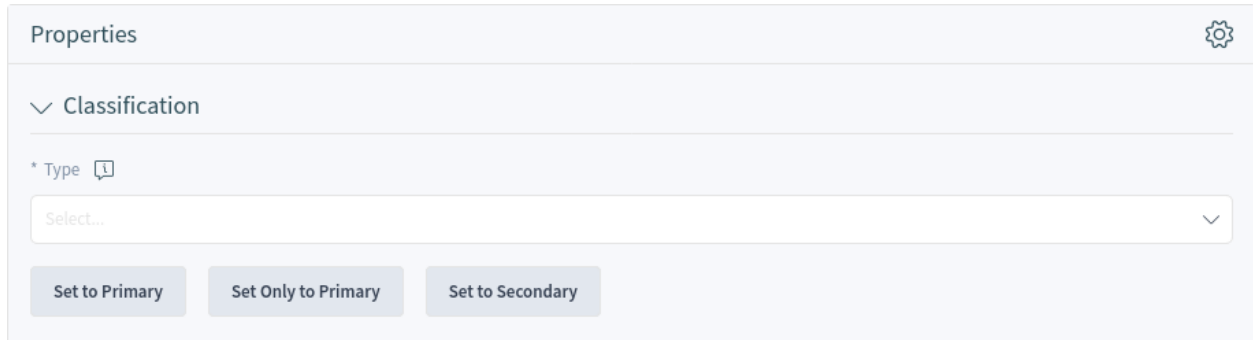
If this option is checked, the secondary tickets of the old primary ticket will be moved to the new primary ticket. Without this, the secondary status of the secondary tickets will be unset.

Keep secondary tickets as linked tickets

If this option is checked, the existing links between the tickets will be kept.

16.2.4 Ticket Create Screens

If the *Primary / Secondary* dynamic field is added to the ticket create screen, the agent can set the primary/secondary status of the new ticket at creation time.



The screenshot shows a configuration window titled "Properties" with a gear icon in the top right corner. Under the "Classification" section, there is a field labeled "* Type" with a tooltip icon. Below this is a dropdown menu with "Select..." and a downward arrow. At the bottom, there are three buttons: "Set to Primary", "Set Only to Primary", and "Set to Secondary".

Fig. 1: Primary / Secondary Dynamic Field on Ticket Create Screen

16.2.5 Dashboard Widget

This package adds a new *Primary / Secondary* widget to the dashboard. The widget lists the primary tickets and the secondary tickets of each primary ticket. The agent can hover the mouse over the number of secondary tickets to display a popup window that lists the secondary tickets.

Using the new widget is the same as the usage of other widgets on the dashboard.

16.3 External Interface

This package has no external interface.

READY2ADOPT ITSM PROCESSES

Use the following predefined processes in the IT service management disciplines of service design, service operation and service transition:

Service Design:

- Availability Management (proactive)
- Availability Management (reactive)
- Catalog Management
- Continuity Management
- Service Design Coordination (individual design)
- Service Level Management
- Supplier Management

Service Operation:

- Access Management
- Event Management
- Incident Management
- Problem Management
- Request Fulfillment Management
- Service Validation and Testing

Service Transition:

- Change Evaluation
- Knowledge Management
- Release and Deployment Management
- Service Asset and Configuration Management
- Service Transition Planning and Support

Benefits

- Continual service improvement (CSI) through pre-defined ITIL processes.
- Higher efficiency and quality through proven ITIL-inspired processes.

Target Groups

- Customer service

- Facility management
- IT
- IT service
- Security management
- Service providers

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSReady2AdoptITSMProcesses

Note: This feature requires the *Configuration Management* feature.

17.1 Administrator Interface

This package has no administrator interface, but it adds the possibility to import Ready2Adopt processes.

17.1.1 Processes & Automation

After installation of the package, a list of several properly defined processes will be available in the *Process Management* screen.

Process Management

This package includes several examples of predefined processes that can help in some specific cases.

The following Ready2Adopt processes are available from the *Ready2Adopt Processes* widget:

```
Service Design::Availability Management (proactive)
Service Design::Availability Management (reactive)
Service Design::Catalogue Management
Service Design::Continuity Management
Service Design::Service Design Coordination (individual design)
Service Design::Service Level Management
Service Design::Supplier Management
Service Operation::Access Management
Service Operation::Event Management
Service Operation::Incident Management
Service Operation::Problem Management
Service Operation::Request Fulfillment Management
Service Operation::Service Validation and Testing
Service Transition::Change Evaluation
Service Transition::Knowledge Management
Service Transition::Release and Deployment Management
Service Transition::Service Asset and Configuration Management
Service Transition::Service Transition Planning and Support
```

To import a Ready2Adopt process:

1. Go to the *Process Management* screen in the administrator interface.
2. Select a process from the *Ready2Adopt Processes* widget in the left sidebar.
3. Click on the *Import Ready2Adopt process* button.
4. Deploy all processes.

During the import process, the system takes care of creating the needed dynamic fields and/or any needed updates to the system configuration.

17.2 Agent Interface

This package has no agent interface.

17.3 External Interface

This package has no external interface.

READY2ADOPT WEB SERVICES

Not only do you use a single instance of **OTRS**, but you may also use additional **OTRS** installations and other tools. This can make accessing data inconvenient. Luckily, the following web services provide a workaround:

- Jira connector
- Bugzilla connector
- OTRS-OTRS connector

Benefits

- All data is centrally accessible in **OTRS**.
- Execute arbitrary complex mappings through the XSLT mapping module.

Target Groups

- Customer service
- Development
- IT
- IT service
- Security management
- Service providers

Available in Service Package

- GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSReady2AdoptWebServices

Note: This feature requires the *Ticket Invoker* feature.

18.1 Administrator Interface

This package has no administrator interface, but it adds the possibility to import Ready2Adopt web services.

18.1.1 Processes & Automation

After installation of the package, a list of several properly defined web services will be available in the *Web Service Management* screen.

Web Services

This package includes several examples of predefined web services created by experts.

The following Ready2Adopt web services are available from the *Ready2Adopt Web Services* widget:

- `BugzillaConnector` to create or update bugs on a remote Bugzilla server.
- `JIRAConnector` to create or update issues on a remote JIRA server.
- `OTRSConnector` to create or update tickets on a remote OTRS server.

To import a Ready2Adopt web service:

1. Go to the *Web Service Management* screen in the administrator interface.
2. Select a web service from the *Ready2Adopt Web Services* widget in the left sidebar.
3. Click on the *Import Ready2Adopt web service* button.

During the import process, the system takes care of creating the needed dynamic fields and/or any needed updates to the system configuration.

18.2 Agent Interface

This package has no agent interface.

18.3 External Interface

This package has no external interface.

RESTRICT CUSTOMER DATA VIEW

Service organizations that go global or have a wide range of products have to outsource their customer service to distribution partners or call centers. When that happens, the security of customer data is very important and only the appropriate third party providing the service should have access to customer information.

This feature makes this kind of defined access possible by assigning customer IDs to partner IDs. It enhances and improves the customer detail view of the standard OTRS version and specifies whose customer data can be accessed by all agents. When customer data is retrieved from any LDAP back end or the local database, the feature checks for a partner ID in the agent's user data which must originate from an LDAP agent back end; then, only returns those customer users whose customer IDs are assigned to the agent's partner ID. Therefore, agents of a partner ID can only see customer data of belonging to the assigned customer ID. If there is no customer ID assigned to a partner ID, the agents are not allowed to see any customer data. If no partner ID has been created, all agents can see all customer data.

The first configuration steps after the installation are the configuration of the LDAP server connection and the mapping of partner IDs to customer IDs with the help of a graphical interface. The administration of those connections can also be managed in a separate dashboard.

Benefits

- Specific allocation of access permissions allows protection of your customer data.
- Simplifies the collaboration with subsidiaries, subcontractors and partners.

Target Groups

- International companies
- Call centers
- Companies offering a wide range of products

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSRestrictCustomerDataView

19.1 Administrator Interface

This package offers the possibility to restrict the access of certain agents to data of certain customers following a mapping table.

19.1.1 Users, Groups & Roles

After installation of the package, a new module *Partner ID Customer ID* appears in the *Users, Groups & Roles* group in the administrator interface. Here you can define which partners should be assigned to customers.

PartnerID CustomerID

Use this screen to manage mappings between partners and customers. The management screen is available in the *PartnerID CustomerID* module of the *Users, Groups & Roles* group.

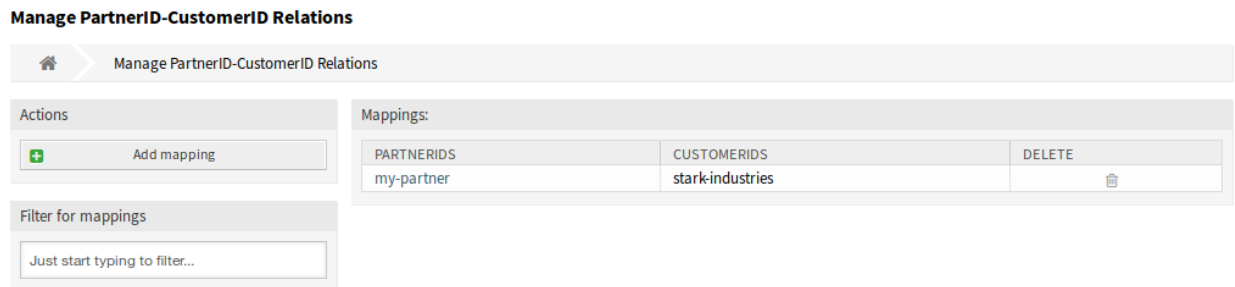


Fig. 1: Manage PartnerID-CustomerID Relations Screen

Manage PartnerID CustomerID Relations

Note: It is necessary to setup the LDAP server configuration in the system configuration.

To setup the LDAP server configuration:

1. Go to the *System Configuration* screen.
2. Select *OTRSRestrictCustomerDataView* in the *Navigation* widget.
3. Navigate to *Core* → *LDAP* in the navigation tree.
4. Set the parameters for the LDAP connection.

If the LDAP server can be reached, the `customer_id` relations to the `partner_id` which is pulled from the LDAP server, will be stored in the database tables which are configured in the `PartnerIDMapping` section within the system configuration.

To see the `PartnerIDMapping` settings:

1. Go to the *System Configuration* screen.

2. Select *OTRSRestrictCustomerDataView* in the *Navigation* widget.
3. Navigate to *Core* → *PartnerIDMapping* in the navigation tree.
4. See the settings.

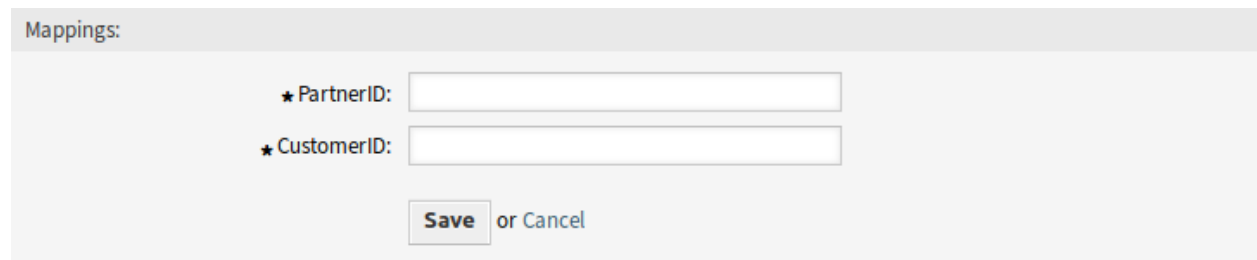
Following the relations stored in the mapping tables, access to data of certain customers is restricted for agents which have a partner ID and relations in the database. Those agents can only see customers which have customer IDs that are allowed in the database.

If a partner ID of an agent is stored in the database, but no related customer IDs are assigned, the agent won't be able to see any customer information. Agents with no given partner ID will still be able to see every customer data.

If the LDAP server configuration and the partner ID mapping is properly set, you can proceed with the management screen.

To create a new mapping:

1. Click on the *Add mapping* button in the left sidebar.
2. Add a partner ID into the first field.
3. Add a customer into the second field. Just start typing, the field has auto-completion support. The added customer IDs will be displayed below the text field.
4. Click on the *Save* button.



Mappings:

★ PartnerID:

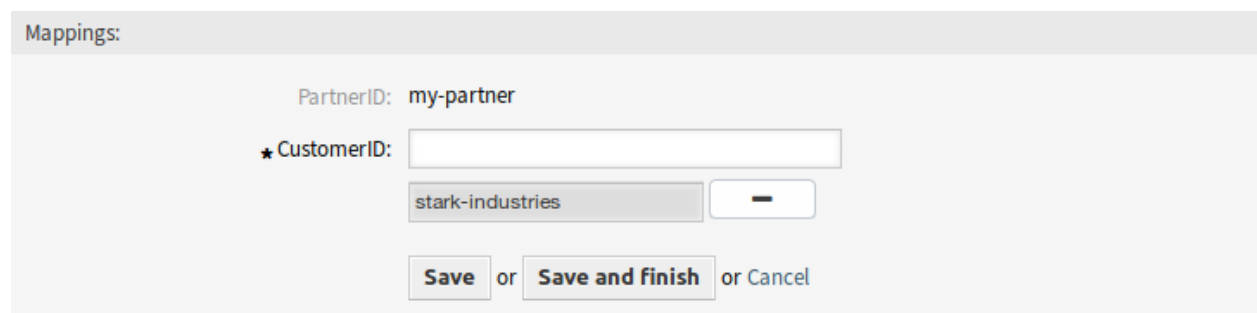
★ CustomerID:

or

Fig. 2: Add Mapping Screen

To edit a mapping:

1. Click on a mapping in the list of mappings.
2. Add or remove customers.
3. Click on the *Save* or *Save and finish* button.



Mappings:

PartnerID: my-partner

★ CustomerID:

stark-industries

or or

Fig. 3: Edit Mapping Screen

The *PartnerID* field is read-only. If you would like to edit the partner ID, you can delete the mapping and create a new one with the other partner ID.

To delete a mapping:

1. Click on the trash icon in the last column of the overview table.
2. Click on the *OK* button in the confirmation dialog.



PARTNERIDS	CUSTOMERIDS	DELETE
my-partner	stark-industries	

Fig. 4: Delete Mapping Screen

Note: If several mappings are added to the system, use the filter box to find a particular mapping by just typing the name to filter.

Mapping Settings

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

PartnerID *

The ID of a partner as stored in the LDAP.

CustomerID *

The internal ID of a customer as stored in the OTRS database.

19.2 Agent Interface

This package has no agent interface.

19.3 External Interface

This package has no external interface.

SAP SOLUTION MANAGER CONNECTOR

This package includes extension modules for the generic interface that provides two-way communication and ticket synchronization between OTRS and SAP Solution Manager.

Note: SAP Solution Manager (reachable via local or remote network) is required for this feature.

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSSAPSolutionManagerConnector

20.1 Administrator Interface

This chapter describes the new features that are available in the administrator interface after installation of the package.

20.1.1 Bidirectional Communication

The modules included in this package provide communication between OTRS and a *SAP Solution Manager* system (from now on referred to as *SolMan* in this document), however it is necessary to first setup both systems independently in order make sure that both systems work correctly and then execute all the necessary steps in the SolMan system to request and provide actions trough a non-SAP web service.

With this package (and the necessary customized configuration) OTRS will be able to communicate with an existing SolMan by pretending to be another SolMan system, using the same function names, data structures and value types.

OTRS does not implement the full set of SolMan API. Please refer to the [Web Services](#) chapter to check which ones are available in OTRS.

Prior to any attempt to creating an OTRS web service with this package modules is necessary to know how SolMan handle concepts like priorities, states (`SAPUserStatus`), article types (`TextTypes`), etc. and the values for this concepts in the used instance of SolMan system, as well as in OTRS.

OTRS and the SolMan system should be able to communicate over the network to send and receive SOAP messages between them. Please be sure to open the necessary ports on both servers and any network device between them (like a switch or firewall).

In order to use this package and the contained modules is necessary to create or modify a web service within OTRS. Please refer to the [Web Services](#) chapter for more information.

20.1.2 Automatic Creation of Customer Users and Agents in OTRS

Tickets exchanged between OTRS and SolMan also transfer ticket owner and customer user.

If an incoming ticket creation/update contains data of an unknown agent, a new agent will be automatically created in OTRS. The same behavior exists similarly for customer users.

20.1.3 Multiple SolMan Instances

It is possible to connect to multiple SolMan instances from within the same OTRS system. Just use one web service per instance.

The system GUID of each system is stored separately and so is the ticket synchronization state.

Processes & Automation

This package makes it possible to connect OTRS with SolMan using web services.

Web Services

As mentioned in the feature list, a basic knowledge of both OTRS and SolMan web services is assumed.

An example web service is provided with the package in `doc/SolManConnectorExample.yml`. It might be a good idea to keep this web service installed (but disabled) as a duplicate on your system as a reference.

For necessary and recommended modifications of each web service part see sections below. Modifications refer to the provided example.

Requester Transport

Be sure to use `HTTP::SOAPSolMan` transport. This is necessary for correct namespace generation.

Set the endpoint (URI) of your SolMan instance and set authentication, proxy and SSL options if necessary.

The `SOAPAction` options, namespace and name scheme should not be changed.

Requester Invokers

This package includes three different invoker controllers that provide functionality for all supported function calls.

- The `SolMan::IncidentCreate` controller handles initial transfer of a ticket to SolMan (e.g. `ProcessIncident`, `ReplicateIncident`).
- The `SolMan::IncidentUpdate` controller handles follow-up transfer of a ticket to SolMan (e.g. `AcceptIncidentProcessing`, `AddInfo`, `CloseIncident`, `RejectIncidentSolution`, `VerifyIncidentSolution`).

- The `SolMan::RequestSystemGuid` controller is used to request system GUID information from SolMan.

It is important to differentiate between `IncidentCreate` and `IncidentUpdate` controllers. Otherwise the invoker names do not make a difference handling-wise in OTRS (except `CloseIncident` which will croak if you try to update an already closed ticket).

If an update-type invoker is triggered for a ticket without any previous sync activity, they will automatically trigger a create type invoker (`ProcessIncident` if it exists or alternatively `ReplicateIncident`).

The invoker for requesting the SolMan system GUID is triggered automatically whenever this information is not known in OTRS.

Every regular invoker will transfer all existing articles that have not been transferred to this point.

By default, the `AddInfo` invoker will be triggered for every article created in OTRS via an event trigger. It is highly recommended to configure event triggers for all relevant invokers (e.g. for state and priority changes) and to restrict them via event conditions (e.g. to tickets in specific queues).

Requester Mapping

Mapping between OTRS and SolMan is done in OTRS via XSLT configuration.

The provided configuration contains working XSLT templates. For standard requirements it is only necessary to modify the outbound mapping template in regard to:

- OTRS file path for XSL helper templates
- OTRS → SolMan state mapping
- OTRS → SolMan priority mapping
- Dynamic field selection and name mapping
- OTRS → SolMan article type mapping

The templates have to be modified individually per invoker but are (by default) identical for all regular invokers.

The data structure provided by the invokers is similar to the structure of the OTRS ticket connector but with some modifications and extra data. This is a quick reference:

```
<Customer>
  <*> customer attributes, e.g. 'UserEmail' </*>
</Customer>
<Owner>
  <*> owner (agent) attributes, e.g. 'UserEmail' </*>
</Owner>
<SolMan>
  <IncidentId> incident id </IncidentId>
  <IncidentGuid> incident guid </IncidentGuid>
  <LocalSystemGuid> OTRS system guid </LocalSystemGuid>
  <RemoteSystemGuid> SolMan system guid </RemoteSystemGuid>
</SolMan>
<Ticket>
  <Article> # can occur 0-n times
    <Attachment> # can occur 0-n times
      <*> attachment attributes, e.g. 'Filename' </*> # 'Content' is base64-
      ↪ encoded automatically
    </Attachment>
    <DynamicField> # can occur 0-n times
```

(continues on next page)

(continued from previous page)

```

        <Name> dynamic field name </Name>
        <Value> dynamic field value(s) </Value>
    </DynamicField>
    <*> article attributes, e.g. 'From' </*>
</Article>
<DynamicField> # can occur 0-n times
    <Name> ticket dynamic field name </Name>
    <Value> ticket dynamic field value(s) </Value>
</DynamicField>
    <*> ticket attributes, e.g. 'TicketNumber' </*>
</Ticket>

```

The default XSLT template types are provided as individual files for reference (see `doc/ExampleMapping-Invoker-*.xsl` provided by the package).

Requester Error Handling

Some error handling modules are pre-configured used to reschedule requests if a specific problem occurs. Rescheduled requests are executed in 1 minute intervals for up to 5 tries.

By defaults all transport errors cause a reschedule as well as data preparation where the SolMan system GUID could not be retrieved.

Also requests are rescheduled if SolMan returns an error code, except if this code is 01, 02, 03 or 04.

It is recommended to change retry options to your needs and to check if the error code specific handling matches your requirements.

Provider Transport

To avoid confusion you should use the `HTTP::SOAPSolMan` transport. Contrary to the requester this is not strictly necessary.

Modify message length if necessary.

The `SOAPAction` options, namespace and name scheme should not be changed.

Provider Operations

This package includes three different operation controllers that provide functionality for all supported function calls.

- The `SolMan::IncidentCreate` controller handles initial transfer of a ticket from SolMan (e.g. `ProcessIncident`, `ReplicateIncident`).
- The `SolMan::IncidentUpdate` controller handles follow-up transfer of a ticket from SolMan (e.g. `AcceptIncidentProcessing`, `AddInfo`, `CloseIncident`, `RejectIncidentSolution`, `VerifyIncidentSolution`).
- The `SolMan::RequestSystemGuid` controller is used to provide system GUID information to SolMan.

It is important to differentiate between `IncidentCreate` and `IncidentUpdate` controllers. Otherwise the invoker names do not make a difference handling-wise in OTRS.

If a ticket has pending updates that need to be transferred from OTRS to SolMan (e.g. new articles), update from SolMan are temporarily blocked.

Due to the nature of SolMan, the sent attachments cannot be assigned to more than one article. Articles sent within a request will automatically attached to the last article created within this request. If the request does not contain any article, a dummy article will be created for the attachments.

Provider Mapping

Mapping between OTRS and SolMan is done in OTRS via XSLT configuration.

The provided configuration contains working XSLT templates. For standard requirements it is only necessary to modify the inbound mapping template in regard to:

- OTRS file path for XSL helper templates
- OTRS target queue
- SolMan → OTRS state mapping
- SolMan → OTRS priority mapping
- Dynamic field selection and name mapping
- SolMan → OTRS article type mapping

The templates have to be modified individually per operation but are (by default) identical for all regular operations.

The data structure expected by the operations is similar to the structure of the OTRS ticket connector but with some modifications and extra data. This is a quick reference:

```
<Attachment> # can occur 0-n times
  <*> attachment attributes, e.g. 'Filename' </*> # 'Content' is base64-decoded,
  ↪ automatically
</Attachment>
<Ticket>
  <Article> # can occur 0-n times
    <*> article attributes, e.g. 'From' </*>
  </Article>
  <DynamicField> # can occur 0-n times
    <Name> ticket dynamic field name </Name>
    <Value> ticket dynamic field value(s) </Value>
  </DynamicField>
  <*> ticket attributes, e.g. 'TicketNumber' </*>
</Ticket>
<SolMan>
  <IncidentGuid> incident guid </IncidentGuid>
  <LocalSystemGuid> OTRS system guid </LocalSystemGuid>
  <RemoteSystemGuid> SolMan system guid </RemoteSystemGuid>
  <Person> # can occur 0-n times
    <Meta>
      <ID> OTRS id of agent / or login of customer user </ID>
      <ExternalID> SolMan id of person </ExternalID>
      <Type> CustomerUser, Owner or Agent </Type>
    </Meta>
    <*> agent or customer user attributes, e.g. 'UserEmail' </*>
  </Person>
</SolMan>
```

The default XSLT template types are provided as individual files for reference (see `doc/ExampleMapping-Operation-*.xsl` provided by the package).

20.2 Agent Interface

This package has no agent interface.

20.3 External Interface

This package has no external interface.

SPECIFIC TICKET NOTIFICATIONS

As part of standard functionality, agents with administrator permissions can create automated notifications to be sent to a certain group of recipients of agents and/or customers. These are standard notifications, e.g. for when the owner of the ticket has changed or the ticket has been moved to another queue.

Now, with the feature, agents without administrator privileges can also create individual, event-based notifications, albeit with the limitation that they can only be sent to their own email address. For example, an IT-service employee can configure individual notifications in order to be informed as soon as there is a change in the priority of a specific customer's ticket.

Benefits

- Stay informed regarding ticket changes, even when working outside of the agent interface.
- React to changes more quickly.
- More personalized configuration of notifications, avoiding a flood of information.

Target Groups

- Internal and external IT
- Customer service
- Sales
- Finance

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSSpecificTicketNotifications

21.1 Administrator Interface

This package has no administrator interface, but extends *Ticket Notifications* screen with new buttons and adds a new group to the system.

21.1.1 Users, Groups & Roles

After installation of the package a new group is added to the system.

Groups

Access to the personal notification module is managed on a role-based access concept. After installation of the package a new group is added to the system. The group management screen is available in the *Groups* module of the *Users, Groups & Roles* group.

The screenshot shows the 'Group Management' interface. On the left, there are three sections: 'Actions' with a '+ Add Group' button, 'Filter for Groups' with a search input field containing 'Just start typing to filter...', and a 'Hint' section with text: 'The admin group is to get in the admin area and the stats group to get stats area. Create new groups to handle access permissions for different groups of agent (e. g. purchasing department, support department, sales department, ...). It's useful for ASP solutions.' The main content area is a table titled 'List (9 total)' with the following data:

NAME	COMMENT	VALIDITY	CHANGED	CREATED
admin	Group of all administrators.	valid	11/11/2018 15:29 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)
custom_notifications	Group for all custom notification users.	valid	07/30/2019 12:32 (Europe/Budapest)	07/30/2019 11:56 (Europe/Budapest)
stats	Group for statistics access.	valid	11/11/2018 15:29 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)
users	Group for default access.	valid	11/11/2018 15:29 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)

Fig. 1: Group Management Screen

New Group

After installation of the package the following group is added to the system:

custom_notifications

Every agent who is a member of this group gets a *Create Notification* button in the *Ticket Notifications* widget of the *Personalization* menu.

Note: The primary administrator user (`root@localhost`) is added to all groups with permission *rw* by default.

See also:

To set the correct permissions for other agents, check the following relations:

- *Agents* *Groups*
- *Roles* *Groups*

21.1.2 Communication & Notifications

After installation of the package a new functionality will be available in the *Ticket Notifications* screen of the *Communication & Notifications* group in the administrator interface.

Ticket Notifications

This screen gets two new buttons in the left sidebar and a new column in the overview table.

The screenshot shows the 'Ticket Notification Management' interface. On the left, there is a sidebar with 'Actions' (Add Notification, Export Notifications, Add personal notification, List my personal notifications only), 'Filter for Notifications' (Just start typing to filter...), and 'Configuration Import' (Browse..., No file selected., Overwrite existing notifications?, Import Notification configuration). The main area is a table with the following data:

NAME	COMMENT	VALIDITY	CHANGED	CREATED	PERSONAL	EXPORT	COPY	DELETE
Customer ticket article notification		valid	11/11/2018 15:29 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)				
Customer ticket create notification		valid	05/30/2019 15:44 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)				
Customer ticket state update (closed)		valid	11/11/2018 15:29 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)				
Customer ticket state update (other than closed)		valid	11/11/2018 15:29 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)				
Supervisor	To view newly created a...	valid	07/30/2019 13:47 (Europe/Budapest)	07/30/2019 13:47 (Europe/Budapest)	Yes			
Ticket create notification		valid	11/11/2018 15:29 (Europe/Budapest)	11/11/2018 15:29 (Europe/Budapest)				

Fig. 2: Ticket Notification Management Screen

Administrators are still able to see all available notification entries. To be able to differentiate between entries of administrators and agents, there is a new column *Personal* in the notification overview. The entries of agents are marked in orange and have the value *Yes* in the column *Personal*.

Furthermore it is still possible to add other agents as email recipients in every new or modified notification entry, including notification entries of agents.

Manage Personal Ticket Notifications

To add a personal ticket notification:

1. Click on the *Add personal notification* button in the left sidebar.
2. Fill in the required fields.
3. Click on the *Save* button.

To edit a personal ticket notification:

1. Click on a ticket notification in the list of ticket notifications in a row that is displayed in orange.

2. Modify the fields.
3. Click on the *Save* or *Save and finish* button.

To delete a personal ticket notification:

1. Click on the trash icon in the list of ticket notifications in a row that is displayed in orange.
2. Click on the *Confirm* button.

To list only the personal notifications:

1. Click on the *List my personal notifications only* button in the left sidebar.
2. Click on the *List all notifications* button to restore the overview.

Adding a notification can be done with the same procedure that is described in the [Ticket Notifications](#) chapter of the administration manual. There is only one difference: the *Recipients* section.



Fig. 3: Ticket Notification Settings - Recipients

In case of adding a personal ticket notification, only the personal email address of the administrator or agent can be selected in the *Recipients* section.

21.2 Agent Interface

After installation of the package it is possible to create own notifications, like administrators can do, except for the recipient addresses. Every event which is created by an agent will send emails to the agents own primary email only.

21.2.1 Personalization

This package extends the *Ticket Notifications* widget in the *Notification Settings* menu item of the *Personalization* menu.

Notification Settings

Every agent who is a member of the group `custom_notifications` is able to create specific ticket notifications. The *Create Notification* button in the bottom of the *Ticket Notifications* widget points to the *Ticket Notifications* screen of the administrator interface.

To add a personal ticket notification:

1. Click on the *Create Notification* button and login to the administrator interface with your password.
2. Click on the *Add Notification* button as an agent or on the *Add personal notification* button as an administrator in the left sidebar if the *Add Notification* screen is not opened.
3. Fill in the required fields.
4. Click on the *Save* button.



Fig. 4: Ticket Notifications Widget

The *Ticket Notification Management* screen does not contain the administrative elements, if an agent opens it.



Fig. 5: Ticket Notification Management Screen

To edit a personal ticket notification:

1. Click on a ticket notification in the list of ticket notifications. If you are an administrator, the personal ticket notifications are displayed in orange rows.
2. Modify the fields.
3. Click on the *Save* or *Save and finish* button.

Alternatively the agent can use the gear icon in the ticket notification list from the agent interface. This icon provides a direct link to the specific ticket notification to be edited.

To delete a personal ticket notification:

1. Click on the trash icon in the list of ticket notifications.
2. Click on the *Confirm* button.

The agent is able to create his own notifications, like administrators can do, except the recipient addresses. Every event that an agent creates, will send emails to its own primary email. This recipient address is mandatory and cannot be switched to another one.

The related agent is only able to see, create, edit and/or delete his own notification entries. It is not possible to see or modify notification entries of other agents.

21.3 External Interface

This package has no external interface.

TICKET ALLOCATION

Service organizations that need to handle a high number of tickets in a short time span and call centers that want to assign tickets automatically to staff members will both be excited about this feature. New tickets are automatically assigned to agents with the smallest number of tickets to work on or to agents with suitable skills. Your service team will efficiently operate at full capacity and will be able to answer more tickets without feeling overburdened. This feature turns **OTRS** into a call center software.

In the generic agent job configuration you can define the event that will trigger the ticket allocation. For example, the trigger could be the creation of a ticket, the reaching of a reminder time, or the closing of a ticket. Moreover, you can decide if only the ticket that triggered the event is allocated or if all relevant tickets are allocated to a specific agent.

You can also define which tickets are relevant. To avoid flooding your agents with tickets, you can restrict the number of allocated tickets per agent. Ticket ordering can also be configured with an additional *order module*. By default, it is the ticket creation time, but you can also choose queue priority or service times. Additionally, a maximum number of allocated tickets per queue can be defined.

Last but not least, you can also create competence groups in order to react more quickly to emergency tickets or to assign tickets to a team with special knowledge of tricky issues right from the start. By default the queue, priority, type, and group competences are active; however, you can also add SLA and service in the system configuration.

Benefits

- Optimal capacity utilization of your service team by automatic allocation of tickets.
- Simple handling of an increased number of tickets.

Target Groups

- IT service
- Sales
- Facility management
- Internal IT
- And many more

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSTicketAllocation

22.1 Administrator Interface

The ticket allocation is the core feature of this package. It will automatically set an owner and a configurable lock for tickets. The events that trigger an allocation check can be defined in the *Events* widget of the *Generic Agent* job management screen. Each event can be configured to allocate only the triggering ticket or all relevant tickets. Which tickets are relevant for ticket allocation can be defined by setting the ticket filters in the generic agent job.

The possible owners of a ticket have to be in the configured group of the ticket queue. Agents can be excluded from the allocation by being member in one of the configured blacklist groups. The allocation can be influenced by the competence feature (if enabled).

The competences will get summed for each agent. The group-competences will be mapped to the queue group of the ticket, and all other competences to its corresponding ticket attribute (queue, priority etc.). If all/some agents have the same competence score or the competences feature is disabled, the agent with the least amount of tickets will be set as the new ticket owner. The competences can be set for each *Agents* individually.

It is possible to define a maximum number of tickets per agent via the system configuration to prevent flooding the agent with tickets. If the configured value is reached for an agent no new ticket will get allocated to her/him. It is possible to loose this rule for tickets that get manually allocated to an agent via the system configuration, too.

By default only on-line agents will get tickets allocated. This behavior can also be changed by a system configuration setting.

22.1.1 Ticket Settings

After installation of the package a new module will be available in the *Ticket Settings* group of the administrator interface.

Competence Levels

Use this screen to add competence levels to automatic allocation. The competence level management screen is available in the *Competence Levels* module of the *Ticket Settings* group.

Competence Level Management

Actions

+ Add Competence Level

Filter for Competence Level

Just start typing to filter...

NAME	LEVEL	VALIDITY	CHANGED	CREATED
1 very low	1	valid	02/17/2021 08:10	02/17/2021 08:10
2 low	2	valid	02/17/2021 08:10	02/17/2021 08:10
3 normal	3	valid	02/17/2021 08:10	02/17/2021 08:10
4 high	4	valid	02/17/2021 08:10	02/17/2021 08:10
5 very high	5	valid	02/17/2021 08:10	02/17/2021 08:10

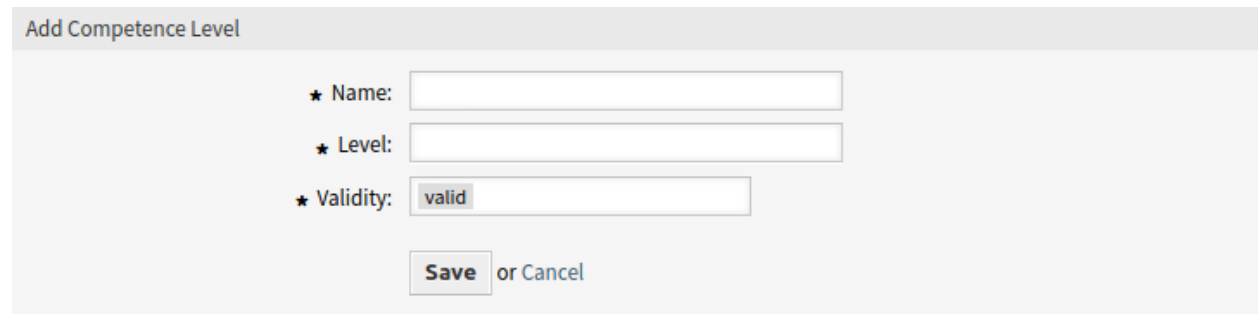
Fig. 1: Competence Level Management Screen

Manage Competence Levels

Note: When creating a customized list of competence levels, please keep in mind that they are sorted alphabetically in the competence level selection box in the user interface.

To add a competence level:

1. Click on the *Add Competence Level* button in the left sidebar.
2. Fill in the required fields.
3. Click on the *Save* button.



Add Competence Level

★ Name:

★ Level:

★ Validity:

or

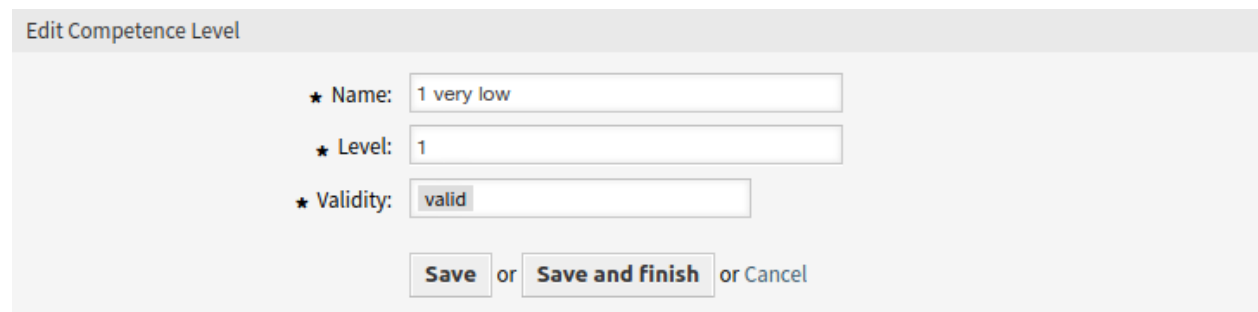
Fig. 2: Add Competence Level Screen

Warning: Competence levels can not be deleted from the system. They can only be deactivated by setting the *Validity* option to *invalid* or *invalid-temporarily*.

Note: It's recommended to limit your system to 5 competence levels or less and reuse the current 5 to keep the use of the ticket allocation system.

To edit a competence level:

1. Click on a competence level in the list of competence levels.
2. Modify the fields.
3. Click on the *Save* or *Save and finish* button.



Edit Competence Level

★ Name:

★ Level:

★ Validity:

or or

Fig. 3: Edit Competence Level Screen

Note: If several competence levels are added to the system, use the filter box to find a particular competence level by just typing the name to filter.

Competence Level Settings

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Name *

The name of this resource. Any type of characters can be entered to this field including uppercase letters and spaces. The name will be displayed in the overview table.

Level *

The level of the competence.

Validity *

Set the validity of this resource. Each resource can be used in OTRS only, if this field is set to *valid*. Setting this field to *invalid* or *invalid-temporarily* will disable the use of the resource.

22.1.2 Processes & Automation

After installation of the package a new generic agent job will be added to the system.

Generic Agent

The ticket allocation functionality is based on the event based ticket actions feature from the generic agent. The job can be executed for all matching tickets or just the ticket that has triggered the event. Additionally it is possible to define custom order modules and pre action modules for the generic agent job. This package provides one order and pre action module to allocate tickets. It uses the triggers to execute the allocation process.

After installation of the package a generic agent job called *TicketAllocation* is added to the system. This job is a basic ticket allocation configuration that can be customized and extended.

The filtering of tickets has not changed from the default generic agent behavior. For the most cases the default event configuration of the ticket allocation job did not have to be changed. The events should be self-explanatory. They represent the different ticket actions and the login/logoff of agents.

By default the job is executed via the OTRS daemon to avoid blocking the agent and preventing infinite allocation loops. Additionally it is necessary to set the prevent infinite loops to *Yes*.

By default the *TicketAllocation* pre action module will set the ticket lock. This will trigger another `TicketLockUpdate` event, which will start a new job and so on. The `TicketLockUpdate` event is required to make it possible to react on automatic and manual ticket unlocks to reallocate tickets. It is recommended to keep these settings.

In the *Execute Custom Modules* widget are two select fields which define if an *Order Module* and/or *Pre Action Module* should get executed. This package adds one order module and pre action module to the system.

The order module is not mandatory to use for the allocation functionality. There are a couple of system configuration settings that can influence the order of the tickets. If the order should get manipulated it is necessary to use the *TicketAllocation* order module. The pre action module is necessary to manipulate the

▼ Events

Event Triggers:

TYPE	EVENT	ALL TICKETS	DELETE
Ticket	TicketCreate	<input type="checkbox"/>	
Ticket	TicketQueueUpdate	<input type="checkbox"/>	
Session	SessionCreate	<input type="checkbox"/>	
Session	SessionRemove	<input type="checkbox"/>	
Ticket	TicketStateUpdate	<input type="checkbox"/>	
Ticket	TicketLockUpdate	<input type="checkbox"/>	

Additionally or alternatively to a periodic execution, you can define ticket events that will trigger this job. The checkbox defines whether the job should search for all matching tickets or just take the one that has triggered the event.

Add Event Trigger:

To add a new event select the event object and event name.

Event execution:

Defines how the job should get executed. Please refer to the package documentation for more information. Attention: Misconfiguration can lead to endless loops.

Prevent infinite loops:

Defines whether the job can get executed in another event or not. Attention: Misconfiguration can lead to endless loops.

Fig. 4: Events Section

Execute Custom Modules

Order module:

Use a module for custom ordering of the tickets that matched the filter criteria.

Pre action module:

Use a module for condition based setting of ticket actions. Attention: Ticket actions may be overwritten by this module.

Execute Custom Module:

Param 1 key: Param 1 value:

Param 2 key: Param 2 value:

Param 3 key: Param 3 value:

Param 4 key: Param 4 value:

Param 5 key: Param 5 value:

Param 6 key: Param 6 value:

Fig. 5: Execute Custom Modules Section

ticket attributes that the tickets should get set. There are some system configuration settings to influence the allocation process, too.

It is possible to set additional attributes the processed tickets should get set. The *TicketAllocation* pre action module will set the owner and the lock (if configured) for the tickets, if there are no errors in the allocation process and a matching agent can be found. If no allocation can be made, it is possible to set default values in the *Update/Add Ticket Attributes* widget of the generic agent for the owner and lock. If an allocation is made these values will be overwritten.

22.1.3 Users, Groups & Roles

This package adds a new configuration screen to the agent preferences screen, which is accessible from the *Agent Management* screen of the administrator interface.

Agents

With this package it is possible to define different competences for agents.

The competences can be enabled and disabled via the system configuration. By default the ticket competences group, priority, queue and type are enabled. Role, SLA and service are disabled but can be enabled via the system configuration.

To set the competences for an agent:

1. Open the *Agents Management* screen in the administrator interface.
2. Select an agent from the list.

3. Click on the *Edit personal preferences for this agent* button in the left sidebar.
4. Click on the *Competences* module in the agent personal preferences.

The screenshot displays the 'Competences' configuration interface, which is organized into four distinct sections, each with a header and a table-like structure for defining user-specific settings.

- Group competence(s):** This section contains two rows. The first row has 'admin' selected in the dropdown and '1 very low' in the value field. The second row has 'users' selected and '5 very high' in the value field. A plus icon is visible at the bottom left of the list area. To the right, there is a text prompt 'Define group competences for this user.' and a checkmark button.
- Priority competence(s):** This section is currently empty, with a plus icon at the bottom left. To the right, there is a text prompt 'Define priority competences for this user.' and a checkmark button.
- Queue competence(s):** This section contains one row with 'Misc' selected in the dropdown and '5 very high' in the value field. A plus icon is visible at the bottom left. To the right, there is a text prompt 'Define queue competences for this user.' and a checkmark button.
- Type competence(s):** This section is currently empty, with a plus icon at the bottom left. To the right, there is a text prompt 'Define type competences for this user.' and a checkmark button.

Fig. 6: Competences Screen

The competences have no direct influence on the behavior of the system. They can be used in other functionalities to make decisions based on the configured values and influence the behavior.

By default the relevant tickets will be allocated ordered by their internal ticket ID and their creation time. This package provides an order module for changing the order of the relevant tickets, based on different system configuration settings. The ticket order can be influenced by configurable queue priorities and service times.

Additionally it is possible to set a maximum number of tickets per queue in a row that should get allocated. If the limit is reached, the next different queue will be preferred. After that, the counter for the queue is reset.

See also:

Take a look at the configuration options in the system configuration for this package.

22.2 Agent Interface

Agents are able to edit they own competences, if this is enabled in the system configuration.

22.2.1 Personalization

Agents are able to edit they own competences, if this is enabled in the system configuration.

Competences

This menu item contains the same settings for the agent, that a system administrator can set for the *Agents*.

Note: The system configuration `OTRSTicketAllocation::AllowAgentEditOwnCompetences` has to be enabled to access this menu item from the *Personalization*.

22.3 External Interface

This package has no external interface.

Competences

Start typing to filter...

Group competence(s)

admin

1 very low

x v



users

5 very high

x v



Add New

Please select



Priority competence(s)

Add New

Please select



Queue competence(s)

Misc

5 very high

x v



Add New

Please select



Type competence(s)

Add New

Please select



Cancel Save

Fig. 7: Competences Selection

TICKET INVOKER

This package adds the functionality of two new invokers `TicketCreate` and `TicketUpdate` for web services.

In addition, the package follows HTTP redirect responses (HTTP code 301, 307 and 308) for outbound communication of HTTP::REST and HTTP::SOAP based web services.

Further the package allows to send empty body HTTP requests (POST, PUT, PATCH) for outbound communication of HTTP::REST based web services.

Available in Service Package

- GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSTicketInvoker

23.1 Administrator Interface

This package adds a new settings screen to the *Web Service Management* screen to configure the invokers.

23.1.1 Processes & Automation

This package adds two new invokers to *Invokers* section of the *Web Service Management* screen.

Web Services

The actions that can be performed when you are using OTRS as a requester are called *invokers*. Each invoker belongs to a controller (controllers are collections of operations or invokers). Usually invokers from the same controller need similar settings and share the same configuration dialogs. Each invoker can have independent configuration dialogs if needed.

After installation of the package, two new invokers will be available in the *Invokers* section. Selecting an invoker from the drop-down list will open a new settings window.

Request Example

`TicketCreate` and `TicketUpdate` invoker returns the complete ticket and article data based on ticket ID and article ID of the triggered event.

Prepare the invocation of the configured remote web service. Event data:

```
my $Result = $InvokerObject->PrepareRequest (
    Data => {                                     # data payload
        TicketID => 123,
        ArticleID => 123,                         # optional
    },
);
```

Invoker result:

```
{
    Data => {
        Ticket => {
            Title      => 'some ticket title',
            Queue      => 'some queue name',
            Lock       => 'some lock name',         # optional
            Type       => 'some type name',        # optional
            Service    => 'some service name',     # optional
            SLA        => 'some SLA name',         # optional
            State      => 'some state name',
            Priority    => 'some priority name',
            Owner      => 'some user login',       # optional
            Responsible => 'some user login',     # optional
            CustomerUser => 'some customer user login',
            PendingTime { # optional
                Year    => 2011,
                Month   => 12,
                Day     => 03,
                Hour    => 23,
                Minute  => 05,
            },
        },
        Article => {
            SenderType    => 'some sender type name', # optional
            AutoResponseType => 'some auto response type', # optional
            From          => 'some from string',      # optional
            Subject       => 'some subject',
            Body          => 'some body'
            ContentType   => 'some content type',     # ContentType or MIMEType
            ↳and Charset is required
            MimeType      => 'some mime type',
            Charset       => 'some charset',
            TimeUnit     => 123,                     # optional
        },
        DynamicField => [ # optional
            {
                Name    => 'some name',
                Value   => 'Value',                 # value type depends on
            },
            ↳the dynamic field
            # ...
        ],
    },
};
```

(continues on next page)

(continued from previous page)

```

Attachment => [
  {
    Content      => 'content'                # base64 encoded
    ContentType => 'some content type'
    Filename    => 'some fine name'
  },
  # ...
],
},
};

```

Note: The invoker will return the newest article of the ticket if no article ID is given.

Note: The invoker will not return dynamic fields with undefined values, because the same named operations `TicketCreate` and `TicketUpdate` does not handle dynamic fields with undefined values.

Advanced Filtering for Outgoing Data

For outgoing data, it is possible to define what kind of ticket fields, article fields or dynamic fields the request should contain. Furthermore it is possible to filter by article type and article sender type.

The different filter options can be selected within every single invoker configuration and are listed in section *Settings for outgoing request data*.

In the left part of the screen on the action column you have the options to go back to web service (discarding all changes since the last save) and delete. If you click on the last one, a dialog will open and ask you if you like to remove the invoker. Click on the *Delete* button to confirm the removal of the invoker and its configuration or click on the *Cancel* button to close the delete dialog.

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

General Invoker Data

Name *

The name of this resource. Any type of characters can be entered to this field including uppercase letters and spaces. The name will be displayed in the overview table.

Description

Add additional information to this resource. It is recommended to always fill this field as a description of the resource with a full sentence for better clarity, because the comment will be also displayed in the overview table.

Invoker backend

This invoker back end module will be called to prepare the data to be sent to the remote system, and to process its response data. The field is read-only, it was selected in the previous screen.

Invoker Details

General invoker data

★ Name:
 The name is typically used to call up an operation of a remote web service.

Description:

Invoker backend:
 This OTRS invoker backend module will be called to prepare the data to be sent to the remote system, and to process its response data.

Settings for outgoing request data

Ticket fields:

Only the selected ticket fields will be considered for the request data.

Article fields:

Only the selected article fields will be considered for the request data.

Ticket dynamic fields:

Only the selected ticket dynamic fields will be considered for the request data.

Article dynamic fields:

Only the selected article dynamic fields will be considered for the request data.

Number of articles:

The outgoing request data will only contain the configured number of articles. If left empty, only 1 article will be sent.

Communication channels:

The outgoing request data will only consider articles of the selected communication channels. If left empty, articles created by all communication channels will be used.

Customer visibility:

The outgoing request data will only consider articles created with the selected customer visibility.

Sender Types:

The outgoing request data will only consider articles created by the selected sender types. If left empty, articles created by all sender types will be used.

Mapping

Mapping for outgoing request data:

The data from the invoker of OTRS will be processed by this mapping, to transform it to the kind of data the remote system expects.

Mapping for incoming response data:

The response data will be processed by this mapping, to transform it to the kind of data the invoker of OTRS expects.

Settings for incoming response data

Remote TicketID dynamic field:

The selected ticket dynamic field is being used to store the remote TicketID. If left empty, the remote TicketID will not be stored, unless you define a system configuration value for this web service.

Ticket dynamic fields:

Only the selected ticket dynamic fields are being considered for processing the incoming response data. If left empty, no dynamic field will be processed.

or Cancel

Settings for Outgoing Request Data

Ticket fields

A multi-select menu containing the available ticket attributes (fields), that can be submitted to a remote system. Just the fields that are selected will be included in outgoing requests.

Article fields

A multi-select menu containing the available article attributes (fields), that can be submitted to a remote system. Just the fields that are selected will be included in outgoing requests.

Ticket dynamic fields

A multi-select menu containing the available ticket dynamic fields, that can be submitted to a remote system. Just the dynamic fields that are selected will be included in outgoing requests.

Article dynamic fields

A multi-select menu containing the available article dynamic fields, that can be submitted to a remote system. Just the dynamic fields that are selected will be included in outgoing requests.

Number of articles

A text field containing the maximum number of articles, that will be transmitted during an outgoing request. The articles will be selected from newest (latest) to oldest. If no number is given, just the latest article will be transmitted.

Communication channels

The outgoing request data will only consider articles of the selected communication channels. If left empty, articles created by all communication channels will be used.

Customer visibility

The outgoing request data will only consider articles created with the selected customer visibility.

Sender Types

The outgoing request data will only consider articles created by the selected sender types. If left empty, articles created by all sender types will be used.

Mapping

Normally there are two mapping configuration sections for each invoker, one for the incoming data and another one for the outgoing data. You can choose different mapping types (back ends) for each mapping direction, since their configuration is independent from each other and also independent from the invoker back end. The normal and most common practice is that the invoker uses the same mapping type in both cases, with inverted configuration. The complete mapping configuration is done in a separate screen, which depends on the mapping type.

Mapping for outgoing request data

The data from the invoker will be processed by this mapping, to transform it to the kind of data the remote system expects.

Mapping for incoming response data

The response data will be processed by this mapping, to transform it to the kind of data the invoker expects.

Settings for Incoming Response Data

It is possible to automatically save certain data of incoming responses to local dynamic fields. The different filter options can be selected within every single invoker configuration.

Remote Ticket ID dynamic field

A drop-down menu containing the available ticket dynamic fields in the system. If such a dynamic field is selected, the received ticket ID from the remote system will be used, to be saved within the selected dynamic field.

Ticket dynamic fields

A multi-select menu containing the available ticket dynamic fields in the system. All selected dynamic fields, that also available in the response data and containing values, will be saved to the local dynamic fields.

The dynamic field values of the response data will be used from the following data structure:

```
<Ticket>
  <DynamicField>..</DynamicField>
</Ticket>
```

and/or

```
<Ticket>
  <Article>
    <DynamicField>..</DynamicField>
  </Article>
</Ticket>
```

The system configuration option `GenericInterface::Invoker::Settings::ResponseDynamicField` was added as a fallback for the dynamic fields that should contain the result ticket ID of the related response data. It shall be used, if the configuration was not added via the invoker GUI and both configurations should not be used at the same time!

Event Triggers

Event triggers are events within OTRS such as `TicketCreate`, `ArticleSend`, etc. These can act as triggers to execute the invoker. Each invoker needs to have at least one event trigger registered, or the invoker will be useless, because it will never be called. Additionally a set of rules (conditions) for each event can be defined to have more control over the triggering of the events. These rules depend on the data of the object associated with the event. The asynchronous property of the event triggers define if the OTRS process will handle the invoker or if it will be delegated to the OTRS daemon.

Note: The OTRS daemon is a separate set of process that executes tasks in the background. Using this the OTRS process itself will not be affected if the remote system takes a long time to respond, if it is not available or if there are network problems. If you do not use the OTRS daemon using web services can make OTRS slow or non-responsive. Therefore it is highly recommend to use asynchronous event triggers as often as possible.

In order to access this section you have to save the current invoker by clicking on the *Save* button.

Event

This invoker will be triggered by the configured events.

Add Event Trigger

To add a new event select the event object and event name and click on the plus button. Asynchronous event triggers are handled by the OTRS daemon in background (recommended). Synchronous event triggers would be processed directly during the web request.

To add an event trigger:

1. Select the event family from the first list.
2. Select the event name from the second list.
3. Set the asynchronous property. If unchecked means that the event trigger will not be asynchronous.
4. Click on the plus button. A new event trigger will be created and it will be listed on the invoker event triggers list.

From the event triggers list each events shows if it contains conditions or not. The edit button next to the condition property allows to add or edit the current conditions of the event.

To delete an event trigger, simply locate the event trigger to be deleted in the event triggers list and click on the trash icon at the end of the row. This will open a dialog that asks you if you are sure to delete the event trigger. Click on the *Delete* button to remove the event trigger from the list, or click on the *Cancel* button to close the dialog.

Sometimes defining an event to trigger an invoker could result in many unnecessary or wrong request to a remote server. Event conditions could be set to restrict the triggering of the invoker in such cases.

To access the event settings screen where the conditions can be defined is necessary to be in the invoker screen and from there click on the edit icon next to the condition status on the event where this condition should take effect.

Within the event settings screen in the action bar there is a button to go back to the invoker screen as well as a button to remove all the event conditions. By default the screen is pre-populated with the first condition. Update the type of linking between conditions if more than one condition is planned, then change the type of linking from *Condition 1* if more than one field is planned. Both linking fields accept *and*, *or* or *xor* as values.

Fill the field name, set the matching type (*String* for exact match, *Regexp* for regular expression or *Validation Module*) and set the value to match (in case of validation module the full class name like `Kernel::GenericInterface::Event::Validation::ValidateDemo`).

To add more fields to the condition, click on the plus button in the fields header. To remove a field, click on the minus button in the field row. It is necessary to keep at least one field per condition.

To add more conditions click on the button below the last condition box. To remove a condition, click on the minus button in the condition header. It is necessary to keep at least one condition in the set. To remove all conditions use the button in the sidebar.

OTRS as Requester - HTTP::REST

If at least two invokers are added the requester transport screen for HTTP::REST is extended with two fields.

Controller mapping for Invoker ‘<InvokerName>’ *

In this setting a resource path is set. This path must be defined according to the needs of the remote web service and following its definition.

Path can contain variables in the form of `:<VariableName>` for each variable name that matches the current data (to be sent), will be replaced by the corresponding data value. This matched variable names and values will be removed from the current data. Depending on the HTTP request command the remaining data could be sent as a JSON string in the request body or as query parameters within the URI.

General Settings

Event: TicketCreate

Event type: Ticket

Asynchronous:

▼ Ticket Event Conditions

Conditions can only operate on non-empty fields.

Type of Linking between Conditions: and

Condition 1

Type of Linking: and

Fields

Name: Type: String Value:

Add New Condition

Save

Save or Save and finish or Cancel

Fig. 2: Add Web Service Invoker Event Screen

Examples for data Var1 = One, Var2 = Two, Var3 = Three and Var4 = Four.

- Controller mapping: `/Resource`, after replacements: `/Resource`, remaining data: Var1 = One, Var2 = Two, Var3 = Three and Var4 = Four
- Controller mapping: `/Resource/:Var1`, after replacements: `/Resource/One`, remaining data: Var2 = Two, Var3 = Three and Var4 = Four
- Controller mapping: `/Resource/:Var1?Param1=:Var2&Var3=:Var3`, after replacements: `/Resource/One?Param1=Two&Var3=Three`, remaining data: Var4 = Four

Valid request command for Invoker ‘<InvokerName>’

This determines the HTTP request method to use. Possible options: CONNECT, DELETE, GET, HEAD, OPTIONS, PATCH, POST, PUT and TRACE. If no command is selected, *Default command* is used.

23.2 Agent Interface

This package has no agent interface.

23.3 External Interface

This package has no external interface.

TICKET TIME UNIT DROPDOWN

For service organizations that must meet strict service level agreements (SLAs), the documentation of time units is fundamental. For this purpose, the standard version of **OTRS** provides a field without any unit that can be filled in however you wish. However, the correct documentation of the working time with this field requires a company-wide understanding and knowledge of time and billing intervals. Unfortunately, and especially in global companies with several locations, this is not always the case. This can cause working times to be documented incorrectly and be a source of confusion for customers.

The problem can be solved by using this feature. It makes it possible to set time units in a drop-down list like minutes or hours.

It is possible to define the following values:

- Selectable values
- Displayed name of value (e.g. *hour* or *minute*)
- Set default value

These definitions allow you to take into account the individual time and billing charge needs of your company. For example, when a quarter of an hour has already started, it could be calculated as a 15-minute interval automatically.

Benefits

- Reduction of error ratio when entering time units.
- Unmistakable definition of intervals and default values reduces documentation time.

Target Groups

- IT service management
- Internal IT service
- Customer service/support
- Globally active companies

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSTicketTimeUnitDropdown

24.1 Administrator Interface

This package has no administrator interface.

Note: For using this feature, it is necessary to activate the drop-down lists via system configuration.

To activate the drop-down lists:

1. Go to the *System Configuration* screen.
2. Select *OTRSTicketTimeUnitDropdown* in the *Navigation* widget.
3. Navigate to *Frontend* → *Agent* in the navigation tree.
4. Activate the settings for the drop-down lists.

To use a custom time unit:

1. Change the value of the `Name` key to the custom time unit name.
2. Add possible values to `Settings` key, where the first entry is the time in seconds and the second entry is the label to be displayed in the drop-down list.

It is possible to set default value for each setting using the `Default` key and any valid value from the **second** entry of the `Settings` key. This default value will be pre-selected in the drop-down list on the ticket screens.

Now open the *New Phone Ticket* screen or any other screen using time units on the *Agent Interface* and use the drop-down lists for hours and minutes to set the time units instead of an input field.

24.2 Agent Interface

This package transforms the *Time Units* input field into drop-down lists. The drop-down lists can be configured in the system configuration of the *Administrator Interface*.

To use the configured time units drop-down lists:

1. Open the *New Phone Ticket* screen or any other screen using time units.
2. Scroll down to the bottom of the form and check the new *Time Units* field.

24.3 External Interface

This package has no external interface.

TIME AND QUOTA MANAGEMENT

This feature is ideal for call centers, IT service providers or any company that wants to assign time or quantity-based quotas to their customers.

The feature provides a dashboard widget in the detail view of the respective customer that displays time unit quotas or volume quotas with reference numbers to tickets and the current status of the quota. The time unit and volume quotas are maintained in the CMDB and they can be assigned to the respective customers.

As an alternative to calculating times, the amount of incident reports, for example, can also be taken as the basis for calculating the number still available.

Benefits

- Simple time recording and transparent documentation in one system.
- Having an overview of current quotas facilitates planning and implementation.

Target Groups

- IT service management
- Call centers
- Service providers
- All companies

Available in Service Package

- SILVER, GOLD, TITANIUM, PLATINUM

Package Name in OTRS Package Manager

- OTRSTimeAndQuotaManagement

Note: This feature requires the *Configuration Management* feature.

This feature requires the *Ticket Time Unit Dropdown* feature.

25.1 Administrator Interface

This module adds additional functionality to set time contingents for customers.

25.1.1 Administration

After installation of the package some new classes will be available in the *General Catalog*.

General Catalog

This package adds some new classes to the *General Catalog*. The general catalog management screen is available in the *General Catalog* module of the *Administration* group.

List
CATALOG CLASS
ITSM::ConfigItem::Class
ITSM::ConfigItem::Computer::Type
ITSM::ConfigItem::DeploymentState
ITSM::ConfigItem::Hardware::Type
ITSM::ConfigItem::Location::Type
ITSM::ConfigItem::Network::Type
ITSM::ConfigItem::QuotaManagement::NotifyBy
ITSM::ConfigItem::QuotaManagement::RenewalFrequency
ITSM::ConfigItem::Software::LicenceType
ITSM::ConfigItem::Software::Type
ITSM::ConfigItem::YesNo
ITSM::Core::IncidentState
ITSM::Service::Type
ITSM::SLA::Type

Fig. 1: General Catalog Class List Screen

New Classes

ITSM::ConfigItem::Class

A class for configuration item classes.

Two new classes are added to this class to store the quotas:

- Time Unit Quota
- Volume Quota

See also:

The class definition of configuration item classes can be managed in the *Configuration Items* module of the *CMDB Settings* group.

ITSM::ConfigItem::QuotaManagement::NotifyBy

A class for specifying the percentage of the quota when notification should be sent.

ITSM::ConfigItem::QuotaManagement::RenewalFrequency

A class for specifying the renewal frequency of the quota.

The screenshot shows a web form titled "Edit Catalog Item" for the class "ITSM::ConfigItem::QuotaManagement::RenewalFrequency". The form contains the following fields and controls:

- Catalog Class:** ITSM::ConfigItem::QuotaManagement::RenewalFrequency
- Name:** Each Month (indicated as mandatory with a star)
- Renew Interval:** 1
- Renew Time Unit:** Month
- Validity:** valid
- Comment:** (empty text area)
- Buttons:** Save, Save and finish, and Cancel.

Fig. 2: Quota Management Renewal Frequency

25.2 Agent Interface

This package extends the quota functionality to be able to set different quota for each customer.

25.2.1 Configuration Items

Quotas will be managed in specific configuration item classes in the *General Catalog*. This simplifies the handling of quotas. There is versioning of quota configuration items so that the latest version of the quota is always up-to-date and the quota history remains audit-proof.

The quotas are now divided into time-based and volume-based quotas. Time-based quotas are reduced by time units accounted in each article while volume-based quotas are reduced by 1 per tickets. This allows our customers to choose the mode in which they offer their contracts.

Since time-based and volume-based quotas have major differences in their specific handling, there are two separate system configuration settings specifically designed for each use case. Quotas can be added like any other configuration item. The class definition is the same for both quota types.

The following settings are available when adding or editing this resource. The fields marked with an asterisk are mandatory.

Customer Company *

The customer where the quota is assigned to.

Quota Total *


Total amount of quota. This amount will be renewed if needed.

Used Quota *

How much quota is used by the customer at the creation time of the configuration item. Set to 0 if no quota is used yet.

∨ Properties

* Customer Company

* Quota Total

* Used Quota


* Remaining Quota

Last Used Reference

Notification Percentage

+

* Notification Sent

Automatic Renewal

+

Attachments


 Drop files here or click to select files

Fig. 3: Quota Properties

Remaining Quota *

How much quota is remaining for the customer. Set the same amount as *Quota Total* if no quota is used yet.

Last Used Reference

Reference to the ticket where the quota is used. Leave it empty if no quota is used yet.

Notification Percentage

A notification will be sent if the quota usage reached this percentage.

Notification Sent *

Whether a notification was sent. Set to *No* if no quota is used yet.

Automatic Renewal

Quota renewal possibility after the configured renewal frequency.

Renewal Repetition Limit

How many times should the quota be renewed.

Renewal Time Limit

What is the last date for the quota renewal.

Last Renewal

The date of the last quota renewal.

Renewal Count

How many renewal happened.

The newly created configuration item will be displayed in the *Configuration Item Overview* screen and in the *Customers* detail view of the relevant customer.

If any renewal option has been set, the quota will be reset after the configured time period.

25.2.2 Customers

The quota lists are displayed in the customer detail view as separate widgets.

The *Time Unit Quota List* widget displays the total quota, the used quota and the remaining quota. The *Last Used Reference* column displays the ticket number and article ID that is used for changing the quota in format *TicketNumber#ArticleID*.

Time Unit Quota List (1)						Select Preset		
Name	Quota Total (Hours)	Used Quota (Hours)	Remaining Quota (Hours)	Last Used Reference	Changed			
General Quota	100	20.00	80.00	2022052610000016#43	15 minutes ago			

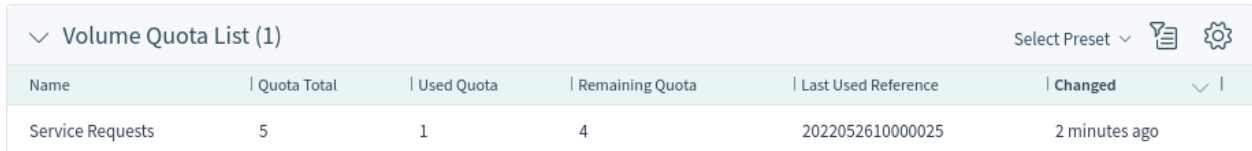
Fig. 4: Time Unit Quota List

In the example above the customer has 100 units quota. A ticket was created for a customer user who belongs to this customer, and 20 time units was accounted. The value `2022052610000016#43` in the *Last Used Reference* column means that ticket number `2022052610000016` and article ID `43` changed the quota last.

To change the time unit quota of a customer:

1. Create a new ticket for the customer or add an article to an existing ticket of the customer.
2. Add any number to the *Time Units* field.
3. Go to the customer detail view and verify that the quota is reduced by the time unit added to the article.

The *Volume Quota List* widget displays the total quota, the used quota and the remaining quota. The *Last Used Reference* column displays the ticket number that is used for changing the quota.



Name	Quota Total	Used Quota	Remaining Quota	Last Used Reference	Changed
Service Requests	5	1	4	2022052610000025	2 minutes ago

Fig. 5: Volume Quota List

In the example above the customer has 5 service requests quota. A ticket was created for a customer user who belongs to this customer, and the quota was reduced by 1. The value 2022052610000025 in the *Last Used Reference* column means that ticket number 2022052610000025 changed the quota last.

To change the volume quota of a customer:

1. Create a new ticket for the customer.
2. Go to the customer detail view and verify that the quota is reduced by 1.

25.3 External Interface

This package has no external interface.