Documentation

OTRS 3.2 - Admin Manual

Build Date:

2012-08-07

OTRS 3.2 - Admin Manual

Copyright © 2003-2012 OTRS AG

René Bakker, Stefan Bedorf, Michiel Beijen, Shawn Beasley, Hauke Böttcher, Jens Bothe, Udo Bretz, Martin Edenhofer, Carlos Javier García, Martin Gruner, Manuel Hecht, Christopher Kuhn, André Mindermann, Marc Nilius, Elva María Novoa, Henning Oschwald, Martha Elia Pascual, Thomas Raith, Carlos Fernando Rodríguez, Stefan Rother, Burchard Steinbild, Michael Thiessmeier, Daniel Zamorano.

This work is copyrighted by OTRS AG.

You may copy it in whole or in part as long as the copies retain this copyright statement.

The source code of this document can be found at github, in the repository doc-admin.

UNIX is a registered trademark of X/Open Company Limited. Linux is a registered trademark of Linus Torvalds.

MS-DOS, Windows, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, Windows 2003, Windows Vista and Windows 7 are registered trademarks of Microsoft Corporation. Other trademarks and registered trademarks are: SUSE and YaST of SUSE Linux GmbH, Red Hat and Fedora are registered trademarks of Red Hat, Inc. Mandrake is a registered trademark of MandrakeSoft, SA. Debian is a registered trademark of Software in the Public Interest, Inc. MySQL and the MySQL Logo are registered trademarks of Oracle Corporation and/or its affiliates.

All trade names are used without the guarantee for their free use and are possibly registered trade marks.

OTRS AG essentially follows the notations of the manufacturers. Other products mentioned in this manual may be trademarks of the respective manufacturer.

Table of Contents

Preface xi
1. Introduction 1
    1. Trouble Ticket Systems - The Basics 1
        1.1. What is a trouble ticket system, and why do you need one? 1
        1.2. What is a trouble ticket? 2
    2. OTRS Help Desk 2
        2.1. Basics 2
        2.2. Features 2
        2.3. Hardware and software requirements 11
        2.4. Community 12
        2.5. Professional Services for OTRS 13
2. Installation 14
    1. The simple way - Installation of pre-built packages 14
        1.1. Installing the RPM on a SUSE Linux server 14
        1.2. Installing OTRS on a Red Hat Enterprise Linux or CentOS system 16
        1.3. Installing OTRS on a Debian system 20
        1.4. Installing OTRS on a Ubuntu system 20
        1.5. Installing OTRS on Microsoft Windows systems 20
    2. Installation from source (Linux, Unix) 20
        2.1. Preparing the installation from source 20
        2.2. Installation of Perl modules 21
        2.3. Configuring the Apache web server 22
        2.4. Configuring the database 23
        2.5. Setting up the cron jobs for OTRS 24
    3. The simple way - Using the web installer (works only with MySQL) 28
    4. Upgrading the OTRS Framework 31
    5. Upgrading Windows Installer 36
    6. Additional applications 36
        6.1. FAQ 37
3. First steps 38
    1. Agent web interface 38
    2. Customer web interface 38
    3. Public web interface 39
    4. First login 40
    5. The web interface - an overview 40
    6. What is a queue? 42
    7. User preferences 43
4. Administration 45
    1. The ADMIN area of OTRS 45
        1.1. Basics 45
        1.2. Agents, Groups and Roles 45
        1.3. Customers and Customer Groups 53
        1.4. Queues 56
        1.5. Salutations, signatures, attachments and responses 58
        1.6. Auto responses 65
        1.7. Email addresses 68
        1.8. Notifications 69
        1.9. SMIME 71
        1.10. PGP 72
        1.11. States 72
        1.12. SysConfig 73

    - 1.13. Using mail accounts ... 74
    - 1.14. Filtering incoming email messages ... 75
    - 1.15. Executing automated jobs with the GenericAgent ... 77
    - 1.16. Admin email ... 78
    - 1.17. Session management ... 79
    - 1.18. System Log ... 80
    - 1.19. SQL queries via the SQL box ... 81
    - 1.20. Package Manager ... 81
    - 1.21. Web Services ... 82
    - 1.22. Dynamic Fields ... 83
  - 2. System Configuration ... 83
    - 2.1. OTRS config files ... 83
    - 2.2. Configuring the system through the web interface ... 84
  - 3. Backing up the system ... 85
    - 3.1. Backup ... 85
    - 3.2. Restore ... 85
  - 4. Email settings ... 86
    - 4.1. Sending/Receiving emails ... 86
    - 4.2. Secure email with PGP ... 90
    - 4.3. Secure email with S/MIME ... 93
  - 5. Using external backends ... 96
    - 5.1. Customer data ... 96
    - 5.2. Customer user backend ... 96
    - 5.3. Backends to authenticate Agents and Customers ... 104
    - 5.4. Customizing the customer self-registration ... 108
  - 6. Ticket settings ... 110
    - 6.1. Ticket States ... 110
    - 6.2. Ticket priorities ... 113
    - 6.3. Ticket Responsibility & Ticket Watching ... 114
  - 7. Time related functions ... 116
    - 7.1. Setting up business hours, holidays and time zones ... 116
    - 7.2. Automated Unlocking ... 117
  - 8. Customizing the PDF output ... 118
  - 9. Stats module ... 118
    - 9.1. Handling of the module by the agent ... 118
    - 9.2. Administration of the stats module by the OTRS administrator ... 128
    - 9.3. Administration of the stats module by the system administrator ... 128
  - 10. Dynamic Fields ... 130
    - 10.1. Introduction ... 130
    - 10.2. Configuration ... 131
  - 11. Generic Interface ... 151
    - 11.1. Generic Interface Layers ... 151
    - 11.2. Generic Interface Communication Flow ... 153
    - 11.3. Web Services ... 156
    - 11.4. Web Service Graphical Interface ... 156
    - 11.5. Web Service Command Line Interface ... 173
    - 11.6. Web Service Configuration ... 175
    - 11.7. Connectors ... 182
  - 12. OTRS Scheduler ... 193
    - 12.1. Scheduler Graphical Interface ... 193
    - 12.2. Scheduler Command Line Interface ... 195
- 5. Customization ... 199
  - 1. Access Control Lists (ACLs) ... 199
    - 1.1. Introduction ... 199

- 1.2. Examples ... 199
- 1.3. Reference ... 201
- 2. Process Management ... 205
  - 2.1. Introduction ... 205
  - 2.2. Example process ... 205
  - 2.3. Implementing the example ... 205
  - 2.4. Process configuration reference ... 231
- 3. Creating your own themes ... 250
- 4. Localization of the OTRS frontend ... 250

6. Performance Tuning ... 251
- 1. OTRS ... 251
  - 1.1. TicketIndexModule ... 251
  - 1.2. TicketStorageModule ... 251
  - 1.3. Archiving Tickets ... 252
- 2. Database ... 252
  - 2.1. MySQL ... 253
  - 2.2. PostgreSQL ... 253
- 3. Webserver ... 253
  - 3.1. Pre-established database connections ... 253
  - 3.2. Preloaded modules - startup.pl ... 253
  - 3.3. Reload Perl modules when updated on disk ... 253
  - 3.4. Choosing the Right Strategy ... 254
  - 3.5. mod_gzip/mod_deflate ... 254

A. Additional Resources ... 255
- 1. Website OTRS Group ... 255
- 2. Mailing lists ... 255
- 3. User Forums ... 256
- 4. Bug tracking ... 256
- 5. Commercial Support ... 257

B. Configuration Options Reference ... 258
- 1. DynamicFields ... 258
  - 1.1. DynamicFields::Backend::Registration ... 258
  - 1.2. DynamicFields::ObjectType::Registration ... 259
  - 1.3. Frontend::Admin::ModuleRegistration ... 260
  - 1.4. Frontend::Agent::Preferences ... 263
- 2. Framework ... 263
  - 2.1. Core ... 263
  - 2.2. Core::Cache ... 272
  - 2.3. Core::LinkObject ... 272
  - 2.4. Core::Log ... 273
  - 2.5. Core::MIME-Viewer ... 275
  - 2.6. Core::MirrorDB ... 276
  - 2.7. Core::PDF ... 276
  - 2.8. Core::Package ... 279
  - 2.9. Core::PerformanceLog ... 281
  - 2.10. Core::ReferenceData ... 282
  - 2.11. Core::SOAP ... 282
  - 2.12. Core::Sendmail ... 283
  - 2.13. Core::Session ... 285
  - 2.14. Core::SpellChecker ... 288
  - 2.15. Core::Stats ... 289
  - 2.16. Core::Stats::Graph ... 290
  - 2.17. Core::Time ... 294
  - 2.18. Core::Time::Calendar1 ... 297

2.19. Core::Time::Calendar2 300
2.20. Core::Time::Calendar3 302
2.21. Core::Time::Calendar4 305
2.22. Core::Time::Calendar5 307
2.23. Core::Time::Calendar6 310
2.24. Core::Time::Calendar7 312
2.25. Core::Time::Calendar8 315
2.26. Core::Time::Calendar9 317
2.27. Core::Web 320
2.28. Core::WebUserAgent 331
2.29. Crypt::PGP 332
2.30. Crypt::SMIME 333
2.31. CustomerInformationCenter 334
2.32. Frontend::Admin::AdminCustomerUser 335
2.33. Frontend::Admin::ModuleRegistration 335
2.34. Frontend::Agent 345
2.35. Frontend::Agent::Dashboard 350
2.36. Frontend::Agent::LinkObject 354
2.37. Frontend::Agent::ModuleMetaHead 354
2.38. Frontend::Agent::ModuleNotify 354
2.39. Frontend::Agent::ModuleRegistration 356
2.40. Frontend::Agent::NavBarModule 362
2.41. Frontend::Agent::Preferences 362
2.42. Frontend::Agent::SearchRouter 366
2.43. Frontend::Agent::Stats 366
2.44. Frontend::Customer 369
2.45. Frontend::Customer::Auth 376
2.46. Frontend::Customer::ModuleMetaHead 382
2.47. Frontend::Customer::ModuleNotify 382
2.48. Frontend::Customer::ModuleRegistration 383
2.49. Frontend::Customer::Preferences 384
2.50. Frontend::Public 387
2.51. Frontend::Public::ModuleRegistration 387
3. GenericInterface 388
3.1. Core::Ticket 388
3.2. Frontend::Admin::ModuleRegistration 388
3.3. GenericInterface::Invoker 391
3.4. GenericInterface::Invoker::ModuleRegistration 392
3.5. GenericInterface::Mapping::ModuleRegistration 393
3.6. GenericInterface::Operation::ModuleRegistration 393
3.7. GenericInterface::Operation::TicketCreate 395
3.8. GenericInterface::Operation::TicketUpdate 396
3.9. GenericInterface::Transport::ModuleRegistration 397
3.10. GenericInterface::Webservice 397
4. ProcessManagement 398
4.1. Core 398
4.2. Core::Ticket 400
4.3. Frontend::Admin::ModuleRegistration 401
4.4. Frontend::Agent::ModuleRegistration 404
4.5. Frontend::Agent::NavBarModule 404
4.6. Frontend::Agent::Ticket::ViewProcess 404
4.7. Frontend::Agent::Ticket::ViewZoom 405
4.8. Frontend::Customer::ModuleRegistration 406
5. Scheduler 406

5.1. Core ... 406
5.2. Core::Log ... 407
5.3. Core::Web ... 407
5.4. Frontend::Admin::ModuleRegistration ... 408
5.5. Frontend::Agent::ModuleNotify ... 408
6. Ticket ... 408
6.1. Core ... 408
6.2. Core::FulltextSearch ... 409
6.3. Core::LinkObject ... 410
6.4. Core::PostMaster ... 411
6.5. Core::Stats ... 419
6.6. Core::Ticket ... 420
6.7. Core::TicketACL ... 435
6.8. Core::TicketBulkAction ... 435
6.9. Core::TicketDynamicFieldDefault ... 436
6.10. Core::TicketWatcher ... 441
6.11. Frontend::Admin::ModuleRegistration ... 442
6.12. Frontend::Agent ... 449
6.13. Frontend::Agent::CustomerSearch ... 456
6.14. Frontend::Agent::Dashboard ... 457
6.15. Frontend::Agent::ModuleMetaHead ... 463
6.16. Frontend::Agent::ModuleNotify ... 463
6.17. Frontend::Agent::ModuleRegistration ... 463
6.18. Frontend::Agent::Preferences ... 475
6.19. Frontend::Agent::SearchRouter ... 480
6.20. Frontend::Agent::Ticket::ArticleAttachmentModule ... 481
6.21. Frontend::Agent::Ticket::ArticleComposeModule ... 481
6.22. Frontend::Agent::Ticket::ArticleViewModule ... 482
6.23. Frontend::Agent::Ticket::ArticleViewModulePre ... 482
6.24. Frontend::Agent::Ticket::MenuModule ... 483
6.25. Frontend::Agent::Ticket::MenuModulePre ... 489
6.26. Frontend::Agent::Ticket::ViewBounce ... 492
6.27. Frontend::Agent::Ticket::ViewBulk ... 493
6.28. Frontend::Agent::Ticket::ViewClose ... 496
6.29. Frontend::Agent::Ticket::ViewCompose ... 502
6.30. Frontend::Agent::Ticket::ViewCustomer ... 505
6.31. Frontend::Agent::Ticket::ViewEmailNew ... 505
6.32. Frontend::Agent::Ticket::ViewEscalation ... 508
6.33. Frontend::Agent::Ticket::ViewForward ... 509
6.34. Frontend::Agent::Ticket::ViewFreeText ... 512
6.35. Frontend::Agent::Ticket::ViewHistory ... 518
6.36. Frontend::Agent::Ticket::ViewMailbox ... 518
6.37. Frontend::Agent::Ticket::ViewMerge ... 520
6.38. Frontend::Agent::Ticket::ViewMove ... 521
6.39. Frontend::Agent::Ticket::ViewNote ... 523
6.40. Frontend::Agent::Ticket::ViewOwner ... 529
6.41. Frontend::Agent::Ticket::ViewPending ... 535
6.42. Frontend::Agent::Ticket::ViewPhoneInbound ... 542
6.43. Frontend::Agent::Ticket::ViewPhoneNew ... 545
6.44. Frontend::Agent::Ticket::ViewPhoneOutbound ... 548
6.45. Frontend::Agent::Ticket::ViewPrint ... 551
6.46. Frontend::Agent::Ticket::ViewPriority ... 552
6.47. Frontend::Agent::Ticket::ViewQueue ... 558
6.48. Frontend::Agent::Ticket::ViewResponsible ... 560

6.49. Frontend::Agent::Ticket::ViewSearch 566
6.50. Frontend::Agent::Ticket::ViewStatus 575
6.51. Frontend::Agent::Ticket::ViewZoom 576
6.52. Frontend::Agent::TicketOverview 578
6.53. Frontend::Agent::ToolBarModule 581
6.54. Frontend::Customer 585
6.55. Frontend::Customer::ModuleMetaHead 588
6.56. Frontend::Customer::ModuleRegistration 588
6.57. Frontend::Customer::Preferences 591
6.58. Frontend::Customer::Ticket::ViewNew 592
6.59. Frontend::Customer::Ticket::ViewPrint 596
6.60. Frontend::Customer::Ticket::ViewSearch 596
6.61. Frontend::Customer::Ticket::ViewZoom 598
6.62. Frontend::Customer::TicketOverview 602
6.63. Frontend::Queue::Preferences 603
6.64. Frontend::SLA::Preferences 603
6.65. Frontend::Service::Preferences 604
C. Credits 605
D. GNU Free Documentation License 607
0. PREAMBLE 607
1. APPLICABILITY AND DEFINITIONS 607
2. VERBATIM COPYING 608
3. COPYING IN QUANTITY 608
4. MODIFICATIONS 609
5. COMBINING DOCUMENTS 610
6. COLLECTIONS OF DOCUMENTS 610
7. AGGREGATION WITH INDEPENDENT WORKS 611
8. TRANSLATION 611
9. TERMINATION 611
10. FUTURE REVISIONS OF THIS LICENSE 611
How to use this License for your documents 611

List of Tables

2.1. Description of several cron job scripts. 25
4.1. Default groups available on a fresh OTRS installation 47
4.2. Rights associated with OTRS Groups 49
4.3. Events for Auto answers 67
4.4. Function of the different X-OTRS-headers 75
4.5. The following fields will be added into the system: 133
4.6. List of Init Scripts And Supported Operating Systems 195
A.1. Mailinglists 255

List of Examples

4.1. Sort spam mails into a specific queue 77
4.2. .fetchmailrc 88
4.3. Example jobs for the filter module Kernel::System::PostMaster::Filter::Match 89
4.4. Example job for the filter module Kernel::System::PostMaster::Filter::CMD 90
4.5. Configuring a DB customer backend 96
4.6. Using company tickets with a DB backend 99
4.7. Configuring an LDAP customer backend 99
4.8. Using Company tickets with an LDAP backend 101
4.9. Using more than one customer backend with OTRS 101
4.10. Authenticate agents against a DB backend 104
4.11. Authenticate agents against an LDAP backend 104
4.12. Authenticate Agents using HTTPBasic 105
4.13. Authenticate Agents against a Radius backend 106
4.14. Customer user authentication against a DB backend 106
4.15. Customer user authentication against an LDAP backend 106
4.16. Customer user authentication with HTTPBasic 107
4.17. Customer user authentication against a Radius backend 108
4.18. Definition of a value series - one element 126
4.19. Definition of a value series - two elements 126
4.20. Activate Field1 in New Phone Ticket Screen. 143
4.21. Activate Field1 in New Phone Ticket Screen as mandatory. 144
4.22. Activate several fields in New Phone Ticket Screen. 145
4.23. Deactivate some fields in New Phone Ticket Screen. 146
4.24. Activate Field1 in Ticket Zoom Screen. 147
4.25. Activate Field1 in Ticket Overview Small Screens. 148
4.26. Activate Field1 in TicketCreate event. 149
4.27. Activate Field1 in the User preferences. 150
4.28. Example To Start The OTRS Scheduler Form An Init.d Script 195
4.29. Example To Start The OTRS Scheduler 196
4.30. Example To Force Stop The OTRS Scheduler 196
4.31. Example To Register The OTRS Scheduler Into the Widows SCM 197
4.32. Example To Start The OTRS Scheduler 197
4.33. Example To Force Stop The OTRS Scheduler 198
5.1. ACL allowing movement into a queue of only those tickets with ticket priority 5.... 199
5.2. ACL allowing movement into a queue of only those tickets with ticket priority 5 stored in the database. 200
5.3. ACL disabling the closing of tickets in the raw queue, and hiding the close button. 200
5.4. ACL removing always state closed successful. 201
5.5. ACL only showing Hardware services for tickets that are created in queues that start with "HW". 201
5.6. Reference showing all possible important ACL settings. 202

Preface

This book is intended for use by OTRS administrators. It also serves as a good reference for OTRS newbies.

The following chapters describe the installation, configuration, and administration of the OTRS software. The first third of the text describes key functionality of the software, while the remainder serves as a reference to the full set of configurable parameters.

This book continues to be a work in progress, given a moving target on new releases. We need your feedback in order to make this a high quality reference document: one that is usable, accurate, and complete. Please write to us if you find content missing in this book, if things are not explained sufficiently, or even if you see spelling mistakes, grammatical errors, or typos. Any kind of feedback is highly appreciated and should be made via our bug tracking system on ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂. Thanks in advance for your contributions!

Chapter 1. Introduction

1. Trouble Ticket Systems - The Basics

This chapter offers a brief introduction to trouble ticket systems, along with an explanation of the core concept of a trouble ticket. A quick example illustrates the advantages of using such a system.

1.1. What is a trouble ticket system, and why do you need one?

The following example describes what a trouble ticket system is, and how you might benefit from using such a system at your company.

Let's imagine that Max is a manufacturer of video recorders. Max receives many messages from customers needing help with the devices. Some days, he is unable to respond promptly or even acknowledge the messages. Some customers get impatient and write a second message with the same question. All messages containing support requests are stored in a single inbox folder. The requests are not sorted, and Max responds to the messages using a regular email program.

Since Max cannot reply fast enough to all the messages, he is assisted by the developers Joe and John in this. Joe and John use the same mail system, accessing the same inbox. They don't realize that Max often gets two identical requests from one frustrated customer. Sometimes they both end up responding separately to the same request, with the customer receiving two different answers. Furthermore, Max is unaware of the details of their responses. He is also unaware of the details of the customer problems and their resolutions, such as which problems occur with high frequency, or how much time and money he has to spend on customer support.

At a meeting, a colleague tells Max about trouble ticket systems and how they can solve Max's problems with customer support. After looking for information on the Internet, Max decides to install OTRS on a computer that is accessible from the web by both his customers and his employees. Now, the customer requests are no longer sent to Max's private inbox but to the mail account that is used for OTRS. The ticket system is connected to this mailbox and saves all requests in its database. For every new request, the system automatically generates an answer and sends it to the customer so that the customer knows that his request has arrived and will be answered soon. OTRS generates an explicit reference, the ticket number, for every single request. Customers are now happy because their requests are acknowledged and it is not necessary to send a second message with the same question. Max, John, and Joe can now log into OTRS with a simple web browser and answer the requests. Since the system locks a ticket that is answered, no message is edited twice.

Let's imagine that Mr. Smith makes a request to Max's company, and his message is processed by OTRS. John gives a brief reply to his question. But Mr. Smith has a follow-up question, which he posts via a reply to John's mail. Since John is busy, Max now answers Mr. Smith's message. The history function of OTRS allows Max to see the full sequence of communications on this request, and he responds with a more detailed reply. Mr. Smith does not know that multiple service representatives were involved in resolving his request, and he is happy with the details that arrived in Max's last reply.

Of course, this is only a short preview of the possibilities and features of trouble ticket systems. But if your company has to attend to a high volume of customer requests through emails and phone calls, and if different service representatives need to respond at different times, a ticket system can be of great assistance. It can help streamline work flow processes, add efficiencies, and improve your overall productivity. A ticket system helps you to flexibly structure your Support or Help Desk environment. Communications between customers and service staff become more transparent. The net result is an increase in service effectiveness. And no doubt, satisfied customers will translate into better financial results for your company.

1.2. What is a trouble ticket?

A trouble ticket is similar to a medical report created for a hospital patient. When a patient first visits the hospital, a medical report is created to hold all necessary personal and medical information on him. Over multiple visits, as he is attended to by the same or additional doctors, the attending doctor updates the report by adding new information on the patient's health and the ongoing treatment. This allows any other doctors or the nursing staff to get a complete picture on the case at hand. When the patient recovers and leaves the hospital, all information from the medical report is archived and the report is closed.

Trouble ticket systems such as OTRS handle trouble tickets like normal email. The messages are saved in the system. When a customer sends a request, a new ticket is generated by the system which is comparable to a new medical report being created. The response to this new ticket is comparable to a doctor's entry in the medical report. A ticket is closed if an answer is sent back to the customer, or if the ticket is separately closed by the system. If a customer responds again on an already closed ticket, the ticket is reopened with the new information added. Every ticket is stored and archived with complete information. Since tickets are handled like normal emails, attachments and contextual annotations will also be stored with each email. In addition, information on relevant dates, employees involved, working time needed for ticket resolution, etc. are also saved. At any later stage, tickets can be sorted, and it is possible to search through and analyze all information using different filtering mechanisms.

2. OTRS Help Desk

This chapter describes the features of OTRS Help Desk (OTRS). You will find information about the hardware and software requirements for OTRS. Additionally, in this chapter you will learn how to get commercial support for OTRS, should you require it, and how to contact the community.

2.1. Basics

OTRS Help Desk (OTRS) is a web application that is installed on a web server and can be used with a web browser.

OTRS is separated into several components. The basic component is the OTRS framework which contains all central functions for the application and the ticket system. Through the web interface of the OTRS framework, it is possible to install additional applications such as ITSM modules, integrations with Network Monitoring solutions, a knowledge base (FAQ), et cetera.

2.2. Features

OTRS has many features. The following list gives an overview of the features included in the central framework.

The features of OTRS

- Web interface:
  - Easy and initial handling with any modern web browser, even with mobile phones or other mobile computers.
  - A web interface to administer the system via the web is available.

  - A web interface to handle customer requests by employees/agents via the web is integrated.
  - A web interface for customers is available to write new tickets, check the state and answer existing tickets and search through their own tickets.
  - The web interface can be customized with different themes; own themes can be integrated.
  - Support for many languages.
  - The appearance of output templates can be customized (dtl).
  - Mails from and into the system can contain multiple attachments.
- Mail interface:
  - Support for mail attachments (MIME support).
  - Automatic conversion of HTML into plain text messages (increased security for sensitive content and enables faster searching).
  - Mail can be filtered with the X-OTRS headers of the system or via mail addresses, e.g. for spam messages.
  - PGP support, creation and import of own keys, signing and encrypting outgoing mail, signed and encrypted messages can be displayed.
  - Support for viewing and encrypting S/MIME messages, handling of S/MIME certificates.
  - Auto answers for customers, configurable for every queue.
  - Email notifications for agents about new tickets, follow-ups or unlocked tickets.
  - Follow-ups by references or In-Reply-To header entries.
- Tickets:
  - Expanded queue view, fast overview of new requests in a queue.
  - Tickets can be locked.
  - Creation of own auto response templates.
  - Creation of own auto responders, configurable for every queue.
  - Ticket history, overview of all events for a ticket (changes of ticket states, replies, notes, etc.).
  - Print view for tickets.
  - Adding own (internal or external) notes to a ticket (text and attachments).
  - Ticket zooming.
  - Access control lists for tickets can be defined.

  - Forwarding or bouncing tickets to other mail addresses.
  - Transferring tickets between queues.
  - Setting or changing the priority of a ticket.
  - The working time for every ticket can be counted.
  - Up-coming tasks for a ticket can be defined (pending features).
  - Bulk actions on tickets are possible.
  - Automatic and timed actions on tickets are possible with the "GenericAgent".
  - Full text search on all tickets is possible.
- System:
  - OTRS runs on many operating systems (Linux, Solaris, AIX, FreeBSD, OpenBSD, Mac OS 10.x, Microsoft Windows).
  - ASP support (active service providing).
  - Linking several objects is possible, e.g. tickets and FAQ entries.
  - Integration of external back-ends for the customer data, e.g. via AD, eDirectory or OpenLDAP.
  - Setting up an own ticket identifier, e.g. Call#, Ticket# or Request#.
  - The integration of your own ticket counter is possible.
  - Support of several database systems for the central OTRS back-end, e.g. MySQL, PostgreSQL, Oracle, MSSQL).
  - Framework to create stats.
  - utf-8 support for the front- and back-end.
  - Authentication for customers via database, LDAP, HTTPAuth or Radius.
  - Support of user accounts, user groups, and roles.
  - Support of different access levels for several systems components or queues.
  - Integration of standard answer texts.
  - Support of sub queues.
  - Different salutations and signatures can be defined for every queue.
  - Email notifications for admins.
  - Information on updates via mail or the web interface.
  - Escalation for tickets.

  - Support for different time zones.
  - Simple integration of own add-ons or applications with the OTRS API.
  - Simple creation of own front-ends, e.g. for X11, console.

2.2.1. New features of OTRS 3.2

2.2.1.1. More customer focused

- The new "Customer Information Center" provides a great dashboard-like view on a customer (company). You can see
  - Escalated, reminder, new, and open tickets of the customer company.
  - Customer users (contacts) belonging to this customer company, with their individual ticket count and shortcuts for creating new tickets for them.
  - An overall ticket status view of the customer company.
- New "switch to customer" feature makes it possible for an agent with the required permissions to look into the customers's panel with their rights.

2.2.1.2. More customizable

2.2.1.2.1. Process Management

- The new process management makes it possible to represent processes within OTRS.

2.2.1.2.2. Customer Interface Improvements

- The customer web interface now fully supports AJAX and ACLs.
- It now requires JavaScript and is not compatible with Internet Explorer 6 or earlier versions.
- In the Customer Interface, you can now set the default ticket type for new tickets. Additionally, you can now also hide the ticket type and use a default value for all tickets created via the customer interface.

2.2.1.2.3. Agent Interface Improvements

- Agents can now search for tickets based on escalation time.
- New option to show DynamicFields by default in ticket search.
- Screen usage optimizations in the ticket screens to avoid scrolling in popup windows. For each ticket screen, the size of the richtext editor can now be configured separately.
- It is now possible to move tickets to another queue from within the TicketAction dialogs (TicketNote, TicketClose etc.) after activating a configuration option. This is turned off by default.
- Ticket search will now directly jump to the ticket zoom screen if only one ticket is found.
- New ability to hide the Article Type from TicketActionCommon-based screens which can be helpful to fit more data in the browser window.

- There is a new out-of-office dashboard widget that lists all currently unavailable agent colleagues.

- New CKEditor 4 makes working with rich text content (such as HTML emails) easier and more stable.

2.2.1.2.4. Administration improvements

- Event Based notifications can now be sent out only for specific Article Sender Types.

- The Statistics engine in OTRS now understands 'Weeks' in addition to days, months and years. This grants the ability to, for instance, create a report for tickets 'created last week', or generate a report that shows tickets created per queue per week.

- It is possible to place customized DTL (template) files in `Custom/Kernel/Output/HTML`, so that they override the system's default DTL files just as how this already works for Perl files.

- In AdminSMIME it is now possible to display human readable certificate contents.

- SysConfig now supports config setting types Date and DateTime.

2.2.1.3. Better scalability

2.2.1.3.1. Ticket Archiving Improved

- When tickets are archived, the information which agent read the ticket and articles can be removed, as well as the ticket subscriptions of agents. This is active by default and helps reduce the amount of data in the database on large systems with many tickets and agents.

- There is also a new script to remove this data from existing archived tickets.

- Archived tickets are now always shown as 'read' by the agent.

2.2.1.3.2. Performance Improvements

- Session management is up to 10 times faster, especially with many active users.

- It is now possible to limit the number of concurrent agents and/or users to avoid server capacity overload.

- Significant reduction in the number of executed database statements in ticket overviews and ticket masks in agent and customer frontend.

  - This will reduce the load on database servers, especially on large systems. In some cases OTRS will become visibly more responsive (if the system was slowed down by the DB load or latency).

- Improved performance of LDAP user synchronization.

- Improved cache performance with many cache files.

2.2.1.4. More Interoperable

2.2.1.4.1. FAQ Connector for the GenericInterface

- It is now possible to access the data of the FAQ module (OTRS knowledge database) via web service (GenericInterface). This can be useful to embed FAQ articles on your company website, for example.

2.2.2. New features of OTRS 3.1

2.2.2.1. GENERIC INTERFACE - A Web Service Framework

- GI is a flexible framework to allow web service interconnections of OTRS with third party applications.
- OTRS can act in both ways - as a provider (server, requested from remote) or requester (client, requesting remotely).
- Simple web service connections can be created without programming by configuring the Generic Interface.
- Complex scenarios can be realized by plugging in custom OTRS extensions that add perl code to the GI infrastructure on different architectural layers.
- ✃✃✃✃✃✃✃✃✃✃ expose parts of OTRS to Generic Interface web services.For example, a ticket connector exposes the ticket create/update function, so that they can be used in a web service regardless which network transport is used.
- A scheduler daemon process supports asynchronous event handling. This is useful to asynchronously start web service requests from OTRS to another system, after the agent''s request has been answered (e.g. when a ticket has been created). Otherwise, it might block the response, resulting in increased response times for the agent.

With the Generic Interface, new web services can be configured easily by using existing OTRS modules, without additional code. They can be combined to create a new web service. When configuring a new web service connection, the administrator has to add:

- A new web service in the admin GUI
- The basic meta data (Transport type (SOAP), URL etc.) and
- Existing operations (part of a connector) and specify for each operation how the data must be mapped (inbound and outbound)

A Generic Interface Debugger will help the OTRS administrator to check how requests are coming in and how they are handled through the different layers.

2.2.2.1.1. Current Features

- Network transports: SOAP/HTTP. Others like REST and JSON are scheduled to be added in the future depending on customers demand.
- Configurable data mapping Graphical User Interface for key/value transformations with respect to incoming and outgoing data.
- Graphical debugger to check the configuration and flow of information of configured web services.
- A ticket connector allowing the use of OTRS as a web service for ticket handling.

2.2.2.1.2. Future Features

- Additional network transports (REST, JSON).
- The GI will replace the iPhoneHandle as the backend for mobile apps.

- Additional connectors will be added to provide more parts of OTRS for use with web services (e.g. to allow the creation, update, or deletion of agents, users, services or CIs).

2.2.2.2. DYNAMIC FIELDS

The DynamicFields Feature replaces the existing ticket and article FreeText and FreeTime fields with a dynamic structure that will also allow to create custom forms in OTRS.

- An unlimited amount of fields can be configured using an own graphical user interface for administration.
- The fields can have different types that can be used for both, tickets and articles. Available by default are:
  - Text
  - Multiline text
  - Checkbox
  - Dropdown
  - Multi-select
  - Date
  - Date and time
- New custom field types (e.g. custom field type dropdown with an external data source) can be added with small effort as the fields are created in a modular, pluggable way.
- A future scenario is, that DynamicFields can be used for objects other than tickets or in custom modules. For example, a custom module adding objects to handle "orders" in OTRS could use the DynamicFields to attach properties/data to these orders.
- A database update script will transform historic FreeText fields and related configuration settings into the new structure.

2.2.2.3. TICKET MANAGEMENT IMPROVEMENTS

2.2.2.3.1. Ticket creation improved

- Multiple email addresses can now be specified as 'To:', 'CC:' or 'BCC:' when creating a new phone or email ticket.

2.2.2.3.2. Inbound phone call support

- Inbound phone calls can now be registered within an existing tickets (until now, only outbound calls were registered).

2.2.2.3.3. Ticket overview preview improved

- It is now possible to exclude articles of certain sender types (e.g. articles from internal agents) in the SysConfig from being displayed in the overview preview mode.
- A certain article type can be configured which will display articles of that type as expanded by default when the view is accessed.

2.2.2.3.4. Ticket move improved

- The screen shown after moving a ticket is now configurable. Options are the ticket zoom view (LastScreenView) or the ticket list (LastScreenOverview).

2.2.2.3.5. Bulk action improved

- With the new bulk action, outbound emails can now be sent from multiple tickets at the same time. As tickets can have different queues, and these queues each can have different templates, salutations and signatures, these are not used in the Bulk Action email.

- An additional bulk action allows configuring the ticket type for selected tickets.

2.2.2.3.6. Configurable Reject Sender Email Address

- The feature allows configuring an email address instead of the administrator address to reject the creation of new tickets by email. This feature can be used in all cases where customers are not allowed to create new tickets by email.

2.2.2.4. PROCESS AUTOMATION

2.2.2.4.1. Escalation events added

- OTRS will now create events for each of the available escalation types (response, update and resolution). This allows performing actions (such as notifications) before the escalation occurs, in the moment it occurs, and in the moment that the escalation ends.

2.2.2.4.2. Notification mechanism improved

- A new generic agent notification module allows the OTRS administrator to define messages that will be shown in the agent web front-end when agents log into the system.

2.2.2.4.3. Time calculation improved

- All kinds of times will now be calculated by and based on the application server only solving the issues that were caused by variances between the clock times of application and data base servers.

2.2.2.4.4. GenericAgent improved

- The GenericAgent can now filter for tickets change time.

- In addition, the GenericAgent can set the ticket responsible for matched tickets.

2.2.2.5. USER INTERFACE, RICH TEXT EDITOR, CHARSET

2.2.2.5.1. User interface performance improved

- The speed for rendering and article display was improved, thanks to Stelios Gikas <stelios.gikas@noris.net>!

2.2.2.5.2. Rich Text Editor Update

- IOS5 support added.

- Block quotes can be left with the enter key.

- Update from CKEditor 3.4 to CKEditor 3.6, so improvements refer to the releases of CKEditor 3.5 and CKEditor 3.6.

- IE9 support improved.
- Resizable dialogs.

2.2.2.5.3. Unicode Support - Non-UTF-8 Internal Encodings Dropped

- UTF-8 is now the only allowed internal charset of OTRS.
- All language files are now formatted in UTF-8, which simplifies their handling and future improvements of the translation mechanism.

2.2.2.6. DATABASE DRIVER SUPPORT

2.2.2.6.1. PostgreSQL DRIVER compatibility improved

- PostgreSQL 9.1 support added.
- A new legacy driver is now available for PostgreSQL 8.1 or earlier versions.

2.2.2.6.2. MS SQL DRIVER compatibility improved

- The MS SQL driver now stores binary data in VARBINARY rather than deprecated type TEXT as well as NVARCHAR to store text strings rather than VARCHAR (for improved Unicode support).

2.2.2.7. MAIL INTEGRATION

2.2.2.7.1. Mail handling improved

- When connecting to IMAP mail accounts, it is now possible to handle emails from a specific email folder, aside from the INBOX folder.
- OTRS can now connect to IMAP servers using Transport Layer Security (TLS), which is useful for modern restricted environments.

2.2.3. Top new features of OTRS 3.0

Context

- User Centered redesign of the Graphical User Interface which results in a dramatic shift from a comprehensive but static to a more powerful and dynamic application using state-of-the art technologies like Ajax, xHTML and optimized CSS.

New Ticket and Article Indicator

- This new feature has been implemented on both ticket and article level. At a glance, an agent is now able to check for any updates within a ticket or on the article level to check for new and unread articles. You benefit from increased transparency and decreased response times.

Optimized Fulltext Search

- The new search feature allows you to flexibly customize the way you browse the information base. Options provided by the new search feature range from single search-string searches to complex multi-string boolean search operations including various operators. You benefit from fully customizable searches adjusted according to your needs.

New Ticket Zoom View

- The redesign based on Ajax technology allows agents to display complex and linked information structures in real-time while keeping the agents' current working environment. The agent will benefit from increased orientation and increased workflow efficiency.

Global Ticket Overviews

- Well known from OTRS 2.4, the global ticket overviews have been optimized to achieve increased inter- activity. Depending on the use case and preferences of your agents, they can easily change the ticket overviews layout according to their specific needs. Options are small, medium, and large, with each providing a different degree of information details.

Accessibility

- The redesign includes common accessibility standards WCAG and WAI-ARIA which also allows disabled users to better interact with OTRS Help Desk. The US Rehabilitation Acts Section 508 has been fulfilled.

New Customer Interface

- The customer web front-end can be integrated to your organization's intranet and is fully integrated into the redesigned help desk system.

Archive Feature

- OTRS 3.0 now offers a new archiving feature. With a separated archive, you'll benefit from a reduced time spent for searches and increased display of results.

2.3. Hardware and software requirements

OTRS can be installed on many different operating systems. OTRS can run on linux and on other unix derivates (e.g. OpenBSD or FreeBSD). You can also run it on Microsoft Windows. OTRS does not have excessive hardware requirements. We recommend using a machine with at least a 2 GHz Xeon or comparable CPU, 2 GB RAM, and a 160 GB hard drive for a small setup.

To run OTRS, you'll also need to use a web server and a database server. Apart from that, you should install perl and/or install some additional perl modules on the OTRS machine. The web server and Perl must be installed on the same machine as OTRS. The database back-end may be installed locally or on another host.

For the web server, we recommend using the Apache HTTP Server, because its module mod_perl greatly improves the performance of OTRS. Apart from that, OTRS should run on any web server that can execute Perl scripts.

You can deploy OTRS on different databases. You can choose between MySQL, PostgreSQL, Oracle, or Microsoft SQL Server. If you use MySQL you have the advantage that the database and some system settings can be configured during the installation, through a web front-end.

For Perl, we recommend using at least version 5.8.8. You will need some additional modules which can be installed either with the Perl shell and CPAN, or via the package manager of your operating system (rpm, yast, apt-get).

Software requirements

2.3.1. Perl support

- Perl 5.8.8 or higher

2.3.2. Web server support

- Apache2 + mod_perl2 or higher (recommended, mod_perl is really fast!)
- Webserver with CGI support (CGI is not recommended)
- Microsoft Internet Information Server (IIS) 6 or higher

2.3.3. Database support

- MySQL 5.0 or higher
- PostgreSQL 7.0 or higher (8.2 or higher recommended)
- Oracle 10g or higher
- Microsoft SQL Server 2005 or higher

The section in the manual about installation of Perl modules describes in more detail how you can set up those which are needed for OTRS.

If you install a binary package of OTRS, which was built for your operating system (rpm, Windows-Installer), either the package contains all Perl modules needed or the package manager of your system should take care of the dependencies of the Perl modules needed.

2.3.4. Web browser support

To use OTRS, you'll be OK if you use a modern browser with JavaScript support enabled. We support the following browsers:

- Internet Explorer 8.0 or higher (agent interface)

  Internet Explorer 7.0 or higher (customer interface)
- Mozilla Firefox 3.6 or higher
- Google Chrome
- Opera 10 or higher
- Safari 4 or higher

We recommend always using the latest version of your browser, because it has the best JavaScript and rendering performance. Dramatical performance varieties between the used browsers can occur with big data or big systems. We are happy to consult you on that matter.

2.4. Community

OTRS has a large user community. Users and developers discuss OTRS and exchange information on related issues through the mailing-lists. You can use the mailing lists to discuss installation, configuration, usage, localization and development of OTRS. You can report software bugs in our bug tracking system.

The homepage of the OTRS community is: ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂.

2.5. Professional Services for OTRS

Whether you need help in configuring or customizing OTRS or you want to be on the safe side, don't hesitate to contact us: We offer a wide range of professional services such as world-wide enterprise support, consulting and engineering including process design, implementation, customization, application support, and fully managed service.

Our Service Contracts guarantee instant help and professional support as well as support assessment and last but not least free access to OTRS Feature Add-ons - useful additional features for your OTRS.

The OTRS Group offers specific training programs in different countries. You can either participate in one of our public OTRS Administrator trainings which take place regularly, or benefit from an inhouse training that covers all the specific needs of your company.

Chapter 2. Installation

This chapter describes the installation and basic configuration of the central OTRS framework. It covers information on installing OTRS from source, or with a binary package such as an RPM or a Windows exectuable.

Topics covered here include configuration of the web and database servers, the interface between OTRS and the database, the installation of additional Perl modules, setting proper access rights for OTRS, setting up the cron jobs for OTRS, and some basic settings in the OTRS configuration files.

Follow the detailed steps in this chapter to install OTRS on your server. You can then use its web interface to login and administer the system.

1. The simple way - Installation of pre-built packages

If available for your platform you should use pre-built packages to install OTRS, since it is the simplest and most convenient method. You can find them in the download area at ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ . The following sections describe the installation of OTRS with a pre-built or binary package on SUSE, Red Hat and Microsoft Windows systems. Only if you are unable to use the pre-built packages for some reason should you follow the manual process.

1.1. Installing the RPM on a SUSE Linux server

This section describes the installation of our RPM package on a SUSE Linux server. We have tested against all recent SLES and openSUSE versions. Before you start the installation, please visit ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ and make sure you use the latest OTRS RPM package available.

1.1.1. Preparing the database for OTRS

You can use OTRS using different database back-ends: MySQL, PostgreSQL, Oracle or Microsoft SQL Server. The most popular database to deploy OTRS on is MySQL. This chapter shows the steps you need to take to configure MySQL on a SUSE-based server. Of course you can install the database on a dedicated database server if needed for scalability or other purposes.

> Note
>
> If you follow this chapter on openSUSE 12.3 and up you'll actually not install MySQL but MariaDB instead, a MySQL compatible fork of the MySQL code. This is no problem, it will work just as well (and even a little better at some points).

Install MySQL by executing the following command as root:

```
linux:~ # zypper install mysql perl-DBD-mysql
```

This will install MySQL with the default options on your system. You'll need to change the defaults in order to make it suitable for OTRS. With a text editor open the file `/etc/my.cnf` and change the line with max_allowed_packet on it, and add a line below, like this:

```
max_allowed_packet=20M
query_cache_size=32M
```

Now execute rcmysql restart to re-start the database server and activate these changes. Then run /usr/bin/mysql_secure_installation and follow the on-screen instructions to set a database root password, remove anonymous access and remove the test database. Lastly, run chkconfig -a mysql in order to make sure mysql is automatically started at server startup time.

1.1.2. Installing OTRS

Install OTRS with via the command line using zypper. This will also pull in some dependencies such as the apache web server and some Perl modules. Make sure you copied the OTRS RPM file to the current directory.

```
otrs-sles:~ # zypper install otrs-3.2.*.rpm
....
Retrieving package otrs-3.2.3-01.noarch (1/26), 17.5 MiB (74.3 MiB unpacked)
Installing: otrs-3.2.3-01 [done]
Additional rpm output:
Check OTRS user ... otrs added.

Next steps:

[start database and Apache]
 Make sure your database is running and execute 'rcapache2 restart'.

[install the OTRS database]
 Use a webbrowser and open this link:
 http://myserver.example.com/otrs/installer.pl

[OTRS services]
 Start OTRS 'rcotrs start-force' (rcotrs {start|stop|status|restart|start-force|

       stop-force}).

((enjoy))

 Your OTRS Team
 http://otrs.org/

otrs-sles:~ #
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂

The OTRS installation is done. Start your web server to load the OTRS specific changes in its configuration, as shown in the script below. Also run chkconfig to make sure OTRS is automatically started when the server reboots.

```
otrs-sles:~ # chkconfig -a apache2
apache2                   0:off  1:off  2:off  3:on   4:off  5:on   6:off
otrs-sles:~ # rcapache2 start
Starting httpd2 (prefork) httpd2-prefork: Could not reliably determine the server's fully
 qualified domain name, using 10.x.x.x for ServerName

                                                                                    done
otrs-sles:~ #
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂

1.1.3. Installation of additional perl modules

OTRS needs some more modules than can be installed by the RPM. You can post-install them manually. You can check what modules you are missing by running the bin/

`otrs.CheckModules.pl` script located in the `/opt/otrs` directory. Some modules are only needed for optional functionality, such as communication with IMAP(S) servers or PDF generation. On SLES you should add an external repository to the zypper configuration in order to get the modules needed for your system. Choose the module needed for your OS version from here: http://download.opensuse.org/repositories/devel:/languages:/perl/. Add the repository like this for SLES 11 SP2:

```
            zypper ar -f -n perl http://download.opensuse.org/repositories/devel:/languages:/
perl/SLE_11_SP2 Perl
```

On openSUSE 12.3 the extra repository is only needed for the Mail::IMAPClient module, which you'd only need if you need to collect mails from an IMAP server secured with TLS. The corresponding line would look like this:

```
            zypper ar -f -n perl http://download.opensuse.org/repositories/devel:/languages:/
perl/openSUSE_12.3/ Perl
```

The first time you use zypper after you added this repository, you will be prompted to add its key. Now you can install missing modules like below.

```
otrs-sles:/opt/otrs # zypper install -y "perl(YAML::LibYAML)"
Refreshing service 'susecloud'.
Retrieving repository 'perl' metadata [\]

New repository or package signing key received:
Key ID: DCCA98DDDCEF338C
Key Name: devel:languages:perl OBS Project <devel:languages:perl@build.opensuse.org>
Key Fingerprint: 36F0AC0BCA9D8AF2871703C5DCCA98DDDCEF338C
Key Created: Wed Oct 10 22:04:18 2012
Key Expires: Fri Dec 19 22:04:18 2014
Repository: perl

Do you want to reject the key, trust temporarily, or trust always? [r/t/a/?] (r): a
Retrieving repository 'perl' metadata [done]
Building repository 'perl' cache [done]
Loading repository data...
Reading installed packages...
'perl(YAML::LibYAML)' not found in package names. Trying capabilities.
Resolving package dependencies...

The following NEW package is going to be installed:
  perl-YAML-LibYAML

The following package is not supported by its vendor:
  perl-YAML-LibYAML

Retrieving package perl-YAML-LibYAML-0.38-12.4.x86_64 (1/1), 75.0 KiB (196.0 KiB unpacked)
Retrieving: perl-YAML-LibYAML-0.38-12.4.x86_64.rpm [done (55.7 KiB/s)]
Installing: perl-YAML-LibYAML-0.38-12.4 [done]
```

The next step is to configure OTRS using the web installer, as described in this section.

1.2. Installing OTRS on a Red Hat Enterprise Linux or CentOS system

This section describes the installation of our RPM package on a Red Hat Enterprise Linux (RHEL) or CentOS server. We ship separate RPMs for versions 5 and 6 of RHEL and CentOS. Before you start the installation, please visit ✃✃✃✃✃✃✃✃✃✃✃✃✃✃✃✃✃✃✃✃ ✃✃✃✃✃✃✃✃✃ and make sure you use the latest OTRS RPM package available.

1.2.1. Preparing the database for OTRS

You can use OTRS using different database back-ends: MySQL, PostgreSQL, Oracle or Microsoft SQL Server. The most popular database to deploy OTRS on is MySQL. This chapter shows the steps you need to take to configure MySQL on a SUSE-based server. Of course you can install the database on a dedicated database server if needed for scalability or other purposes.

Install MySQL by executing the following command as root:

```
        [root@otrs-centos6 ~]# yum -y install mysql-server
```

This will install MySQL with the default options on your system. You'll need to change the defaults in order to make it suitable for OTRS. With a text editor open the file `/etc/my.cnf` and add the next two lines under the [mysqld] section:

```
        max_allowed_packet=20M
        query_cache_size=32M
```

Now execute service mysqld start to re-start the database server and activate these changes. Then run /usr/bin/mysql_secure_installation and follow the on-screen instructions to set a database root password, remove anonymous access and remove the test database. Lastly, run chkconfig mysqld on in order to make sure mysql is automatically started at server startup time.

1.2.2. Installing OTRS

Install OTRS with via the command line using yum. This will also pull in some dependencies such as the apache web server and some Perl modules. Make sure you copied the OTRS RPM file to the current directory.

```
[root@otrs-centos6 ~]# yum install --nogpgcheck otrs-3.2.*.rpm
...
Dependencies Resolved

================================================================================
 Package                     Arch   Version                Repository       Size
================================================================================
Installing:
 otrs                        noarch 3.2.3-01               /otrs-3.2.3-01.noarch
                                                                            74 M
Installing for dependencies:
 apr                         x86_64 1.3.9-5.el6_2          updates         123 k
 ...
 procmail                    x86_64 3.22-25.1.el6          base            163 k

Transaction Summary
================================================================================
Install      26 Package(s)

Total size: 80 M
Total download size: 6.0 M
Installed size: 88 M
Downloading Packages:
(1/25): apr-1.3.9-5.el6_2.x86_64.rpm                     | 123 kB     00:00
...
(25/25): procmail-3.22-25.1.el6.x86_64.rpm               | 163 kB     00:00
--------------------------------------------------------------------------------
```

```
Total                                           887 kB/s | 6.0 MB     00:06
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : apr-1.3.9-5.el6_2.x86_64                               1/26
  ...
  Installing : otrs-3.2.3-01.noarch                                   26/26
Check OTRS user ... otrs added.

Next steps:

[httpd services]
 Restart httpd 'service httpd restart'

[install the OTRS database]
 Make sure your database server is running.
 Use a web browser and open this link:
 http://myserver.example.com/otrs/installer.pl

[OTRS services]
 Start OTRS 'service otrs start' (service otrs {start|stop|status|restart).

((enjoy))

 Your OTRS Team

Installed:
  otrs.noarch 0:3.2.3-01

Dependency Installed:
  ...

Complete!
[root@otrs-centos6 ~]#
```

* * *

The OTRS installation is now done. Now you should make sure that Apache is started and that it starts whenever the server reboots.

```

[root@otrs-centos6 ~]# chkconfig httpd on
[root@otrs-centos6 ~]# service httpd start
Starting httpd: httpd: Could not reliably determine the server's fully qualified domain name,
 using 10.x.x.x for ServerName [  OK  ]
[root@otrs-centos6 ~]#
```

* * *

1.2.3. Installation of additional perl modules

OTRS needs some more modules than can be installed by the RPM. You can post-install them manually. You can check what modules you are missing by running the `bin/otrs.CheckModules.pl` script located in the `/opt/otrs` directory. Some modules are only needed for optional functionality, such as communication with IMAP(S) servers or PDF generation. On Red Hat or CentOS we recommend installing these modules from the EPEL repository, a repository maintained by the Fedora project, which provides high quality packages for RHEL and derivatives. Check for more information the EPEL web site.

If you're on RHEL 6 or CentOS 6, you can get the latest package for EPEL from this site. You can add this repository to yum it in one go by copying the RPM URL you find on this page and executing this command:

```
 [root@otrs-centos6 otrs]# yum -y install http://download.fedoraproject.org/pub/epel/6/i386/
epel-release-6-8.noarch.rpm
Loaded plugins: security
Setting up Install Process
epel-release-6-8.noarch.rpm                                   |  14 kB     00:00
Examining /var/tmp/yum-root-7jrJef/epel-release-6-8.noarch.rpm: epel-release-6-8.noarch
Marking /var/tmp/yum-root-7jrJef/epel-release-6-8.noarch.rpm to be installed
Resolving Dependencies
--> Running transaction check
---> Package epel-release.noarch 0:6-8 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====================================================================================
 Package             Arch          Version     Repository                       Size
=====================================================================================
Installing:
 epel-release        noarch        6-8         /epel-release-6-8.noarch         22 k

Transaction Summary
=====================================================================================
Install       1 Package(s)

Total size: 22 k
Installed size: 22 k
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : epel-release-6-8.noarch                                           1/1
  Verifying  : epel-release-6-8.noarch                                           1/1

Installed:
  epel-release.noarch 0:6-8

Complete!
[root@otrs-centos6 otrs]#
```

The first time you use yum after you added this repository, you will be prompted to add its key. Now you can install missing modules like below.

```
[root@otrs-centos6 otrs]# yum -y install "perl(Text::CSV_XS)"
Loaded plugins: security
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package perl-Text-CSV_XS.x86_64 0:0.85-1.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====================================================================================
 Package                   Arch            Version               Repository     Size
=====================================================================================
Installing:
 perl-Text-CSV_XS          x86_64          0.85-1.el6            epel           71 k

Transaction Summary
=====================================================================================
Install       1 Package(s)

Total download size: 71 k
```

```
Installed size: 154 k
Downloading Packages:
perl-Text-CSV_XS-0.85-1.el6.x86_64.rpm                        |  71 kB     00:00
warning: rpmts_HdrFromFdno: Header V3 RSA/SHA256 Signature, key ID 0608b895: NOKEY
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
Importing GPG key 0x0608B895:
 Userid : EPEL (6) <epel@fedoraproject.org>
 Package: epel-release-6-8.noarch (@/epel-release-6-8.noarch)
 From   : /etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
Is this ok [y/N]: y
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : perl-Text-CSV_XS-0.85-1.el6.x86_64                                1/1
  Verifying  : perl-Text-CSV_XS-0.85-1.el6.x86_64                                1/1

Installed:
  perl-Text-CSV_XS.x86_64 0:0.85-1.el6

Complete!
[root@otrs-centos6 otrs]#
```

The next step is to configure OTRS using the web installer, as described in this section.

1.3. Installing OTRS on a Debian system

On the OTRS Wiki you can find detailed instructions for setting up OTRS on a Debian system: http://wiki.otrs.org/index.php?title=Installation_on_Debian_5.04_lenny .

1.4. Installing OTRS on a Ubuntu system

On the OTRS Wiki you can find detailed instructions for setting up OTRS on an Ubuntu system: http://wiki.otrs.org/index.php?title=Installation_on_Ubuntu_Lucid_Lynx_(10.4) .

1.5. Installing OTRS on Microsoft Windows systems

Installing OTRS on a Microsoft Windows system is very easy. Download the latest installer for Win32 from ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ and save the file to your local file system. Then simply double-click on the file to execute the installer, and follow the few installation steps to setup the system. After that you will be able to login as OTRS administrator and configure the system according to your needs. To log in as OTRS administrator use the user name 'root@localhost' and the default password 'root'.

Warning

Please change the password for the 'root@localhost' account as soon as possible.

Important

The Windows installer for OTRS contains all needed components for OTRS, i.e. the Apache web server, the MySQL database server, Perl (with all needed modules) and cron for Windows. For that reason you should only install OTRS on Windows systems that don't already have an installation of Apache or another web server, or a MySQL database installation.

2. Installation from source (Linux, Unix)

2.1. Preparing the installation from source

If you want to install OTRS from source, first download the source archive as .tar.gz, .tar.bz2, or .zip file from ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁

Unpack the archive (for example, using tar) into the directory `/opt`, and rename the directory from otrs-x.x.x to otrs (see Script below).

```
linux:/opt# tar xf /tmp/otrs-x.x.x.tar.gz
linux:/opt# mv otrs-x.x.x otrs
linux:/opt# ls
otrs
linux:/opt#
```

✁✁✁✁✁✁✁ ✁✁✁✁✁ ✁✁✁✁✁ ✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁✁

OTRS should NOT be run with root rights. Next you should add a new user for OTRS. The home directory of this new user should be `/opt/otrs`. If your web server is not running with the same user rights as the new 'otrs' user, which is the case on most systems, you will need to add the new 'otrs' user to the group of the web server user (see Script below).

```
linux:/opt# useradd -r -d /opt/otrs/ -c 'OTRS user' otrs
linux:/opt# usermod -G nogroup otrs
linux:/opt#
```

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁ ✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁ ✁✁ ✁✁ ✁ ✁✁✁✁✁✁

Next, you have to copy some sample configuration files. The system will later use the copied files. The files are located in `/opt/otrs/Kernel` and `/opt/otrs/Kernel/Config` and have the suffix .dist (see Script below).

```
linux:/opt# cd otrs/Kernel/
linux:/opt/otrs/Kernel# cp Config.pm.dist Config.pm
linux:/opt/otrs/Kernel# cd Config
linux:/opt/otrs/Kernel/Config# cp GenericAgent.pm.dist GenericAgent.pm
```

✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁ ✁✁✁✁✁✁

The last step to prepare the installation of OTRS is to set the proper access rights for the files. You can use the script otrs.SetPermissions.pl, which is located in the bin directory, in the home directory of the 'otrs' user. You can execute the script with the following parameters:

`otrs.SetPermissions.pl` { `Home directory of the OTRS user` } { --otrs-user= `OTRS user` } { --web-user= `Web server user` } [--otrs-group= `Group of the OTRS user`] [--web-group= `Group of the web server user`]

If your web server is running with the same user rights as user 'otrs', the command to set the proper access rights is otrs.SetPermissions.pl /opt/otrs --otrs-user=otrs --web-user=otrs. On SUSE systems the web server is running with the user rights of 'wwwrun'. On Debian-based systems this is 'www-data'. You would use the command otrs.SetPermissions.pl /opt/otrs --otrs-user=otrs --web-user=wwwrun --otrs-group=nogroup --web-group=www to set the proper access rights.

2.2. Installation of Perl modules

OTRS requires some additional Perl modules, as described in Table 3-1. If you install OTRS from source, you will need to install these modules manually. This can be done either with the package manager of your Linux distribution (yast, apt-get) or, as described in this section, through the Perl shell and CPAN. If you're using ActiveState Perl, for instance on Windows, you could use PPM, the built-in Perl Package Manager. We recommend using your package manager if possible.

You can verify which modules you need to install with otrs.CheckModules.pl. This script is located in the `bin` directory, in the home directory of the 'otrs' user (see Script below). Please note that some modules are optional.

```
linux:~# cd /opt/otrs/bin/
linux:/opt/otrs/bin# ./otrs.CheckModules.pl
   o CGI.............................ok (v3.60)
   o Crypt::PasswdMD5................ok (v1.3)
   o Crypt::SSLeay...................Not installed! (Optional - Required for Generic Interface
 SOAP SSL connections.)
   o CSS::Minifier...................ok (v0.01)
   o Date::Format....................ok (v2.22)
   o Date::Pcalc.....................ok (v1.2)

   ...
```

* * *

You should strive to install the missing modules from your Linux distribution's package management system. By doing so, the packages will be automatically updated when new versions are available or when security issues are found. Please refer to your distribution's documentation on how to install additional packages. If the (correct version of) the module is not available from the package repositories, you can also install from CPAN, the Comprehensive Perl Archive Network.

To install one of the modules from above via CPAN, you have to execute the command perl -e shell -MCPAN. The Perl shell will be started in interactive mode and the CPAN module will be loaded. If CPAN is already configured, you can install the modules with the command install followed by the name of the module. CPAN takes care of the dependencies of a module to other Perl modules and will let you know if other modules are needed.

Execute also the commands perl -cw bin/cgi-bin/index.plperl -cw bin/cgi-bin/customer.pl and perl -cw bin/otrs.PostMaster.pl after changing into the directory `/opt/otrs`. If the output of both commands is "syntax OK", your Perl is properly set up (see Script below).

```
linux:~# cd /opt/otrs
linux:/opt/otrs# perl -cw bin/cgi-bin/index.pl
cgi-bin/installer.pl syntax OK
linux:/opt/otrs# perl -cw bin/cgi-bin/customer.pl
cgi-bin/customer.pl syntax OK
linux:/opt/otrs# perl -cw bin/otrs.PostMaster.pl
bin/otrs.PostMaster.pl syntax OK
linux:/opt/otrs#
```

2.3. Configuring the Apache web server

First of all, you should install the Apache2 web server and mod_perl; you'd typically do this from your systems package manager. Below you'll find the commands needed to set up Apache on the most popular Linux distributions.

```
   # rhel / centos:
   linux:# yum install httpd mod_perl

   # suse:
   linux:# zypper install apache2-mod_perl
```

```
    # debian/ubuntu:
    linux:# apt-get install apache2 libapache-mod-perl2
```

To access the web interface of OTRS via a short URL, Alias and ScriptAlias entries are needed. Most Apache installations have a `conf.d` directory included. On Linux systems you can usually find this directory under `/etc/apache` or `/etc/apache2`. Log in as root, change to the `conf.d` directory and copy the appropriate template in `/opt/otrs/scripts/apache2-httpd.include.conf` to a file called `otrs.conf` in the Apache configuration directory.

Restart your web server to load the new configuration settings. On most systems you can start/restart your web server with the command /etc/init.d/apache2 restart (see Script below).

```
linux:/etc/apache2/conf.d# /etc/init.d/apache2 restart
Forcing reload of web server: Apache2.
linux:/etc/apache2/conf.d#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂

Now your web server should be configured for OTRS.

2.4. Configuring the database

2.4.1. Installing the OTRS database manually

The recommended way to configure the database is to run the Web Installer. If you can't use this for some reason, you can also configure the database manually, as described in this chapter. If you can't use the web installer to setup the OTRS database, you have to set it up manually. Scripts with the SQL statements to create and configure the database are located in `scripts/database`, in the home directory of the 'otrs' user (see Script below).

```
linux:~# cd /opt/otrs/scripts/database/
linux:/opt/otrs/scripts/database# ls
otrs-initial_insert.db2.sql         otrs-schema.mysql.sql
otrs-schema.oracle.sql
otrs-initial_insert.mssql.sql       otrs-schema-post.db2.sql
otrs-initial_insert.mysql.sql       otrs-schema.postgresql.sql
otrs-initial_insert.oracle.sql
otrs-initial_insert.postgresql.sql  otrs-schema-post.mssql.sql
otrs-initial_insert.xml             otrs-schema-post.mysql.sql
otrs-schema.db2.sql                 otrs-schema-post.oracle.sql
 otrs-schema-post.postgresql.sql
otrs-schema.mssql.sql               otrs-schema.xml
linux:/opt/otrs/scripts/database#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂

To setup the database for the different database back-ends, the .sql files must be processed in a specific order.

Create the OTRS database manually step by step

1. Creating the DB: Create the database that you want to use for OTRS, with your database client or your database interface.

2. Creating the tables: With the otrs-schema.DatabaseType.sql files (e.g. `otrs-schema.oracle.sql`, `otrs-schema.postgresql.sql`) you can create the tables in your OTRS database.

3. Inserting the initial system data: OTRS needs some initial system data to work properly (e.g. the different ticket states, ticket and notification types). Depending on the type of database that you are using, you will need to use one of the following files: `otrs-initial_insert.mysql.sql`, `otrs-initial_insert.oracle.sql`, `otrs-initial_insert.postgresql.sql` or `otrs-initial_insert.mssql.sql` .

4. Creating references between tables: The last step is to create the references between the different tables in the OTRS database. Use the otrs-schema-post.DatabaseType.sql file to create these (e.g. `otrs-schema-oracle.post.sql`, `otrs-schema-post.postgresql.sql`).

After you have finished the database setup, you should check and set proper access rights for the OTRS database. It should be enough to grant access to one user. Depending on the database server you are using, setting up the access rights differs, but it should be possible either with your database client or your graphical database front-end.

If your database and the access rights are configured properly, you have to tell OTRS which database back-end you want to use and how the ticket system can connect to the database. Open the file `Kernel/Config.pm` located in the home directory of the 'otrs' user, and change the parameters shown in the script below according to your needs.

```
    # DatabaseHost
    # (The database host.)
    $Self->{'DatabaseHost'} = 'localhost';

    # Database
    # (The database name.)
    $Self->{Database} = 'otrs';

    # DatabaseUser
    # (The database user.)
    $Self->{DatabaseUser} = 'otrs';

    # DatabasePw
    # (The password of database user.)
    $Self->{DatabasePw} = 'some-pass';
```

2.5. Setting up the cron jobs for OTRS

OTRS needs some cron jobs to work properly. The cron jobs should be run with the same user rights that were specified for the OTRS modules. That means that the cron jobs must be inserted into the crontab file of the 'otrs' user.

All scripts with the cron jobs are located in `var/cron`, in the home directory of the 'otrs' user (see Script below).

```
linux:~# cd /opt/otrs/var/cron
linux:/opt/otrs/var/cron# ls
aaa_base.dist                generic_agent.dist       rebuild_ticket_index.dist
cache.dist                   pending_jobs.dist        session.dist
fetchmail.dist               postmaster.dist          unlock.dist
generic_agent-database.dist  postmaster_mailbox.dist
linux:/opt/otrs/var/cron#
```

These scripts have a suffix of '.dist'. You should copy them to files with the suffix removed. If you use bash, you might want to use the command listed in Script below.

```
linux:/opt/otrs/var/cron# for foo in *.dist; do cp $foo `basename $foo .dist`; done
linux:/opt/otrs/var/cron# ls
aaa_base                generic_agent-database.dist  rebuild_ticket_index
aaa_base.dist           generic_agent.dist           rebuild_ticket_index.dist
cache                   pending_jobs                 session
cache.dist              pending_jobs.dist            session.dist
fetchmail               postmaster                   unlock
fetchmail.dist          postmaster.dist              unlock.dist
generic_agent           postmaster_mailbox
generic_agent-database  postmaster_mailbox.dist
linux:/opt/otrs/var/cron#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂
✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂ ✂✂✂✂✂

Table 3-2 describes the different cron jobs.

Table 2.1. Description of several cron job scripts.

Script	Function
aaa_base	Sets the basics for the crontab of the 'otrs' user.
cache	Removes expired cache entries from disk. Clears the loader cache for CSS and JavaScript files.
fetchmail	Used only if new mails will be fetched with fetchmail into the ticket system.
generic_agent	Executes the jobs of the GenericAgent that are not stored in the database but in own config files.
generic_agent-database	Executes the jobs of the GenericAgent that are stored in the database.
pending_jobs	Checks system for pending tickets, and closes them or sends reminders if needed.
postmaster	Checks the message queue of the ticket system, and delivers messages that are still in the queues.
postmaster_mailbox	Fetches the mails from the POP3 accounts that were specified in the admin area, in the section for "PostMaster Mail Accounts".
rebuild_ticket_index	Rebuilds the ticket index, which improves the speed of the QueueView.
session	Removes old and no longer needed session IDs.
unlock	Unlocks tickets in the system.

To setup all cron jobs, the script `bin/Cron.sh` located in the home directory of the 'otrs' user can be used. When this script is executed, it needs a parameter to specify whether you want to install, remove, or reinstall the cron jobs. The following parameters can be used:

`Cron.sh {start} {stop} {restart} [OTRS user]`

Because the cron jobs need to be installed in the crontab file of the 'otrs' user, you need to be logged in as 'otrs'. If you are logged in as root, you can switch to 'otrs' with the command su otrs. Execute the commands specified in Script below to install the cron jobs.

Warning

Please note that other crontab entries of the 'otrs' user will be overwritten or removed by the `Cron.sh` script. Please change the `Cron.sh` script to retain other crontab entries as needed.

```
linux:/opt/otrs/var/cron# cd /opt/otrs/bin/
linux:/opt/otrs/bin# su otrs
linux:~/bin$ ./Cron.sh start
/opt/otrs/bin
Cron.sh - start/stop OTRS cronjobs
Copyright (C) 2001-2009 OTRS AG, http://otrs.org/
(using /opt/otrs) done
linux:~/bin$ exit
exit
linux:/opt/otrs/bin#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂ ✂✂✂✂✂

The command crontab -l -u otrs, which can be executed as root, shows you the crontab file of the 'otrs' user, and you can check if all entries are placed correctly (see Script below).

```
linux:/opt/otrs/bin# crontab -l -u otrs
# --
# cron/aaa_base - base crontab package
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# Who gets the cron emails?
MAILTO="root@localhost"

# --
# cron/cache - delete expired cache
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# delete expired cache weekly (Sunday mornings)
20 0 * * 0  $HOME/bin/otrs.CacheDelete.pl --expired >> /dev/null
30 0 * * 0  $HOME/bin/otrs.LoaderCache.pl -o delete >> /dev/null

# --
# cron/fetchmail - fetchmail cron of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# fetch every 5 minutes emails via fetchmail
#*/5 * * * * /usr/bin/fetchmail -a >> /dev/null

# --
# cron/generic_agent - otrs.GenericAgent.pl cron of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# start generic agent every 20 minutes
*/20 * * * * $HOME/bin/GenericAgent.pl >> /dev/null
# example to execute GenericAgent.pl on 23:00 with
# Kernel::Config::GenericAgentMove job file
#0 23 * * * $HOME/bin/otrs.GenericAgent.pl -c "Kernel::Config::GenericAgentMove" >> /dev/null
# --
# cron/generic_agent - GenericAgent.pl cron of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# start generic agent every 10 minutes
*/10 * * * * $HOME/bin/otrs.GenericAgent.pl -c db >> /dev/null
# --
# cron/pending_jobs - pending_jobs cron of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# check every 120 min the pending jobs
45 */2 * * * $HOME/bin/otrs.PendingJobs.pl >> /dev/null
# --
# cron/postmaster - postmaster cron of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# check daily the spool directory of OTRS
#10 0 * * * * test -e /etc/init.d/otrs & /etc/init.d/otrs cleanup >> /dev/null; test -e /etc/
rc.d/init.d/otrs && /etc/rc.d/init.d/otrs cleanup >> /dev/null
10 0 * * * $HOME/bin/otrs.CleanUp.pl >> /dev/null
# --
# cron/postmaster_mailbox - postmaster_mailbox cron of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# fetch emails every 10 minutes
*/10 * * * * $HOME/bin/otrs.PostMasterMailbox.pl >> /dev/null
# --
# cron/rebuild_ticket_index - rebuild ticket index for OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# just every day
01 01 * * * $HOME/bin/otrs.RebuildTicketIndex.pl >> /dev/null

# --
# cron/session - delete old session ids of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# delete every 120 minutes old/idle session ids
55 */2 * * * $HOME/bin/otrs.DeleteSessionIDs.pl --expired >> /dev/null

# --
# cron/unlock - unlock old locked ticket of the OTRS
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This software comes with ABSOLUTELY NO WARRANTY.
# --
# unlock every hour old locked tickets
35 * * * * $HOME/bin/otrs.UnlockTickets.pl --timeout >> /dev/null

linux:/opt/otrs/bin#
```

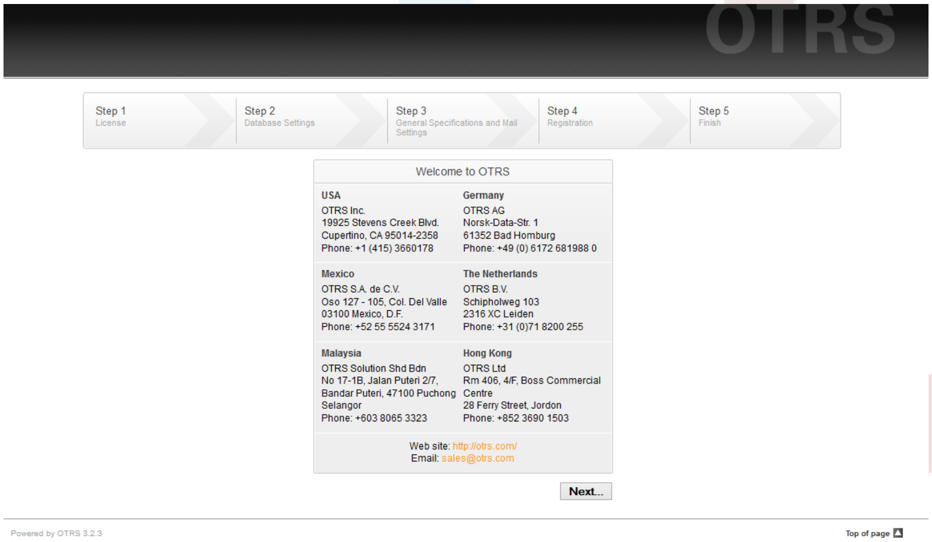
✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁✁

3. The simple way - Using the web installer (works only with MySQL)

If you use MySQL as the database back-end, you can use the OTRS web installer: ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ .

When the web installer starts, please follow the following steps to setup your system:

1. Check out the information about the OTRS offices and click on next to continue (see Figure below).


OTRS
Step 1
License
Step 2
Database Settings
Step 3
General Specifications and Mail Settings
Step 4
Registration
Step 5
Finish

Welcome to OTRS

USA
OTRS Inc.
19925 Stevens Creek Blvd.
Cupertino, CA 95014-2358
Phone: +1 (415) 3660178

Germany
OTRS AG
Norsk-Data-Str. 1
61352 Bad Homburg
Phone: +49 (0) 6172 681988 0

Mexico
OTRS S.A. de C.V.
Oso 127 - 105, Col. Del Valle
03100 Mexico, D.F.
Phone: +52 55 5524 3171

The Netherlands
OTRS B.V.
Schipholweg 103
2316 XC Leiden
Phone: +31 (0)71 8200 255

Malaysia
OTRS Solution Shd Bdn
No 17-1B, Jalan Puteri 2/7,
Bandar Puteri, 47100 Puchong
Selangor
Phone: +603 8065 3323

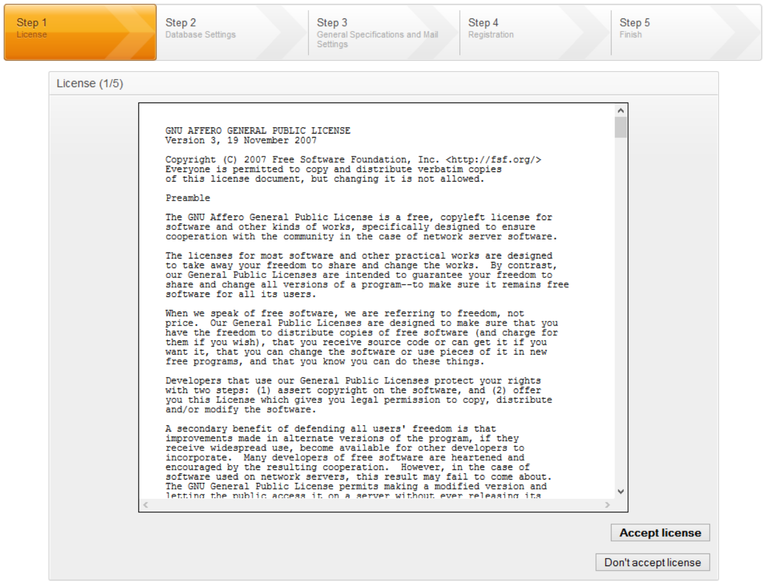
Hong Kong
OTRS Ltd
Rm 406, 4/F, Boss Commercial Centre
28 Ferry Street, Jordon
Phone: +852 3690 1503

Web site: http://otrs.com/
Email: sales@otrs.com

Next...
Powered by OTRS 3.2.3
Top of page

✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁

2. Read the GNU Affero General Public License (see Figure below) and accept it, by clicking the corresponding button at the bottom of the page.

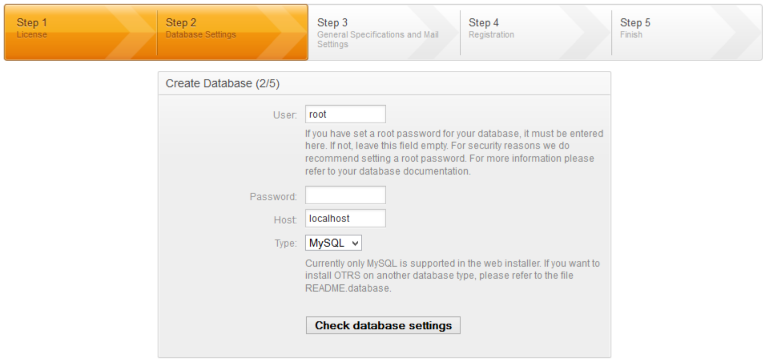

Step 1
License
Step 2
Database Settings
Step 3
General Specifications and Mail Settings
Step 4
Registration
Step 5
Finish
License (1/5)

```
GNU AFFERO GENERAL PUBLIC LICENSE
Version 3, 19 November 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU Affero General Public License is a free, copyleft license for
software and other kinds of works, specifically designed to ensure
cooperation with the community in the case of network server software.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
our General Public Licenses are intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.

When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

Developers that use our General Public Licenses protect your rights
with two steps: (1) assert copyright on the software, and (2) offer
you this License which gives you legal permission to copy, distribute
and/or modify the software.

A secondary benefit of defending all users' freedom is that
improvements made in alternate versions of the program, if they
receive widespread use, become available for other developers to
incorporate.  Many developers of free software are heartened and
encouraged by the resulting cooperation.  However, in the case of
software used on network servers, this result may fail to come about.
The GNU General Public License permits making a modified version and
letting the public access it on a server without ever releasing its
```

Accept license
Don't accept license

✁✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁

3. Provide the username and password of the administrator, the DNS name of the computer which hosts OTRS and the type of database system to be used. After that, check the settings (see Figure below).


Step 1
License
Step 2
Database Settings
Step 3
General Specifications and Mail Settings
Step 4
Registration
Step 5
Finish
Check database settings

Create Database (2/5)

User: root

If you have set a root password for your database, it must be entered here. If not, leave this field empty. For security reasons we do recommend setting a root password. For more information please refer to your database documentation.

Password:

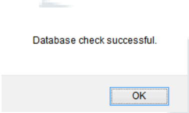
Host: localhost

Type: MySQL

Currently only MySQL is supported in the web installer. If you want to install OTRS on another database type, please refer to the file README.database.

* * *

You will be notified if the check was successful. Press OK to continue (see Figure below).



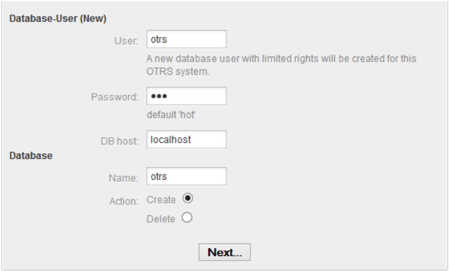
Database check successful.

* * *

4. Create a new database user, choose a name for the database and click on 'Next' (see Figure below).

Warning

It is never a good idea to use default passwords. Please change the default password for the OTRS database!



Database-User (New)

User: otrs

A new database user with limited rights will be created for this OTRS system.

Password: •••

default 'hot'

DB host: localhost

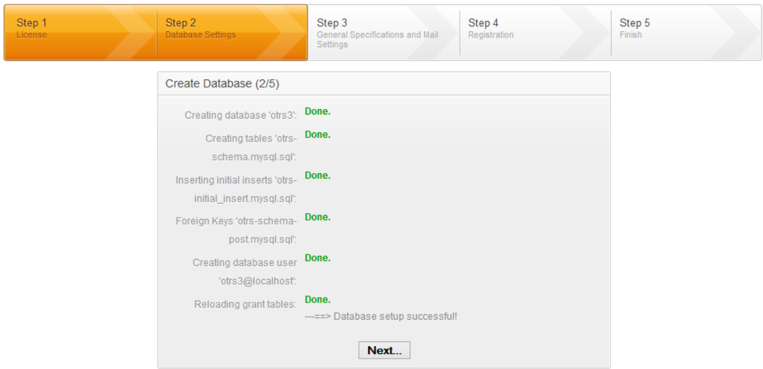
Database

Name: otrs

Action: Create ● Delete ○

* * *

If the database and its user were successfully created, you will receive a setup notification, as shown in Figure. Click 'Next' to go to the next screen.


Step 1
License
Step 2
Database Settings
Step 3
General Specifications and Mail Settings
Step 4
Registration
Step 5
Finish

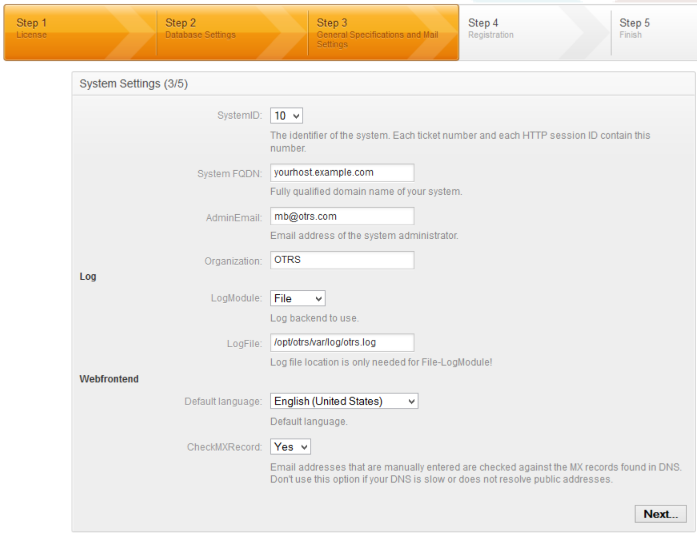
Create Database (2/5)

Creating database 'otrs3': Done.
Creating tables 'otrs-schema.mysql.sql': Done.
Inserting initial inserts 'otrs-initial_insert.mysql.sql': Done.
Foreign Keys 'otrs-schema-post.mysql.sql': Done.
Creating database user 'otrs3@localhost': Done.
Reloading grant tables: Done.
---==> Database setup successful!

Next...

[illegible]

5. Provide all the required system settings and click on 'Next' (see Figure below).


Step 1
License
Step 2
Database Settings
Step 3
General Specifications and Mail Settings
Step 4
Registration
Step 5
Finish

System Settings (3/5)

SystemID: 10
The identifier of the system. Each ticket number and each HTTP session ID contain this number.
System FQDN: yourhost.example.com
Fully qualified domain name of your system.
AdminEmail: mb@otrs.com
Email address of the system administrator.
Organization: OTRS

Log

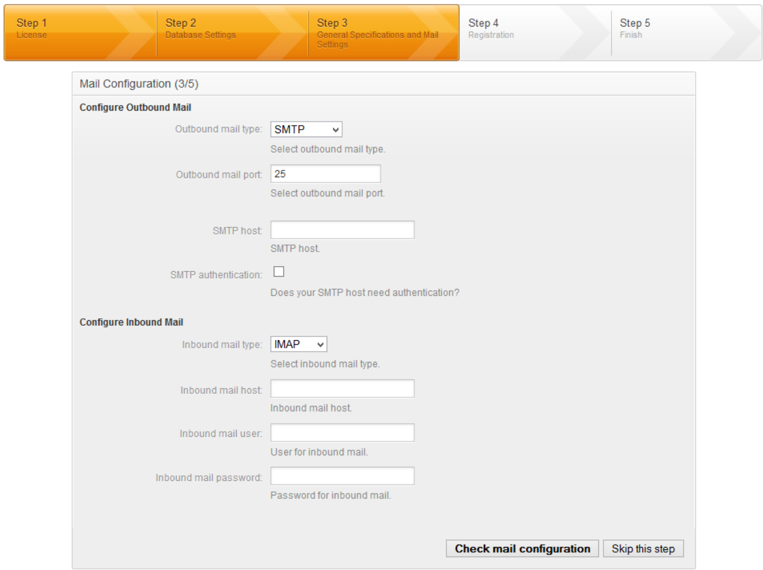
LogModule: File
Log backend to use.
LogFile: /opt/otrs/var/log/otrs.log
Log file location is only needed for File-LogModule!

Webfrontend

Default language: English (United States)
Default language.
CheckMXRecord: Yes
Email addresses that are manually entered are checked against the MX records found in DNS. Don't use this option if your DNS is slow or does not resolve public addresses.

[illegible]

6. If desired, you can provide the needed data to configure your inbound and outbound mail, or skip this step by pressing the right button at the bottom of the screen (see Figure below).



Step 1
License
Step 2
Database Settings
Step 3
General Specifications and Mail Settings
Step 4
Registration
Step 5
Finish

Mail Configuration (3/5)

Configure Outbound Mail

Outbound mail type: SMTP
Select outbound mail type.

Outbound mail port: 25
Select outbound mail port.

SMTP host:
SMTP host.

SMTP authentication: ☐
Does your SMTP host need authentication?

Configure Inbound Mail

Inbound mail type: IMAP
Select inbound mail type.

Inbound mail host:
Inbound mail host.

Inbound mail user:
User for inbound mail.

Inbound mail password:
Password for inbound mail.

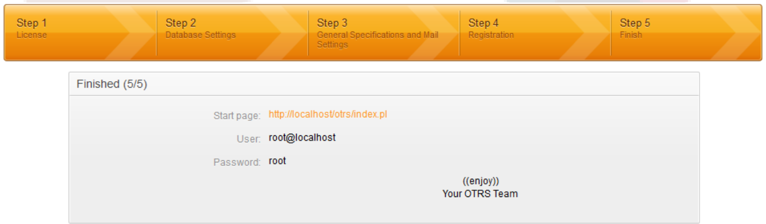
Check mail configuration
Skip this step

✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂

Congratulations! Now the installation of OTRS is finished and you should be able to work with the system (see Figure below). To log into the web interface of OTRS, use the address ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ from your web browser. Log in as OTRS administrator, using the username 'root@localhost' and the password 'root'. After that, you can configure the system to meet your needs.

Warning

Please change the password for the 'root@localhost' account as soon as possible.



Step 1
License
Step 2
Database Settings
Step 3
General Specifications and Mail Settings
Step 4
Registration
Step 5
Finish

Finished (5/5)

Start page: http://localhost/otrs/index.pl
User: root@localhost
Password: root

((enjoy))
Your OTRS Team

✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂

4. Upgrading the OTRS Framework

These instructions are for people upgrading OTRS from version ✂✂✂ to ✂✂✂, and apply both for RPM and source code (tarball) upgrades.

If you are running a lower version of OTRS you have to follow the upgrade path to 3.1 first (1.1->1.2->1.3->2.0->2.1->2.2->2.3->2.4->3.0->3.1->3.2 ...)!

Please note that if you upgrade from OTRS 2.2 or earlier, you must take an extra step; please read http://bugs.otrs.org/show_bug.cgi?id=6798.

Please note that for upgrades from 3.2.0.beta1, an additional step 9 is needed!

Within a single minor version you can skip patch level releases if you want to upgrade. For instance you can upgrade directly from OTRS 3.2.1 to version 3.2.4. If you need to do such a "patch level upgrade", you should skip steps 9, 13, 15, 16 and 17.

1. Stop all relevant services.

   e. g. (depends on used services):

   ```
   shell> /etc/init.d/cron stop
   shell> /etc/init.d/postfix stop
   shell> /etc/init.d/apache stop
   ```

2. Backup everything below $OTRS_HOME (default: OTRS_HOME=/opt/otrs):

   - `Kernel/Config.pm`
   - `Kernel/Config/GenericAgent.pm`
   - `Kernel/Config/Files/ZZZAuto.pm`
   - `var/*`
   - `as well as the database`

3. Make sure that you have backed up everything ;-)

4. Setup new system (optional)

   If possible, try this install on a separate machine for testing first.

5. Install the new release (tar or RPM).

   - With the tarball:

     ```
     shell> cd /opt
     shell> tar -xzf otrs-x.x.x.tar.gz
     shell> ln -s otrs-x.x.x otrs
     ```

     Restore old configuration files.

     - `Kernel/Config.pm`
     - `Kernel/Config/GenericAgent.pm`
     - `Kernel/Config/Files/ZZZAuto.pm`

   - With the RPM:

     ```
     shell> rpm -Uvh otrs-x.x.x.-01.rpm
     ```

     In this case the RPM update automatically restores the old configuration files.

6. Own themes

Note: The OTRS themes between 3.1 and 3.2 are NOT compatible, so don't use your old themes!

Themes are located under $OTRS_HOME/Kernel/Output/HTML/*/*.dtl (default: OTRS_HOME=/opt/otrs).

7. Set file permissions.

   If the tarball is used, execute:

   ```
   shell> cd /opt/otrs/
   shell> bin/otrs.SetPermissions.pl
   ```

   with the permissions needed for your system setup.

8. Check needed Perl modules

   Verify that all needed perl modules are installed on your system and install any modules that might be missing.

   ```
   shell> /opt/otrs/bin/otrs.CheckModules.pl
   ```

9. Schema update:

   Note: new tables created in the MySQL UPGRADING process will be created with the default table storage engine set in your MySQL server. In MySQL 5.5 the new default type is InnoDB. If existing tables, e.g. "users", have the table storage engine, e.g. MyISAM, then an error will be displayed when creating the foreign key constraints.

   You have two options: (1) you can change the default storage engine of MySQL back to MyISAM so that new tables will have the same engine as the existing tables, or (2) change the existing tables to use InnoDB as storage engine.

   Any problems with regards to the storage engine will be reported by the otrs.CheckDB.pl script, so please run it to check for possible issues.

   ```
   shell> cd /opt/otrs/

   # MySQL:
   shell> bin/otrs.CheckDB.pl
   shell> cat scripts/DBUpdate-to-3.2.mysql.sql | mysql -p -f -u root otrs

   # PostgreSQL 8.2+:
   shell> cat scripts/DBUpdate-to-3.2.postgresql.sql | psql otrs

   # PostgreSQL, older versions:
   shell> cat scripts/DBUpdate-to-3.2.postgresql_before_8_2.sql | psql otrs
   ```

   Note: If you use PostgreSQL 8.1 or earlier, you need to activate the new legacy driver for these older versions. Do this by adding a new line to your `Kernel/Config.pm` like this:

   ```
   $Self->{DatabasePostgresqlBefore82} = 1;
   ```

Run the migration script (as user 'otrs', NOT as root):

```
shell> scripts/DBUpdate-to-3.2.pl
```

Do not continue the upgrading process if this script does not work properly for you. Otherwise data loss may occur.

10. Database Upgrade During Beta Phase

    This step is ONLY needed if you upgrade from 3.2.0.beta1!

    Please apply the required database changes as follows:

```
MySQL:
 shell> cat scripts/DBUpdate-3.2.beta.mysql.sql | mysql -p -f -u root otrs

PostgreSQL 8.2+:
 shell> cat scripts/DBUpdate-3.2.beta.postgresql.sql | psql otrs

PostgreSQL, older versions:
 shell> cat scripts/DBUpdate-3.2.beta.postgresql_before_8_2.sql | psql otrs
```

11. Refresh the configuration and delete caches. Please run:

```
shell> bin/otrs.RebuildConfig.pl
shell> bin/otrs.DeleteCache.pl
```

12. Restart your services.

    e. g. (depends on used services):

```
shell> /etc/init.d/cron start
shell> /etc/init.d/postfix start
shell> /etc/init.d/apache start
```

    Now you can log into your system.

13. Check 'Cache::Module' setting

    The file cache backend 'FileRaw' was removed in favor of the faster 'FileStorable'. The DBUpdate-to-3.2.pl automatically updates the config setting 'Cache::Module', but you need to change it manually if you defined this setting in Kernel/Config.pm directly. It needs to be changed from 'Kernel::System::Cache::FileRaw' to 'Kernel::System::Cache::FileStorable'.

14. Check installed packages

    In the package manager, check if all packages are still marked as correctly installed or if any require reinstallation or even a package upgrade.

15. Cleanup metadata of archived tickets

    Note: This step only applies if you use the ticket archiving feature of OTRS.

With OTRS 3.2, when tickets are archived, the information which agent read the ticket and articles can be removed, as well as the ticket subscriptions of agents. This is active by default and helps reduce the amount of data in the database on large systems with many tickets and agents.

If you also want to cleanup this information for existing archived tickets, please run this script:

```
shell> bin/otrs.CleanupTicketMetadata.pl --archived
```

If you want to KEEP this information instead, please set these SysConfig settings to "No":

```
Ticket::ArchiveSystem::RemoveSeenFlags
Ticket::ArchiveSystem::RemoveTicketWatchers
```

16. Review (Modify) ACLs for Dynamic Fields

Note: This step only applies if you use ACLs to limit Dynamic Fields Dropdown or Multiselect possible values.

Now in OTRS 3.2 the Possible and PossibleNot ACL sections for Dynamic Fields Dropdown and Multiselect must refer to the key (internal values) rather than the value (shown values).

Example:

```
For the defined field "Dropdown1"  with possible values:
1 => 'A',
2 => 'B',
3 => 'C',

ACLs prior OTRS 3.2 should look like:

$Self->{TicketAcl}->{'Limit Dropdown1 entries'} = {
   Properties => {},
   Possible => {
       Ticket => {
           # White list entries with VALUES containing 'B' and 'C'
           DynamicField_Dropdown1 => [ 'B', 'C' ],
       },
   },
};

ACLs must be modified to:

$Self->{TicketAcl}->{'Limit Dropdown1 entries'} = {
   Properties => {},
   Possible => {
       Ticket => {
           # White list entries with VALUES containing 'B' and 'C' (now using KEYS)
           DynamicField_Dropdown1 => [ '2', '3' ],
       },
   },
};

By doing this change ACLs will look much more consistent, since Possible and PossibleDatabase
sections already use Keys instead of Values, please look at the following example:

$Self->{TicketAcl}->{'Limit Dropdown1 entries based in Dropdown2'} = {
    Properties => {
        Ticket => {
            # Match on the DeopDown2 KEY '1'
            DynamicField_Dwondown2 => ['1'],
        },
    },
    Possible => {
        Ticket => {
            # White list Dropdown1 entries with VALUES containing 'B' and 'C' (now using
 KEYS)
            DynamicField_Dropdown1 => ['1', '2'],
        },
    },
};
```

17. Adapt custom event handler modules

    Note: this only applies if you have any custom developed event handler modules.

    Since OTRS 3.2, the data payload for event handler modules is no longer copied into the %Param hash. You need to explicitly access it through $Param{Data}.

```
Old:

# get ticket
my %Ticket = $Self->{TicketObject}->TicketGet(
    TicketID      => $Param{TicketID},
    UserID        => 1,
);

New:

# get ticket
my %Ticket = $Self->{TicketObject}->TicketGet(
    TicketID      => $Param{Data}->{TicketID},
    UserID        => 1,
);
```

18. Well done!

5. Upgrading Windows Installer

There is currently no in-place upgrade tool available for OTRS installations that were done with the Windows Installer. The upgrade process basically consists of backing up the database and the filesystem, uninstalling OTRS, installing the new version, restoring the database and running the upgrade procedure if needed.

Upgrading is described in FAQ# 4200351, and there is also an informative YouTube video available.

6. Additional applications

You can install additional application packages to extend the functionality of the OTRS framework. This can be done via the package manager from the Admin page, which downloads the applications from an online repository and manages package dependencies. It is also possible to install packages from local files.

6.1. FAQ

The FAQ is the Knowledge Base component. It supports editing and viewing of FAQ articles. Articles can be viewed and restricted to agents, customer users, or anonymous users. These can also be structured into groups, and be read in different languages.

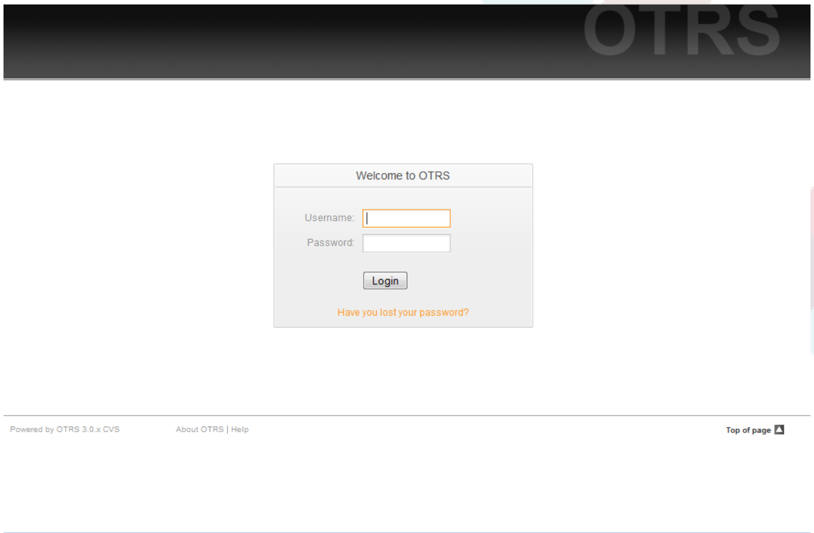
Chapter 3. First steps

The goal of this chapter is to provide a brief overview of OTRS and the structure of its web interface. The terms 'agents', 'customers', and 'administrators' are introduced. We also login as the OTRS admininstrator and take a closer look at the user preferences available on every account.

1. Agent web interface

The agent web interface allows agents to answer customer requests, create new tickets for customers or other agents, write tickets about telephone calls with customers, write FAQ entries, edit customer data, etc.

Supposing your OTRS host is reachable via the URL ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ , then the OTRS login screen can be reached by using the address ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ in a web browser (see Figure below).

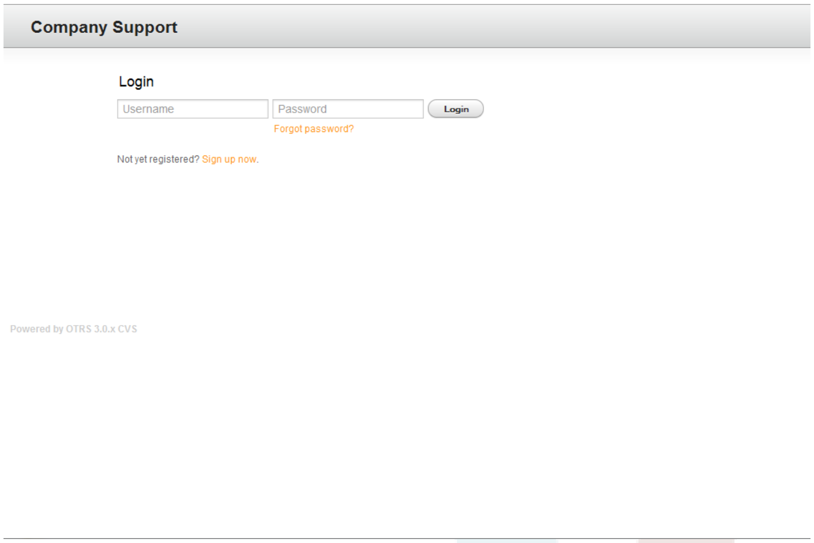

OTRS
Welcome to OTRS
Username:
Password:
Login
Have you lost your password?
Powered by OTRS 3.0.x CVS
About OTRS | Help
Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂

2. Customer web interface

Customers have a separate web interface in OTRS through which they can create new accounts, change their account settings, create and edit tickets, get an overview on tickets that they have created, etc.

Continuing the above example, the customer login screen can be reached by using the URL ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ with a web browser (see Figure below).

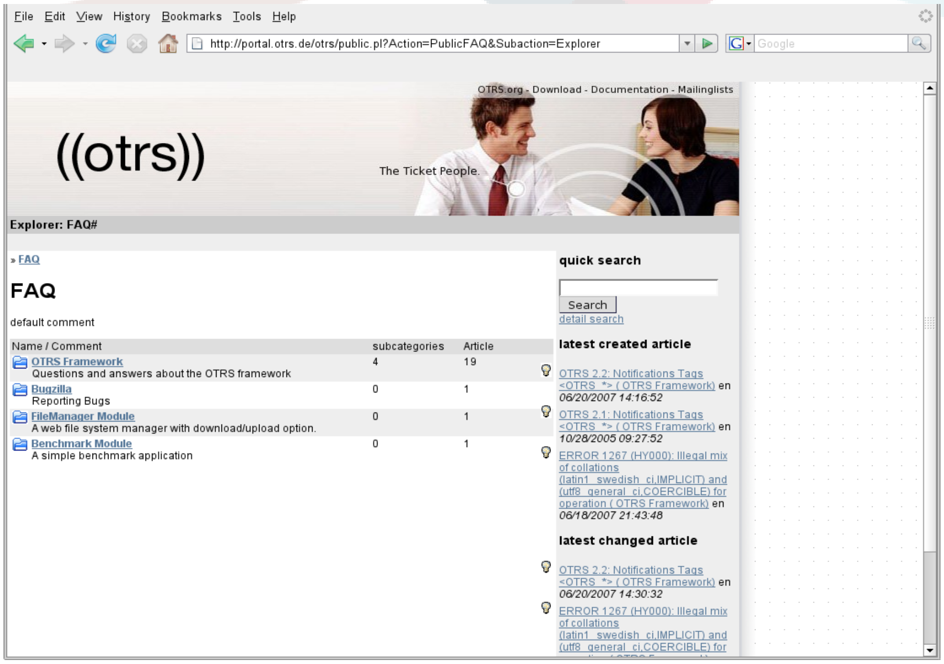

Company Support
Login
Username
Password
Login
Forgot password?
Not yet registered? Sign up now.
Powered by OTRS 3.0.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂

3. Public web interface

In addition to the web interfaces for agents and customers, OTRS also has a public web interface which is available through the FAQ-Module. This module needs to be installed separately. It provides public access to the FAQ system and lets visitors search through FAQ entries without any special authorization.

In our example, the public web interface can be reached via either of the following URLs: ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ , ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂


File Edit View History Bookmarks Tools Help
http://portal.otrs.de/otrs/public.pl?Action=PublicFAQ&Subaction=Explorer
Google
OTRS.org - Download - Documentation - Mailinglists
((otrs))
The Ticket People.
Explorer: FAQ#
» FAQ
quick search
Search
detail search

FAQ

default comment

Name / Comment	subcategories	Article
OTRS Framework Questions and answers about the OTRS framework	4	19
Bugzilla Reporting Bugs	0	1
FileManager Module A web file system manager with download/upload option.	0	1
Benchmark Module A simple benchmark application	0	1

latest created article

OTRS 2.2: Notifications Tags <OTRS_*> (OTRS Framework) en 06/20/2007 14:16:52

OTRS 2.1: Notifications Tags <OTRS_*> (OTRS Framework) en 10/28/2005 09:27:52

ERROR 1267 (HY000): Illegal mix of collations (latin1_swedish_ci,IMPLICIT) and (utf8_general_ci,COERCIBLE) for operation (OTRS Framework) en 06/18/2007 21:43:48

latest changed article

OTRS 2.2: Notifications Tags <OTRS_*> (OTRS Framework) en 06/20/2007 14:30:32

ERROR 1267 (HY000): Illegal mix of collations (latin1_swedish_ci,IMPLICIT) and (utf8_general_ci,COERCIBLE) for

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂

4. First login

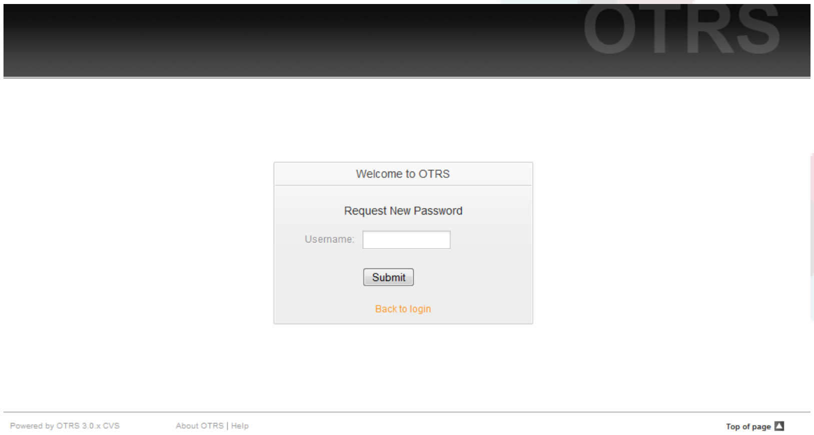
Access the login screen as described in the section Agent web interface . Enter a user name and password. Since the system has just been installed and no users have yet been created, login as OTRS administrator first, using 'root@localhost' for username and 'root' for password.

Warning

This account data is valid on every newly installed OTRS system. You should change the password for the OTRS administrator as soon as possible! This can be done via the preferences screen for the OTRS administrator account.

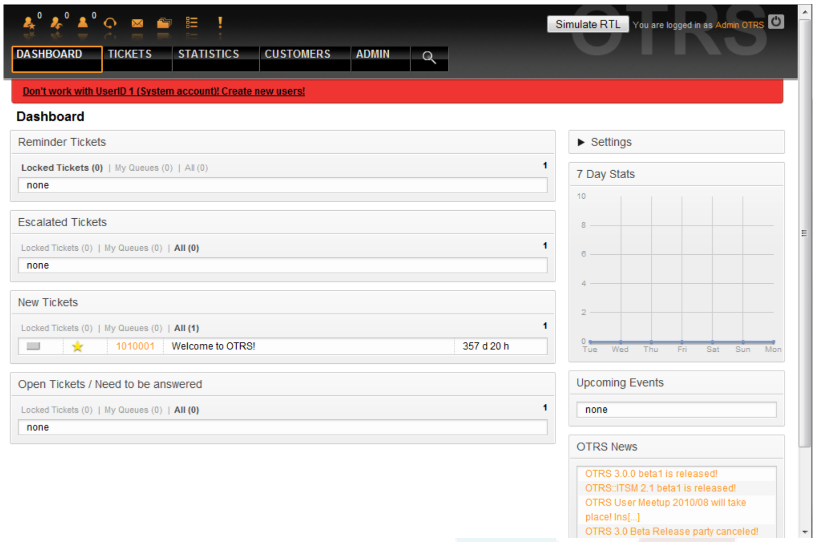
If you don't want to login as OTRS administrator, just enter the user name and password for your normal agent account.

In case you have forgotten your password, you can request the system for a new password. Simply press the link below the Login button, enter the mail address that is registered for your OTRS account into the input field, and press the Submit button (see Figure).



5. The web interface - an overview

Upon successfully logging into the system, you are presented with the Dashboard page (see Figure below). The Dashboard is completely customizable. It shows your locked tickets, allows direct access through menus to the queue, status and escalation views, and also holds options for creation of new phone and e-mail tickets. It also presents a quick summary of the tickets which are pending, escalated, new, and open.



Simulate RTL
You are logged in as Admin OTRS
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Dashboard
Reminder Tickets
Locked Tickets (0) | My Queues (0) | All (0)
none
Escalated Tickets
Locked Tickets (0) | My Queues (0) | All (0)
none
New Tickets
Locked Tickets (0) | My Queues (0) | All (1)
1010001
Welcome to OTRS!
357 d 20 h
Open Tickets / Need to be answered
Locked Tickets (0) | My Queues (0) | All (0)
none
Settings
7 Day Stats
10
8
6
4
2
0
Tue
Wed
Thu
Fri
Sat
Sun
Mon
Upcoming Events
none
OTRS News

Don't work with UserID 1 (System account)! Create new users!

OTRS 3.0.0 beta1 is released!
OTRS::ITSM 2.1 beta1 is released!
OTRS User Meetup 2010/08 will take place! Ins[...]
OTRS 3.0 Beta Release party canceled!

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂

To improve clarity, the general web interface is separated into different areas. The top row of each page shows some general information such the current username, the logout button, icons listing the number of locked tickets with direct access to them, links to create a new phone/e-mail ticket, etc. There are also icons to go to the queue, status, and escalation views.

Below the icons row is the navigation bar. It shows a menu that enables you to navigate to different areas or modules of the system, lettng you execute some global actions. Clicking on the Dashboard button takes you to the dashboard which is the default start page after login. If you click on the Tickets button, you will get a submenu with options to change the ticket's view, create a new ticket (phone/e-mail) or search for a specific ticket. The Statistics button presents a menu that allows you to choose from an overview of the registered statistics, creating a new one or importing an existing one. The Customers button leads you to the Customer Managment screen. By clicking the Admin button, you can access all of the administrator modules, which allows you to create new agents, queues, etc. There is also a Search button to make ticket searches.

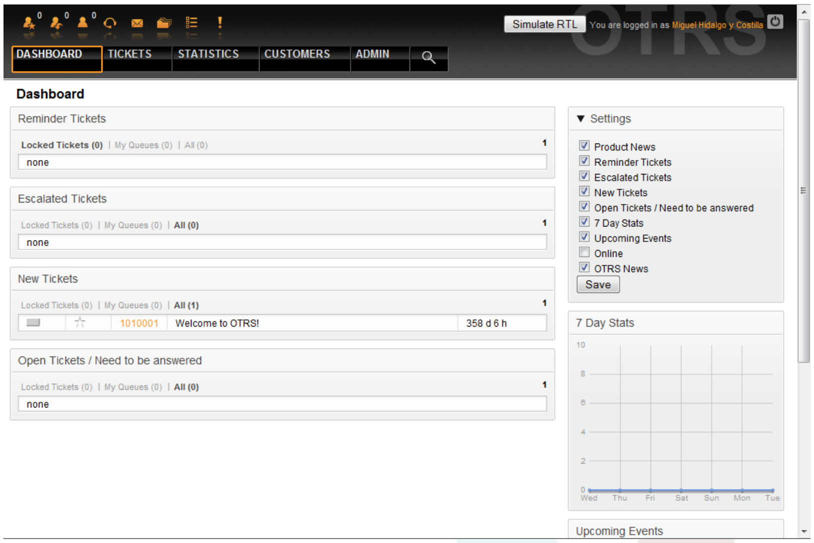
If any associated applications are also installed, e.g. the File Manager or the Web Mailer, buttons to reach these applications are also displayed.

The red bar below the navigation bar shows different system messages. If you are logged in as OTRS administrator, you get a message warning you not to work using this system account.

Below the title of the section you are currently in, there are several subsections, each in a separate box. These boxes can be relocated within the same column by clicking on and dragging the box header, and dropping them elsewhere.

In the left column, you can see information on some tickets clasified as - reminder, escalated, new, and open. In each of the categories, you are also able to see all of the tickets that you are allowed to access, how many tickets you have locked, and how many are located in "My Queues". "My Queues" are queues that you identify in your user configuration account preferences as those you have a special interest in tracking.

In the right column is the Settings button. Click on it to expand the section and see the various settings, as shown in Figure. You can then check or uncheck the individual settings options and save your changes. This section is fixed, so you can not drag and drop it.


Simulate RTL
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Dashboard
Reminder Tickets
Escalated Tickets
New Tickets
Open Tickets / Need to be answered
▼ Settings
Save
7 Day Stats
10
8
6
4
2
0
Wed
Thu
Fri
Sat
Sun
Mon
Tue
Upcoming Events

Reminder Tickets

Locked Tickets (0) | My Queues (0) | All (0) 1

none

Escalated Tickets

Locked Tickets (0) | My Queues (0) | All (0) 1

none

New Tickets

Locked Tickets (0) | My Queues (0) | All (1) 1

	☆	1010001	Welcome to OTRS!	358 d 6 h

Open Tickets / Need to be answered

Locked Tickets (0) | My Queues (0) | All (0) 1

none

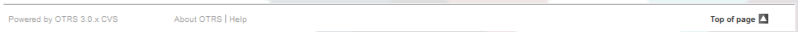
▼ Settings

☑ Product News
☑ Reminder Tickets
☑ Escalated Tickets
☑ New Tickets
☑ Open Tickets / Need to be answered
☑ 7 Day Stats
☑ Upcoming Events
☐ Online
☑ OTRS News

* * *

Below the settings area, you can see a section with a graph of ticket activity over the past 7 days. Further below is a section displaying Upcoming Events and OTRS News.

Finally at the bottom of the page, the site footer is displayed (see Figure below). It contains links to directly access the OTRS official website, or go to the Top of the page.


Powered by OTRS 3.0.x CVS
About OTRS | Help
Top of page

* * *

6. What is a queue?

On many mail systems, it is common for all messages to flow into an Inbox file, where they remain stored. New messages are appended at the end of the Inbox file. The mail client program used to read and write mails reads this Inbox file and presents the content to the user.

A queue in OTRS is somewhat comparable to an Inbox file, since it too can store many messages. A queue also has features beyond those of an Inbox mail file. As an OTRS agent or user, one needs to remember which queue a ticket is stored in. Agents can open and edit tickets in a queue, and also move tickets from one queue to another. But why would they move tickets?

To explain it more practically, remember the example of Max's company described in an example of a ticket system. Max installed OTRS in order to allow his team to better manage support for company customers buying video recorders.

One queue holding all requests is enough for this situation. However, after some time Max decides to also sell DVD recorders. Now, the customers have questions not only about the video recorder, but also about the new product. More and more emails get into the single queue of Max's OTRS and it's difficult to have a clear picture of what's happening.

Max decides to restructure his support system, and adds two new queues. So now three queues are being used. New messages arriving at the ticket system are stored into the old queue titled "raw". Of the two new queues, one titled "video recorder" is exclusively for video recorder requests, while the other one titled "dvd recorder" is exclusively for dvd recorder requests.

Max asks Sandra to watch the "raw" queue and sort (dispatch) the messages either into "video recorder" or "dvd recorder" queue, depending on the customer request. John only has access to the "video recorder" queue, while Joe can only answer tickets in the "dvd recorder" queue. Max is able to edit tickets in all queues.

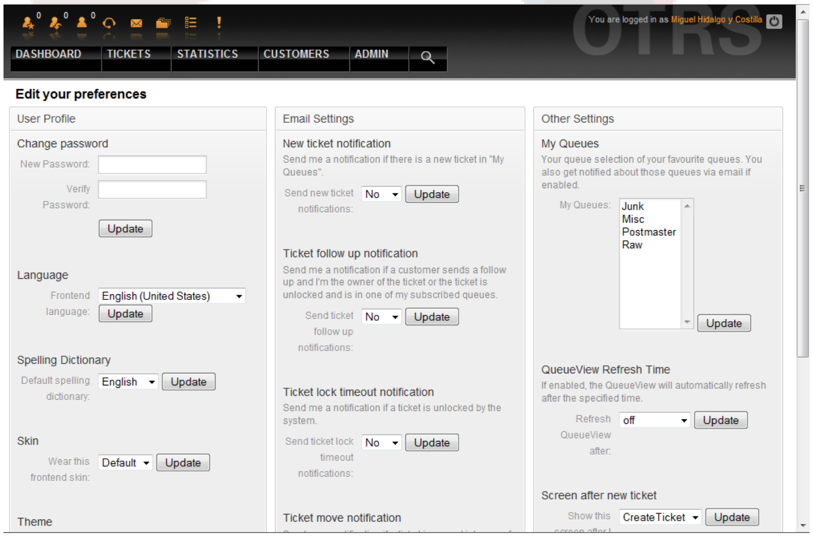
OTRS supports access management for users, groups, and roles, and it is easy to setup queues that are accessible only to some user accounts. Max could also use another way to get his requests into the different queues, with filter rules. Otherwise, if two different mail addresses are used, Sandra only has to dispatch those emails into the two other queues, which can't be dispatched automatically.

Sorting your incoming messages into different queues helps you to keep the support system structured and tidy. Because your agents are arranged into different groups with different access rights on queues, the system can be optimized even further. Queues can be used to define work flow processes or to create the structure of a company. Max could implement, for example, another queue called "sales", which could contain the sub queues "requests", "offers", "orders", "billing", etc. Such a queue structure could help Max to optimize his order transactions.

Improved system structures, such as through the proper design of queues, can lead to significant time and cost savings. Queues can help to optimize the processes in your company.

7. User preferences

OTRS users such as customers, agents and the OTRS administrator can configure their account preferences as per their needs. Agent can access the configuration screen by clicking on their login name at the top right corner of the web interface (see Figure below), and customers must click on the "Preferences" link (see Figure below).



You are logged in as Miguel Hidalgo y Costilla

Edit your preferences

User Profile

Change password

New Password:

Verify Password:

Language

Frontend language: English (United States)

Spelling Dictionary

Default spelling dictionary: English

Skin

Wear this frontend skin: Default

Theme

Email Settings

New ticket notification

Send me a notification if there is a new ticket in "My Queues".

Send new ticket notifications: No

Ticket follow up notification

Send me a notification if a customer sends a follow up and I'm the owner of the ticket or the ticket is unlocked and is in one of my subscribed queues.

Send ticket follow up notifications: No

Ticket lock timeout notification

Send me a notification if a ticket is unlocked by the system.

Send ticket lock timeout notifications: No

Ticket move notification

Other Settings

My Queues

Your queue selection of your favourite queues. You also get notified about those queues via email if enabled.

My Queues: Junk
Misc
Postmaster
Raw

QueueView Refresh Time

If enabled, the QueueView will automatically refresh after the specified time.

Refresh QueueView after: off

Screen after new ticket

Show this screen after: CreateTicket

* * *

An agent can configure 3 different categories of preferences: user profile, email settings, and other settings. The default posibilities are:

User Profile

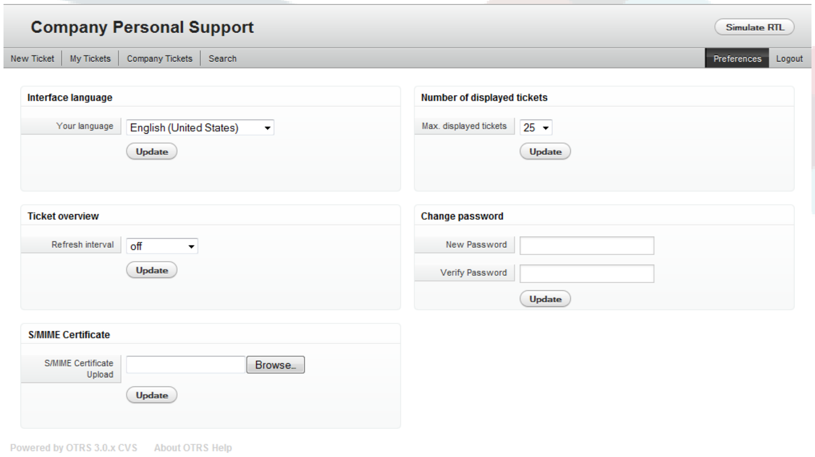
- Change the current password.
- Adjust the interface language.
- Switch the frontend skin.
- Shift the frontend theme.
- Activate and configure the out-of-office time.

Email Settings

- Select events that trigger email notifications to the agent.

Other Settings

- Select the queues you want to monitor in "My Queues".
- Set the refresh period for the queue view.
- Set the screen to be displayed after a ticket is created.


Company Personal Support
Simulate RTL
New Ticket
My Tickets
Company Tickets
Search
Preferences
Logout
Update
Update
Update
Update
Update
Browse..
Powered by OTRS 3.0.x CVS
About OTRS Help

Interface language

Your language: English (United States)

Number of displayed tickets

Max. displayed tickets: 25

Ticket overview

Refresh interval: off

Change password

New Password:

Verify Password:

S/MIME Certificate

S/MIME Certificate Upload:

* * *

A customer can select the web interface language, set the refresh interval for the ticket overview, and choose the maximum amount of shown tickets. It is also possible to set a new password.

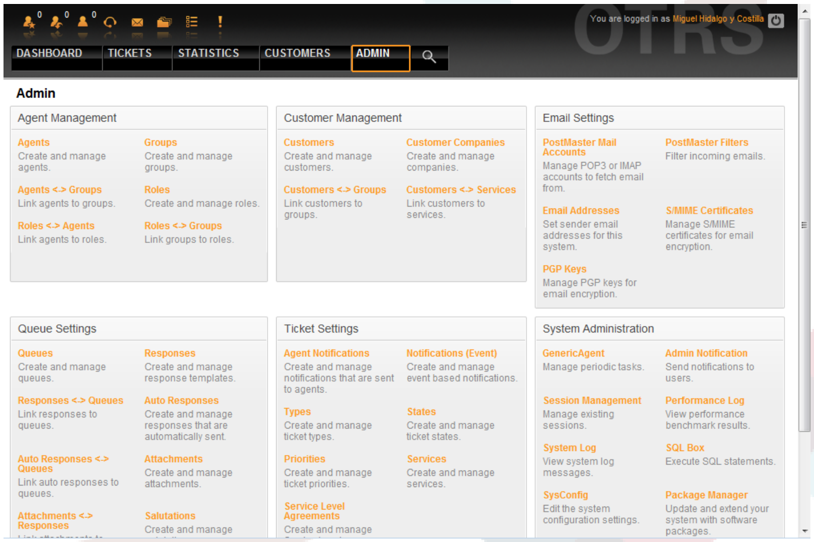
Chapter 4. Administration

1. The ADMIN area of OTRS

1.1. Basics

The following system configuration settings are available to OTRS administrators by accessing the Admin page of the OTRS web interface - adding agents, customers and queues, ticket and mail settings, installing additional packages such as FAQ and ITSM, and much more.

Agents who are members of the ✂✂✂✂✂ group can access the Admin area by clicking the ✂✂✂✂✂ link in the navigation bar (see Figure below). Agents without sufficiently elevated access rights will not be able to access this link.



You are logged in as Miguel Hidalgo y Costilla

Admin

Agent Management

Agents Create and manage agents.
Groups Create and manage groups.
Agents <-> Groups Link agents to groups.
Roles Create and manage roles.
Roles <-> Agents Link agents to roles.
Roles <-> Groups Link groups to roles.

Customer Management

Customers Create and manage customers.
Customer Companies Create and manage companies.
Customers <-> Groups Link customers to groups.
Customers <-> Services Link customers to services.

Email Settings

PostMaster Mail Accounts Manage POP3 or IMAP accounts to fetch email from.
PostMaster Filters Filter incoming emails.
Email Addresses Set sender email addresses for this system.
S/MIME Certificates Manage S/MIME certificates for email encryption.
PGP Keys Manage PGP keys for email encryption.

Queue Settings

Queues Create and manage queues.
Responses Create and manage response templates.
Responses <-> Queues Link responses to queues.
Auto Responses Create and manage responses that are automatically sent.
Auto Responses <-> Queues Link auto responses to queues.
Attachments Create and manage attachments.
Attachments <-> Responses
Salutations Create and manage

Ticket Settings

Agent Notifications Create and manage notifications that are sent to agents.
Notifications (Event) Create and manage event based notifications.
Types Create and manage ticket types.
States Create and manage ticket states.
Priorities Create and manage ticket priorities.
Services Create and manage services.
Service Level Agreements Create and manage

System Administration

GenericAgent Manage periodic tasks.
Admin Notification Send notifications to users.
Session Management Manage existing sessions.
Performance Log View performance benchmark results.
System Log View system log messages.
SQL Box Execute SQL statements.
SysConfig Edit the system configuration settings.
Package Manager Update and extend your system with software packages.

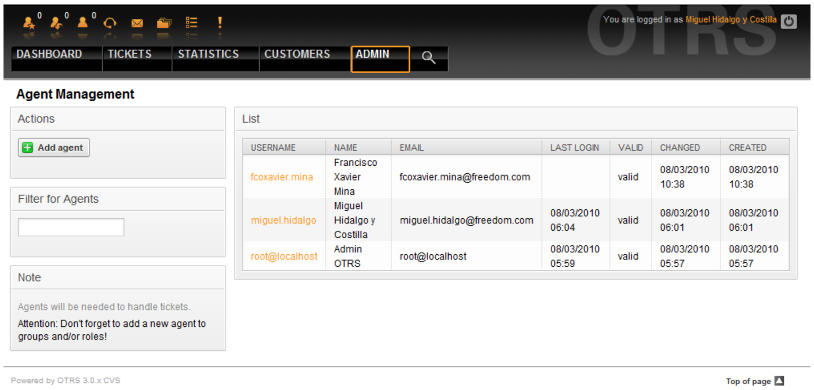
✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂

1.2. Agents, Groups and Roles

1.2.1. Agents

By clicking the link ✂✂✂✂✂✂, you get access to the agent management screen of OTRS (see Figure below). Administrators can add, change or deactivate agent accounts. Furthermore they can also manage agent preferences, including the language and notification settings for the individual agent's interface.

> **Note**
>
> An OTRS agent account may be deactivated but not deleted. Deactivation is done by setting the Valid flag to ✂✂✂✂✂✂✂ or ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂.



DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla

Agent Management

Actions

Add agent

Filter for Agents

Note

Agents will be needed to handle tickets.

Attention: Don't forget to add a new agent to groups and/or roles!

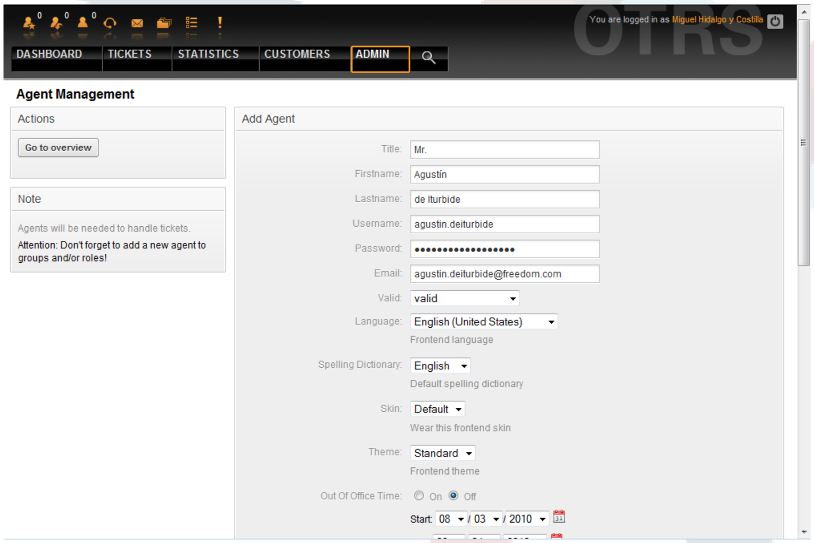
List

USERNAME	NAME	EMAIL	LAST LOGIN	VALID	CHANGED	CREATED
fcoxavier.mina	Francisco Xavier Mina	fcoxavier.mina@freedom.com		valid	08/03/2010 10:38	08/03/2010 10:38
miguel.hidalgo	Miguel Hidalgo y Costilla	miguel.hidalgo@freedom.com	08/03/2010 06:04	valid	08/03/2010 06:01	08/03/2010 06:01
root@localhost	Admin OTRS	root@localhost	08/03/2010 05:59	valid	08/03/2010 05:57	08/03/2010 05:57

Powered by OTRS 3.0.x CVS
Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

To register an agent, click on the "Add agent" button, enter the required data and press the Submit button at the bottom of the screen, as shown in Figure.



DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla

Agent Management

Actions

Go to overview

Note

Agents will be needed to handle tickets.

Attention: Don't forget to add a new agent to groups and/or roles!

Add Agent

Title: Mr.
Firstname: Agustín
Lastname: de Iturbide
Username: agustin.deiturbide
Password: ••••••••••••••••••
Email: agustin.deiturbide@freedom.com
Valid: valid
Language: English (United States)
Frontend language
Spelling Dictionary: English
Default spelling dictionary
Skin: Default
Wear this frontend skin
Theme: Standard
Frontend theme
Out Of Office Time: ○ On ● Off
Start: 08 / 03 / 2010

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂ ✂✂✂✂✂✂

After the new agent account has been created, you should make the agent a member of one or more groups or roles. Information about groups and roles is available in the Groups and Roles sections of this chapter.

1.2.2. Groups

Every agent's account should belong to at least one group or role. In a brand new installation, there are three pre-defined groups available, as shown in Table 5-1.

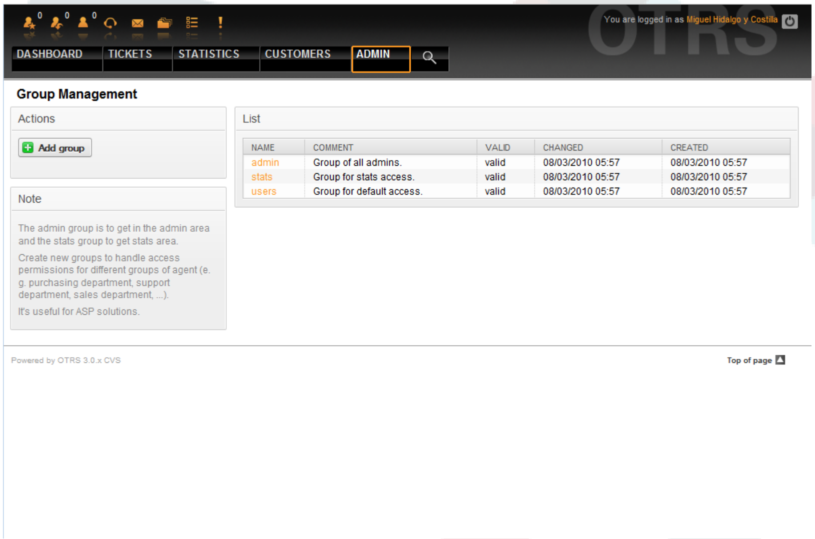
Table 4.1. Default groups available on a fresh OTRS installation

Group	Description
admin	Allowed to perform administrative tasks in the system.
stats	Qualified to access the stats module of OTRS and generate statistics.
users	Agents should belong to this group, with read and write permissions. They can then access all functions of the ticket system.

Note

In a brand new OTRS installation, the ✁✁✁✁✁ group initially does not have any members. The agent 'root@localhost' belongs by default to the admin and stats groups.

You can access the group management page (see Figure below) by clicking the ✁✁✁✁✁✁ link in the admin area.


0
0
0
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Group Management
Actions
Add group
List
Note
Powered by OTRS 3.0.x CVS
Top of page

NAME	COMMENT	VALID	CHANGED	CREATED
admin	Group of all admins.	valid	08/03/2010 05:57	08/03/2010 05:57
stats	Group for stats access.	valid	08/03/2010 05:57	08/03/2010 05:57
users	Group for default access.	valid	08/03/2010 05:57	08/03/2010 05:57

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, ...).

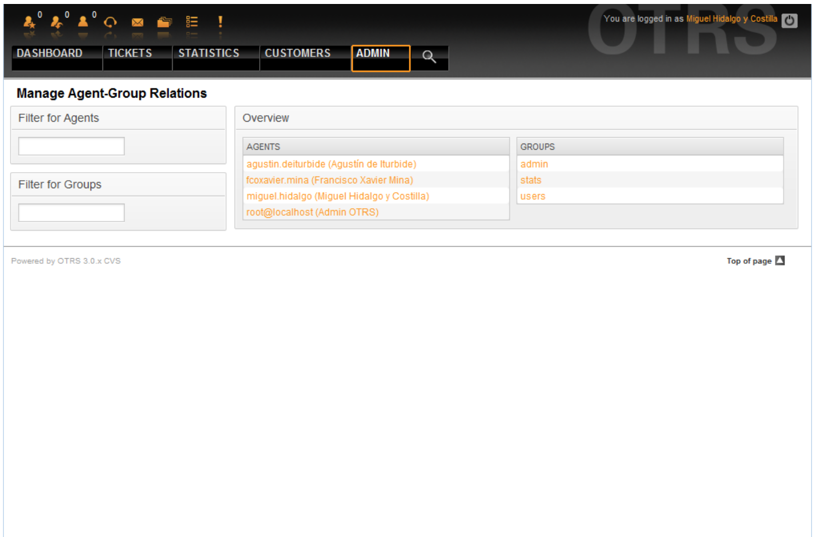
It's useful for ASP solutions.

✁✁✁✁✁✁✁ ✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁

Note

As with agents, an OTRS group may be deactivated but not deleted. Deactivation is done by setting the Valid flag to ✁✁✁✁✁✁✁ or ✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁.

To add an agent to a group, or to change the agents who belong to a group, you can use the link ✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁ from the Admin page (see Figure below).


0
0
0
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Powered by OTRS 3.0.x CVS
Top of page

Manage Agent-Group Relations

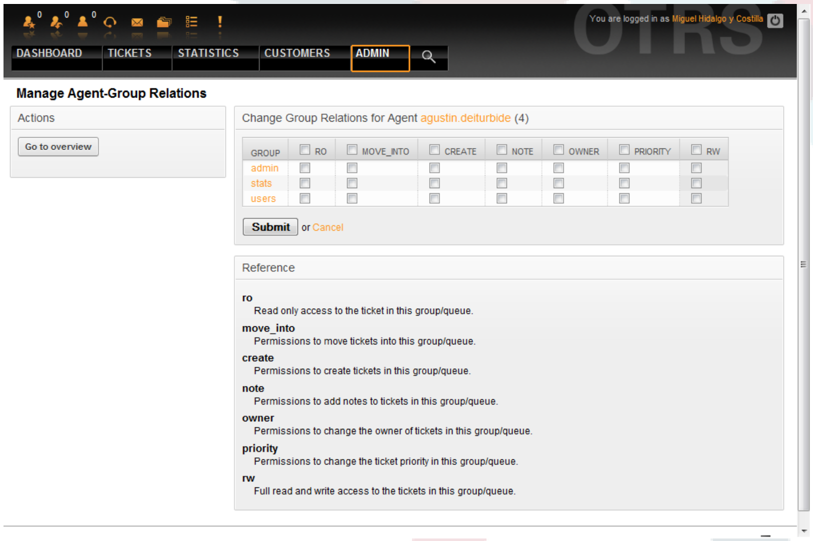
Filter for Agents

Filter for Groups

Overview

AGENTS	GROUPS
agustin.deiturbide (Agustín de Iturbide)	admin
fcoxavier.mina (Francisco Xavier Mina)	stats
miguel.hidalgo (Miguel Hidalgo y Costilla)	users
root@localhost (Admin OTRS)	

An overview of all groups and agents in the system is displayed on this page. You can also use the available filters to find a specific entity. If you want to change the groups that an agent is a member of, just click on the agent's name (see Figure below). To change the agents associated with a group, just click on the group you want to edit (see Figure below).


0
0
0
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Go to overview
Submit
or Cancel

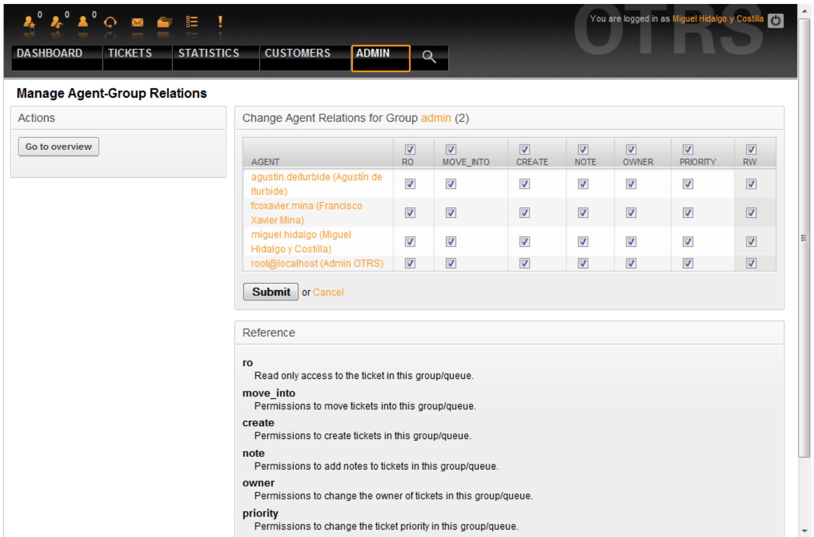
Manage Agent-Group Relations

Change Group Relations for Agent agustin.deiturbide (4)

GROUP	☐ RO	☐ MOVE_INTO	☐ CREATE	☐ NOTE	☐ OWNER	☐ PRIORITY	☐ RW
admin	☐	☐	☐	☐	☐	☐	☐
stats	☐	☐	☐	☐	☐	☐	☐
users	☐	☐	☐	☐	☐	☐	☐

Reference

ro
Read only access to the ticket in this group/queue.
move_into
Permissions to move tickets into this group/queue.
create
Permissions to create tickets in this group/queue.
note
Permissions to add notes to tickets in this group/queue.
owner
Permissions to change the owner of tickets in this group/queue.
priority
Permissions to change the ticket priority in this group/queue.
rw
Full read and write access to the tickets in this group/queue.



Manage Agent-Group Relations

Actions

Go to overview

Change Agent Relations for Group admin (2)

AGENT	☑ RO	☑ MOVE_INTO	☑ CREATE	☑ NOTE	☑ OWNER	☑ PRIORITY	☑ RW
agustin.deiturbide (Agustín de Iturbide)	☑	☑	☑	☑	☑	☑	☑
fcoxavier.mina (Francisco Xavier Mina)	☑	☑	☑	☑	☑	☑	☑
miguel.hidalgo (Miguel Hidalgo y Costilla)	☑	☑	☑	☑	☑	☑	☑
root@localhost (Admin OTRS)	☑	☑	☑	☑	☑	☑	☑

Submit or Cancel

Reference

ro
Read only access to the ticket in this group/queue.

move_into
Permissions to move tickets into this group/queue.

create
Permissions to create tickets in this group/queue.

note
Permissions to add notes to tickets in this group/queue.

owner
Permissions to change the owner of tickets in this group/queue.

priority
Permissions to change the ticket priority in this group/queue.

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂

Each group has a set of rights associated with it, and each group member (agent) may have some combination of these rights for themselves. A list of the permissions / rights is shown in Table 5-2.

Table 4.2. Rights associated with OTRS Groups

Right	Description
ro	Read only access to the tickets, entries and queues of this group.
move into	Right to move tickets or entries between queues or areas that belong to this group.
create	Right to create tickets or entries in the queues or areas of this group.
owner	Right to update the owner of tickets or entries in queues or areas that belong to this group.
priority	Right to change the priority of tickets or entries in queues or areas that belong to this group.
rw	Full read and write access on tickets or entries in the queues or areas that belong to this group.

Note

By default, the QueueView only lists tickets in queues that an agent has ✂✂ access to, i.e., the tickets the agent needs to work on. If you want to change this behaviour, you can set `Ticket::Frontend::AgentTicketQueue###ViewAllPossibleTickets` to ✂✂✂.

1.2.3. Roles

Roles are a powerful feature to manage the access rights of many agents in a very simple and quick manner. They are particularly useful for large, complex support systems with a lot of agents, groups and queues. An example below explains when they should be used.

Suppose that you have a system with 100 agents, 90 of them with access to a single queue called "support" where all support requests are handled. The "support" queue contains multiple sub queues. The other 10 agents have permission to access all queues of the system. These 10 agents dispatch tickets, watch the raw queue and move spam messages into the "junk" queue.

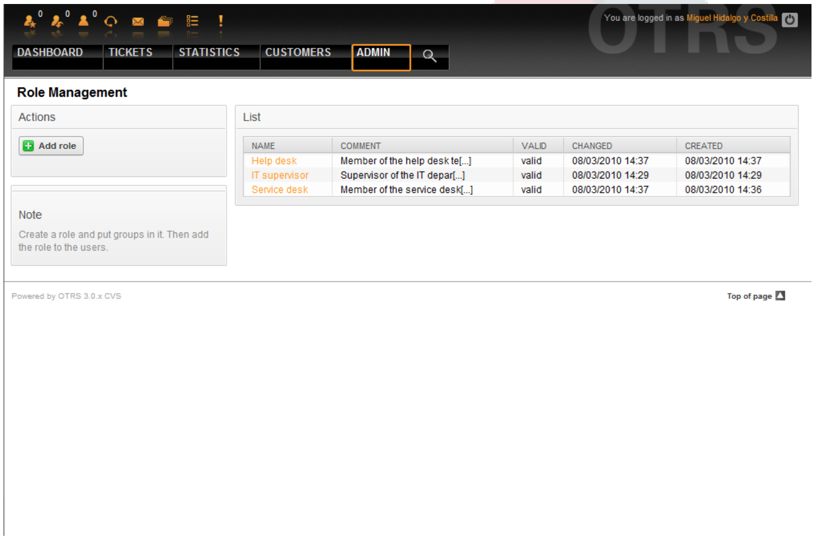
The company now opens a new department that sells some products. Order request and acceptance, order confirmation, bills, etc. must be processed, and some of the company's agents are supposed to do this using OTRS. The different agents have to get access to the new queues that must be created.

Because it would take a long time to change the access rights for the individual agents manually, roles that define the different access levels can be created. The agents can then be added to one or more roles, with their access rights being modified automatically. If a new agent account is created, it is also possible to add this account to one or more roles.

Note

Roles are really useful when dealing with complex organizations and when maintaining larger OTRS installations. Proper care is advised though. Mixing Agent to Group with Agent to Role mappings can make for a complex access control scheme, that is difficult to understand and maintain. If you wish to use only roles and disable the Agents <-> Groups option in the Admin area, you can do so by modifying the Frontend::Module###AdminUserGroup in the SysConfig. Be aware that this won't remove already existing Agents to Group assignments!

You can access the role management section (see Figure below) by clicking the ✂✂✂✂✂ link on the Admin page.



You are logged in as Miguel Hidalgo y Costilla

Role Management

Actions

List

NAME	COMMENT	VALID	CHANGED	CREATED
Help desk	Member of the help desk te[...]	valid	08/03/2010 14:37	08/03/2010 14:37
IT supervisor	Supervisor of the IT depar[...]	valid	08/03/2010 14:29	08/03/2010 14:29
Service desk	Member of the service desk[...]	valid	08/03/2010 14:37	08/03/2010 14:36

Note

Create a role and put groups in it. Then add the role to the users.

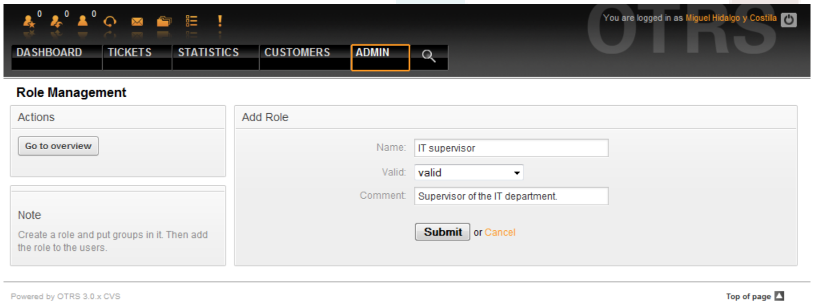
Powered by OTRS 3.0.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

Note

As with agent and groups, roles once created can be deactivated but not deleted. To deactivate, set the Valid option to ✂✂✂✂✂✂✂ or ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂.

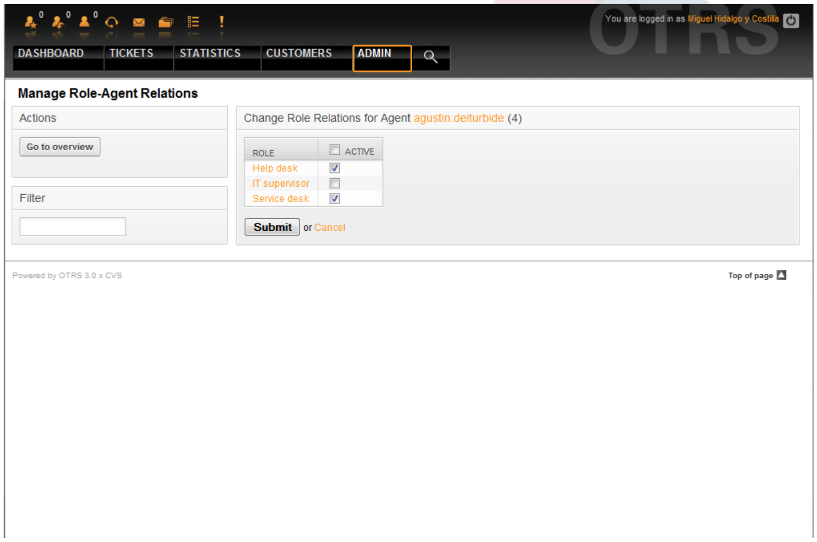
An overview of all roles in the system is displayed. To edit a role's settings, click on the role's name. In a fresh new OTRS installation, there are no roles defined by default. To register one, click on the "Add role" button, provide the needed data and submit it (see Figure below).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Role Management
Actions
Go to overview
Note
Create a role and put groups in it. Then add the role to the users.
Add Role
Submit
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

Name: IT supervisor
Valid: valid
Comment: Supervisor of the IT department.

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂ ✂✂✂✂✂

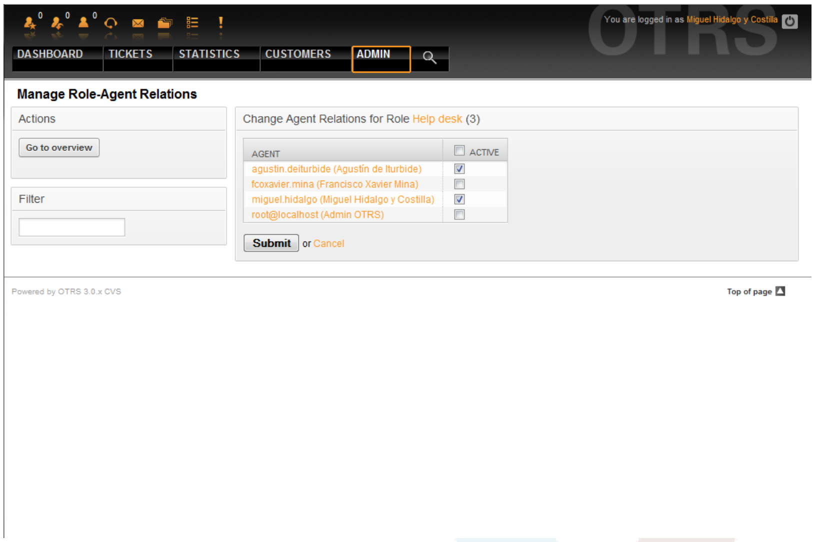
To get an overview of all roles and agents in the system, click on the link Roles <-> Agents on the Admin page. You can also use filters to find a specific element. If you want to change the roles associated with an agent, just click on the agent's name (see Figure below). To change the agents associated with a role, click on the role you want to edit (see Figure below).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Manage Role-Agent Relations
Actions
Go to overview
Filter
Submit
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

Change Role Relations for Agent agustin.deiturbide (4)

ROLE	☐ ACTIVE
Help desk	☑
IT supervisor	☐
Service desk	☑

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂ ✂✂✂✂✂✂


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Actions
Go to overview
Filter
Submit
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

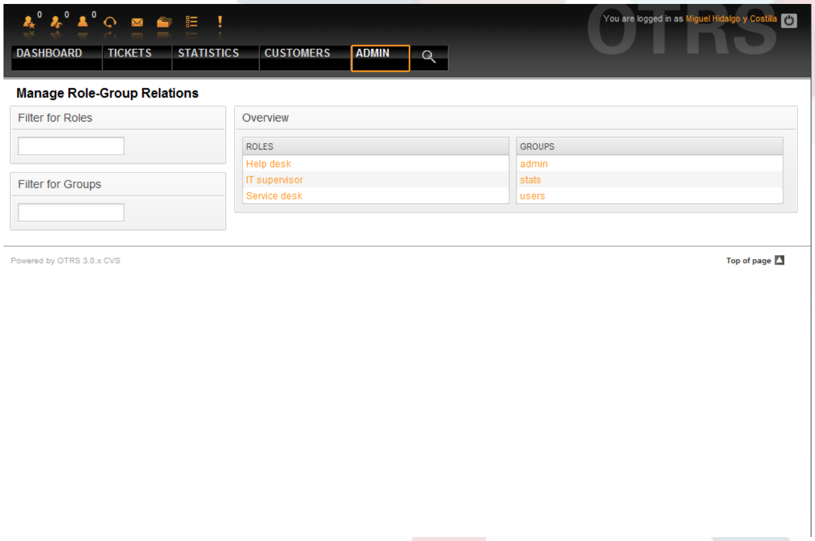
Manage Role-Agent Relations

Change Agent Relations for Role Help desk (3)

AGENT	☐ ACTIVE
agustin.deiturbide (Agustín de Iturbide)	☑
fcoxavier.mina (Francisco Xavier Mina)	☐
miguel.hidalgo (Miguel Hidalgo y Costilla)	☑
root@localhost (Admin OTRS)	☐

✃✃✃✃✃✃✃✃ ✃✃✃✃✃✃✃ ✃✃✃✃ ✃✃✃✃✃✃✃ ✃✃✃✃✃✃✃✃✃✃✃ ✃✃✃✃✃ ✃ ✃✃✃✃✃✃✃✃✃ ✃✃✃✃✃

To get an overview of all roles and groups in the system, click on the link Roles <-> Groups on the Admin page. You will see a similar screen as the one shown in the Figure. You can also use filters to find a specific entity.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Filter for Roles
Filter for Groups
Powered by OTRS 3.0.x CVS
Top of page

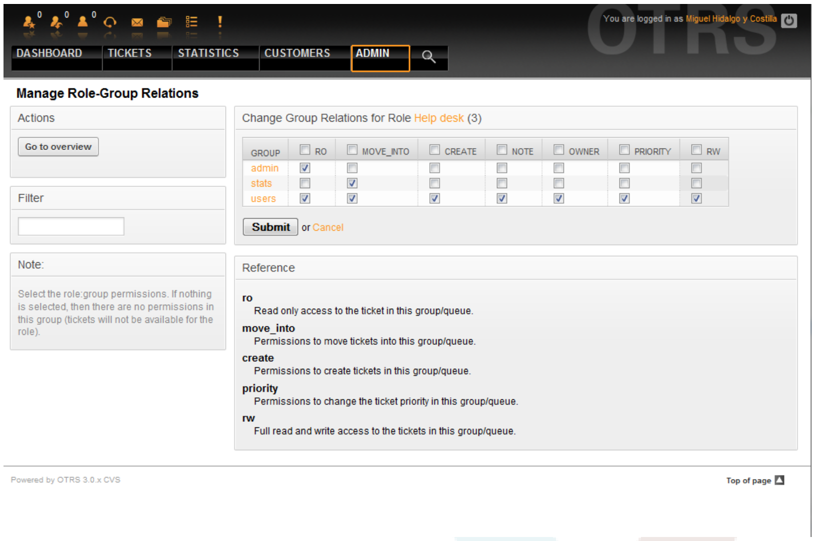
Manage Role-Group Relations

Overview

ROLES	GROUPS
Help desk	admin
IT supervisor	stats
Service desk	users

✃✃✃✃✃✃✃✃ ✃✃✃✃✃✃✃ ✃✃✃✃✃✃✃✃✃✃✃✃ ✃✃✃✃✃✃✃✃✃✃

To define the different access rights for a role, click on the name of a role or a group (see below the Figures 5.13 and 5.14, respectively).


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Actions
Go to overview
Filter
Submit
Powered by OTRS 3.0.x CVS
Top of page

Manage Role-Group Relations

Change Group Relations for Role Help desk (3)

GROUP	☐ RO	☐ MOVE_INTO	☐ CREATE	☐ NOTE	☐ OWNER	☐ PRIORITY	☐ RW
admin	☑	☐	☐	☐	☐	☐	☐
stats	☐	☑	☐	☐	☐	☐	☐
users	☑	☑	☑	☑	☑	☑	☑

Submit or Cancel

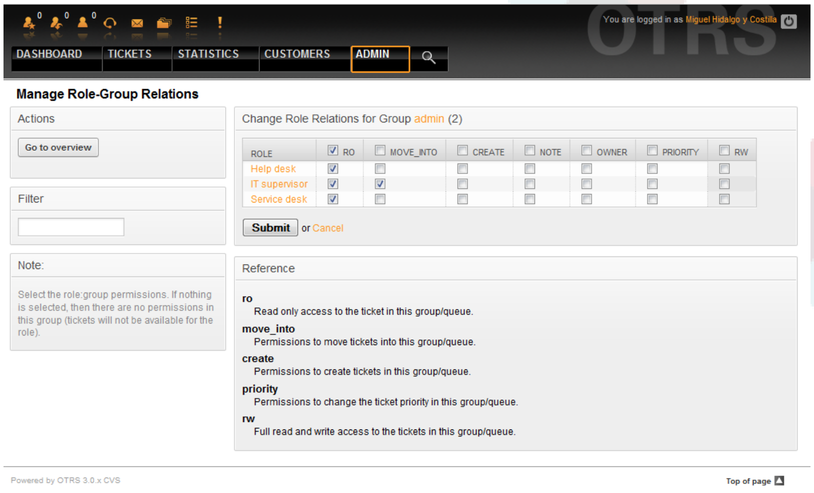
Note:

Select the role:group permissions. If nothing is selected, then there are no permissions in this group (tickets will not be available for the role).

Reference

ro
Read only access to the ticket in this group/queue.
move_into
Permissions to move tickets into this group/queue.
create
Permissions to create tickets in this group/queue.
priority
Permissions to change the ticket priority in this group/queue.
rw
Full read and write access to the tickets in this group/queue.

* * *


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Actions
Go to overview
Filter
Submit
Powered by OTRS 3.0.x CVS
Top of page

Manage Role-Group Relations

Change Role Relations for Group admin (2)

ROLE	☑ RO	☐ MOVE_INTO	☐ CREATE	☐ NOTE	☐ OWNER	☐ PRIORITY	☐ RW
Help desk	☑	☐	☐	☐	☐	☐	☐
IT supervisor	☑	☑	☐	☐	☐	☐	☐
Service desk	☑	☐	☐	☐	☐	☐	☐

Submit or Cancel

Note:

Select the role:group permissions. If nothing is selected, then there are no permissions in this group (tickets will not be available for the role).

Reference

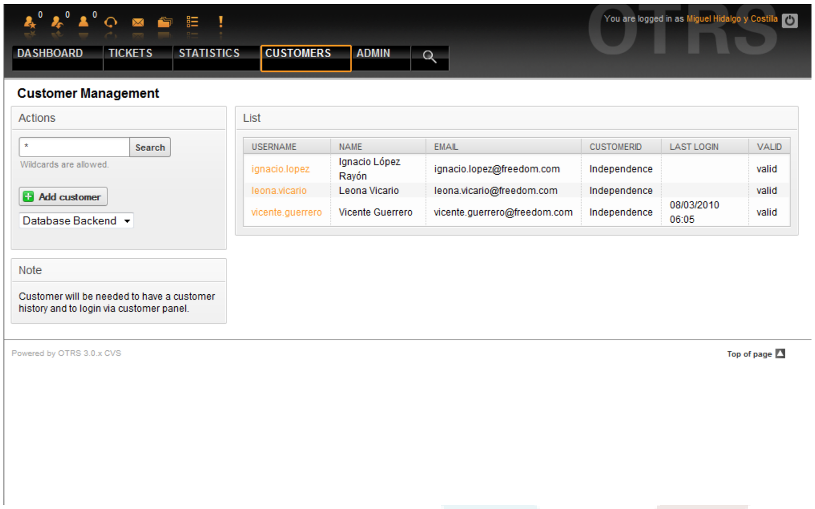
ro
Read only access to the ticket in this group/queue.
move_into
Permissions to move tickets into this group/queue.
create
Permissions to create tickets in this group/queue.
priority
Permissions to change the ticket priority in this group/queue.
rw
Full read and write access to the tickets in this group/queue.

* * *

1.3. Customers and Customer Groups

1.3.1. Customers

OTRS supports different types of users. Using the link "Customers" (via the navigation bar, or the Admin page), you can manage the accounts of your customers (see Figure below), who can log into the system via the Customers interface (customer.pl). Through this interface, your customers can not only create tickets but also review their past tickets for new updates. It is important to know that a customer is needed for the ticket history in the system.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Customer Management
Actions
Search
Wildcards are allowed.
Add customer
Database Backend
List
Note
Powered by OTRS 3.0.x CVS
Top of page

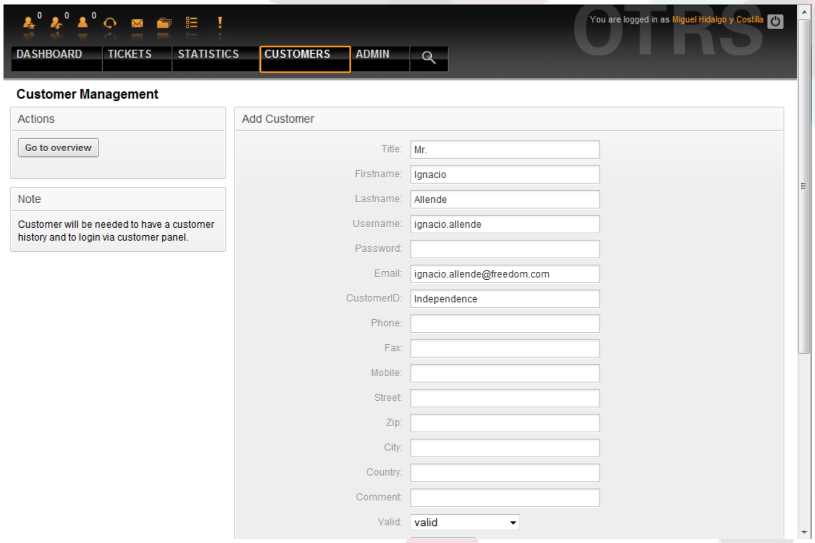
USERNAME	NAME	EMAIL	CUSTOMERID	LAST LOGIN	VALID
ignacio.lopez	Ignacio López Rayón	ignacio.lopez@freedom.com	Independence		valid
leona.vicario	Leona Vicario	leona.vicario@freedom.com	Independence		valid
vicente.guerrero	Vicente Guerrero	vicente.guerrero@freedom.com	Independence	08/03/2010 06:05	valid

Customer will be needed to have a customer history and to login via customer panel.

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

You can search for a registered customer, or edit their settings by clicking on their name. You also have the possibility to change the customer back-end, for further information please refer to the chapter about external back-ends.

To create a new customer account, click on the "Add customer" button (see Figure below). Some of the fields are mandatory, i.e., they have to contain values, so if you leave one of those empty, it will be highlighted in red.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Customer Management
Actions
Go to overview
Note
Add Customer

Customer will be needed to have a customer history and to login via customer panel.

Title:	Mr.
Firstname:	Ignacio
Lastname:	Allende
Username:	ignacio.allende
Password:	
Email:	ignacio.allende@freedom.com
CustomerID:	Independence
Phone:	
Fax:	
Mobile:	
Street:	
Zip:	
City:	
Country:	
Comment:	
Valid:	valid

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂

Customers can access the system by providing their username and password. The CustomerID is needed by the system to identify the user and associated tickets. Since the email address is a unique value, it can be used as the ID.

Note

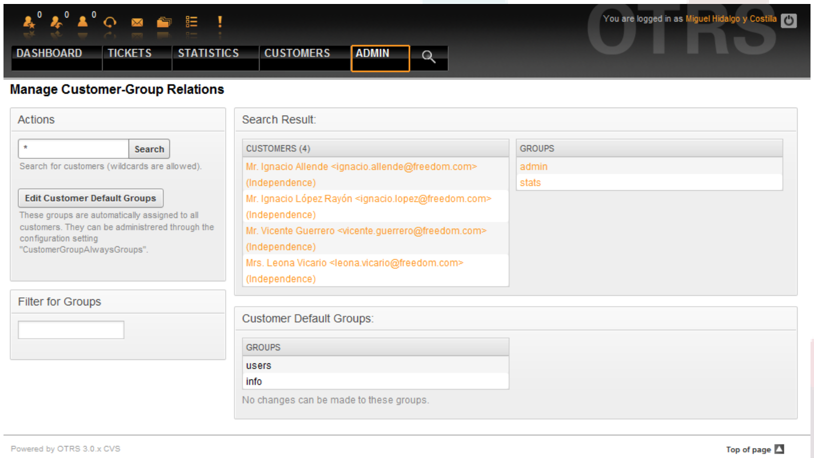
As with agents, groups and roles, customers can not be deleted from the system, only deactivated by setting the Valid option to ✂✂✂✂✂✂✂ or ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂.

1.3.2. Customer Groups

Customer users can also be added to a group, which can be useful if you want to add customers of the same company with access to one or a few queues. First create the group to which your customers will belong, via the Group management module. Then add the queues and select the new group for the queues.

The next step is to activate the customer group support. This can be done with the configuration parameter CustomerGroupSupport, from the Admin SysConfig option. Using the parameter CustomerGroupAlwaysGroups, you can specify the default groups for a newly added customer, so that every new account will be automatically added to these groups.

Through the link "Customers <-> Groups" you can manage which customer shall belong to the different groups (see Figure below).



You are logged in as Miguel Hidalgo y Costilla

Manage Customer-Group Relations

Actions

Search for customers (wildcards are allowed).

These groups are automatically assigned to all customers. They can be administrered through the configuration setting "CustomerGroupAlwaysGroups".

Filter for Groups

Search Result:

CUSTOMERS (4)
Mr. Ignacio Allende <ignacio.allende@freedom.com> (Independence)
Mr. Ignacio López Rayón <ignacio.lopez@freedom.com> (Independence)
Mr. Vicente Guerrero <vicente.guerrero@freedom.com> (Independence)
Mrs. Leona Vicario <leona.vicario@freedom.com> (Independence)

GROUPS
admin
stats

Customer Default Groups:

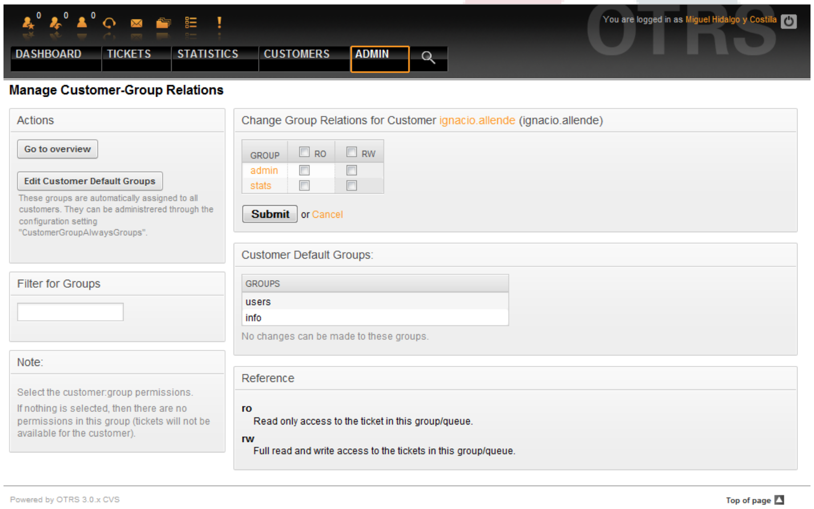
GROUPS
users
info

No changes can be made to these groups.

Powered by OTRS 3.0.x CVS

* * *

To define the different groups a customer should be part of and vice versa, click on the corresponding customer username or group (see below the Figures 5.16 and 5.17, respectively).



You are logged in as Miguel Hidalgo y Costilla

Manage Customer-Group Relations

Actions

These groups are automatically assigned to all customers. They can be administrered through the configuration setting "CustomerGroupAlwaysGroups".

Filter for Groups

Note:

Select the customer:group permissions.

If nothing is selected, then there are no permissions in this group (tickets will not be available for the customer).

Change Group Relations for Customer ignacio.allende (ignacio.allende)

GROUP	☐ RO	☐ RW
admin	☐	☐
stats	☐	☐

Customer Default Groups:

GROUPS
users
info

No changes can be made to these groups.

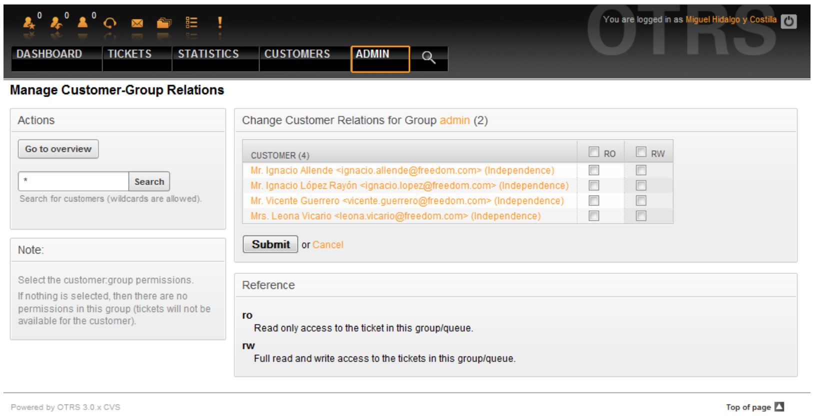
Reference

ro
Read only access to the ticket in this group/queue.

rw
Full read and write access to the tickets in this group/queue.

Powered by OTRS 3.0.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂



You are logged in as Miguel Hidalgo y Costilla

Manage Customer-Group Relations

Actions

*

Search for customers (wildcards are allowed).

Note:

Select the customer:group permissions.

If nothing is selected, then there are no permissions in this group (tickets will not be available for the customer).

Change Customer Relations for Group admin (2)

CUSTOMER (4)	☐ RO	☐ RW
Mr. Ignacio Allende <ignacio.allende@freedom.com> (Independence)	☐	☐
Mr. Ignacio López Rayón <ignacio.lopez@freedom.com> (Independence)	☐	☐
Mr. Vicente Guerrero <vicente.guerrero@freedom.com> (Independence)	☐	☐
Mrs. Leona Vicario <leona.vicario@freedom.com> (Independence)	☐	☐

or Cancel

Reference

ro
Read only access to the ticket in this group/queue.

rw
Full read and write access to the tickets in this group/queue.

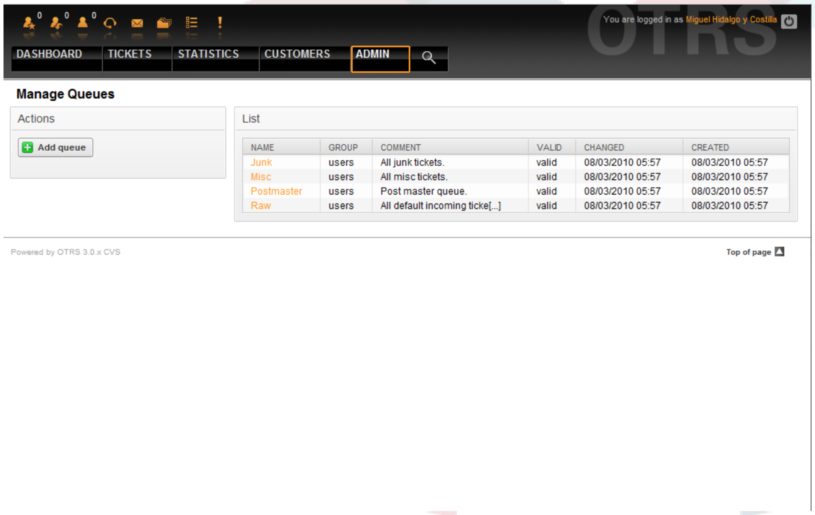
Powered by OTRS 3.0.x CVS

Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂ ✂✂✂✂✂✂

1.4. Queues

Clicking on the link "Queues" of the Admin page, you can manage the queues of your system (see Figure below). In a new OTRS installation there are 4 default queues: Raw, Junk, Misc and Postmaster. All incoming messages will be stored in the "Raw" queue if no filter rules are defined. The "Junk" queue can be used to store spam messages.



You are logged in as Miguel Hidalgo y Costilla

Manage Queues

Actions

List

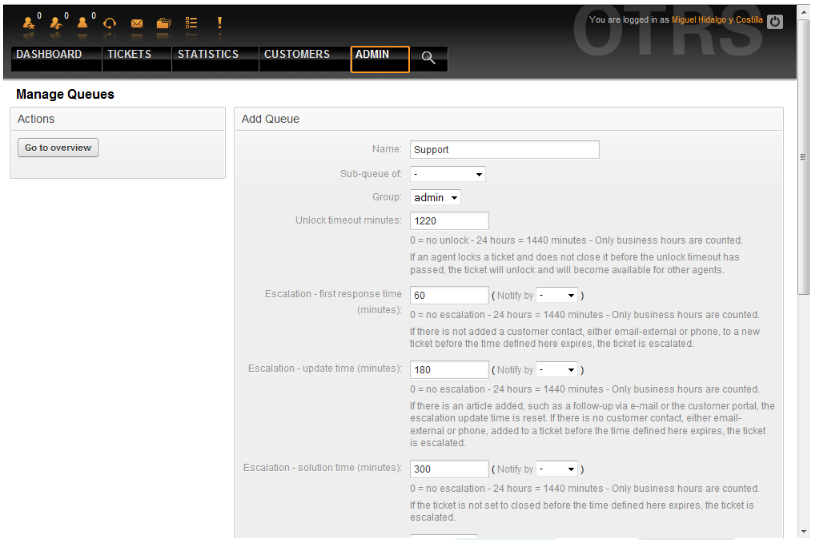
NAME	GROUP	COMMENT	VALID	CHANGED	CREATED
Junk	users	All junk tickets.	valid	08/03/2010 05:57	08/03/2010 05:57
Misc	users	All misc tickets.	valid	08/03/2010 05:57	08/03/2010 05:57
Postmaster	users	Post master queue.	valid	08/03/2010 05:57	08/03/2010 05:57
Raw	users	All default incoming ticke[...]	valid	08/03/2010 05:57	08/03/2010 05:57

Powered by OTRS 3.0.x CVS

Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

Here you can add queues (see Figure below) and modify them. You can specify the group that should use the queue. You can also set the queue as a sub-queue of an existing queue.



You are logged in as Miguel Hidalgo y Costilla

Manage Queues

Actions

Add Queue

Name: Support

Sub-queue of: -

Group: admin

Unlock timeout minutes: 1220

0 = no unlock - 24 hours = 1440 minutes - Only business hours are counted.

If an agent locks a ticket and does not close it before the unlock timeout has passed, the ticket will unlock and will become available for other agents.

Escalation - first response time (minutes): 60 (Notify by -)

0 = no escalation - 24 hours = 1440 minutes - Only business hours are counted.

If there is not added a customer contact, either email-external or phone, to a new ticket before the time defined here expires, the ticket is escalated.

Escalation - update time (minutes): 180 (Notify by -)

0 = no escalation - 24 hours = 1440 minutes - Only business hours are counted.

If there is an article added, such as a follow-up via e-mail or the customer portal, the escalation update time is reset. If there is no customer contact, either email-external or phone, added to a ticket before the time defined here expires, the ticket is escalated.

Escalation - solution time (minutes): 300 (Notify by -)

0 = no escalation - 24 hours = 1440 minutes - Only business hours are counted.

If the ticket is not set to closed before the time defined here expires, the ticket is escalated.

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂ ✂✂✂✂✂✂

You can define an unlock timeout for a queue - if an agent locks a ticket and does not close it before the unlock timeout has passed, the ticket will be automatically unlocked and made available for other agents to work on.

There are three escalation time settings that can be associated at queue level:

Escalation - First Response Time

- After creation of the ticket, if the time defined here expires without any communication with the customer, either by email or phone, the ticket is escalated.

Escalation - Update Time

- If there is a customer followup either via e-mail or the customer portal, that is recorded in the ticket, the escalation update time is reset. If there is no customer contact before the time defined here expires, the ticket is escalated.

Escalation - Solution Time

- If the ticket is not closed before the time defined here expires, the ticket is escalated.

With 'Ticket lock after a follow-up', you can define if a ticket should be set to 'locked' to the old owner if a ticket that has been closed and later is re-opened. This ensures that a follow up for a ticket is processed by the agent that has previously handled that ticket.

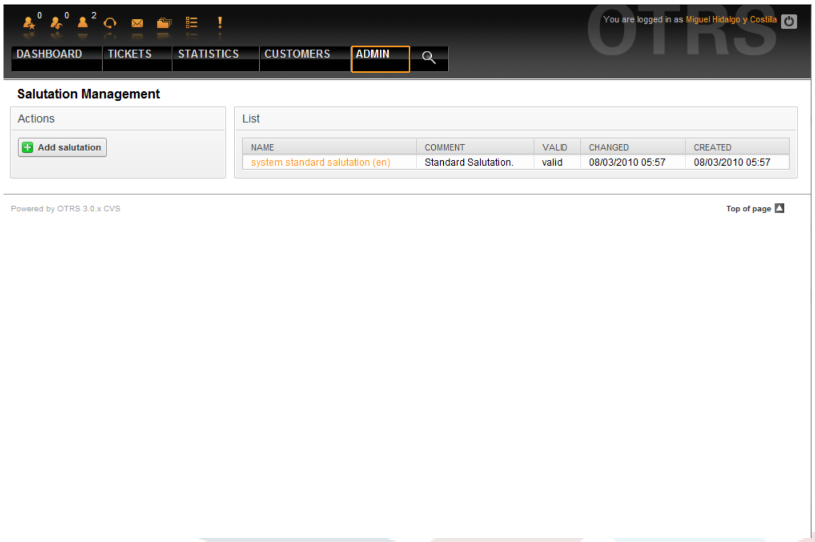
The parameter for the system address specifies the email address that will be used for the outgoing tickets of this queue. There is also the possibility to associate a queue with a salutation and a signature, for the email answers. For more detailed information, please refer to the sections email addresses, salutations and signatures.

> Note
>
> As with agents, groups and customers, queues cannot be deleted, only deactivated, by setting the Valid option to ✂✂✂✂✂✂✂ or ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂.

1.5. Salutations, signatures, attachments and responses

1.5.1. Salutations

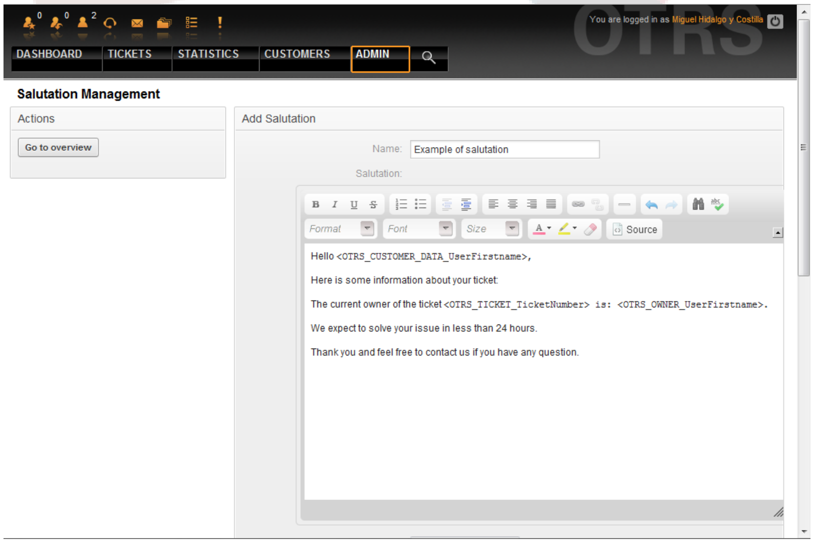
A salutation is a text module for a response. Salutations can be linked to one or more queues, as described in the section about queues. A salutation is used only if a ticket from a queue the salutation is linked to, is answered. To manage the different salutations of your system, use the "Salutations" link of the admin area (see Figure below).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Salutation Management
Actions
Add salutation
List
Powered by OTRS 3.0.x CVS
Top of page

NAME	COMMENT	VALID	CHANGED	CREATED
system standard salutation (en)	Standard Salutation.	valid	08/03/2010 05:57	08/03/2010 05:57

After a default installation there is already one salutation available, "system standard salutation (en)".

To create a new salutation, press the button "Add salutation", provide the required data and submit it (see Figure below).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Salutation Management
Actions
Go to overview
Add Salutation
Format
Font
Size
Source

Name: Example of salutation

Salutation:

Hello <OTRS_CUSTOMER_DATA_UserFirstname>,

Here is some information about your ticket:

The current owner of the ticket <OTRS_TICKET_TicketNumber> is: <OTRS_OWNER_UserFirstname>.

We expect to solve your issue in less than 24 hours.

Thank you and feel free to contact us if you have any question.

It is possible to use variables in salutations. When you respond to a ticket, the variable names will be replaced by their values.

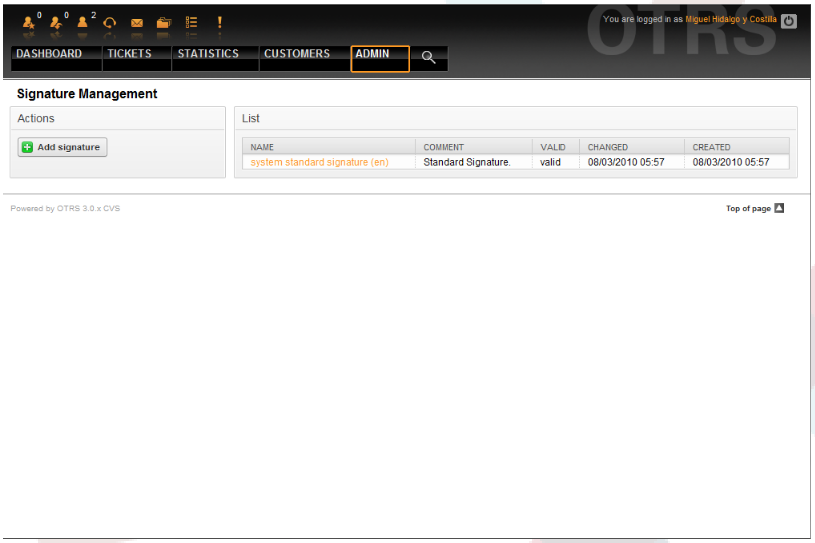
The different variables you can use in responses are listed in the lower part of the salutation screen. If you use, for example, the variable <OTRS_LAST_NAME> the last name of the ticket's sender will be included in your reply.

Note

As with other OTRS entities, salutations cannot be deleted, only deactivated by setting the Valid option to ✂✂✂✂✂✂✂ or ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂.

1.5.2. Signatures

Another text module for a response is the signature. Signatures can be linked to a queue, as described in the section about the queues. Please note that a signature will only be appended to a response text, if it has previously been linked to a queue. You can manage the signatures in your system by accessing the "Signatures" link of the Admin page, (see Figure below).

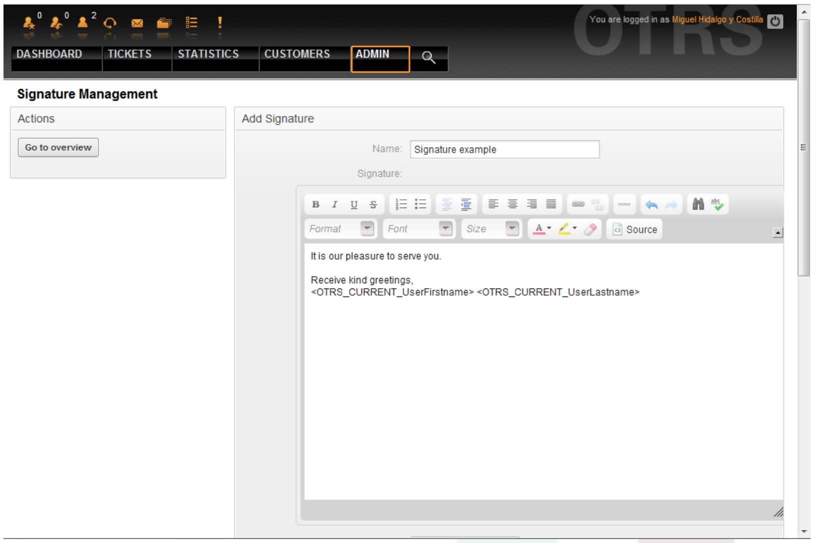

You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Signature Management
Actions
Add signature
List
Powered by OTRS 3.0.x CVS
Top of page

NAME	COMMENT	VALID	CHANGED	CREATED
system standard signature (en)	Standard Signature.	valid	08/03/2010 05:57	08/03/2010 05:57

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

After a fresh installation of OTRS, there is one predefined signature stored in your system, "system standard signature (en)".

To create a new signature, press the button "Add signature", provide the needed data and submit it (see Figure below).


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Actions
Go to overview
Format
Font
Size
Source

Signature Management

Add Signature

Name: Signature example

Signature:

It is our pleasure to serve you.

Receive kind greetings,
<OTRS_CURRENT_UserFirstname> <OTRS_CURRENT_UserLastname>

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁ ✁✁✁ ✁✁✁✁✁✁✁✁✁✁

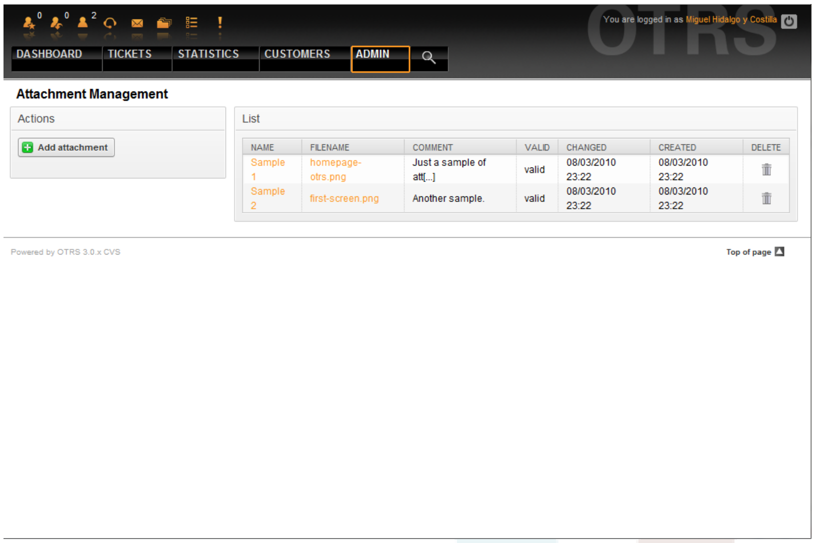
Like salutations, signatures can also contain dynamic content, such as the first and last name of the agent who answers the ticket. Here too, variables can be used to replace the content of the signature text for every ticket. See the lower part of the signatures screen for the variables which can be used. If you include the variable <OTRS_LAST_NAME> in a signature for example, the last name of the agent who answers the ticket will replace the variable.

> ## Note
>
> As with salutations, signatures too cannot be deleted, only deactivated by setting the Valid option to ✁✁✁✁✁✁✁ or ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁.

1.5.3. Attachments

You can also optionally add one or more attachments to a response. If the response is selected, the attachments will be attached to the message composition window. If necessary, the agent can remove the attachment from an individual response before sending it to the customer.

Through the "Attachment" link of the Admin page, you can load the attachments into the database of the system (see Figure below).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Add attachment
Powered by OTRS 3.0.x CVS
Top of page

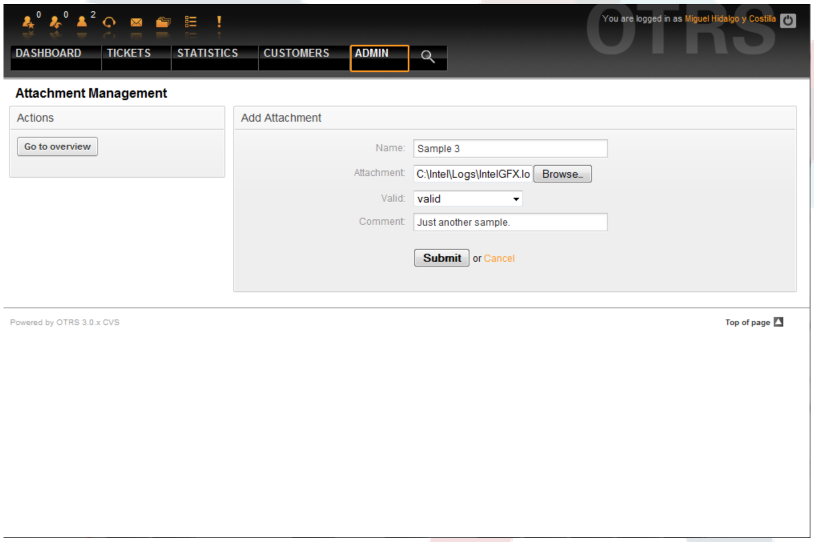
Attachment Management

List

NAME	FILENAME	COMMENT	VALID	CHANGED	CREATED	DELETE
Sample 1	homepage-otrs.png	Just a sample of att[...]	valid	08/03/2010 23:22	08/03/2010 23:22	🗑
Sample 2	first-screen.png	Another sample.	valid	08/03/2010 23:22	08/03/2010 23:22	🗑

✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁

To create a new attachment, press the button "Add attachment", provide the required data and submit it (see Figure below).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Go to overview
Browse...
Submit
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

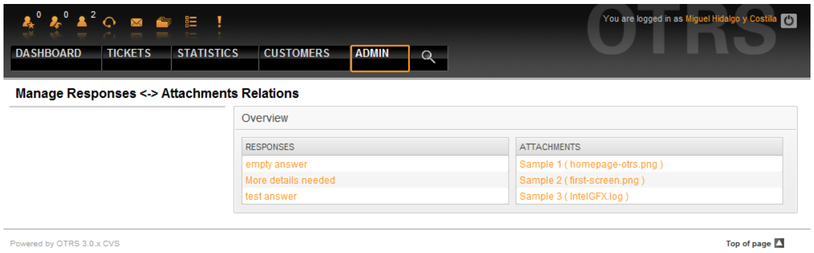
Attachment Management

Add Attachment

Name: Sample 3
Attachment: C:\Intel\Logs\IntelGFX.lo
Valid: valid
Comment: Just another sample.

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁ ✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁

If an attachment is stored it can be linked to one or more responses. Click on the "Attachment <-> Responses" link of the Admin page (see Figure below).


0
0
2
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
OTRS

You are logged in as Miguel Hidalgo y Costilla

Manage Responses <-> Attachments Relations

Overview

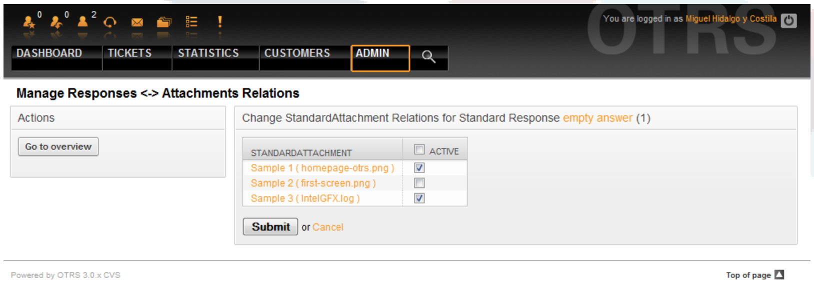
RESPONSES	ATTACHMENTS
empty answer	Sample 1 (homepage-otrs.png)
More details needed	Sample 2 (first-screen.png)
test answer	Sample 3 (IntelGFX.log)

Powered by OTRS 3.0.x CVS

Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂✂✂✂

To associate different attachments with a specific response and vice versa, click on the corresponding response name or attachment (see below the Figures 5.27 and 5.28, respectively).


0
0
2
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
OTRS

You are logged in as Miguel Hidalgo y Costilla

Manage Responses <-> Attachments Relations

Actions
Go to overview

Change StandardAttachment Relations for Standard Response empty answer (1)

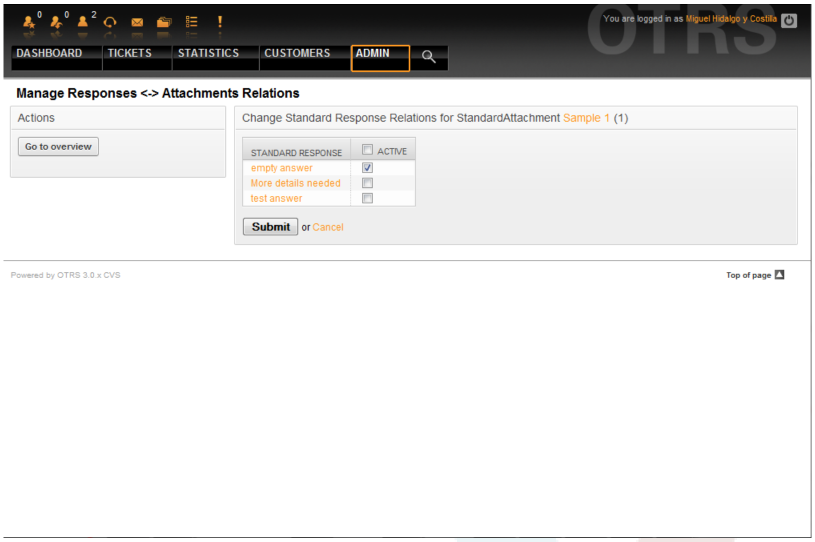
STANDARDATTACHMENT	☐ ACTIVE
Sample 1 (homepage-otrs.png)	☑
Sample 2 (first-screen.png)	☐
Sample 3 (IntelGFX.log)	☑

Submit
or Cancel

Powered by OTRS 3.0.x CVS

Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂


0
0
2
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Go to overview
Submit
Cancel
Powered by OTRS 3.0.x CVS
Top of page

Manage Responses <-> Attachments Relations

Change Standard Response Relations for StandardAttachment Sample 1 (1)

STANDARD RESPONSE	☐ ACTIVE
empty answer	☑
More details needed	☐
test answer	☐

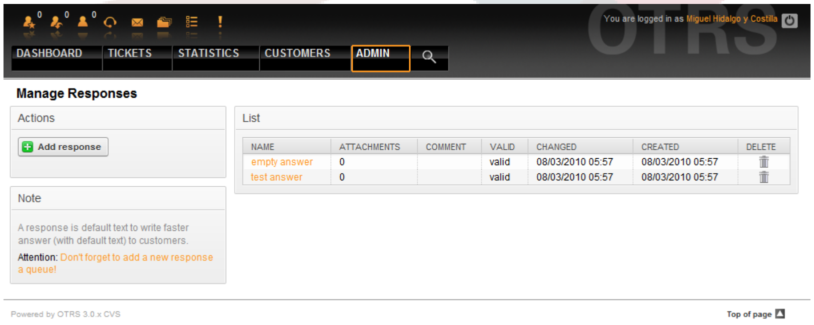
Submit or Cancel

* * *

1.5.4. Responses

To speed up ticket processing and to standardize the look of answers, you can define responses in OTRS. A response can be linked to one or more queues and vice versa. In order to be able to use a response quickly, the different responses are displayed below every ticket in the QueueView or in "My Queues".

For a fresh OTRS installation, the "empty answer" response is set as the default for every queue. Clicking the "Responses" link on the Admin page brings you to the Responses management page (see Figure below).


0
0
0
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Add response
List
Powered by OTRS 3.0.x CVS
Top of page

Manage Responses

NAME	ATTACHMENTS	COMMENT	VALID	CHANGED	CREATED	DELETE
empty answer	0		valid	08/03/2010 05:57	08/03/2010 05:57	🗑
test answer	0		valid	08/03/2010 05:57	08/03/2010 05:57	🗑

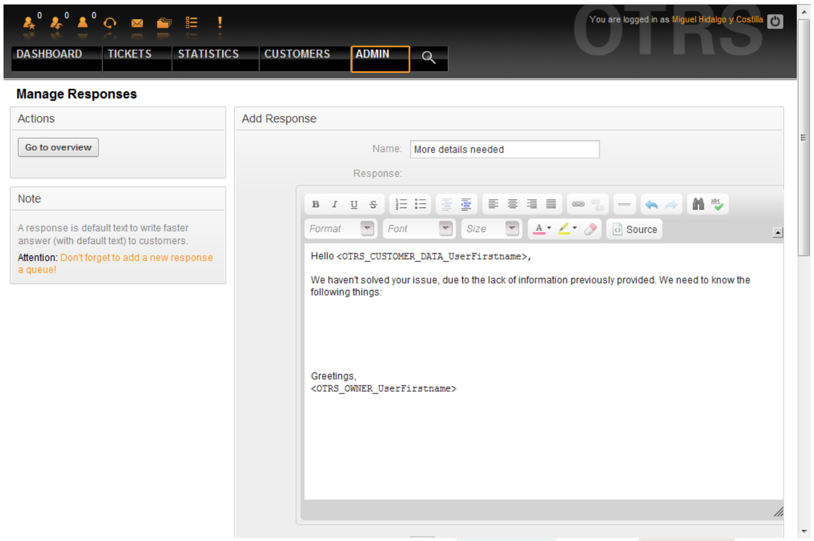
Note

A response is default text to write faster answer (with default text) to customers.

Attention: Don't forget to add a new response a queue!

* * *

To create a new response, click on the "Add response" button, provide the required data and submit it (see Figure below).



DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Actions
Go to overview
Format
Font
Size
Source

Manage Responses

Note

A response is default text to write faster answer (with default text) to customers.

Attention: Don't forget to add a new response a queue!

Add Response

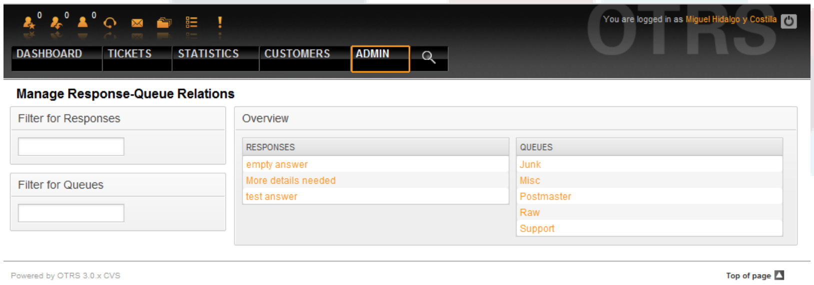
Name: More details needed

Response:

```
Hello <OTRS_CUSTOMER_DATA_UserFirstname>,

We haven't solved your issue, due to the lack of information previously provided. We need to know the following things:



Greetings,
<OTRS_OWNER_UserFirstname>
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂

To add/remove responses to one or more queues, click on the "Responses <-> Queues" link on the Admin page (see Figure below). You can also use filters to get information regarding a specific entity.



DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Filter for Responses
Filter for Queues
Powered by OTRS 3.0.x CVS
Top of page

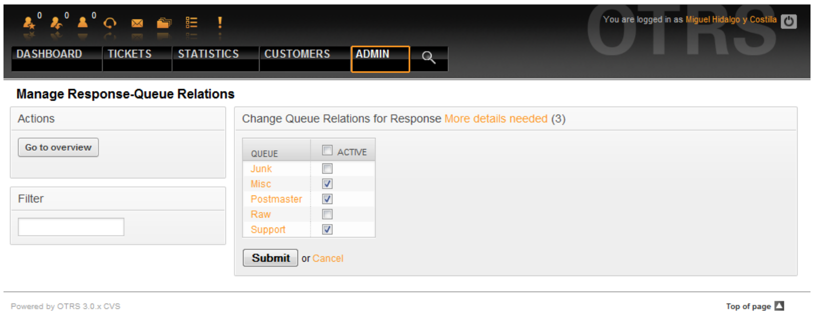
Manage Response-Queue Relations

Overview

RESPONSES	QUEUES
empty answer	Junk
More details needed	Misc
test answer	Postmaster
	Raw
	Support

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

To define the different responses that will be available for a queue and vice versa, click on the corresponding response or queue (see below the Figures 5.32 and 5.33, respectively).

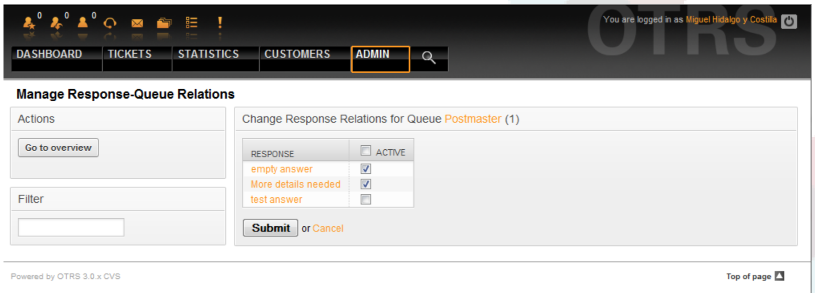

0
0
0
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Go to overview
Filter
Submit
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

Manage Response-Queue Relations

Change Queue Relations for Response More details needed (3)

QUEUE	☐ ACTIVE
Junk	☐
Misc	☑
Postmaster	☑
Raw	☐
Support	☑

* * *


0
0
0
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Go to overview
Filter
Submit
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

Manage Response-Queue Relations

Change Response Relations for Queue Postmaster (1)

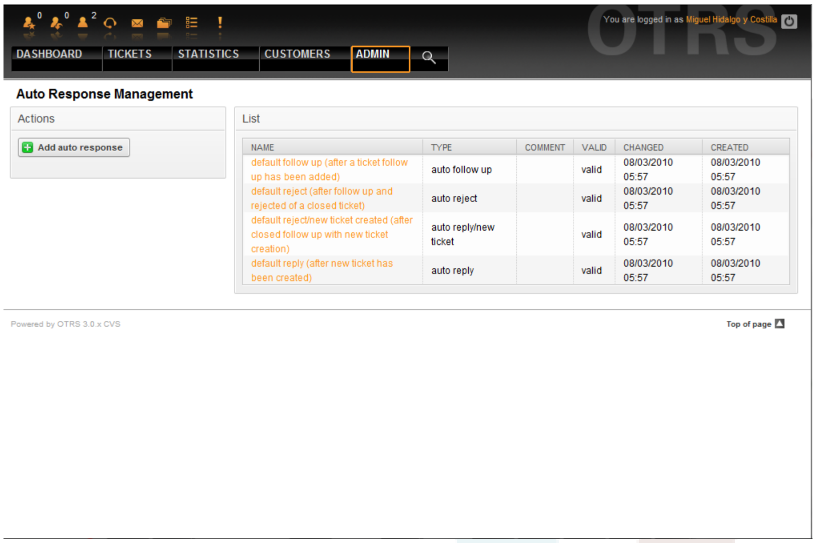
RESPONSE	☐ ACTIVE
empty answer	☑
More details needed	☑
test answer	☐

* * *

The structure of a response is self explanatory. It includes the salutation associated with the queue, then the text of the response, then the quoted ticket text, and finally the signature associated with the queue.

1.6. Auto responses

OTRS allows you to send automatic responses to customers based on the occurrence of certain events, such as the creation of a ticket in a specific queue, the receipt of a follow-up message in regards to a ticket, the closure or rejection of a ticket, etc. To manage such responses, click the link "Auto responses" on the Admin page (see Figure below).


0
0
2
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
OTRS
Actions
Add auto response
Powered by OTRS 3.0.x CVS
Top of page

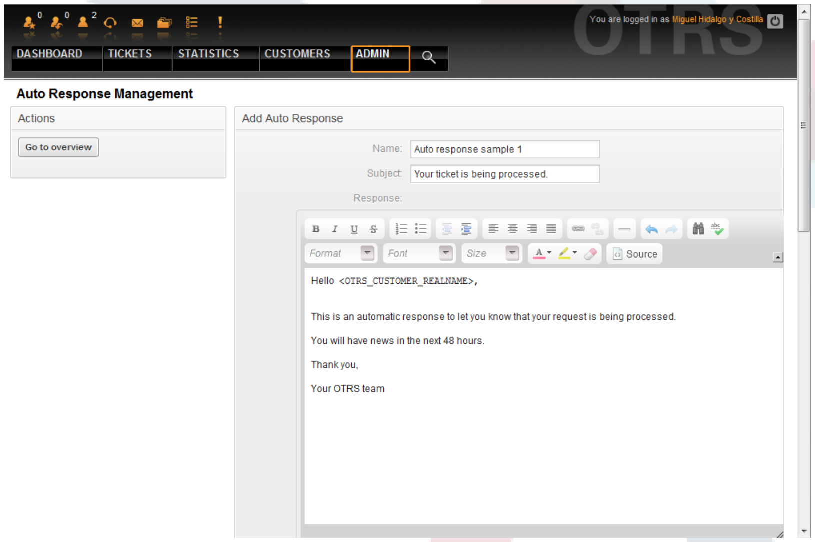
Auto Response Management

List

NAME	TYPE	COMMENT	VALID	CHANGED	CREATED
default follow up (after a ticket follow up has been added)	auto follow up		valid	08/03/2010 05:57	08/03/2010 05:57
default reject (after follow up and rejected of a closed ticket)	auto reject		valid	08/03/2010 05:57	08/03/2010 05:57
default reject/new ticket created (after closed follow up with new ticket creation)	auto reply/new ticket		valid	08/03/2010 05:57	08/03/2010 05:57
default reply (after new ticket has been created)	auto reply		valid	08/03/2010 05:57	08/03/2010 05:57

✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

To create an automatic response, click on the button "Add auto response", provide the needed data and submit it (see Figure below).


0
0
2
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
OTRS
Actions
Go to overview
B
I
U
S
Format
Font
Size
A
Source

Auto Response Management

Add Auto Response

Name: Auto response sample 1

Subject: Your ticket is being processed.

Response:

Hello <OTRS_CUSTOMER_REALNAME>,

This is an automatic response to let you know that your request is being processed.

You will have news in the next 48 hours.

Thank you,

Your OTRS team

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂✂

The subject and text of auto responses can be generated by variables, just as in signatures and salutations. If you insert, for example, the variable <OTRS_CUSTOMER_EMAIL[5]> into the body of the auto answer, the first 5 lines of the customer mail text will be inserted into the auto answer. You will find more details about the valid variables that can be used at the bottom of the screen shown in the Figure.

For every automatic answer, you can specify the event that should trigger it. The system events that are available after a default installation are described in the Table 5-3.

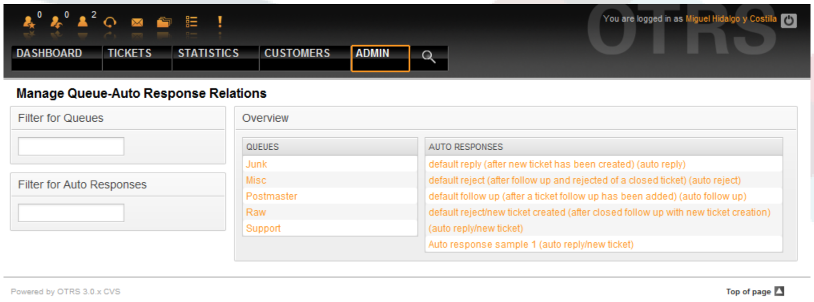
Table 4.3. Events for Auto answers

Name	Description
auto reply	Creation of a ticket in a certain queue.
auto reply/new ticket	Reopening of an already closed ticket, e.g. if a customer replies to such ticket.
auto follow up	Reception of a follow-up for a ticket.
auto reject	Automatic rejection of a ticket, done by the system.
auto remove	Deletion of a ticket, done by the system.

Note

As with other OTRS entities, auto responses too cannot be deleted, only deactivated, by setting the Valid option to ✁✁✁✁✁✁✁ or ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁.

To add an auto response to a queue, use the "Auto Response <-> Queues" link on the Admin page (see Figure below). All system events are listed for every queue, and an auto answer with the same event can be selected or removed via a listbox.



Manage Queue-Auto Response Relations

Filter for Queues

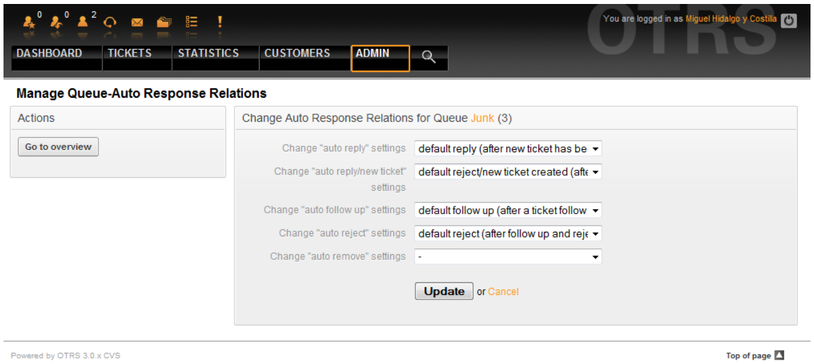
Filter for Auto Responses

Overview

QUEUES	AUTO RESPONSES
Junk	default reply (after new ticket has been created) (auto reply)
Misc	default reject (after follow up and rejected of a closed ticket) (auto reject)
Postmaster	default follow up (after a ticket follow up has been added) (auto follow up)
Raw	default reject/new ticket created (after closed follow up with new ticket creation) (auto reply/new ticket)
Support	Auto response sample 1 (auto reply/new ticket)

✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁

To define the different auto responses that will be available for a queue, click on the corresponding queue name (see Figure below). It is also possible to edit an existing auto response - to do so, click on the response and edit in the same manner as editing a new auto response.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla

Manage Queue-Auto Response Relations

Actions

Go to overview

Change Auto Response Relations for Queue Junk (3)

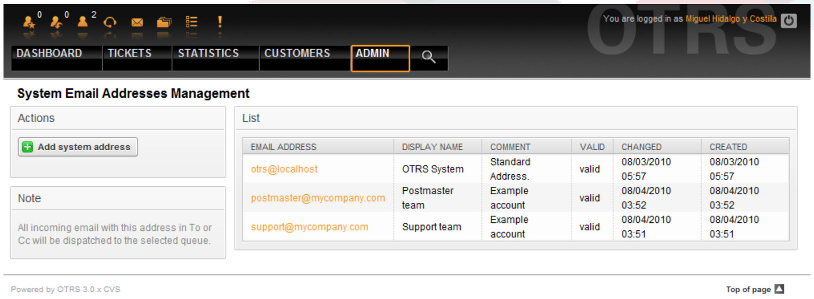
Change "auto reply" settings: default reply (after new ticket has be
Change "auto reply/new ticket" settings: default reject/new ticket created (afte
Change "auto follow up" settings: default follow up (after a ticket follow
Change "auto reject" settings: default reject (after follow up and reje
Change "auto remove" settings: -

Update
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂ ✂✂✂✂✂✂

1.7. Email addresses

To enable OTRS to send emails, you need a valid email address to be used by the system. OTRS is capable of working with multiple email addresses, since many support installations need to use more than one. A queue can be linked to many email addresses, and vice versa. The address used for outgoing messages from a queue can be set when the queue is created. Use the "Email Addresses" link from the Admin page to manage all email addresses of the system (see Figure below).


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla

System Email Addresses Management

Actions

Add system address

Note

All incoming email with this address in To or Cc will be dispatched to the selected queue.

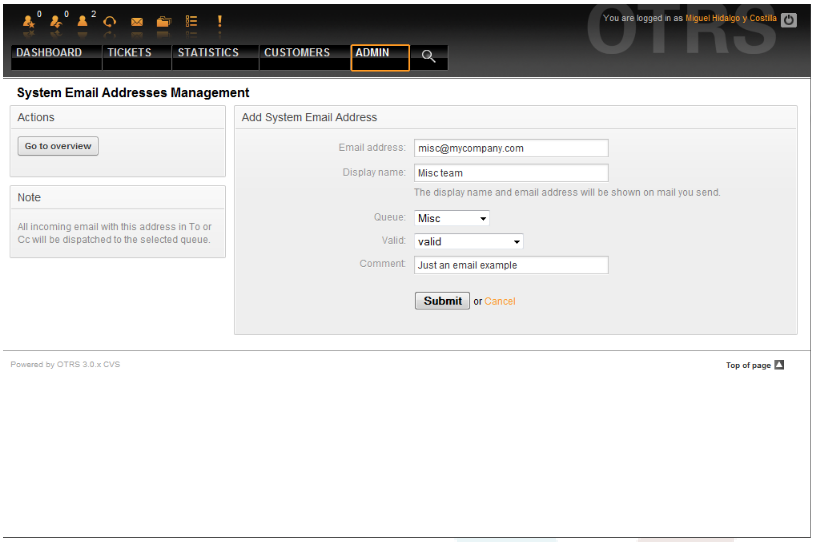
List

EMAIL ADDRESS	DISPLAY NAME	COMMENT	VALID	CHANGED	CREATED
otrs@localhost	OTRS System	Standard Address.	valid	08/03/2010 05:57	08/03/2010 05:57
postmaster@mycompany.com	Postmaster team	Example account	valid	08/04/2010 03:52	08/04/2010 03:52
support@mycompany.com	Support team	Example account	valid	08/04/2010 03:51	08/04/2010 03:51

Powered by OTRS 3.0.x CVS
Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

If you create a new mail address (see Figure below),you can select the queue or sub queue to be linked with it. This link enables the system to sort incoming messages via the address in the To: field of the mail into the right queue.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
OTRS

System Email Addresses Management

Actions

Go to overview

Note

All incoming email with this address in To or Cc will be dispatched to the selected queue.

Add System Email Address

Email address:	misc@mycompany.com
Display name:	Misc team
	The display name and email address will be shown on mail you send.
Queue:	Misc
Valid:	valid
Comment:	Just an email example

Submit
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂

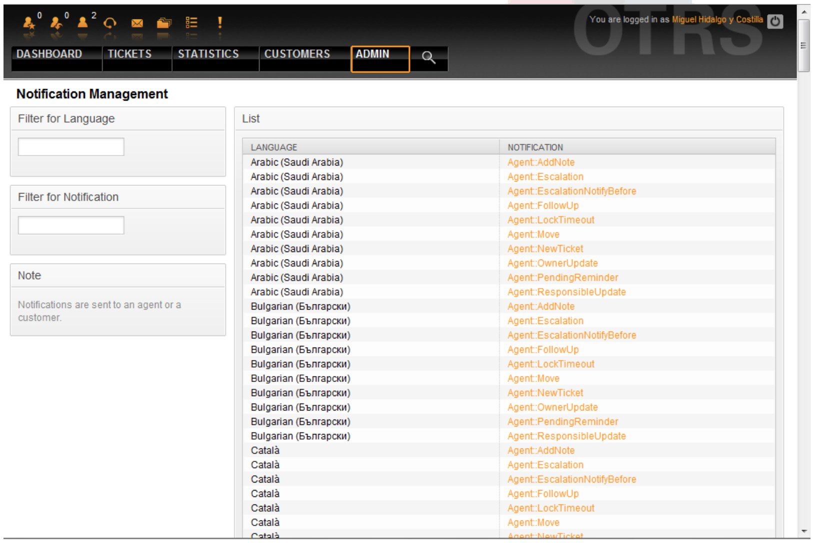
Note

As with other OTRS entities, email addresses cannot be deleted, only deactivated by setting the Valid option to ✂✂✂✂✂✂✂ or ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂.

1.8. Notifications

OTRS allows notifications to be sent to agents and customers, based on the occurrence of certain events. Agents can set the system events for their own notifications via the preferences link.

Through the "Agent Notifications" link on the Admin page, you can manage the notifications of your system (see Figure below). You can use filters to list only certain notifications.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
OTRS

Notification Management

Filter for Language

Filter for Notification

Note

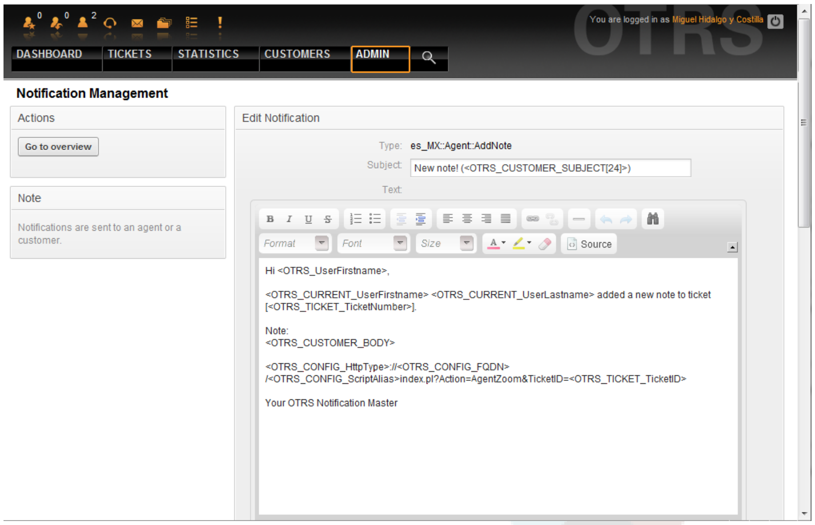
Notifications are sent to an agent or a customer.

List

LANGUAGE	NOTIFICATION
Arabic (Saudi Arabia)	Agent::AddNote
Arabic (Saudi Arabia)	Agent::Escalation
Arabic (Saudi Arabia)	Agent::EscalationNotifyBefore
Arabic (Saudi Arabia)	Agent::FollowUp
Arabic (Saudi Arabia)	Agent::LockTimeout
Arabic (Saudi Arabia)	Agent::Move
Arabic (Saudi Arabia)	Agent::NewTicket
Arabic (Saudi Arabia)	Agent::OwnerUpdate
Arabic (Saudi Arabia)	Agent::PendingReminder
Arabic (Saudi Arabia)	Agent::ResponsibleUpdate
Bulgarian (Български)	Agent::AddNote
Bulgarian (Български)	Agent::Escalation
Bulgarian (Български)	Agent::EscalationNotifyBefore
Bulgarian (Български)	Agent::FollowUp
Bulgarian (Български)	Agent::LockTimeout
Bulgarian (Български)	Agent::Move
Bulgarian (Български)	Agent::NewTicket
Bulgarian (Български)	Agent::OwnerUpdate
Bulgarian (Български)	Agent::PendingReminder
Bulgarian (Български)	Agent::ResponsibleUpdate
Català	Agent::AddNote
Català	Agent::Escalation
Català	Agent::EscalationNotifyBefore
Català	Agent::FollowUp
Català	Agent::LockTimeout
Català	Agent::Move
Català	Agent::NewTicket

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

You can customize the subject and the text of the notifications. Click on the notification you want to change from the list, and its content will get loaded for editing (see Figure). Please note that there is a notification with the same name for each of the available languages.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Notification Management
Actions
Go to overview
Note
Edit Notification
Format
Font
Size
Source

Notifications are sent to an agent or a customer.

Type: es_MX::Agent::AddNote

Subject: New note! (<OTRS_CUSTOMER_SUBJECT[24]>)

Text:

Hi <OTRS_UserFirstname>,

<OTRS_CURRENT_UserFirstname> <OTRS_CURRENT_UserLastname> added a new note to ticket [<OTRS_TICKET_TicketNumber>].

Note:
<OTRS_CUSTOMER_BODY>

<OTRS_CONFIG_HttpType>://<OTRS_CONFIG_FQDN>/<OTRS_CONFIG_ScriptAlias>index.pl?Action=AgentZoom&TicketID=<OTRS_TICKET_TicketID>

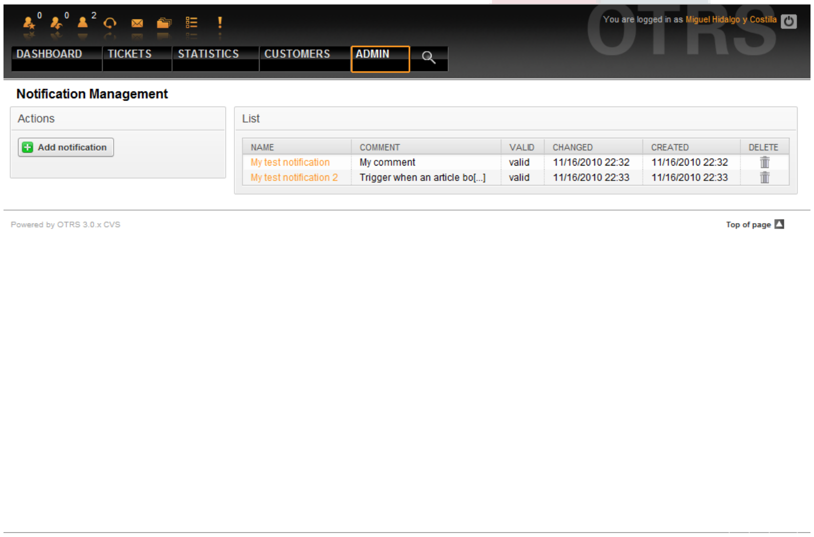
Your OTRS Notification Master

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂

Just as with signatures and salutations, it is possible to dynamically create the content of a notification, by using special variables. You can find a list of variables at the bottom of the screen shown in the Figure.

It is also possible to create notifications based on events. You can specify in detail when and to whom you want such a notification to be sent. You can choose from a wide variety of parameters, such as: recipient group(s), agent(s), role(s), email address(es), type of event triggering the notification, ticket-type, state, priority, queue, lock, service, SLA, etc.

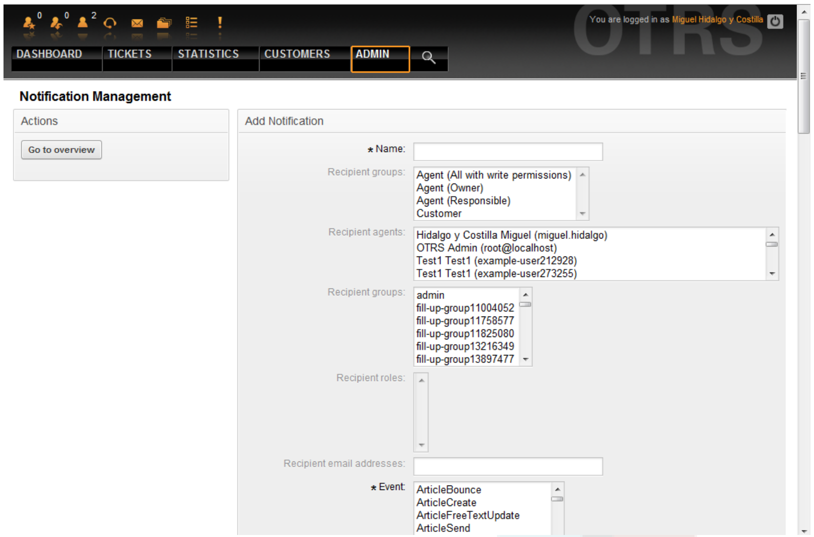
In order to see a list of all event based notifications, click on the link "Notifications (Event)" on the Admin page (see Figure).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Notification Management
Actions
Add notification
List
Powered by OTRS 3.0.x CVS
Top of page

NAME	COMMENT	VALID	CHANGED	CREATED	DELETE
My test notification	My comment	valid	11/16/2010 22:32	11/16/2010 22:32	
My test notification 2	Trigger when an article bo[...]	valid	11/16/2010 22:33	11/16/2010 22:33	

✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

As shown in Figure, you can create a new notification by clicking on the Add button (see Figure).



DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Go to overview

Notification Management

Actions

Add Notification

* Name:

Recipient groups: Agent (All with write permissions), Agent (Owner), Agent (Responsible), Customer

Recipient agents: Hidalgo y Costilla Miguel (miguel.hidalgo), OTRS Admin (root@localhost), Test1 Test1 (example-user212928), Test1 Test1 (example-user273255)

Recipient groups: admin, fill-up-group11004052, fill-up-group11758577, fill-up-group11825080, fill-up-group13216349, fill-up-group13897477

Recipient roles:

Recipient email addresses:

* Event: ArticleBounce, ArticleCreate, ArticleFreeTextUpdate, ArticleSend

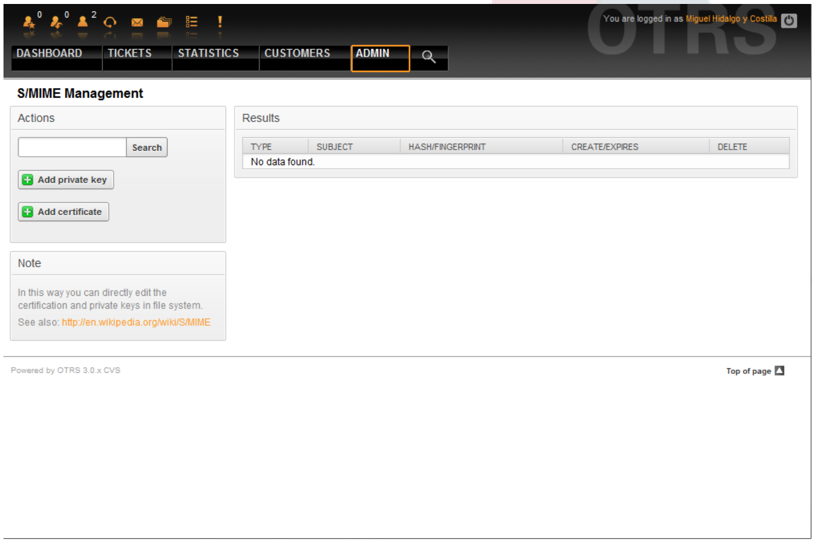
✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

Please note that the content of the event based notifications can also be dynamically created by using the special variables listed at the bottom of the screen shown in the Figure.

1.9. SMIME

OTRS can process incoming S/MIME encoded messages and sign outgoing mails. Before this feature can be used, you need to activate it and change some configuration parameters in the SysConfig.

The "S/MIME Certificates" link on the Admin page allows you to manage your S/MIME certificates (see Figure below). You can add or remove certificates, and also search through the SMIME data.



DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Search
Add private key
Add certificate
Powered by OTRS 3.0.x CVS
Top of page

S/MIME Management

Actions

Results

TYPE	SUBJECT	HASH/FINGERPRINT	CREATE/EXPIRES	DELETE
No data found.				

Note

In this way you can directly edit the certification and private keys in file system.

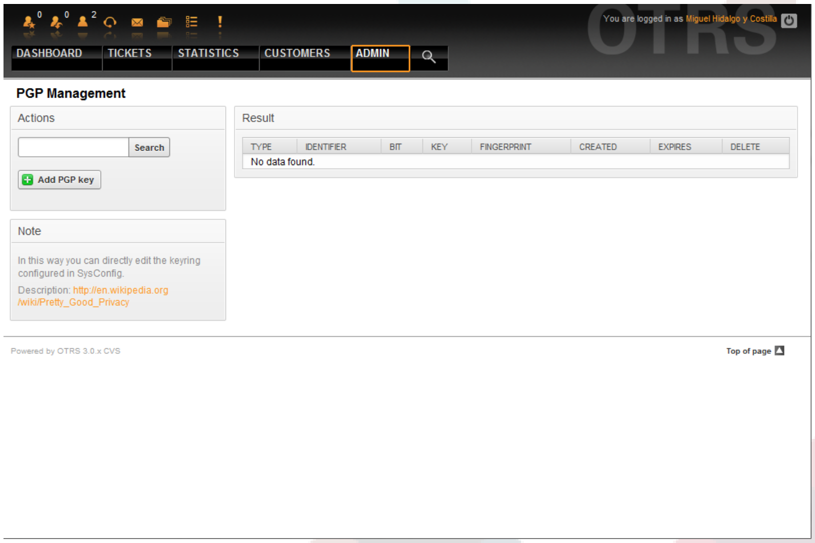
See also: http://en.wikipedia.org/wiki/S/MIME

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

1.10. PGP

OTRS handles PGP keys, which allows you to encrypt/decrypt messages and to sign outgoing messages. Before this feature can be used, you need to activate it and change some configuration parameters in the SysConfig.

Through the "PGP Keys" link on the Admin page, it is possible to manage the key ring of the user who shall be used for PGP with OTRS (see Figure below), e.g. the local OTRS user or the web server user. It is possible to add and remove keys and signatures, and you can search through all data in your key ring.


0
0
2
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
PGP Management
Actions
Search
Add PGP key
Result
Note
Powered by OTRS 3.0.x CVS
Top of page

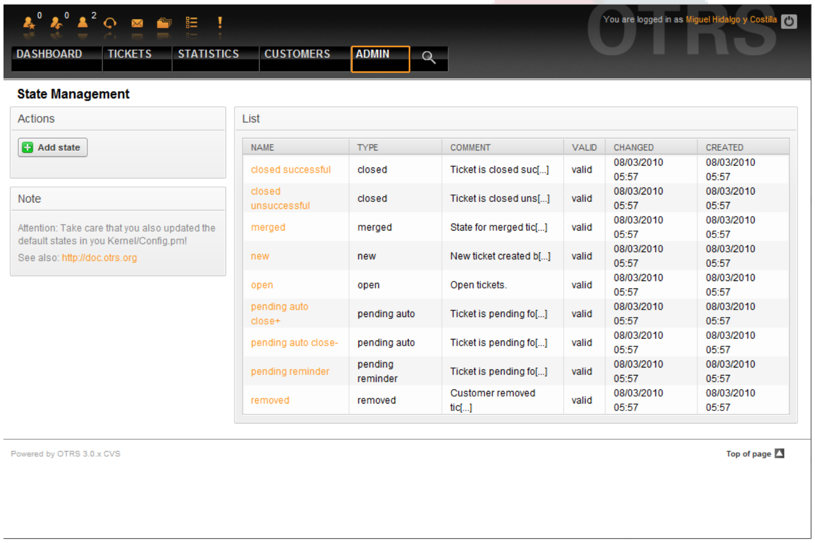
TYPE	IDENTIFIER	BIT	KEY	FINGERPRINT	CREATED	EXPIRES	DELETE
No data found.							

In this way you can directly edit the keyring configured in SysConfig.

Description: http://en.wikipedia.org/wiki/Pretty_Good_Privacy

1.11. States

Through the "States" link on the Admin page, you can manage the different ticket states you want to use in the system (see Figure below).


0
0
2
You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
State Management
Actions
Add state
List
Note
Powered by OTRS 3.0.x CVS
Top of page

NAME	TYPE	COMMENT	VALID	CHANGED	CREATED
closed successful	closed	Ticket is closed suc[...]	valid	08/03/2010 05:57	08/03/2010 05:57
closed unsuccessful	closed	Ticket is closed uns[...]	valid	08/03/2010 05:57	08/03/2010 05:57
merged	merged	State for merged tic[...]	valid	08/03/2010 05:57	08/03/2010 05:57
new	new	New ticket created b[...]	valid	08/03/2010 05:57	08/03/2010 05:57
open	open	Open tickets.	valid	08/03/2010 05:57	08/03/2010 05:57
pending auto close+	pending auto	Ticket is pending fo[...]	valid	08/03/2010 05:57	08/03/2010 05:57
pending auto close-	pending auto	Ticket is pending fo[...]	valid	08/03/2010 05:57	08/03/2010 05:57
pending reminder	pending reminder	Ticket is pending fo[...]	valid	08/03/2010 05:57	08/03/2010 05:57
removed	removed	Customer removed tic[...]	valid	08/03/2010 05:57	08/03/2010 05:57

Attention: Take care that you also updated the default states in you Kernel/Config.pm!

See also: http://doc.otrs.org

* * *

After a default setup, there are some states defined:

- closed successful
- closed unsuccessful
- merged
- new
- open
- pending auto close+
- pending auto close-
- pending reminder
- removed

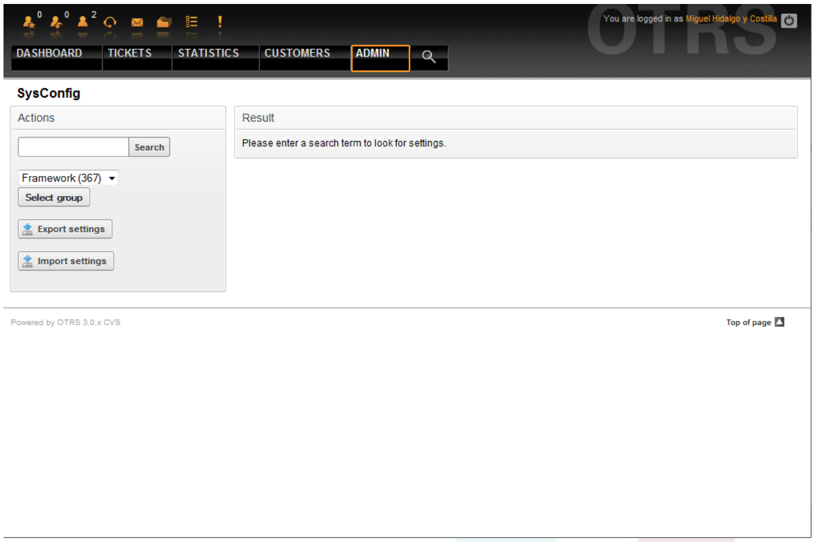
Every state is linked to a type, which needs to be specified if a new state is created. By default the state types are:

- closed
- merged
- new
- open
- pending auto
- pending reminder
- removed

1.12. SysConfig

The SysConfig link leads to the section where many OTRS configuration options are maintained.

The SysConfig link on the Admin page loads the graphical interface for system configuration (see Figure below). You can upload your own configuration files for the system, as well as backup all your current settings into a file. Almost all configuration parameters of the OTRS framework and installed applications can be viewed and changed through this interface. Since all configuration parameters are sorted into groups and sub groups, it is possible to navigate quickly through the vast number of existing parameters. It is also possible to perform a full-text search through all of the configuration parameters.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
OTRS
SysConfig
Actions
Search
Framework (367)
Select group
Export settings
Import settings
Result
Powered by OTRS 3.0.x CVS
Top of page

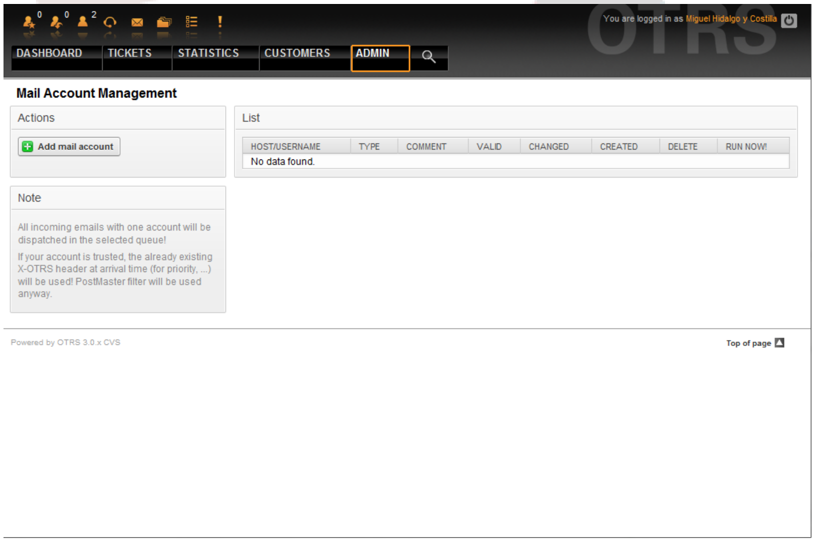
Please enter a search term to look for settings.

✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂
✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂

The graphical interface for system configuration is described in more detail in the chapter "Configuring the system through the web interface".

1.13. Using mail accounts

There are several possibilities to transport new emails into the ticket system. One way is to use a local MTA and the otrs.PostMaster.pl script that pipes the mails directly into the system. Another possibility is the use of mail accounts which can be administrated through the web interface. The "PostMaster Mail Accounts" link on the Admin page loads the management console for the mail accounts (see Figure below). OTRS supports the mail protocols: POP3, POP3S, IMAP and IMAPS.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
OTRS
Mail Account Management
Actions
Add mail account
List
Note
Powered by OTRS 3.0.x CVS
Top of page

HOST/USERNAME	TYPE	COMMENT	VALID	CHANGED	CREATED	DELETE	RUN NOW!
No data found.							

All incoming emails with one account will be dispatched in the selected queue!

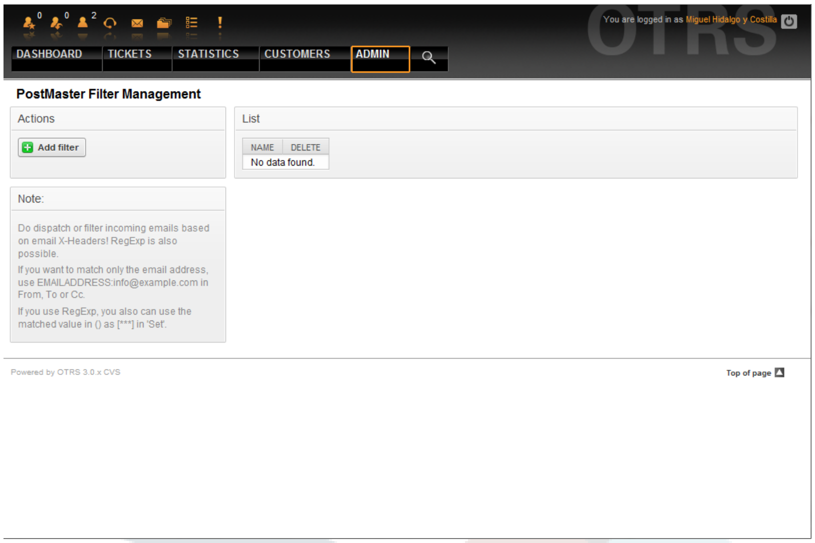
If your account is trusted, the already existing X-OTRS header at arrival time (for priority, ...) will be used! PostMaster filter will be used anyway.

✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

See the section about PostMaster Mail Accounts for more details.

1.14. Filtering incoming email messages

OTRS has the capability to filter incoming email messages. For example, it is possible to put certain emails automatically into specified queues, or to set a specific state or ticket type for some mails. The filters apply to all incoming mails. You can manage your filters via the link "PostMaster Filter" on the Admin page (see Figure below).



PostMaster Filter Management

No data found.

Note:

Do dispatch or filter incoming emails based on email X-Headers! RegExp is also possible.

If you want to match only the email address, use EMAILADDRESS:info@example.com in From, To or Cc.

If you use RegExp, you also can use the matched value in () as [***] in 'Set'.

Powered by OTRS 3.0.x CVS

* * *

A filter consists of one or more criteria that must be met in order for the defined actions to be executed on the email. Filter criteria may be defined for the headers or the body of an email, e.g. search for specific header entries, such as a sender address, or on strings in the body. Even regular expressions can be used for extended pattern matching. If your filter matches, you can set fields using the X-OTRS headers in the GUI. These values will be applied when creating the ticket or follow-up message in OTRS. The Table 5-4 lists the different X-OTRS headers and their meaning.

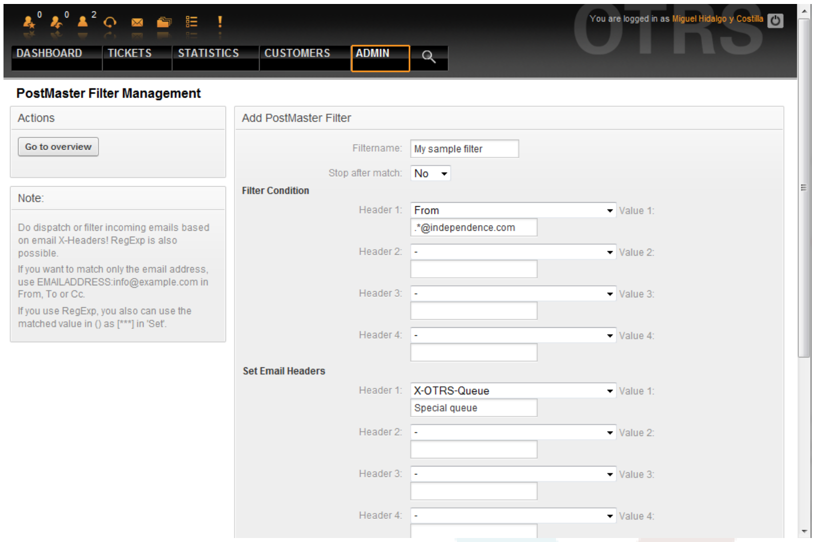
Note: You also can use X-OTRS-FollowUp-* headers to set values for follow up emails.

Table 4.4. Function of the different X-OTRS-headers

Name	Possible values	Description
X-OTRS-Priority:	1 very low, 2 low, 3 normal, 4 high, 5 very high	Sets the priority of a ticket.
X-OTRS-Queue:	Name of a queue in the system.	Sets the queue where the ticket shall be sorted. If set in X-OTRS header, all other filter rules that try to sort a ticket into a specific queue are ignored. If you use a sub-queue, specify it as "Parent::Sub".
X-OTRS-Lock:	lock, unlock	Sets the lock state of a ticket.
X-OTRS-Ignore:	Yes or True	If this X-OTRS header is set to "Yes", the incoming message

Name	Possible values	Description
		will completely be ignored and never delivered to the system.
X-OTRS-State:	new, open, closed successful, closed unsuccessful, ...	Sets the next state of the ticket.
X-OTRS-State-PendingTime:	e. g. 2010-11-20 00:00:00	Sets the pending time of a ticket (you also should sent a pending state via X-OTRS-State).
X-OTRS-Type:	default (depends on your setup)	Sets the type of a ticket (if Ticket::Type is activated).
X-OTRS-Service:	(depends on your setup)	Sets the service of a ticket (if Ticket::Service is active). If you want to set a sub-service you should specify it as "Parent::Sub".
X-OTRS-SLA:	(depends on your setup)	Sets the SLA of a ticket (if Ticket::Service support is active).
X-OTRS-CustomerUser:	CustomerUser	Sets the customer user for the ticket.
X-OTRS-CustomerNo:	CustomerNo	Sets the customer ID for this ticket.
X-OTRS-SenderType:	agent, system, customer	Sets the type of the ticket sender.
X-OTRS-ArticleType:	email-external, email-internal, email-notification-ext, email-notification-int, phone, fax, sms, webrequest, note-internal, note-external, note-report	Sets the article type for the incoming ticket.
X-OTRS-DynamicField-<DynamicFieldName>:	Depends on Dynamic Field configuration (Text: Notebook, Date: 2010-11-20 00:00:00, Integer: 1)	Saves an additional info value for the ticket on <DynamicFieldName> Dynamic Field.
X-OTRS-Loop:	True	If this X-OTRS header is set, no auto answer is delivered to the sender of the message (mail loop protection).

You should specify a name for every filter rule. Filter criteria can be specified in the section "Filter Condition". Choose via the listboxes for "Header 1", "Header 2" and so on for the parts of the messages where you would like to search, and specify on the right side the values you wish to filter on. In the section "Set Email Headers", you can choose the actions that are triggered if the filter rules match. You can select for "Header 1", "Header 2" and so on to select the X-OTRS-Header and set the associated values (see Figure below).



PostMaster Filter Management

Add PostMaster Filter

Filtername: My sample filter

Stop after match: No

Filter Condition

Header 1: From Value 1: .*@independence.com

Header 2: - Value 2:

Header 3: - Value 3:

Header 4: - Value 4:

Set Email Headers

Header 1: X-OTRS-Queue Value 1: Special queue

Header 2: - Value 2:

Header 3: - Value 3:

Header 4: - Value 4:

Note:

Do dispatch or filter incoming emails based on email X-Headers! RegExp is also possible.

If you want to match only the email address, use EMAILADDRESS:info@example.com in From, To or Cc.

If you use RegExp, you also can use the matched value in () as [***] in 'Set'.

✂✂✂✂✂✂✂ ✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂

Example 4.1. Sort spam mails into a specific queue

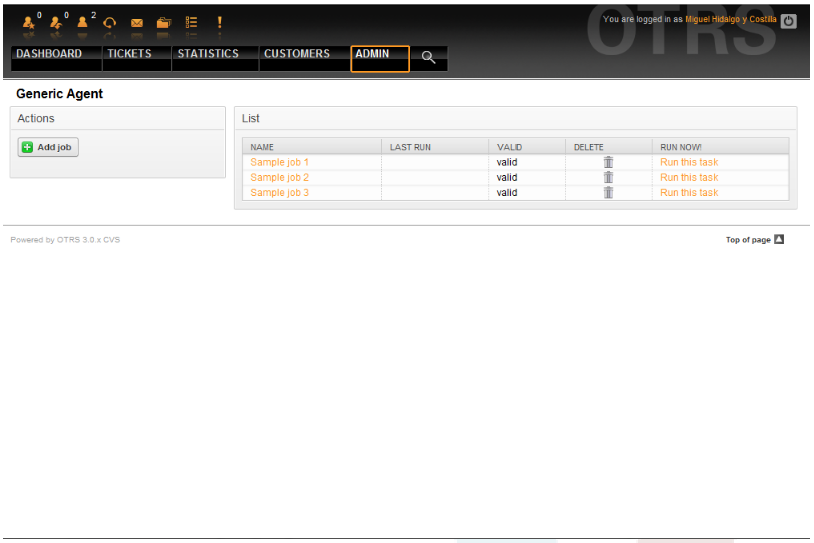
A useful filter rule would be to let OTRS automatically move mails marked for spam ,by using a spam detection tool such as SpamAssassin, into the "Junk" queue. SpamAssassin adds the "X-Spam-Flag" header to every checked mail. When the mail is marked as spam, the Header is set to "Yes". So the filter criteria would be "X-Spam-Flag: Yes". To create a filter rule with this criteria you can insert the name as, for example, "spam-mails". In the section for "Filter Condition", choose "X-Spam-Flag:" for "Header 1" from the listbox. Insert "Yes" as value for this header. Now the filter criteria is specified. To make sure that all spam mails are placed into the "Junk" queue, choose in the section for "Set Email Headers", the "X-OTRS-Queue:" entry for "Header 1". Specify "Junk" as value for this header. Finally add the new filter rule to activate it for new messages in the system.

There are additional modules, that can be used to filter incoming messages more specifically. These modules might be useful when dealing with larger, more complex systems.

1.15. Executing automated jobs with the GenericAgent

The GenericAgent is a tool to execute tasks automatically. In its absence such tasks would need to be done manually by an agent. The GenericAgent, for example, can close or move tickets, send notifications on escalated tickets, etc.

Click the link "GenericAgent" on the Admin page (see Figure below). A table with all automated jobs, that are currently configure to run in the system is displayed. These jobs can then be edited in order to run them manually or can be removed entirely.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
OTRS
Actions
Add job
List
Top of page

You are logged in as Miguel Hidalgo y Costilla

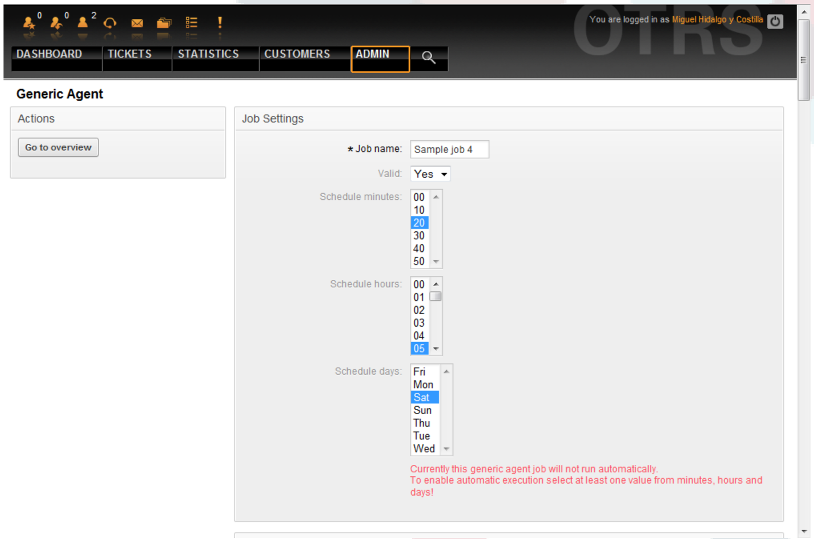
Generic Agent

NAME	LAST RUN	VALID	DELETE	RUN NOW!
Sample job 1		valid		Run this task
Sample job 2		valid		Run this task
Sample job 3		valid		Run this task

Powered by OTRS 3.0.x CVS

* * *

Click the "Add job" button to create a new job. You first need to supply a name for the job in addition to the times when the job should be executed. Different criteria to target the tickets to work on and regarding what changes to apply to those tickets can also be set (see Figure below).


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
OTRS
Actions
Go to overview
Job Settings

You are logged in as Miguel Hidalgo y Costilla

Generic Agent

* Job name: Sample job 4

Valid: Yes

Schedule minutes: 00, 10, 20, 30, 40, 50 (20 selected)

Schedule hours: 00, 01, 02, 03, 04, 05 (05 selected)

Schedule days: Fri, Mon, Sat, Sun, Thu, Tue, Wed (Sat selected)

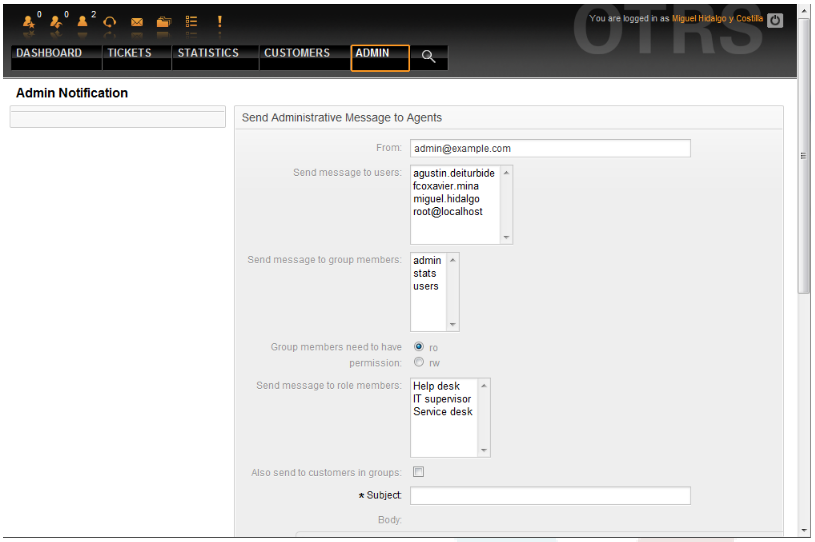
Currently this generic agent job will not run automatically.
To enable automatic execution select at least one value from minutes, hours and days!

* * *

On completion of the job creation, all affected tickets by the job are listed. This list helps you to verify that the job is working as intended. At this point no changes have been made to these tickets yet. The job will only be activated once it is saved into the job list.

1.16. Admin email

OTRS administrators can send messages to specific users or groups. The "Admin Notification" link on the Admin page opens the screen where the agents and groups that should be notified can be selected (see Figure below).



You are logged in as Miguel Hidalgo y Costilla

Admin Notification

Send Administrative Message to Agents

From: admin@example.com

Send message to users:
agustin.deiturbide
fcoxavier.mina
miguel.hidalgo
root@localhost

Send message to group members:
admin
stats
users

Group members need to have permission: ● ro ○ rw

Send message to role members:
Help desk
IT supervisor
Service desk

Also send to customers in groups: ☐

* Subject:

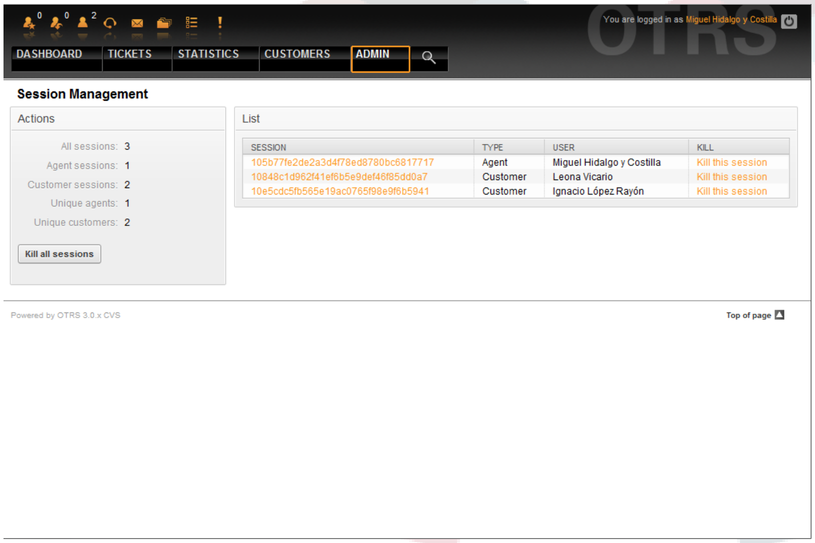
Body:

✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂

It is possible to specify the sender, subject and body text of the notification. You can also select the agents, groups and roles who should receive the message.

1.17. Session management

You can see all logged in users and their session details by clicking the "Session Management" link in the admin area (see Figure below).



You are logged in as Miguel Hidalgo y Costilla

Session Management

Actions

All sessions: 3
Agent sessions: 1
Customer sessions: 2
Unique agents: 1
Unique customers: 2

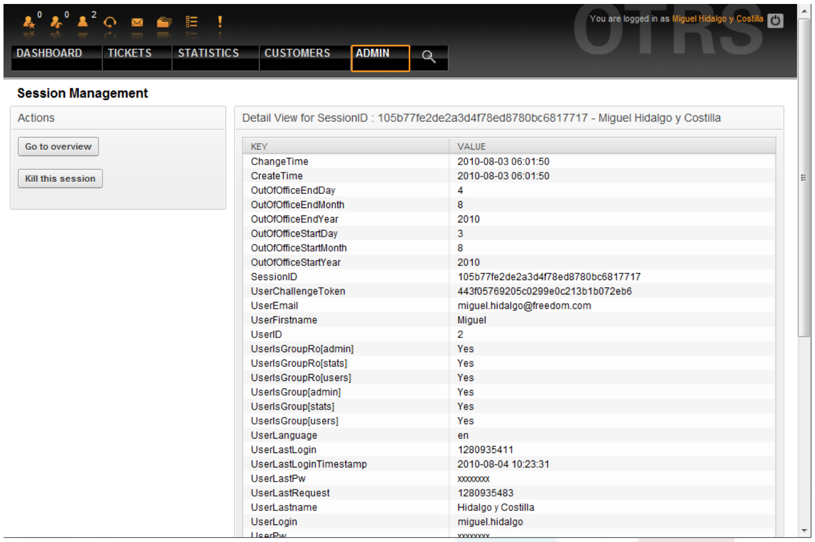
List

SESSION	TYPE	USER	KILL
105b77fe2de2a3d4f78ed8780bc6817717	Agent	Miguel Hidalgo y Costilla	Kill this session
10848c1d962f41ef6b5e9def46f85dd0a7	Customer	Leona Vicario	Kill this session
10e5cdc5fb565e19ac0765f98e9f6b5941	Customer	Ignacio López Rayón	Kill this session

Powered by OTRS 3.0.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂

Some statistics about all active sessions are displayed, e.g. how many agents and customer users are logged in and the number of active sessions. Any individual session can be removed by clicking on the ✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂ link on the right-hand side of the list. You also have the option to ✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂, which can be useful if you want to take the system offline. Detailed information for every session is available, too (see Figure below).

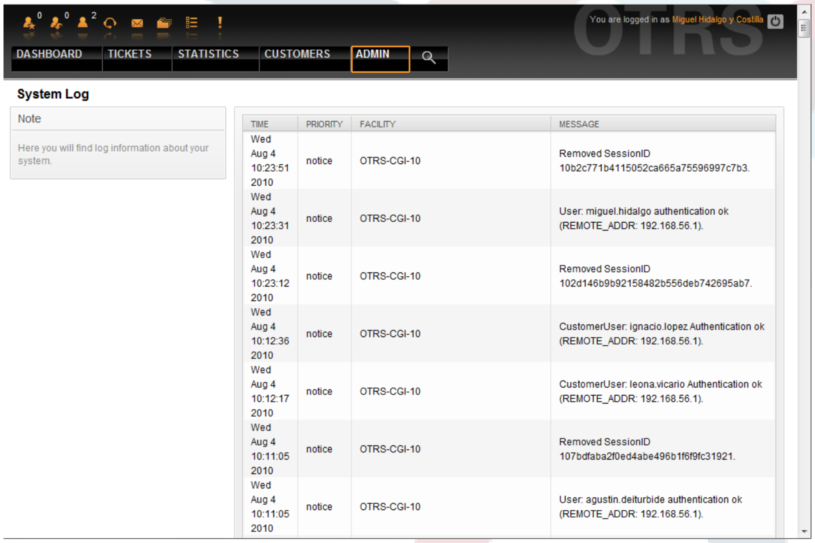

DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Session Management
Actions
Go to overview
Kill this session

Detail View for SessionID : 105b77fe2de2a3d4f78ed8780bc6817717 - Miguel Hidalgo y Costilla

KEY	VALUE
ChangeTime	2010-08-03 06:01:50
CreateTime	2010-08-03 06:01:50
OutOfOfficeEndDay	4
OutOfOfficeEndMonth	8
OutOfOfficeEndYear	2010
OutOfOfficeStartDay	3
OutOfOfficeStartMonth	8
OutOfOfficeStartYear	2010
SessionID	105b77fe2de2a3d4f78ed8780bc6817717
UserChallengeToken	443f05769205c0299e0c213b1b072eb6
UserEmail	miguel.hidalgo@freedom.com
UserFirstname	Miguel
UserID	2
UserIsGroupRo[admin]	Yes
UserIsGroupRo[stats]	Yes
UserIsGroupRo[users]	Yes
UserIsGroup[admin]	Yes
UserIsGroup[stats]	Yes
UserIsGroup[users]	Yes
UserLanguage	en
UserLastLogin	1280935411
UserLastLoginTimestamp	2010-08-04 10:23:31
UserLastPw	xxxxxxxx
UserLastRequest	1280935483
UserLastname	Hidalgo y Costilla
UserLogin	miguel.hidalgo
UserPw	xxxxxxxx

1.18. System Log

The "System Log" link on the Admin page shows the log entries of the system, reverse chronologically sorted with most recent first (see Figure below).


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
System Log
Note

Here you will find log information about your system.

TIME	PRIORITY	FACILITY	MESSAGE
Wed Aug 4 10:23:51 2010	notice	OTRS-CGI-10	Removed SessionID 10b2c771b4115052ca665a75596997c7b3.
Wed Aug 4 10:23:31 2010	notice	OTRS-CGI-10	User: miguel.hidalgo authentication ok (REMOTE_ADDR: 192.168.56.1).
Wed Aug 4 10:23:12 2010	notice	OTRS-CGI-10	Removed SessionID 102d146b9b92158482b556deb742695ab7.
Wed Aug 4 10:12:36 2010	notice	OTRS-CGI-10	CustomerUser: ignacio.lopez Authentication ok (REMOTE_ADDR: 192.168.56.1).
Wed Aug 4 10:12:17 2010	notice	OTRS-CGI-10	CustomerUser: leona.vicario Authentication ok (REMOTE_ADDR: 192.168.56.1).
Wed Aug 4 10:11:05 2010	notice	OTRS-CGI-10	Removed SessionID 107bdfaba2f0ed4abe496b1f6f9fc31921.
Wed Aug 4 10:11:05 2010	notice	OTRS-CGI-10	User: agustin.deiturbide authentication ok (REMOTE_ADDR: 192.168.56.1).

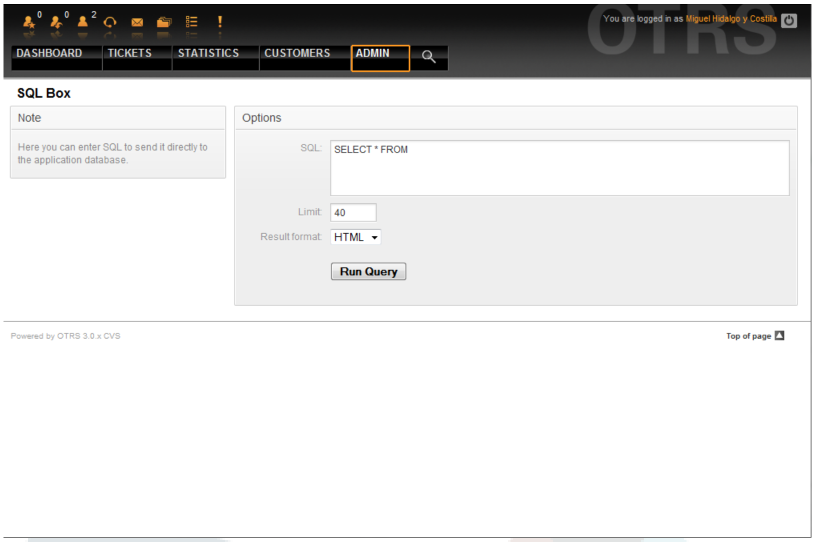
Each line in the log contains a time stamp, the log priority, the system component and the log entry itself.

Note

System logs are available via the web interface only on Linux / Unix systems. On Windows systems, you can see the logs using a text editor by opening the file `[install_dir]otrs\var\log\otrs.log`.

1.19. SQL queries via the SQL box

The "SQL Box" link on the Admin page opens a screen that lets you query the content of the tables in the OTRS database (see Figure below). It is not possible to change the content of the tables, only 'select' queries are allowed.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
SQL Box
Note
Options
Run Query
Powered by OTRS 3.0.x CVS
Top of page

Here you can enter SQL to send it directly to the application database.

SQL: SELECT * FROM

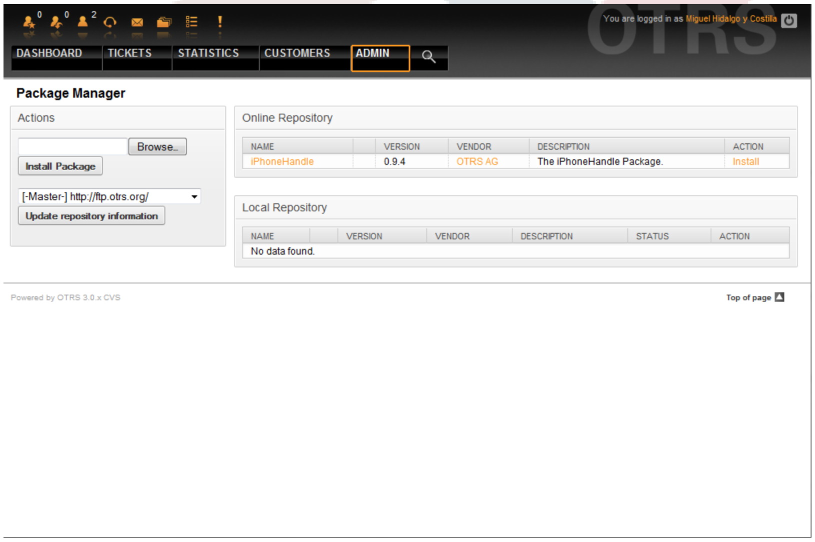
Limit: 40

Result format: HTML

✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂

1.20. Package Manager

Using the "Package Manager" link on the Admin page, you can install and manage packages that extend the functionality of OTRS (see Figure below). See the Additional applications section for a discussion on the extensions that are available from the OTRS repositories.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Package Manager
Actions
Browse...
Install Package
[-Master-] http://ftp.otrs.org/
Update repository information
Powered by OTRS 3.0.x CVS
Top of page

Online Repository

NAME	VERSION	VENDOR	DESCRIPTION	ACTION
iPhoneHandle	0.9.4	OTRS AG	The iPhoneHandle Package.	Install

Local Repository

NAME	VERSION	VENDOR	DESCRIPTION	STATUS	ACTION
No data found.					

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂

The Package Manager shows the OTRS addon packages you currently have installed on your server, together with their version numbers.

You can install packages from a remote host by selecting the repository in the ✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ section, and clicking the ✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂

[illegible] button. The available packages are displayed in the corresponding table. The right side of the screen shows the available packages. To install a package, click on [illegible]. After installation, the package is displayed in the [illegible] [illegible] section.

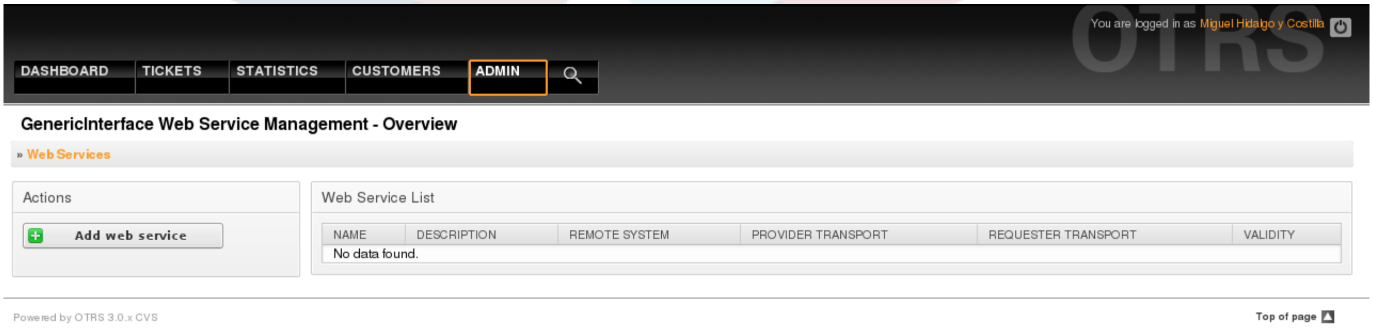
To upgrade an installed package, the list of available packages in the online repository will show [illegible] in the Action column for any package that has a higher version than the one that is installed locally. Just click Upgrade and it will install the new package version on your system.

In some cases, such as when your OTRS system is not connected to the Internet, you can also install those packages that you have downloaded to a local disk. Click the [illegible] button on the Actions side bar, and select the .opm file of the package on your disk. Click [illegible] and then [illegible] [illegible]. After the installation has been completed, the package is displayed in the [illegible] [illegible] section. You can use the same steps for updating a package that is already installed.

In special cases, you might want to configure the Package Manager, e.g., to use a proxy or to use a local repository. Just take a look at the available options in SysConfig under Framework:Core::Package.

1.21. Web Services

The Web Services link leads to the graphical interface where web services (for the OTRS Generic Interface) are created and maintained (see Figure below).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Add web service
Powered by OTRS 3.0.x CVS
Top of page

GenericInterface Web Service Management - Overview

» Web Services

Actions

Web Service List

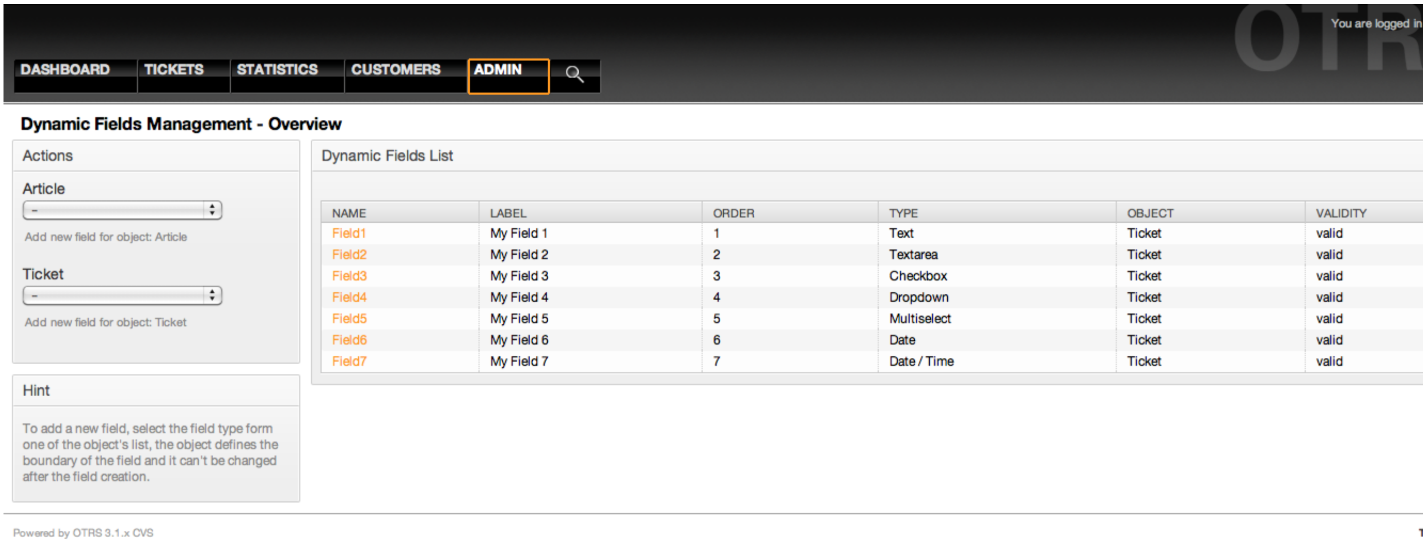
NAME	DESCRIPTION	REMOTE SYSTEM	PROVIDER TRANSPORT	REQUESTER TRANSPORT	VALIDITY
No data found.					

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

The graphical interface for web services configuration is described in more detail in the section "Web Service Graphical Interface".

1.22. Dynamic Fields

Dynamic Fields is the place where you setup and manage custom fields for tickets and articles (see figure below).



Dynamic Fields Management - Overview

Actions

Article

Add new field for object: Article

Ticket

Add new field for object: Ticket

Hint

To add a new field, select the field type form one of the object's list, the object defines the boundary of the field and it can't be changed after the field creation.

Dynamic Fields List

NAME	LABEL	ORDER	TYPE	OBJECT	VALIDITY
Field1	My Field 1	1	Text	Ticket	valid
Field2	My Field 2	2	Textarea	Ticket	valid
Field3	My Field 3	3	Checkbox	Ticket	valid
Field4	My Field 4	4	Dropdown	Ticket	valid
Field5	My Field 5	5	Multiselect	Ticket	valid
Field6	My Field 6	6	Date	Ticket	valid
Field7	My Field 7	7	Date / Time	Ticket	valid

Powered by OTRS 3.1.x CVS

✁✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁

The dynamic fields configuration is described in more detail in the section "Dynamic Fields Configuration".

Each dynamic field type has its own configuration settings and therefore its own configuration screen.

Note

In the OTRS framework, dynamic fields can only be linked to tickets and articles by default, but they can be extended to other objects as well.

2. System Configuration

2.1. OTRS config files

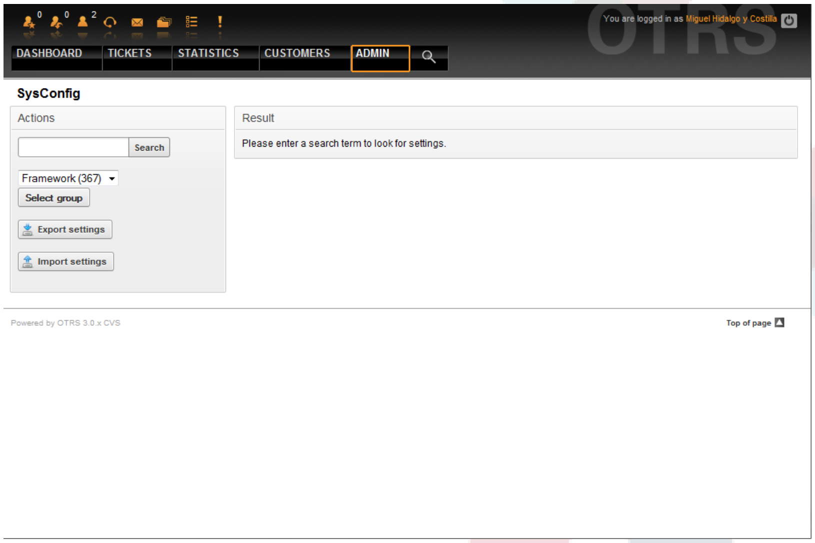
All OTRS configuration files are stored in the directory `Kernel` and in its subdirectories. There is no need to manually change any other file than `Kernel/Config.pm`, because the rest of the files will be changed when the system gets upgraded. Just copy the configuration parameters from the other files into `Kernel/Config.pm` and change them as per your needs. This file will never be touched during the upgrade process, so your manual settings are safe.

In the directory `Kernel/Config/Files` there are some other files that are parsed when the OTRS login page is accessed. If additional applications like the FAQ or the File Manager are installed, the configuration files for those can also be found in the mentioned path.

If the OTRS web interface is accessed, all .xml files in the `Kernel/Config/Files` directory are parsed in alphabetical order, and the settings for the central framework and additional applications will be loaded. Afterwards, the settings in the files `Kernel/Config/Files/ZZZAAuto.pm`, `Kernel/Config/Files/ZZZAuto.pm` and `Kernel/Config/Files/ZZZProcessManagement.pm` (if it exists) will be evaluated. These files are used by the graphical interface for system configuration caching and should never be changed manually. Lastly, the file `Kernel/Config.pm` that contains your individual settings and manually changed configuration parameters, will be parsed. Reading the configuration files in this order makes sure that your specific configuration settings are used by the system.

2.2. Configuring the system through the web interface

Since OTRS 2.0, nearly all configuration parameters of the central framework or additional installed applications, can be changed easily with the graphical interface for system configuration. Log in as OTRS administrator and follow the SysConfig link on the Admin page to execute the new configuration tool (see Figure below).



You are logged in as Miguel Hidalgo y Costilla

Please enter a search term to look for settings.

Powered by OTRS 3.0.x CVS

✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂
✂✂✂✂✂✂✂✂✂✂✂✂✂✂

OTRS currently has over 600 configuration parameters, and there are different ways to quickly access a specific one. With the full text search, all configuration parameters can be scanned for one or more keywords. The full text search not only searches through the names of the configuration parameters, but also through the descriptions of the parameters. This allows an element to be found easily even if its name is unknown.

Furthermore, all configuration parameters are sorted in main groups and sub groups. The main group represents the application that the configuration parameter belongs to, e.g. "Framework" for the central OTRS framework, "Ticket" for the ticket system, "FAQ" for the FAQ system, and so on. The sub groups can be accessed if the application is selected from the groups listbox and the "Select group" button is pressed.

Every configuration parameter can be turned on or off via a checkbox. If the parameter is turned off, the system will ignore this parameter or use a default. It is possible to switch a changed configuration parameter back to the system default using the Reset link. The Update button submits all changes to system configuration parameters.

If you want to save all the changes you made to your system's configuration, for example to setup a new installation quickly, you can use the "Export settings" button, which will create a .pm file. To restore your own settings, just press the "Import settings" and select the .pm created before.

> ### Note
>
> For security reasons, the configuration parameters for the database connection cannot be changed in the SysConfig section. They have to be set manually in `Kernel/Config.pm`.

3. Backing up the system

This chapter describes the backup and restore of the OTRS data.

3.1. Backup

There are two types of data to backup: application files (e.g. the files in `/opt/otrs`), and the data stored in the database.

To simplify backups, the script `scripts/backup.pl` is included with every OTRS installation. It can be run to backup all important data (see Script below).

```
linux:/opt/otrs# cd scripts/
linux:/opt/otrs/scripts# ./backup.pl --help
backup.pl <Revision 1.1> - backup script
Copyright (c) 2001-2005 Martin Edenhofer <martin@otrs.org>
usage: backup.pl -d /data_backup/ [-c bzip2|gzip] [-r 30] [-t nofullbackup]
linux:/opt/otrs/scripts#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂ ✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂

Execute the command specified in the script below to create a backup:

```
linux:/opt/otrs/scripts# ./backup.pl -d /backup/
Backup /backup//2010-09-07_14-28/Config.tar.gz ... done
Backup /backup//2010-09-07_14-28/Application.tar.gz ... done
Dump MySQL rdbms ... done
Compress SQL-file... done
linux:/opt/otrs/scripts#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂

All data was stored in the directory `/backup/2010-09-07_14-28/` (see Script below). Additionally, the data was saved into a .tar.gz file.

```
linux:/opt/otrs/scripts# ls /backup/2010-09-07_14-28/
Application.tar.gz  Config.tar.gz  DatabaseBackup.sql.gz
linux:/opt/otrs/scripts#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂

3.2. Restore

To restore a backup, the saved application data has to be written back into the installation directory, e.g. `/opt/otrs`. Also the database has to be restored.

A script `scripts/restore.pl` (see Script below), which simplifies the restore process, is shipped with every OTRS installation. It supports MySQL and PostgreSQL.

```
linux:/opt/otrs/scripts# ./restore.pl --help
restore.pl <Revision 1.1> - restore script
Copyright (c) 2001-2005 Martin Edenhofer <martin@otrs.org>
usage: restore.pl -b /data_backup/<TIME>/ -d /opt/otrs/
linux:/opt/otrs/scripts#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂

Data that is stored, for example, in the directory `/backup/2010-09-07_14-28/`, can be restored with the command specified in the script below, assuming the OTRS installation is at `/opt/otrs`.

```
linux:/opt/otrs/scripts# ./restore.pl -b /backup/2010-09-07_14-28 -d /opt/otrs/
Restore /backup/2010-09-07_14-28//Config.tar.gz ...
Restore /backup/2010-09-07_14-28//Application.tar.gz ...
create MySQL
decompresses SQL-file ...
cat SQL-file into MySQL database
compress SQL-file...
linux:/opt/otrs/scripts#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂

4. Email settings

4.1. Sending/Receiving emails

4.1.1. Sending emails

4.1.1.1. Via Sendmail (default)

OTRS can send out emails via Sendmail, Postfix, Qmail or Exim). The default configuration is to use Sendmail and should work out-of-the-box.

You can configure the sendmail settings via the graphical configuration frontend (Framework::Core::Sendmail)

4.1.1.2. Via SMTP server or smarthost

OTRS can send emails via SMTP (Simple Mail Transfer Protocol / RFC 821) or Secure SMTP. You will want to use this on non-UNIX platforms (e.g. Windows).

The SMTP server settings can be configured via the SysConfig (Framework::Core::Sendmail). If you don't see SMTPS available as an option, the required Perl modules are missing. In that case, please refer to "Installation of Perl modules required for OTRS" for instructions.

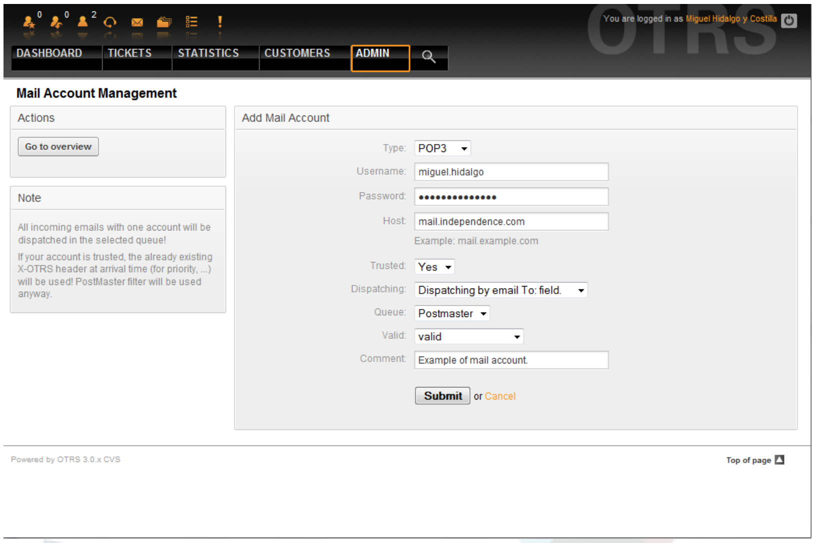
4.1.2. Receiving emails

4.1.2.1. Mail accounts configured via the OTRS GUI

OTRS is able to receive emails from POP3, POP3S, IMAP, and IMAPS mail accounts.

Configure your mail accounts via the PostMaster Mail Accounts link on the Admin page.

If a new mail account is to be created (see Figure below), then its mail server name, login name and password must be specified. Also, you need to select the mail server type, which can be POP3, POP3S, IMAP or IMAPS. If you don't see your server type available as an option, the required Perl modules are missing on your system. In that case, please refer to "Installation of Perl modules required for OTRS" for instructions.



You are logged in as Miguel Hidalgo y Costilla

Mail Account Management

Actions

Note

All incoming emails with one account will be dispatched in the selected queue!

If your account is trusted, the already existing X-OTRS header at arrival time (for priority, ...) will be used! PostMaster filter will be used anyway.

Add Mail Account

Type: POP3
Username: miguel.hidalgo
Password: ••••••••••••••
Host: mail.independence.com
Example: mail.example.com
Trusted: Yes
Dispatching: Dispatching by email To: field.
Queue: Postmaster
Valid: valid
Comment: Example of mail account.

Powered by OTRS 3.0.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂

If you select Yes for the value of the Trusted option, any X-OTRS headers attached to an incoming message are evaluated and executed. Because the X-OTRS header can execute some actions in the ticket system, you should set the Trusted option to Yes only for known senders. X-OTRS-Headers are used by the filter module in OTRS. The X-OTRS headers are explained in this table in more detail. Any postmaster filter rules created are executed, irrespective of the Trusted option's setting.

The distribution of incoming messages can be controlled if they need to be sorted by queue or by the content of the "To:" field. For the Dispatching field, if "Dispatching by selected queue" is selected, all incoming messages will be sorted into the specified queue. The address where the mail was sent to is disregarded in this case. If "Dispatching by email To: field" is selected, the system checks if a queue is linked with the address in the To: field of the incoming mail. You can link an address to a queue in the E-mail address management section of the Admin page. If the address in the To: field is linked with a queue, the new message will be sorted into the linked queue. If no link is found between the address in the To: field and any queue, then the message flows into the "Raw" queue in the system, which is the PostmasterDefaultQueue after a default installation.

All data for the mail accounts are saved in the OTRS database. The `otrs.PostMasterMailbox.pl` script, which is located in the bin directory of your OTRS installation, uses the settings in the database and fetches the mail. You can execute `./bin/otrs.PostMasterMailbox.pl` manually to check if all your mail settings are working properly.

On a normal installation, the mail will be fetched every 10 minutes by the `postmaster_mailbox` cron job. For further information about modifying cron jobs, please refer to the "Setting up the cron jobs for OTRS" section.

Note

When fetching mail, OTRS deletes the mail from the POP or IMAP server. There is no option to also keep a copy on the server. If you want to retain a copy on the server, you should create forwarding rules on your mail server. Please consult your mail server documentation for details.

4.1.2.2. Via command line program and procmail (otrs.PostMaster.pl)

If you cannot use mail accounts to get the email into OTRS, the command line program `bin/otrs.PostMaster.pl` might be a way around the problem. It takes the mails via STDIN and pipes them directly into OTRS. That means email will be available in your OTRS system if the MDA (mail delivery agent, e.g. procmail) executes this program.

To test `bin/otrs.PostMaster.pl` without an MDA, execute the command of the following script.

```
linux:/opt/otrs# cd bin
linux:/opt/otrs/bin# cat ../doc/sample_mails/test-email-1.box | ./otrs.PostMaster.pl
linux:/opt/otrs/bin#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂

If the email is shown in the QueueView, then your setup is working.

Procmail is a very common e-mail filter in Linux environments. It is installed on most systems. If not, have a look at the ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂.

To configure procmail for OTRS (based upon a procmail configured MTA such as sendmail, postfix, exim or qmail), use the `~otrs/.procmailrc.dist` file and copy it to `.procmailrc` and add the lines of the script below.

```
SYS_HOME=$HOME
PATH=/bin:/usr/bin:/usr/local/bin
# --
# Pipe all email into the PostMaster process.
# --
:0 :
| $SYS_HOME/bin/otrs.PostMaster.pl
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂

All email sent to the local OTRS user will be piped into `bin/otrs.PostMaster.pl` and then shown in your QueueView.

4.1.2.3. Fetching emails via POP3 or IMAP and fetchmail for otrs.PostMaster.pl

In order to get email from your mail server, via a POP3 or IMAP mailbox, to the OTRS machine/ local OTRS account and to procmail, use fetchmail.

Note

A working SMTP configuration on the OTRS machine is required.

You can use the `.fetchmailrc.dist` in the home directory of OTRS and copy it to `.fetchmailrc`. Modfiy/change it for your needs (see the Example 7-1 below).

Example 4.2. .fetchmailrc

```
#poll (mailserver) protocol POP3 user (user) password (password) is (localuser)
poll mail.example.com protocol POP3 user joe password mama is otrs
```

Don't forget to set the .fetchmailrc to 710 ("chmod 710 .fetchmailrc")!

With the `.fetchmailrc` from the Example 7-1 above, all email will be forwarded to the local OTRS account, if the command fetchmail -a is executed. Set up a cronjob with this command if you want to fetch the mails regularly.

4.1.2.4. Filtering/dispatching by OTRS/PostMaster modules (for more complex dispatching)

If you use the bin/otrs.PostMaster.pl or bin/otrs.PostMasterMailbox.pl method, you can insert or modify X-OTRS header entries with the PostMaster filter modules. With the X-OTRS headers, the ticket system can execute some actions on incoming mails, sort them into a specific queue, change the priority or change the customer ID, for example. More information about the X-OTRS headers are available in the section about adding mail accounts from the OTRS Admin page.

There are some default filter modules:

> Note
>
> The job name (e.g. $Self->{'PostMaster::PreFilterModule'}->{'JobName'}) needs to be unique!

Kernel::System::PostMaster::Filter::Match is a default module to match on some email header (e.g. From, To, Subject, ...). It can set new email headers (e.g. X-OTRS-Ignore: yes or X-OTRS-Queue: spam) if a filter rule matches. The jobs of the Example 7-2 can be inserted in `Kernel/Config.pm`

Example 4.3. Example jobs for the filter module Kernel::System::PostMaster::Filter::Match

```
    # Job Name: 1-Match
    # (block/ignore all spam email with From: noreply@)
    $Self->{'PostMaster::PreFilterModule'}->{'1-Match'} = {
        Module => 'Kernel::System::PostMaster::Filter::Match',
        Match => {
            From => 'noreply@',
        },
        Set => {
            'X-OTRS-Ignore' => 'yes',
        },
    };
    # Job Name: 2-Match
    # (sort emails with From: sales@example.com and Subject: **ORDER**
    # into queue 'Order')
    $Self->{'PostMaster::PreFilterModule'}->{'2-Match'} = {
        Module => 'Kernel::System::PostMaster::Filter::Match',
        Match => {
            To => 'sales@example.com',
            Subject => '**ORDER**',
        },
        Set => {
            'X-OTRS-Queue' => 'Order',
        },
    };
```

Kernel::System::PostMaster::Filter::CMD is a default module to pipe the email into an external command. The output is given to STDOUT and if the result is true, then set new email header (e.g. X-OTRS-Ignore: yes or X-OTRS-Queue: spam). The Example 7-3 can be used in `Kernel/Config.pm`

Example 4.4. Example job for the filter module Kernel::System::PostMaster::Filter::CMD

```
# Job Name: 5-SpamAssassin
# (SpamAssassin example setup, ignore spam emails)
$Self->{'PostMaster::PreFilterModule'}->{'5-SpamAssassin'} = {
    Module => 'Kernel::System::PostMaster::Filter::CMD',
    CMD => '/usr/bin/spamassassin | grep -i "X-Spam-Status: yes"',
    Set => {
        'X-OTRS-Ignore' => 'yes',
    },
};
```

Of course it's also possible to develop your own PostMaster filter modules.

4.2. Secure email with PGP

OTRS has the capability to sign or encrypt outgoing messages with PGP. Furthermore, encrypted incoming messages can be decrypted. Encryption and decryption are done with the GPL tool GnuPG. To setup GnuPG for OTRS, the following steps have to be performed:

1. Install GnuPG, via the package manager of your operating system.
2. Configure GnuPG for use with OTRS. The necessary directories for GnuPG and a private key have to be created. The command shown in the script below has to be executed as user 'otrs' from a shell.

```
linux:~# su otrs
linux:/root$ cd
linux:~$ pwd
/opt/otrs
linux:~$ gpg --gen-key
gpg (GnuPG) 1.4.2; Copyright (C) 2005 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

gpg: directory `/opt/otrs/.gnupg' created
gpg: new configuration file `/opt/otrs/.gnupg/gpg.conf' created
gpg: WARNING: options in `/opt/otrs/.gnupg/gpg.conf' are not yet active during t
his run
gpg: keyring `/opt/otrs/.gnupg/secring.gpg' created
gpg: keyring `/opt/otrs/.gnupg/pubring.gpg' created
Please select what kind of key you want:
   (1) DSA and Elgamal (default)
   (2) DSA (sign only)
   (5) RSA (sign only)
Your selection? 1
DSA keypair will have 1024 bits.
ELG-E keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
     0 = key does not expire
  <n>  = key expires in n days
  <n>w = key expires in n weeks
  <n>m = key expires in n months
  <n>y = key expires in n years
Key is valid for? (0)
Key does not expire at all
Is this correct? (y/N) y

You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Ticket System
Email address: support@example.com
Comment: Private PGP Key for the ticket system with address support@example.com
You selected this USER-ID:
"Ticket System (Private PGP Key for the ticket system with address support@examp
le.com) <support@example.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

Passphrase: secret
Repeat passphrase: secret

We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
++++++++++.+++++++++++++++++++++++++....+++++.+++++...+++++++++++++++++++++++++.
+++++++++++++++++++++++++.+++++.+++++.+++++++++++++++++++++++++>++++++++++>+++++
.......>+++++<+++++................................+++++

Not enough random bytes available.  Please do some other work to give
the OS a chance to collect more entropy! (Need 280 more bytes)

++++++++++.+++++..++++++++++..+++++....++++++++++++++++++++.+++++++++++++++.++++
++++++++++++++++++++++++++.++++++++++.+++++++++++++++.++++++++++.+++++++++++++++
..+++++>.+++++....>+++++........................................................
...........................................................>+++++<+++++.........
.............+++++^^^
gpg: /opt/otrs/.gnupg/trustdb.gpg: trustdb created
gpg: key 7245A970 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
pub   1024D/7245A970 2006-02-03
      Key fingerprint = 2ED5 BC36 D2B6 B055 7EE1  5833 1D7B F967 7245 A970
 uid                  Ticket System (Private pgp key for ticket system with addre
ss support@example.com) <support@example.com>
sub   2048g/52B97069 2006-02-03

linux:~$
```

* * *

As shown in the script below, the default settings can be applied for most of the required parameters. Only the values for the key owner have to be entered correctly, with a proper password specified for the key.

3. Now OTRS has to be made ready to use PGP. From the Admin console, open the SysConfig interface and search for "PGP". Select the sub group Crypt::PGP from the search results.

   In the screen for the PGP settings, PGP should be activated for OTRS (first option). Also, the path to the gpg program should be set and checked.

The next config setting (PGP::Options) may also require changing. Via this config setting, the parameters that are used for every execution of gpg by the 'otrs' user can be specified. In particular, the directory of the config files for GnuPG of the 'otrs' user is important. In the example `/opt/otrs/.gnupg` is used. This directory was created earlier during the PGP configuration.

Via the next config option (PGP::Key::Password) it is possible to specify the pairs of key IDs and their passwords for own private keys. Because communication partners from outside write to the ticket system with their messages encrypted with your public key, OTRS can decrypt these messages with the ID/passwords specified here.

How to get the id of your own private key? The ID of your own private key is already shown during the key generation (see step 1 from above). It is also possible to get the ID if the command specified in the following script is executed as user 'otrs':

```
linux:~# su otrs
linux:/root$ cd
linux:~$ pwd
/opt/otrs
linux:~$ gpg --list-keys
/opt/otrs/.gnupg/pubring.gpg
----------------------------
pub   1024D/7245A970 2006-02-03
uid                  Ticket System (Private pgp key for ticket system with
address support@example.com) <support@example.com>
sub   2048g/52B97069 2006-02-03

linux:~$
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂ ✂✂ ✂✂ ✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂

The ID of the private key can be found in the line that starts with "sub". It is a hexadecimal string that is eight characters long, in the example above it is "52B97069". The password you have to specify for this key in the ticket system is the same that was given during key generation.

After this data is inserted, the "Update" button can be used to save the settings. OTRS is ready to receive and decrypt encoded messages now.

4. Finally, import a customer's public key. This ensures that encrypted messages can be sent out to this customer. There are two ways to import a public key of a customer.

   The first way is to specify the public key of a customer in the customer management interface.

   The second way is to specify the key via the PGP settings, reachable from the Admin page. On the right section of this screen, all already imported public keys of customers are displayed. After PGP has been activated and configured for OTRS, your own public key should also be listed there. In the left area of the PGP setting screen it is possible to search for keys. Also, a new public key can be uploaded into the system from a file.

   The files with the public key that need to be imported into OTRS have to be GnuPGP compatible key files. In most cases, the key stored in a file is an "ASCII armored key". OTRS can deal with this format.

4.3. Secure email with S/MIME

At first glance, encryption with S/MIME seems a little more complicated than with PGP. First, you have to establish a Certification Authority (CA) for the OTRS system. The subsequent steps are very much like those needed with PGP: configure OTRS, install your own certificate, import other public certificates as needed, etc.

The S/MIME configuration is conducted outside the OTRS web interface for the most part, and should be carried out in a shell by the 'otrs' user. The MIME configuration under Linux is based on SSL (OpenSSL). Therefore, check first of all whether the OpenSSL package is installed on your system. The OpenSSL package includes a script called CA.pl, with which the most important steps of certificate creation can be performed. To simplify the procedure, find out where in the filesystem the CA.pl script is stored and enter the location temporarily into the PATH variable of the shell (see Script below).

```
otrs@linux:~> rpm -ql openssl | grep CA
/usr/share/ssl/misc/CA.pl
otrs@linux:~> export PATH=$PATH:/usr/share/ssl/misc
otrs@linux:~> which CA.pl
/usr/share/ssl/misc/CA.pl
otrs@linux:~> mkdir tmp; cd tmp
otrs@linux:~/tmp>
```

* * *

The script above shows that a new temporary directory ~/tmp has been created, in which the certificate is to be generated.

To create a certificate, perform the following operations in the command line (we assume that the OTRS administrator has to create a SSL certificate for test and learning purposes. In case you already have a certified SSL certificate for the encryption, use it and skip these steps):

1. Establish your own Certification Authority for SSL. You need it to certify the request for your own SSL certificate (see Script below).

   ```
   otrs@linux:~/tmp> CA.pl -newca
   CA certificate filename (or enter to create)

   Making CA certificate ...
   Generating a 1024 bit RSA private key
   ...++++++
   ......++++++
   writing new private key to './demoCA/private/cakey.pem'
   Enter PEM pass phrase:
   Verifying - Enter PEM pass phrase:
   -----
   You are about to be asked to enter information that will be incorporated
   into your certificate request.
   What you are about to enter is what is called a Distinguished Name or a DN.
   There are quite a few fields but you can leave some blank
   For some fields there will be a default value,
   If you enter '.', the field will be left blank.
   -----
   Country Name (2 letter code) [AU]:DE
   State or Province Name (full name) [Some-State]:OTRS-state
   Locality Name (eg, city) []:OTRS-town
   Organization Name (eg, company) [Internet Widgits Pty Ltd]:Your company
   Organizational Unit Name (eg, section) []:
   Common Name (eg, YOUR name) []:OTRS Admin
   Email Address []:otrs@your-domain.tld
   otrs@linux:~/tmp> ls -la demoCA/
   ```

```
total 8
-rw-r--r--  1 otrs otrs 1330 2006-01-08 17:54 cacert.pem
drwxr-xr-x  2 otrs otrs   48 2006-01-08 17:53 certs
drwxr-xr-x  2 otrs otrs   48 2006-01-08 17:53 crl
-rw-r--r--  1 otrs otrs    0 2006-01-08 17:53 index.txt
drwxr-xr-x  2 otrs otrs   48 2006-01-08 17:53 newcerts
drwxr-xr-x  2 otrs otrs   80 2006-01-08 17:54 private
-rw-r--r--  1 otrs otrs   17 2006-01-08 17:54 serial
otrs@linux:~/tmp>
```

* * *

2. Generate a certificate request (see Script below).

```
otrs@linux:~/tmp> CA.pl -newreq
Generating a 1024 bit RSA private key
..........................................++++++
....++++++
writing new private key to 'newreq.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DE\keyreturn
State or Province Name (full name) [Some-State]:OTRS-state
Locality Name (eg, city) []:OTRS-town
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Your company
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:OTRS admin
Email Address []:otrs@your-domain.tld

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Request (and private key) is in newreq.pem
otrs@linux:~/tmp> ls -la
total 4
drwxr-xr-x  6 otrs otrs  232 2006-01-08 17:54 demoCA
-rw-r--r--  1 otrs otrs 1708 2006-01-08 18:04 newreq.pem
otrs@linux:~/tmp>
```

* * *

3. Signing of the certificate request. The certificate request can either be signed and thereby certified by your own CA, or made more credible by being signed by another external certified CA (see Script below).

```
otrs@linux:~/tmp> CA.pl -signreq
Using configuration from /etc/ssl/openssl.cnf
Enter pass phrase for ./demoCA/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number:
            fd:85:f6:9f:14:07:16:c8
        Validity
            Not Before: Jan  8 17:04:37 2006 GMT
            Not After : Jan  8 17:04:37 2007 GMT
        Subject:
            countryName               = DE
            stateOrProvinceName       = OTRS-state
            localityName              = OTRS-town
            organizationName          = Your Company
            commonName                = OTRS administrator
            emailAddress              = otrs@your-domain.tld
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                01:D9:1E:58:C0:6D:BF:27:ED:37:34:14:D6:04:AC:C4:64:98:7A:22
            X509v3 Authority Key Identifier:
                keyid:10:4D:8D:4C:93:FD:2C:AA:9A:B3:26:80:6B:F5:D5:31:E2:8E:DB:A8
                DirName:/C=DE/ST=OTRS-state/L=OTRS-town/O=Your Company/
                CN=OTRS admin/emailAddress=otrs@your-domain.tld
                serial:FD:85:F6:9F:14:07:16:C7

Certificate is to be certified until Jan  8 17:04:37 2007 GMT (365 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
Signed certificate is in newcert.pem
otrs@linux:~/tmp>
```

* * *

4. Generate your own certificate, and all data going with it, using the signed certificate request (see Script below).

```
otrs@linux:~/tmp> CA.pl -pkcs12 "OTRS Certificate"
Enter pass phrase for newreq.pem:
Enter Export Password:
Verifying - Enter Export Password:
otrs@linux:~/tmp> ls -la
total 12
drwxr-xr-x  6 otrs otrs  328 2006-01-08 18:04 demoCA
-rw-r--r--  1 otrs otrs 3090 2006-01-08 18:13 newcert.p12
-rw-r--r--  1 otrs otrs 3791 2006-01-08 18:04 newcert.pem
-rw-r--r--  1 otrs otrs 1708 2006-01-08 18:04 newreq.pem
otrs@linux:~/tmp>
```

* * *

Now that these operations have been performed, the S/MIME setup must be completed in OTRS.

This part of the setup is carried out in the Admin page, choosing the link "SMIME". In case the general S/MIME support in OTRS has not yet been enabled, the mask points this out to the administrator and provides an appropriate link for enabling it.

With the SysConfig group "Crypt::SMIME", you can also enable and configure the general S/MIME support.

Here you can activate S/MIME support, and define the paths for the OpenSSL command and the directory for the certificates. The key file created above must be stored in the directory indicated here. Otherwise OpenSSL cannot use it.

The next step is performed in the S/MIME configuration on the OTRS Admin page. Here, you can import the private key(s) of the OTRS system and the public keys of other communication partners. Enter the public key that has been created in the beginning of this section and added to OTRS.

Obviously, all public S/MIME keys of communication partners can be imported using the customer administration tool as well.

5. Using external backends

5.1. Customer data

OTRS works with many customer data attributes such as username, email address, phone number, etc. These attributes are displayed in both the Agent and the Customer frontends, and also used for the authentication of customers.

Customer data used or displayed within OTRS is highly customizable. The following information is however always needed for customer authentication:

- User login
- Email address
- Customer ID

Use the configuration parameters of the following script in your `Kernel/Config.pm` file, if you want to display customer information in your agent interface.

```
    # Ticket::Frontend::CustomerInfo*
    # (show customer info on Compose (Phone and Email), Zoom and
    # Queue view)
    $Self->{'Ticket::Frontend::CustomerInfoCompose'} = 1;
    $Self->{'Ticket::Frontend::CustomerInfoZoom'} = 1;
    $Self->{'Ticket::Frontend::CustomerInfoQueue'} = 0;
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂
✂✂✂✂✂✂✂✂✂✂✂

5.2. Customer user backend

You can use two types of customer backends, DB and LDAP. If you already have another customer backend (e.g. SAP), it is of course possible to write a module that uses it.

5.2.1. Database (Default)

Example 11-1 shows the configuration of a DB customer backend, which uses customer data stored in the OTRS database.

Example 4.5. Configuring a DB customer backend

```
# CustomerUser (customer database backend and settings)
$Self->{CustomerUser} = {
    Name => 'Database Datasource',
    Module => 'Kernel::System::CustomerUser::DB',
    Params => {
        # if you want to use an external database, add the required settings
#            DSN => 'DBI:odbc:yourdsn',
#            Type => 'mssql', # only for ODBC connections
#            DSN => 'DBI:mysql:database=customerdb;host=customerdbhost',
#            User => '',
#            Password => '',
            Table => 'customer_user',
            # if your frontend is unicode and the charset of your
            # customer database server is iso-8859-1, use these options.
#           SourceCharset => 'iso-8859-1',
#           DestCharset => 'utf-8',

            # CaseSensitive will control if the SQL statements need LOWER()
            #   function calls to work case insensitively. Setting this to
            #   1 will improve performance dramatically on large databases.
            CaseSensitive => 0,
        },
# customer unique id
CustomerKey => 'login',

# customer #
CustomerID => 'customer_id',
CustomerValid => 'valid_id',
    CustomerUserListFields => ['first_name', 'last_name', 'email'],
    CustomerUserSearchFields => ['login', 'last_name', 'customer_id'],
    CustomerUserSearchPrefix => '',
    CustomerUserSearchSuffix => '*',
    CustomerUserSearchListLimit => 250,
    CustomerUserPostMasterSearchFields => ['email'],
    CustomerUserNameFields => ['title','first_name','last_name'],
    CustomerUserEmailUniqCheck => 1,
#    # show not own tickets in customer panel, CompanyTickets
#    CustomerUserExcludePrimaryCustomerID => 0,
#    # generate auto logins
#    AutoLoginCreation => 0,
#    AutoLoginCreationPrefix => 'auto',
#    # admin can change customer preferences
#    AdminSetPreferences => 1,
#    # cache time to live in sec. - cache any database queries
#    CacheTTL => 0,
#    # just a read only source
#    ReadOnly => 1,
    Map => [
        # note: Login, Email and CustomerID needed!
        # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
 readonly, http-link-target
        [ 'UserTitle',      'Title',      'title',      1, 0, 'var', '', 0 ],
        [ 'UserFirstname',  'Firstname',  'first_name', 1, 1, 'var', '', 0 ],
        [ 'UserLastname',   'Lastname',   'last_name',  1, 1, 'var', '', 0 ],
        [ 'UserLogin',      'Username',   'login',      1, 1, 'var', '', 0 ],
        [ 'UserPassword',   'Password',   'pw',         0, 0, 'var', '', 0 ],
        [ 'UserEmail',      'Email',      'email',      1, 1, 'var', '', 0 ],

#        [ 'UserEmail',      'Email', 'email',           1, 1, 'var', '$Env{"CGIHandle"}?
Action=AgentTicketCompose&ResponseID=1&TicketID=$Data{"TicketID"}&ArticleID=
$Data{"ArticleID"}', 0 ],
        [ 'UserCustomerID', 'CustomerID', 'customer_id', 0, 1, 'var', '', 0 ],

#        [ 'UserCustomerIDs', 'CustomerIDs', 'customer_ids', 1, 0, 'var', '', 0 ],
        [ 'UserPhone',        'Phone',       'phone',        1, 0, 'var', '', 0 ],
        [ 'UserFax',          'Fax',         'fax',          1, 0, 'var', '', 0 ],
        [ 'UserMobile',       'Mobile',      'mobile',       1, 0, 'var', '', 0 ],
        [ 'UserStreet',       'Street',      'street',       1, 0, 'var', '', 0 ],
        [ 'UserZip',          'Zip',         'zip',          1, 0, 'var', '', 0 ],
        [ 'UserCity',         'City',        'city',         1, 0, 'var', '', 0 ],
        [ 'UserCountry',      'Country',     'country',      1, 0, 'var', '', 0 ],
        [ 'UserComment',      'Comment',     'comments',     1, 0, 'var', '', 0 ],
        [ 'ValidID',          'Valid',       'valid_id',     0, 1, 'int', '', 0 ],
    ],
    # default selections
    Selections => {
        UserTitle => {
            'Mr.' => 'Mr.',
            'Mrs.' => 'Mrs.',
        },
    },
};
```

If you want to customize the customer data, change the column headers or add new ones to the customer_user table in the OTRS database. As an example, the script below shows how to add a new field for room number.

```
linux:~# mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 116 to server version: 5.0.18-Debian_7-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use otrs;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> ALTER TABLE customer_user ADD room VARCHAR (250);
Query OK, 1 rows affected (0.01 sec)
Records: 1  Duplicates: 0  Warnings: 0

mysql> quit
Bye
linux:~#
```

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁ ✁✁✁✁ ✁✁✁✁✁ ✁✁ ✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁

Now add the new column to the MAP array in `Kernel/Config.pm`, as shown in the following script.

```
   # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
 readonly
   [...]
   [ 'UserRoom',      'Room',      'room',       0, 1, 'var', '', 0 ],
```

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁ ✁✁✁✁ ✁✁✁✁✁ ✁✁ ✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁

It is also possible to edit all of this customer information via the Customers link in the Agent interface.

5.2.1.1. Customer with multiple IDs (Company tickets)

It is possible to assign more than one customer ID to a customer. This can be useful if a customer must access tickets of other customers, e.g. a supervisor wants to watch the tickets of his assistants. If a customer can access the tickets of another customer, the company ticket feature of OTRS is used. Company tickets can be accessed via the "Company Tickets" link in the customer panel.

To use company tickets, a new column with the IDs that should be accessible for a customer, has to be added to the customer_user table in the OTRS database (see Script below).

```
linux:~# mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 124 to server version: 5.0.18-Debian_7-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use otrs;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> ALTER TABLE customer_user ADD customer_ids VARCHAR (250);
Query OK, 1 rows affected (0.02 sec)
Records: 1  Duplicates: 0  Warnings: 0

mysql> quit
Bye
linux:~#
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂

Now the new column has to be added to the MAP array in `Kernel/Config.pm`, as shown in the script below.

```
   # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
 readonly
   [...]
   [ 'UserCustomerIDs', 'CustomerIDs', 'customer_ids', 1, 0, 'var', '', 0 ],
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂

Now, the new column for the multiple customer IDs can be edited via the Agent interface, in the section for the customer management.

To ensure that one customer can access the tickets of other customers, add the IDs of these other users into the new field for the multiple customer IDs. Each ID has to be separated by a semicolon (see Example 11-2 below).

Example 4.6. Using company tickets with a DB backend

The customers A, B and C exist in your system, and A wants to have access to the tickets of B and C via the customer panel. B and C should have no access to the tickets of other users.

To realize this setup, change the customer_user table and the mapping in `Kernel/Config.pm` as described above. Then load the settings for customer A via the Customers link in the Agent interface or via the Admin page. If the settings are displayed, add into the field for CustomerIDs the values "B;C;".

5.2.2. LDAP

If you have a LDAP directory with your customer data, you can use it as the customer backend with OTRS, as shown in Example 11-3.

Example 4.7. Configuring an LDAP customer backend

```
# CustomerUser
# (customer ldap backend and settings)
$Self->{CustomerUser} = {
    Name => 'LDAP Data Source',
    Module => 'Kernel::System::CustomerUser::LDAP',
    Params => {
        # ldap host
        Host => 'bay.csuhayward.edu',
        # ldap base dn
        BaseDN => 'ou=seas,o=csuh',
        # search scope (one|sub)
        SSCOPE => 'sub',
        # The following is valid but would only be necessary if the
        # anonymous user does NOT have permission to read from the LDAP tree
        UserDN => '',
        UserPw => '',
        # in case you want to add always one filter to each ldap query, use
        # this option. e. g. AlwaysFilter => '(mail=*)' or AlwaysFilter => '(objectclass=user)'
        AlwaysFilter => '',
            # if both your frontend and your LDAP are unicode, use this:
            SourceCharset => 'utf-8',
            DestCharset   => 'utf-8',
            # if your frontend is unicode and the charset of your
            # ldap server is iso-8859-1, use these options.
            # SourceCharset => 'iso-8859-1',
            # DestCharset => 'utf-8',
            # Net::LDAP new params (if needed - for more info see perldoc Net::LDAP)
            Params => {
                port => 389,
                timeout => 120,
                async => 0,
                version => 3,
            },
    },
    # customer unique id
    CustomerKey => 'uid',
    # customer #
    CustomerID => 'mail',
    CustomerUserListFields => ['cn', 'mail'],
    CustomerUserSearchFields => ['uid', 'cn', 'mail'],
    CustomerUserSearchPrefix => '',
    CustomerUserSearchSuffix => '*',
    CustomerUserSearchListLimit => 250,
    CustomerUserPostMasterSearchFields => ['mail'],
    CustomerUserNameFields => ['givenname', 'sn'],
    # show not own tickets in customer panel, CompanyTickets
    CustomerUserExcludePrimaryCustomerID => 0,
    # add an ldap filter for valid users (expert setting)
#    CustomerUserValidFilter => '(!(description=locked))',
    # administrator can't change customer preferences
    AdminSetPreferences => 0,
#    # cache time to live in sec. - cache any database queries
#    CacheTTL => 0,
    Map => [
        # note: Login, Email and CustomerID are mandatory!
        # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
 readonly
        [ 'UserTitle',      'Title',      'title',           1, 0, 'var', '', 0 ],
        [ 'UserFirstname',  'Firstname',  'givenname',       1, 1, 'var', '', 0 ],
        [ 'UserLastname',   'Lastname',   'sn',              1, 1, 'var', '', 0 ],
        [ 'UserLogin',      'Username',   'uid',             1, 1, 'var', '', 0 ],
        [ 'UserEmail',      'Email',      'mail',            1, 1, 'var', '', 0 ],
        [ 'UserCustomerID', 'CustomerID', 'mail',            0, 1, 'var', '', 0 ],
#        [ 'UserCustomerIDs', 'CustomerIDs', 'second_customer_ids', 1, 0, 'var', '', 0 ],
        [ 'UserPhone',      'Phone',      'telephonenumber', 1, 0, 'var', '', 0 ],
        [ 'UserAddress',    'Address',    'postaladdress',   1, 0, 'var', '', 0 ],
        [ 'UserComment',    'Comment',    'description',     1, 0, 'var', '', 0 ],
    ],
};
```

If additional customer attributes are stored in your LDAP directory, such as a manager's name, a mobile phone number, or a department, and if you want to display this information in OTRS, just expand the MAP array in `Kernel/Config.pm` with the entries for these attributes, as shown in the following script.

```
   # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
readonly
   [...]
   [ 'UserPhone',      'Phone',      'telephonenumber', 1, 0, 'var', '', 0 ],
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂
✂✂✂✂✂

5.2.2.1. Customer with multiple IDs (Company tickets)

It is possible to assign more than one Customer ID to a customer, when using an LDAP backend. To use company tickets, a new field has to be added to the LDAP directory that contains the IDs accessible by the customer.

If the new field in the LDAP directory has been created, the new entry has to be added to the MAP array in `Kernel/Config.pm`, as shown in the script below.

```
   # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
readonly
   [...]
   [ 'UserCustomerIDs', 'CustomerIDs', 'customer_ids', 1, 0, 'var', '', 0 ],
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂
✂✂✂✂✂

The field for the multiple customer IDs has to be edited directly in the LDAP directory. OTRS can only read from LDAP, not write to it.

To ensure access by a customer to the tickets of other customers, add the customer IDs of the customers whose tickets should be accessed to the new field in your LDAP directory. Each ID has to be separated by a semicolon (see Example 11-4 below).

Example 4.8. Using Company tickets with an LDAP backend

The customers A, B and C exist in your system and A wants to have access to the tickets of B and C via the customer panel. B and C should have no access to tickets of other users.

To realize this setup, change the LDAP directory and the mapping in `Kernel/Config.pm` as described above. Then add into the field for CustomerIDs the values "B;C;" for customer A in your LDAP directory.

5.2.3. Using more than one customer backend with OTRS

If you want to utilize more than one customer data source used with OTRS (e.g. an LDAP and a database backend), the CustomerUser config parameter should be expanded with a number, e.g. "CustomerUser1", "CustomerUser2" (see Example 11-5 below).

Example 4.9. Using more than one customer backend with OTRS

The following configuration example shows usage of both an LDAP and a database customer backend with OTRS.

```
# 1. Customer user backend: DB
# (customer database backend and settings)
$Self->{CustomerUser1} = {
    Name => 'Customer Database',
    Module => 'Kernel::System::CustomerUser::DB',
    Params => {
        # if you want to use an external database, add the
        # required settings
#        DSN => 'DBI:odbc:yourdsn',
#        Type => 'mssql', # only for ODBC connections
#        DSN => 'DBI:mysql:database=customerdb;host=customerdbhost',
#        User => '',
#        Password => '',
        Table => 'customer_user',
    },
    # customer unique id
    CustomerKey = 'login',
    # customer #
    CustomerID = 'customer_id',
    CustomerValid = 'valid_id',
    CustomerUserListFields => ['first_name', 'last_name', 'email'],
    CustomerUserSearchFields => ['login', 'last_name', 'customer_id'],
    CustomerUserSearchPrefix => '',
    CustomerUserSearchSuffix => '*',
    CustomerUserSearchListLimit => 250,
    CustomerUserPostMasterSearchFields => ['email'],
    CustomerUserNameFields => ['title','first_name','last_name'],
    CustomerUserEmailUniqCheck => 1,
#    # show not own tickets in customer panel, CompanyTickets
#    CustomerUserExcludePrimaryCustomerID => 0,
#    # generate auto logins
#    AutoLoginCreation => 0,
#    AutoLoginCreationPrefix => 'auto',
#    # admin can change customer preferences
#    AdminSetPreferences => 1,
#    # cache time to live in sec. - cache any database queries
#    CacheTTL => 0,
#    # just a read only source
#    ReadOnly => 1,
    Map => [

        # note: Login, Email and CustomerID needed!
        # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
 readonly, http-link-target
        [ 'UserTitle',      'Title',       'title',          1, 0, 'var', '', 0 ],
        [ 'UserFirstname',  'Firstname',   'first_name',     1, 1, 'var', '', 0 ],
        [ 'UserLastname',   'Lastname',    'last_name',      1, 1, 'var', '', 0 ],
        [ 'UserLogin',      'Username',    'login',          1, 1, 'var', '', 0 ],
        [ 'UserPassword',   'Password',    'pw',             0, 0, 'var', '', 0 ],
        [ 'UserEmail',      'Email',       'email',          1, 1, 'var', '', 0 ],
        [ 'UserCustomerID', 'CustomerID',  'customer_id',    0, 1, 'var', '', 0 ],
        [ 'UserPhone',      'Phone',       'phone',          1, 0, 'var', '', 0 ],
        [ 'UserFax',        'Fax',         'fax',            1, 0, 'var', '', 0 ],
        [ 'UserMobile',     'Mobile',      'mobile',         1, 0, 'var', '', 0 ],
        [ 'UserStreet',     'Street',      'street',         1, 0, 'var', '', 0 ],
        [ 'UserZip',        'Zip',         'zip',            1, 0, 'var', '', 0 ],
        [ 'UserCity',       'City',        'city',           1, 0, 'var', '', 0 ],
        [ 'UserCountry',    'Country',     'country',        1, 0, 'var', '', 0 ],
        [ 'UserComment',    'Comment',     'comments',       1, 0, 'var', '', 0 ],
        [ 'ValidID',        'Valid',       'valid_id',       0, 1, 'int', '', 0 ],
    ],
    # default selections
    Selections => {
        UserTitle => {
            'Mr.' => 'Mr.',
            'Mrs.' => 'Mrs.',
        },
    },
};

# 2. Customer user backend: LDAP
# (customer ldap backend and settings)
$Self->{CustomerUser2} = {
    Name => 'LDAP Datasource',
    Module => 'Kernel::System::CustomerUser::LDAP',
    Params => {
        # ldap host
        Host => 'bay.csuhayward.edu',
        # ldap base dn
        BaseDN => 'ou=seas,o=csuh',
        # search scope (one|sub)
        SSCOPE => 'sub',
#        # The following is valid but would only be necessary if the
#        # anonymous user does NOT have permission to read from the LDAP tree
        UserDN => '',
        UserPw => '',
        # in case you want to add always one filter to each ldap query, use
        # this option. e. g. AlwaysFilter => '(mail=*)' or AlwaysFilter => '(objectclass=user)'
        AlwaysFilter => '',
        # if both your frontend and your LDAP are unicode, use this:
#        SourceCharset => 'utf-8',
#        DestCharset   => 'utf-8',
        # if your frontend is e. g. iso-8859-1 and the character set of your
        # ldap server is utf-8, use these options:
#        SourceCharset => 'utf-8',
#        DestCharset => 'iso-8859-1',

        # Net::LDAP new params (if needed - for more info see perldoc Net::LDAP)
        Params => {
            port => 389,
            timeout => 120,
            async => 0,
            version => 3,
        },
    },
    # customer unique id
    CustomerKey => 'uid',
    # customer #
    CustomerID => 'mail',
    CustomerUserListFields => ['cn', 'mail'],
    CustomerUserSearchFields => ['uid', 'cn', 'mail'],
    CustomerUserSearchPrefix => '',
    CustomerUserSearchSuffix => '*',
    CustomerUserSearchListLimit => 250,
    CustomerUserPostMasterSearchFields => ['mail'],
    CustomerUserNameFields => ['givenname', 'sn'],
    # show not own tickets in customer panel, CompanyTickets
    CustomerUserExcludePrimaryCustomerID => 0,
    # add a ldap filter for valid users (expert setting)
#    CustomerUserValidFilter => '(!(description=locked))',
    # admin can't change customer preferences
    AdminSetPreferences => 0,
    Map => [
        # note: Login, Email and CustomerID needed!
        # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
 readonly
        [ 'UserTitle',      'Title',      'title',           1, 0, 'var', '', 0 ],
        [ 'UserFirstname',  'Firstname',  'givenname',       1, 1, 'var', '', 0 ],
        [ 'UserLastname',   'Lastname',   'sn',              1, 1, 'var', '', 0 ],
        [ 'UserLogin',      'Username',   'uid',             1, 1, 'var', '', 0 ],
        [ 'UserEmail',      'Email',      'mail',            1, 1, 'var', '', 0 ],
        [ 'UserCustomerID', 'CustomerID', 'mail',            0, 1, 'var', '', 0 ],
#        [ 'UserCustomerIDs', 'CustomerIDs', 'second_customer_ids', 1, 0, 'var', '', 0 ],
        [ 'UserPhone',      'Phone',      'telephonenumber', 1, 0, 'var', '', 0 ],
        [ 'UserAddress',    'Address',    'postaladdress',   1, 0, 'var', '', 0 ],
        [ 'UserComment',    'Comment',    'description',     1, 0, 'var', '', 0 ],
    ],
};
```

It is possible to integrate up to 10 different customer backends. Use the customer management interface in OTRS to view or edit (assuming write access is enabled) all customer data.

5.3. Backends to authenticate Agents and Customers

OTRS offers the option to authenticate agents and customers against different backends.

5.3.1. Authentication backends for Agents

5.3.1.1. DB (Default)

The backend to authenticate agents which is used by default is the OTRS database. Agents can be added and edited via the agent management interface in the Admin page (see Example 11-6 below).

Example 4.10. Authenticate agents against a DB backend

```

    $Self->{'AuthModule'} = 'Kernel::System::Auth::DB';
```

5.3.1.2. LDAP

If an LDAP directory has all your agent data stored, you can use the LDAP module to authenticate your users in OTRS (see Example 11-7 below). This module has only read access to the LDAP tree, which means that you cannot edit your user data via the agent management interface.

Example 4.11. Authenticate agents against an LDAP backend

```

# This is an example configuration for an LDAP auth. backend.
# (Make sure Net::LDAP is installed!)
$Self->{'AuthModule'} = 'Kernel::System::Auth::LDAP';
$Self->{'AuthModule::LDAP::Host'} = 'ldap.example.com';
$Self->{'AuthModule::LDAP::BaseDN'} = 'dc=example,dc=com';
$Self->{'AuthModule::LDAP::UID'} = 'uid';

# Check if the user is allowed to auth in a posixGroup
# (e. g. user needs to be in a group xyz to use otrs)
$Self->{'AuthModule::LDAP::GroupDN'} = 'cn=otrsallow,ou=posixGroups,dc=example,dc=com';
$Self->{'AuthModule::LDAP::AccessAttr'} = 'memberUid';
# for ldap posixGroups objectclass (just uid)
#  $Self->{'AuthModule::LDAP::UserAttr'} = 'UID';
# for non ldap posixGroups objectclass (with full user dn)
#  $Self->{'AuthModule::LDAP::UserAttr'} = 'DN';

# The following is valid but would only be necessary if the
# anonymous user do NOT have permission to read from the LDAP tree
$Self->{'AuthModule::LDAP::SearchUserDN'} = '';
$Self->{'AuthModule::LDAP::SearchUserPw'} = '';

# in case you want to add always one filter to each ldap query, use
# this option. e. g. AlwaysFilter => '(mail=*)' or AlwaysFilter => '(objectclass=user)'
$Self->{'AuthModule::LDAP::AlwaysFilter'} = '';

# in case you want to add a suffix to each login name, then
# you can use this option. e. g. user just want to use user but
# in your ldap directory exists user@domain.
#    $Self->{'AuthModule::LDAP::UserSuffix'} = '@domain.com';

# Net::LDAP new params (if needed - for more info see perldoc Net::LDAP)
$Self->{'AuthModule::LDAP::Params'} = {
    port => 389,
    timeout => 120,
    async => 0,
    version => 3,
};
```

The configuration parameters shown in the script below can be used to synchronize the user data from your LDAP directory into your local OTRS database. This reduces the number of requests to your LDAP server and speeds up the authentication with OTRS. The data synchronization is done when the agent authenticates the first time. Although the data can be syncronized into the local OTRS database, the LDAP directory is the last instance for the authentication, so an inactive user in the LDAP tree can't authenticate to OTRS, even when the account data is already stored in the OTRS database. The agent data in the LDAP directory can't be edited via the web interface of OTRS, so the data has to be managed directly in the LDAP tree.

```

# defines AuthSyncBackend (AuthSyncModule) for AuthModule
# if this key exists and is empty, there won't be a sync.
# example values: AuthSyncBackend, AuthSyncBackend2
$Self->{'AuthModule::UseSyncBackend'} = 'AuthSyncBackend';

# agent data sync against ldap
$Self->{'AuthSyncModule'} = 'Kernel::System::Auth::Sync::LDAP';
$Self->{'AuthSyncModule::LDAP::Host'} = 'ldap://ldap.example.com/';
$Self->{'AuthSyncModule::LDAP::BaseDN'} = 'dc=otrs, dc=org';
$Self->{'AuthSyncModule::LDAP::UID'} = 'uid';
$Self->{'AuthSyncModule::LDAP::SearchUserDN'} = 'uid=sys, ou=user, dc=otrs, dc=org';
$Self->{'AuthSyncModule::LDAP::SearchUserPw'} = 'some_pass';
$Self->{'AuthSyncModule::LDAP::UserSyncMap'} = {
    # DB -> LDAP
    UserFirstname => 'givenName',
    UserLastname  => 'sn',
    UserEmail     => 'mail',
};
[...]

# AuthSyncModule::LDAP::UserSyncInitialGroups
# (sync following group with rw permission after initial create of first agent
# login)
$Self->{'AuthSyncModule::LDAP::UserSyncInitialGroups'} = [
    'users',
];
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂ ✂✂✂✂ ✂✂✂✂ ✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂✂

5.3.1.3. HTTPBasicAuth for Agents

If you want to implement a "single sign on" solution for all your agents, you can use HTTP basic authentication (for all your systems) and the HTTPBasicAuth module for OTRS (see Example 11-8 below).

Example 4.12. Authenticate Agents using HTTPBasic

```
# This is an example configuration for an apache ($ENV{REMOTE_USER})
# auth. backend. Use it if you want to have a singe login through
# apache http-basic-auth
$Self->{'AuthModule'} = 'Kernel::System::Auth::HTTPBasicAuth';

# Note:
#
# If you use this module, you should use as fallback
# the following configuration settings if the user is not authorized
# apache ($ENV{REMOTE_USER})
$Self->{LoginURL} = 'http://host.example.com/not-authorised-for-otrs.html';
$Self->{LogoutURL} = 'http://host.example.com/thanks-for-using-otrs.html';
```

5.3.1.4. Radius

The configuration parameters shown in Example 11-9 can be used to authenticate agents against a Radius server.

Example 4.13. Authenticate Agents against a Radius backend

```
# This is example configuration to auth. agents against a radius server
$Self->{'AuthModule'} = 'Kernel::System::Auth::Radius';
$Self->{'AuthModule::Radius::Host'} = 'radiushost';
$Self->{'AuthModule::Radius::Password'} = 'radiussecret';
```

5.3.2. Authentication backends for Customers

5.3.2.1. Database (Default)

The default user authentication backend for customers in OTRS is the OTRS database. With this backend, all customer data can be edited via the web interface of OTRS (see Example 11-10 below).

Example 4.14. Customer user authentication against a DB backend

```
# This is the auth. module againt the otrs db
$Self->{'Customer::AuthModule'} = 'Kernel::System::CustomerAuth::DB';
$Self->{'Customer::AuthModule::DB::Table'} = 'customer_user';
$Self->{'Customer::AuthModule::DB::CustomerKey'} = 'login';
$Self->{'Customer::AuthModule::DB::CustomerPassword'} = 'pw';
#$Self->{'Customer::AuthModule::DB::DSN'} =
 "DBI:mysql:database=customerdb;host=customerdbhost";
#$Self->{'Customer::AuthModule::DB::User'} = "some_user";
#$Self->{'Customer::AuthModule::DB::Password'} = "some_password";
```

5.3.2.2. LDAP

If you have an LDAP directory with all your customer data, you can use the LDAP module to authenticate your customers to OTRS (see Example 11-11 below). Because this module has only read-access to the LDAP backend, it is not possible to edit the customer data via the OTRS web interface.

Example 4.15. Customer user authentication against an LDAP backend

```
# This is an example configuration for an LDAP auth. backend.
# (make sure Net::LDAP is installed!)
$Self->{'Customer::AuthModule'} = 'Kernel::System::CustomerAuth::LDAP';
$Self->{'Customer::AuthModule::LDAP::Host'} = 'ldap.example.com';
$Self->{'Customer::AuthModule::LDAP::BaseDN'} = 'dc=example,dc=com';
$Self->{'Customer::AuthModule::LDAP::UID'} = 'uid';

# Check if the user is allowed to auth in a posixGroup
# (e. g. user needs to be in a group xyz to use otrs)
$Self->{'Customer::AuthModule::LDAP::GroupDN'} =
 'cn=otrsallow,ou=posixGroups,dc=example,dc=com';
$Self->{'Customer::AuthModule::LDAP::AccessAttr'} = 'memberUid';
# for ldap posixGroups objectclass (just uid)
$Self->{'Customer::AuthModule::LDAP::UserAttr'} = 'UID';
# for non ldap posixGroups objectclass (full user dn)
#$Self->{'Customer::AuthModule::LDAP::UserAttr'} = 'DN';

# The following is valid but would only be necessary if the
# anonymous user does NOT have permission to read from the LDAP tree
$Self->{'Customer::AuthModule::LDAP::SearchUserDN'} = '';
$Self->{'Customer::AuthModule::LDAP::SearchUserPw'} = '';

# in case you want to add always one filter to each ldap query, use
# this option. e. g. AlwaysFilter => '(mail=*)' or AlwaysFilter => '(objectclass=user)'
$Self->{'Customer::AuthModule::LDAP::AlwaysFilter'} = '';

# in case you want to add a suffix to each customer login name, then
# you can use this option. e. g. user just want to use user but
# in your ldap directory exists user@domain.
#$Self->{'Customer::AuthModule::LDAP::UserSuffix'} = '@domain.com';

# Net::LDAP new params (if needed - for more info see perldoc Net::LDAP)
$Self->{'Customer::AuthModule::LDAP::Params'} = {
    port => 389,
    timeout => 120,
    async => 0,
    version => 3,
};
```

5.3.2.3. HTTPBasicAuth for customers

If you want to implement a "single sign on" solution for all your customer users, you can use HTTPBasic authentication (for all your systems) and use the HTTPBasicAuth module with OTRS (no login is needed with OTRS any more). See Example 11-12 below.

Example 4.16. Customer user authentication with HTTPBasic

```
# This is an example configuration for an apache ($ENV{REMOTE_USER})
# auth. backend. Use it if you want to have a singe login through
# apache http-basic-auth
$Self->{'Customer::AuthModule'} = 'Kernel::System::CustomerAuth::HTTPBasicAuth';

# Note:
# If you use this module, you should use the following
# config settings as fallback, if user isn't login through
# apache ($ENV{REMOTE_USER})
$Self->{CustomerPanelLoginURL} = 'http://host.example.com/not-authorised-for-otrs.html';
$Self->{CustomerPanelLogoutURL} = 'http://host.example.com/thanks-for-using-otrs.html';
```

5.3.2.4. Radius

The settings shown in Example 11-13 can be used to authenticate your customers against a Radius server.

Example 4.17. Customer user authentication against a Radius backend

```
# This is a example configuration to auth. customer against a radius server
$Self->{'Customer::AuthModule'} = 'Kernel::System::Auth::Radius';
$Self->{'Customer::AuthModule::Radius::Host'} = 'radiushost';
$Self->{'Customer::AuthModule::Radius::Password'} = 'radiussecret';
```

5.4. Customizing the customer self-registration

It is possible to customize the self-registration for new customers, accessible via the customer.pl panel. New optional or required fields, like room number, address or state can be added.

The following example shows how you can specify a required field in the customer database, in this case to store the room number of a customer.

5.4.1. Customizing the web interface

To display the new field for the room number in the customer.pl web interface, the .dtl file responsible for the layout in this interface has to be modified. Edit the `Kernel/Output/HTML/Standard/CustomerLogin.dtl` file, adding the new field around line 80 (see Script below).

```
[...]
<div class="NewLine">
    <label for="Room">$Text{"Room{CustomerUser}"}</label>
    <input title="$Text{"Room Number"}" name="Room" type="text" id="UserRoom" maxlength="50" />
</div>
[...]
```

* * *

5.4.2. Customer mapping

In the next step, the customer mapping has to be expanded with the new entry for the room number. To ensure that the changes are not lost after an update, put the "CustomerUser" settings from the `Kernel/Config/Defaults.pm` into the `Kernel/Config.pm`. Now change the MAP array and add the new room number field, as shown in the script below.

```
# CustomerUser
# (customer database backend and settings)
$Self->{CustomerUser} = {
    Name => 'Database Backend',
    Module => 'Kernel::System::CustomerUser::DB',
    Params => {
        # if you want to use an external database, add the
        # required settings
#         DSN => 'DBI:odbc:yourdsn',
#         Type => 'mssql', # only for ODBC connections
#         DSN => 'DBI:mysql:database=customerdb;host=customerdbhost',
#         User => '',
#         Password => '',
        Table => 'customer_user',
    },
    # customer unique id
    CustomerKey => 'login',
    # customer #
    CustomerID => 'customer_id',
    CustomerValid => 'valid_id',
    CustomerUserListFields => ['first_name', 'last_name', 'email'],
#    CustomerUserListFields => ['login', 'first_name', 'last_name', 'customer_id', 'email'],
    CustomerUserSearchFields => ['login', 'last_name', 'customer_id'],
    CustomerUserSearchPrefix => '',
    CustomerUserSearchSuffix => '*',
    CustomerUserSearchListLimit => 250,
    CustomerUserPostMasterSearchFields => ['email'],
    CustomerUserNameFields => ['title', 'first_name', 'last_name'],
    CustomerUserEmailUniqCheck => 1,
#    # show not own tickets in customer panel, CompanyTickets
#    CustomerUserExcludePrimaryCustomerID => 0,
#    # generate auto logins
#    AutoLoginCreation => 0,
#    AutoLoginCreationPrefix => 'auto',
#    # admin can change customer preferences
#    AdminSetPreferences => 1,
#    # cache time to live in sec. - cache database queries
#    CacheTTL => 0,
#    # just a read only source
#    ReadOnly => 1,
    Map => [

        # note: Login, Email and CustomerID needed!
        # var, frontend, storage, shown (1=always,2=lite), required, storage-type, http-link,
 readonly, http-link-target
        [ 'UserTitle',      'Title',      'title',       1, 0, 'var', '', 0 ],
        [ 'UserFirstname',  'Firstname',  'first_name',  1, 1, 'var', '', 0 ],
        [ 'UserLastname',   'Lastname',   'last_name',   1, 1, 'var', '', 0 ],
        [ 'UserLogin',      'Username',   'login',       1, 1, 'var', '', 0 ],
        [ 'UserPassword',   'Password',   'pw',          0, 0, 'var', '', 0 ],
        [ 'UserEmail',      'Email',      'email',       1, 1, 'var', '', 0 ],
        [ 'UserCustomerID', 'CustomerID', 'customer_id', 0, 1, 'var', '', 0 ],
        [ 'UserPhone',      'Phone',      'phone',       1, 0, 'var', '', 0 ],
        [ 'UserFax',        'Fax',        'fax',         1, 0, 'var', '', 0 ],
        [ 'UserMobile',     'Mobile',     'mobile',      1, 0, 'var', '', 0 ],
        [ 'UserRoom',       'Room',       'room',        1, 0, 'var', '', 0 ],
        [ 'UserStreet',     'Street',     'street',      1, 0, 'var', '', 0 ],
        [ 'UserZip',        'Zip',        'zip',         1, 0, 'var', '', 0 ],
        [ 'UserCity',       'City',       'city',        1, 0, 'var', '', 0 ],
        [ 'UserCountry',    'Country',    'country',     1, 0, 'var', '', 0 ],
        [ 'UserComment',    'Comment',    'comments',    1, 0, 'var', '', 0 ],
        [ 'ValidID',        'Valid',      'valid_id',    0, 1, 'int', '', 0 ],
    ],
    # default selections
    Selections => {
        UserTitle => {
            'Mr.' => 'Mr.',
            'Mrs.' => 'Mrs.',
        },
    },
};
```

5.4.3. Customizing the customer_user table in the OTRS DB

The last step is to add the new room number column to the customer_user table in the OTRS database (see Script below). In this column, the entries for the room numbers will be stored.

```
linux:~# mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6 to server version: 5.0.18-Debian_7-log
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use otrs;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> ALTER TABLE customer_user ADD room VARCHAR (200);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> quit
Bye
linux:~#
```

* * *

Now the new field for the room should be displayed in the Customer Information panel if filled, and in the Customer User administration screens. Also, new customers should have to insert their room number if they register a new account. If you use OTRS on Microsoft IIS, you should restart the web server to activate the changes made in `Config.pm`.

6. Ticket settings

6.1. Ticket States

6.1.1. Predefined states

OTRS allows you to change predefined ticket states and their types, or even add new ones. Two attributes are important for a state: the state name and the state type.

The default states of OTRS are: 'closed successful', 'closed unsuccessful', 'merged', 'new', 'open', 'pending auto close+', 'pending auto close-', 'pending reminder' and 'removed'.

6.1.1.1. New

Tickets are usually in this state when created from incoming e-mails.

6.1.1.2. Open

This is the default state for tickets assigned to queues and agents.

6.1.1.3. Pending reminder

After the pending time has expired, the ticket owner will receive a reminder email concerning the ticket. If the ticket is not locked, the reminder will be sent to all agents in the queue. Reminder tickets will only be sent out during business hours, and are repeatedly sent every 24 hours until the ticket state is changed by the agent. Time spent by the ticket in this status will still add towards the escalation time calculation.

6.1.1.4. Pending auto close-

Tickets in this status will be set to "Closed Unsuccessful" if the pending time has expired. Time spent by the ticket in this status will still add towards the escalation time calculation.

6.1.1.5. Pending auto close+

Tickets in this status will be set to "Closed Successful" if the pending time has expired. Time spent by the ticket in this status will still add towards the escalation time calculation.

6.1.1.6. Merged

This is the state for tickets that have been merged with other tickets.

6.1.1.7. Closed Successful

This is the end state for tickets that have been successfully resolved. Depending on your configuration, you may or may not be able to reopen closed tickets.

6.1.1.8. Closed Unsuccessful

This is the end state for tickets that have NOT been successfully resolved. Depending on your configuration, you may or may not be able to reopen closed tickets.

6.1.2. Customizing states

Every state has a name (state-name) and a type (state-type). Click on the States link on the Admin page and press the button "Add state" to create a new state. You can freely choose the name of a new state. The state types can not be changed via the web interface. The database has to be directly modified if you want to add new types or change existing names. The default state types should typically not be modified as this can yield unpredictable results. For instance, escalation calculations and the unlock feature are based on specific state types.

The name of an already existing state can be changed, or new states added through this screen. If the state "new" has been changed via the web interface, this change also has to be configured via the config file `Kernel/Config.pm` or via the SysConfig interface. The settings specified in the script below have to be modified to ensure that OTRS works with the changed state for "new".

```
    [...]
    # PostmasterDefaultState
    # (The default state of new tickets.) [default: new]
    $Self->{PostmasterDefaultState} = 'new';

    # CustomerDefaultState
    # (default state of new customer tickets)
    $Self->{CustomerDefaultState} = 'new';
    [...]
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂

If a new state type should be added, the ticket_state_type table in the OTRS database needs to be modified with a database client program, as shown in the script below.

```
linux:~# mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 23 to server version: 5.0.16-Debian_1-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use otrs;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> insert into ticket_state_type (name,comments) values ('own','Own
state type');
```

```
Query OK, 1 row affected (0.00 sec)

mysql> quit
Bye
linux:~#
```

✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂

Now it is possible to use the new state type you just created. After a state has been linked with this new state type, the OTRS configuration also has to be changed to ensure that the new state is usable. Just modify the following options via SysConfig:

Ticket -> Frontend::Agent::Ticket::ViewPhoneNew > AgentTicketPhone###StateDefault - to define the default next state for new phone tickets.

Ticket -> Frontend::Agent::Ticket::ViewPhoneNew > AgentTicketPhone###StateType - to define the available next states for new phone tickets.

Ticket -> Frontend::Agent::Ticket::ViewEmailNew > AgentTicketEmail###StateDefault - to define the default next state for new email tickets.

Ticket -> Frontend::Agent::Ticket::ViewEmailNew > AgentTicketEmail###StateType - to define the available next states for new email tickets.

Ticket -> Frontend::Agent::Ticket::ViewPhoneOutbound > AgentTicketPhoneOutbound###State - to define the default next state for new phone articles.

Ticket -> Frontend::Agent::Ticket::ViewPhoneOutbound > AgentTicketPhoneOutbound###StateType - to define the available next states for new phone articles.

Ticket:Frontend::Agent::Ticket::ViewMove:Ticket::DefaultNextMoveStateType - to define the default next state after moving a ticket.

Ticket -> Frontend::Agent::Ticket::ViewBounce > StateDefault - to define the default next state after bouncing a ticket.

Ticket -> Frontend::Agent::Ticket::ViewBounce > StateType - to define the available next states in the bounce screen.

Ticket -> Frontend::Agent::Ticket::ViewBulk > StateDefault - to define the default next state in a bulk action.

Ticket -> Frontend::Agent::Ticket::ViewBulk > StateType - to define the available next states in the bulk action screen.

Ticket -> Frontend::Agent::Ticket::ViewClose > StateDefault - to define the default next state after closing a ticket.

Ticket -> Frontend::Agent::Ticket::ViewClose > StateType - to define the available next states in the close screen.

Ticket -> Frontend::Agent::Ticket::ViewCompose > StateDefault - to define the default next state in the Compose (reply) screen.

Ticket -> Frontend::Agent::Ticket::ViewCompose > StateType - to define the available next states in the Compose (reply) screen.

Ticket -> Frontend::Agent::Ticket::ViewForward > StateDefault - to define the default next state after forwarding a ticket.

Ticket -> Frontend::Agent::Ticket::ViewForward > StateType - to define the available next states in the Forward screen.

Ticket -> Frontend::Agent::Ticket::ViewForward > StateDefault - to define the default next state of a ticket in the free text screen.

Ticket -> Frontend::Agent::Ticket::ViewForward > StateType - to define the available next states in the free text screen.

Ticket -> Core::PostMaster > PostmasterDefaultState - to define the state of tickets created from emails.

Ticket -> Core::PostMaster > PostmasterFollowUpState - to define the state of tickets after a follow-up has been received.

Ticket -> Core::PostMaster > PostmasterFollowUpStateClosed - to define the state of tickets after a follow-up has been received on an already closed ticket.

Ticket -> Core::Ticket > ViewableStateType - to define the state types that are displayed at various places in the system, for example in the Queueview.

Ticket -> Core::Ticket > UnlockStateType - to define the state types for unlocked tickets.

Ticket -> Core::Ticket > PendingReminderStateType - to define the state type for reminder tickets.

Ticket -> Core::Ticket > PendingAutoStateType - to define the state type for Pending Auto tickets.

Ticket -> Core::Ticket > StateAfterPending - to define the state a ticket is set to after the Pending Auto timer of the configured state has expired.

6.2. Ticket priorities

OTRS comes with five default priority levels that can be modified via the "Priorities" link on the Admin page. When creating a customized list of priorities, please keep in mind that they are sorted alphabetically in the priority selection box in the user interface. Also, OTRS orders tickets by internal database IDs in the QueueView.

Note

As with other OTRS entities, priorities may not be deleted, only deactivated by setting the Valid option to ✂✂✂✂✂✂✂ or ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂.

Important

If a new priority was added or if an existing one was changed, you might also want to modify some values in SysConfig:

- Ticket:Core::Postmaster::PostmasterDefaultPriority - defines the default priority for all incoming emails.
- Ticket:Frontend::Agent:Ticket::ViewPhoneNew:Priority - defines the default priority in the New Phone Ticket screen for agents.
- Ticket:Frontend::Agent:Ticket::ViewEmailNew:Priority - defines the default priority in the New Email Ticket screen for agents.
- Ticket:Frontend::Customer:Ticket::ViewNew:PriorityDefault - defines the default priority in the New Ticket screen in the Customer frontend.

6.3. Ticket Responsibility & Ticket Watching

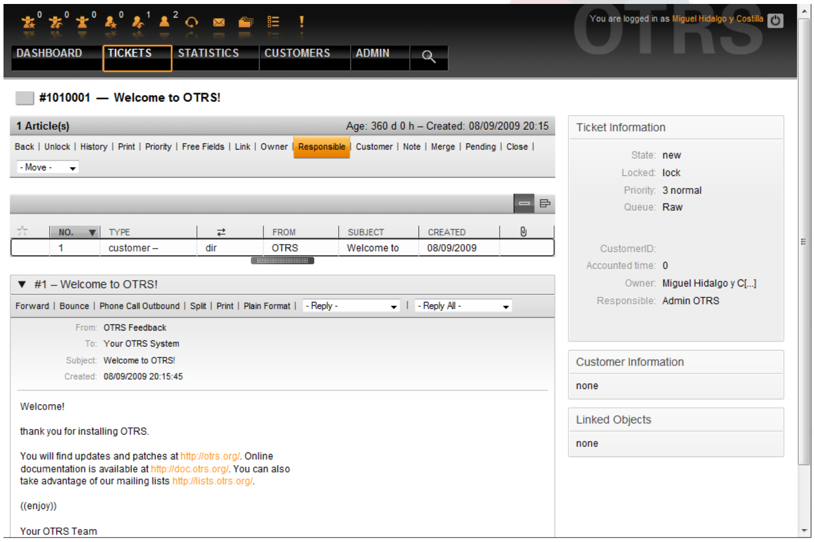
From OTRS 2.1 on, it is possible to assign a person as being responsible for a ticket, in addition to its owner. Moreover, all activities connected with the ticket can be watched by someone other than the ticket owner. These two functionalities are implemented with the TicketResponsible and TicketWatcher features, and facilitate the assignment of tasks and working within hierarchical team structures.

6.3.1. Ticket Responsibility

The ticket responsiblility feature facilitates the complete processing of a ticket by an agent other than the ticket owner. Thus an agent who has locked a ticket can pass it on to another agent, who is not the ticket owner, in order for the second to respond to a customer request. After the request has been dealt with, the first agent can withdraw the ticket responsibility from the second agent.

With the configuration parameter Ticket::Responsible, the ticket responsibility feature can be activated. This will cause 3 new links to appear in the ticket activities menu of a zoomed ticket in the agent interface.

Ticket responsibility can be assigned by calling up the ticket content and clicking on the "Responsible" link in the ticket activities menu of a zoomed ticket in the agent interface (see the Figure below).


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Back | Unlock | History | Print | Priority | Free Fields | Link | Owner | Responsible | Customer | Note | Merge | Pending | Close |
- Move -
Forward | Bounce | Phone Call Outbound | Split | Print | Plain Format | - Reply - | - Reply All -
Ticket Information
Customer Information
Linked Objects

You are logged in as Miguel Hidalgo y Costilla

#1010001 — Welcome to OTRS!

1 Article(s) Age: 360 d 0 h – Created: 08/09/2009 20:15

NO.	TYPE	⇄	FROM	SUBJECT	CREATED
1	customer –	dir	OTRS	Welcome to	08/09/2009

#1 – Welcome to OTRS!

From: OTRS Feedback
To: Your OTRS System
Subject: Welcome to OTRS!
Created: 08/09/2009 20:15:45

Welcome!

thank you for installing OTRS.

You will find updates and patches at http://otrs.org/. Online documentation is available at http://doc.otrs.org/. You can also take advantage of our mailing lists http://lists.otrs.org/.

((enjoy))

Your OTRS Team

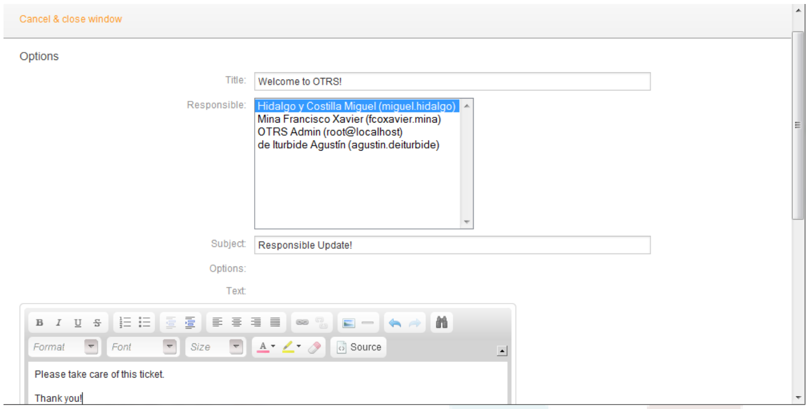
State: new
Locked: lock
Priority: 3 normal
Queue: Raw
CustomerID:
Accounted time: 0
Owner: Miguel Hidalgo y C[...]
Responsible: Admin OTRS

none

none

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂

After clicking on "Responsible", a pop-up dialog to change the responsibility of that ticket will open (see Figure below). This dialog can also be used to send a message to the new responsible agent.


Cancel & close window
Options
Format
Font
Size
Source

Title: Welcome to OTRS!

Responsible:
Hidalgo y Costilla Miguel (miguel.hidalgo)
Mina Francisco Xavier (fcoxavier.mina)
OTRS Admin (root@localhost)
de Iturbide Agustín (agustin.deiturbide)

Subject: Responsible Update!

Options:

Text:

Please take care of this ticket.

Thank you!

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂

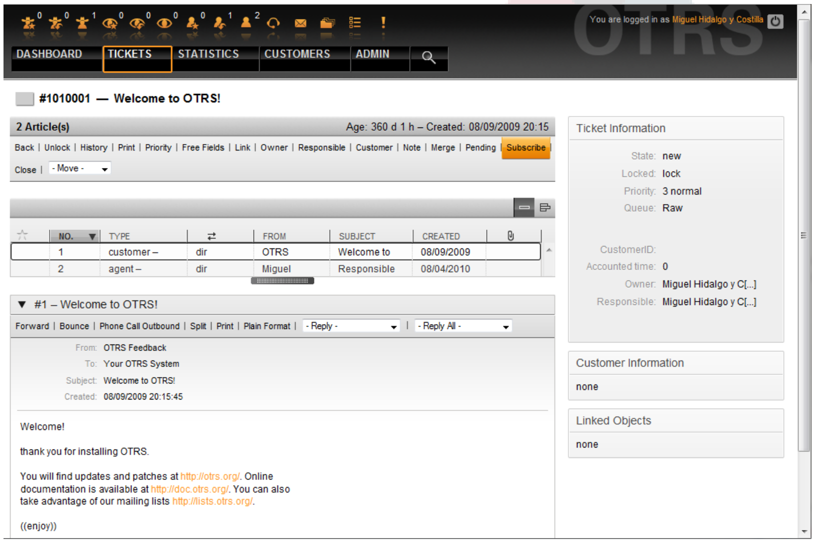
The list of all tickets for which an agent is responsible, can be accessed through the Responsible view of the OTRS agent interface, as soon as the ticket responsibility feature gets activated.

6.3.2. Ticket watching

From OTRS 2.1 on, select agents such as supervisors can watch certain tickets within the system without processing them, by using the TicketWatcher feature.

The TicketWatcher feature can be activated with the configuration parameter Ticket::Watcher which adds new links to your actions toolbar. Using Ticket::WatcherGroup, one or more user groups with permission to watch tickets can also be defined.

In order to watch a ticket, go to its zoomed view and click on the "Subscribe" link in the ticket activities menu (see Figure below).


0
0
1
0
0
0
0
1
2
OTRS
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Back | Unlock | History | Print | Priority | Free Fields | Link | Owner | Responsible | Customer | Note | Merge | Pending | Subscribe |
Close | - Move -
Forward | Bounce | Phone Call Outbound | Split | Print | Plain Format | - Reply - | - Reply All -
Ticket Information
Customer Information
Linked Objects

You are logged in as Miguel Hidalgo y Costilla

#1010001 — Welcome to OTRS!

2 Article(s) Age: 360 d 1 h – Created: 08/09/2009 20:15

NO.	TYPE	⇄	FROM	SUBJECT	CREATED
1	customer –	dir	OTRS	Welcome to	08/09/2009
2	agent –	dir	Miguel	Responsible	08/04/2010

▼ #1 – Welcome to OTRS!

From: OTRS Feedback
To: Your OTRS System
Subject: Welcome to OTRS!
Created: 08/09/2009 20:15:45

Welcome!

thank you for installing OTRS.

You will find updates and patches at http://otrs.org/. Online documentation is available at http://doc.otrs.org/. You can also take advantage of our mailing lists http://lists.otrs.org/.

((enjoy))

State: new
Locked: lock
Priority: 3 normal
Queue: Raw

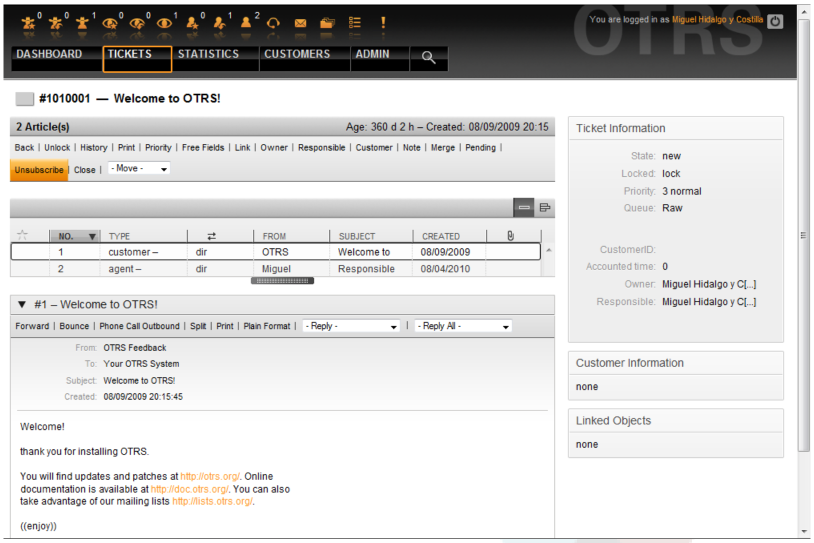
CustomerID:
Accounted time: 0
Owner: Miguel Hidalgo y C[...]
Responsible: Miguel Hidalgo y C[...]

Customer Information: none

Linked Objects: none

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂

If you no longer want to watch a specific ticket, go to its zoomed view and click on the "Unsubscribe" link in the ticket activities menu (see Figure below).


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Back | Unlock | History | Print | Priority | Free Fields | Link | Owner | Responsible | Customer | Note | Merge | Pending |
Unsubscribe | Close | - Move -
Forward | Bounce | Phone Call Outbound | Split | Print | Plain Format | - Reply - | - Reply All -
Ticket Information
Customer Information
Linked Objects

#1010001 — Welcome to OTRS!

2 Article(s) Age: 360 d 2 h – Created: 08/09/2009 20:15

NO.	TYPE	⇄	FROM	SUBJECT	CREATED
1	customer –	dir	OTRS	Welcome to	08/09/2009
2	agent –	dir	Miguel	Responsible	08/04/2010

#1 – Welcome to OTRS!

From: OTRS Feedback
To: Your OTRS System
Subject: Welcome to OTRS!
Created: 08/09/2009 20:15:45

Welcome!

thank you for installing OTRS.

You will find updates and patches at http://otrs.org/. Online documentation is available at http://doc.otrs.org/. You can also take advantage of our mailing lists http://lists.otrs.org/.

((enjoy))

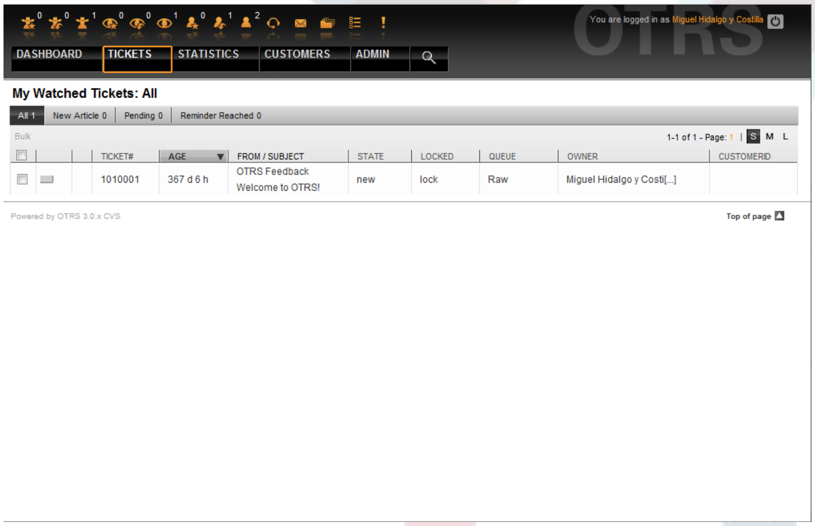
State: new
Locked: lock
Priority: 3 normal
Queue: Raw
CustomerID:
Accounted time: 0
Owner: Miguel Hidalgo y C[...]
Responsible: Miguel Hidalgo y C[...]

Customer Information: none

Linked Objects: none

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂
✂✂✂✂✂✂ ✂✂✂✂✂

The list of all watched tickets can be accessed through the Watched view of the OTRS agent interface (see Figure below), as soon as the ticket watcher feature gets activated.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
All 1
New Article 0
Pending 0
Reminder Reached 0
Bulk
1-1 of 1 - Page: 1
S
M
L
Powered by OTRS 3.0.x CVS
Top of page

My Watched Tickets: All

☐		TICKET#	AGE	FROM / SUBJECT	STATE	LOCKED	QUEUE	OWNER	CUSTOMERID
☐		1010001	367 d 6 h	OTRS Feedback Welcome to OTRS!	new	lock	Raw	Miguel Hidalgo y Costi[...]	

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂

7. Time related functions

7.1. Setting up business hours, holidays and time zones

Some functions in OTRS, like escalations and automatic unlocking of tickets, depend on a proper configuration of business hours, time zones and holidays. You can define these via the SysConfig interface, in Framework > Core::Time. You can also specify different sets of business hours, holidays and time zones as separate 'Calendars' in Framework > Core::Time::Calendar1 through Framework > Core::Time::Calendar9. Calendars can be defined by queue settings, or on SLA levels. This means that, for example, you can specify a calendar with 5 x 8 business hours for your 'standard' SLA, but create a separate calendar with 7 x 24 support for your 'gold' SLA; as well as set a calendar for your 'Support-USA' queue with a different time window than your 'Support-Japan' queue. OTRS can handle up to 99 different calendars.

7.1.1. Business Hours

Set up the working hours for your system in SysConfig Framework > Core::Time::TimeWorkingHours, or for your specific calendar in the calendar's configuration. OTRS can handle a granularity of one hour. Checking the marks in the boxes 8, 9, 10 ... 18 corresponds with business hours of 8 AM - 6 PM.

Only during business hours can tickets escalate, notifications for escalated and pending tickets be sent, and tickets be unlocked.

7.1.2. Fixed date holidays

Holidays that are on a fixed date every year, such as New Year's Day or the Fourth of July, can be specified in TimeVacationDays, or in the corresponding section for the calendars 1-9.

Tickets will not escalate nor get unlocked on dates defined in TimeVacationDays.

Note

By default, OTRS ships with the ✃✃✃✃✃✃ holidays installed.

7.1.3. TimeVacationDaysOneTime

Holidays such as Easter that do not have a yearly fixed date but instead vary each year, can be specified in TimeVacationDaysOneTime.

Tickets will not escalate and will not be unlocked on dates defined in TimeVacationDaysOneTime.

Note

OTRS does not ship with any One-Time holidays pre-installed. This means that you need to add holidays, such as Easter or Thanksgiving, to the system when configuring OTRS.

7.2. Automated Unlocking

Locked tickets can be automatically unlocked by the system. This feature might be useful if, for example, an agent has locked tickets that need to be processed, but he can't work on them for some reason, say because he is out of the office on an emergency. The automated unlock feature unlocks tickets after a given time to ensure that no locked tickets will be forgotten, thereby allowing other agents to process them.

The amount of time before a ticket is unlocked can be specified in the queue settings for every queue. The module `bin/otrs.UnlockTickets.pl`, which is executed periodically as a cron job, performs the automated unlocking of tickets.

Notifications on unlocked tickets are sent out only to those agents that have the queue with the unlocked tickets set in "My queues", and that have activated the notification on unlocked tickets in their personal preferences.

Tickets will be unlocked if all of the following conditions are met:

- There is an ✂✂✂✂✂✂ ✂✂✂✂✂✂✂ defined for the queue the ticket is in.
- The ticket is set to ✂✂✂✂✂✂.
- The ticket state is ✂✂✂✂.

The unlock timer will be reset if an agent adds a new external article to the ticket. It can be of any of the following types: ✂✂✂✂✂✂✂✂✂✂✂✂✂✂, ✂✂✂✂✂, ✂✂✂, ✂✂✂, or ✂✂✂✂✂ ✂✂✂✂✂✂✂✂.

Also, if the last article in the ticket is created by an agent, and a customer adds another one, either via web or email response, the unlock timer will be reset.

The last event that will reset the unlock timer is when the ticket is assigned to another agent.

8. Customizing the PDF output

This section handles the configurable options for PDF output in OTRS.

If you use the Print action from anywhere within the OTRS interface, it will generate a formatted PDF file. You can deactivate this by modifying the configuration parameter PDF to create HTML output instead.

You can adjust the look of the files generated by OTRS by creating your own logo and adding it to PDF::LogoFile. You can use PDF::PageSize to define the standard page size of the generated pdf file (DIN-A4 or Letter), and also PDF::MaxPage to specify the maximum number of pages for a pdf file, which is useful if a user generates a huge output file by mistake.

The Perl CPAN modules PDF::API2 and Compress::Zlib must be installed for the generation of pdf files. In many distributions they are available as packages and can be easily installed, using the respective package manager. In case this is not possible, they have to be installed with CPAN. For further information about installing Perl modules, please refer to the "Installation of Perl modules" section.

9. Stats module

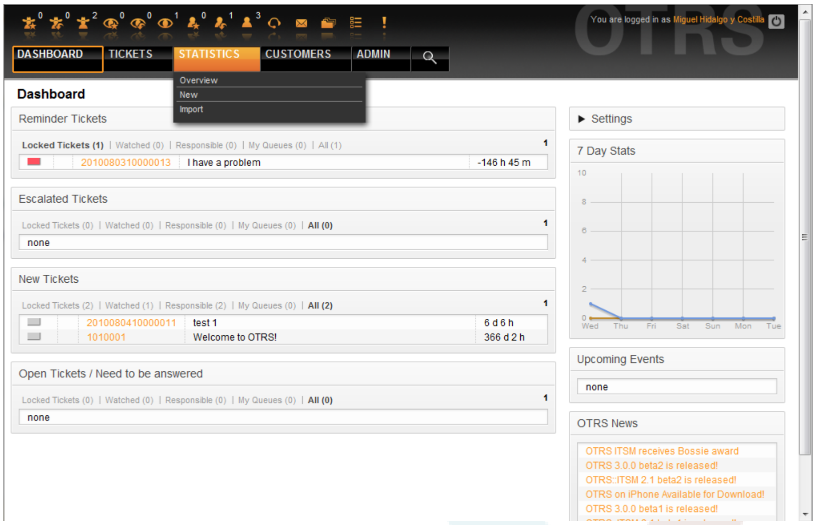
The OTRS stats module holds features to track operational statistics and generates custom reports associated with OTRS usage. The OTRS system uses the term "stat" generically to refer to a report presenting various indicators.

Proper configuration of the OTRS stats module is associated with a multitude of requirements and considerations. These include the various OTRS modules to be evaluated, user permission settings, indicators to be calculated and their complexity levels, ease of configuration of the stats module, speed and efficiency of calculations, and support of a rich set of output variants.

Statistical elements, i.e. files which supplement the functionality of the stats module for specific requirements, can be integrated for calculating complex statistics.

9.1. Handling of the module by the agent

When signed on as an agent, the navigation bar displays the link "Statistics", with various submenu options, as shown in Figure.



You are logged in as Miguel Hidalgo y Costilla

Dashboard

Reminder Tickets

Locked Tickets (1) | Watched (0) | Responsible (0) | My Queues (0) | All (1)

🟥	2010080310000013	I have a problem	-146 h 45 m

Escalated Tickets

Locked Tickets (0) | Watched (0) | Responsible (0) | My Queues (0) | All (0)

none

New Tickets

Locked Tickets (2) | Watched (1) | Responsible (2) | My Queues (0) | All (2)

▭	2010080410000011	test 1	6 d 6 h
▭	1010001	Welcome to OTRS!	366 d 2 h

Open Tickets / Need to be answered

Locked Tickets (0) | Watched (0) | Responsible (0) | My Queues (0) | All (0)

none

Upcoming Events

none

OTRS News

OTRS ITSM receives Bossie award
OTRS 3.0.0 beta2 is released!
OTRS::ITSM 2.1 beta2 is released!
OTRS on iPhone Available for Download!
OTRS 3.0.0 beta1 is released!

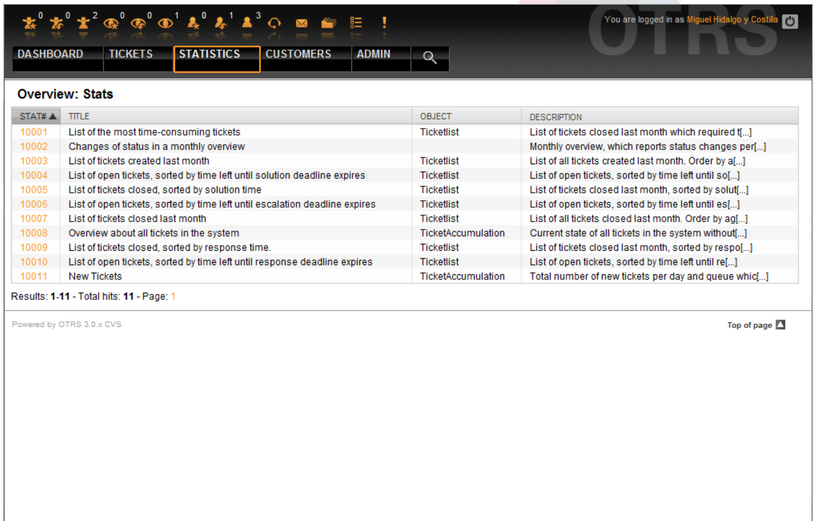
✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂

The different options provided in the statistics menu are:

- ✂✂✂✂✂✂✂✂. Presents a list of different pre-configured reports.
- ✂✂✂. Requires rw rights.
- ✂✂✂✂✂✂. Requires rw rights.

9.1.1. Overview

Selecting the "Statistics" link in the navigation bar, and then the submenu link "Overview", calls up the Overview screen. The Overview screen presents a list of all pre-configured reports the agent can use (see Figure below).



You are logged in as Miguel Hidalgo y Costilla

Overview: Stats

STAT# ▲	TITLE	OBJECT	DESCRIPTION
10001	List of the most time-consuming tickets	Ticketlist	List of tickets closed last month which required t[...]
10002	Changes of status in a monthly overview		Monthly overview, which reports status changes per[...]
10003	List of tickets created last month	Ticketlist	List of all tickets created last month. Order by a[...]
10004	List of open tickets, sorted by time left until solution deadline expires	Ticketlist	List of open tickets, sorted by time left until so[...]
10005	List of tickets closed, sorted by solution time	Ticketlist	List of tickets closed last month, sorted by solut[...]
10006	List of open tickets, sorted by time left until escalation deadline expires	Ticketlist	List of open tickets, sorted by time left until es[...]
10007	List of tickets closed last month	Ticketlist	List of all tickets closed last month. Order by ag[...]
10008	Overview about all tickets in the system	TicketAccumulation	Current state of all tickets in the system without[...]
10009	List of tickets closed, sorted by response time.	Ticketlist	List of tickets closed last month, sorted by respo[...]
10010	List of open tickets, sorted by time left until response deadline expires	Ticketlist	List of open tickets, sorted by time left until re[...]
10011	New Tickets	TicketAccumulation	Total number of new tickets per day and queue whic[...]

Results: **1-11** - Total hits: **11** - Page: 1

Powered by OTRS 3.0.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂

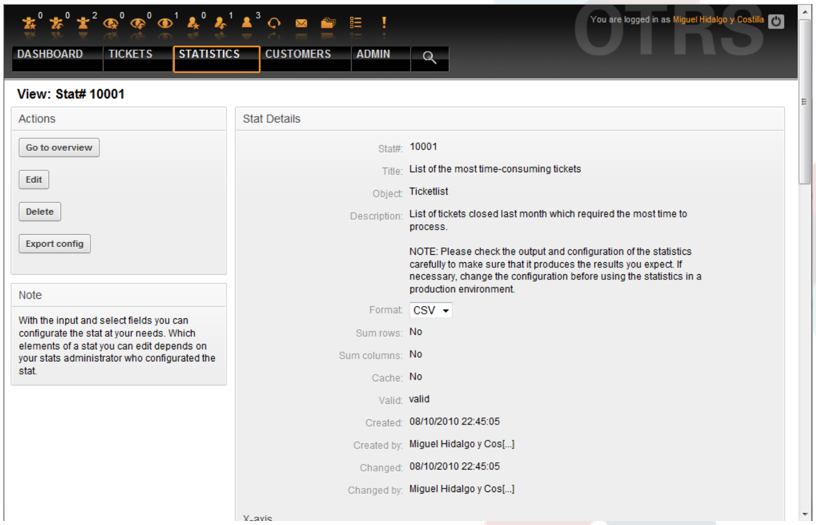
The following information is provided for each of the standard reports listed in the Overview:

- ✂✂✂✂✂. Unique report number.
- ✂✂✂✂✂. Title of the report.
- ✂✂✂✂✂✂. Object used for generating the statistic. In the case of a static statistic, no object is displayed as no dynamic object is used for its generation.
- ✂✂✂✂✂✂✂✂✂✂✂. A brief description of the report.

When the stats module is installed, it comes preloaded with a few sample reports imported into the system. These are shown as a list on the Overview page. If the Overview list extends to more than a single page, the agent can browse through the different pages. The list of reports can be sorted as desired, by clicking the desired column header in the list. To generate a particular report, click on the stat number associated with the report in the Overiew list. This brings up the "View" interface for the report.

9.1.2. Generate and view reports

The view user interface provides the stat's configuration settings (see Figure below).



View: Stat# 10001

With the input and select fields you can configure the stat at your needs. Which elements of a stat you can edit depends on your stats administrator who configurated the stat.

Stat#: 10001
Title: List of the most time-consuming tickets
Object: Ticketlist
Description: List of tickets closed last month which required the most time to process.

NOTE: Please check the output and configuration of the statistics carefully to make sure that it produces the results you expect. If necessary, change the configuration before using the statistics in a production environment.

Format: CSV
Sum rows: No
Sum columns: No
Cache: No
Valid: valid
Created: 08/10/2010 22:45:05
Created by: Miguel Hidalgo y Cos[...]
Changed: 08/10/2010 22:45:05
Changed by: Miguel Hidalgo y Cos[...]

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂

Configuration settings for a particular report can be set within the range of options in the View screen. Either the report creator or any others with the appropriate permissions can make the settings.

The page shows the following:

- Possible actions:
  - ✂✂ ✂✂ ✂✂✂✂✂✂✂✂. Link back to the Overview list of reports.
  - ✂✂✂✂. Edit the current report structure (rw rights required).
  - ✂✂✂✂✂✂. Delete the current report (rw rights required).
  - ✂✂✂✂✂✂ ✂✂✂✂✂✂. Export a report configuration, via file download (rw rights required).

Usage: Export and Import functions allow for the convenient creation and testing of reports on test systems and subsequent easy integration into the production system.

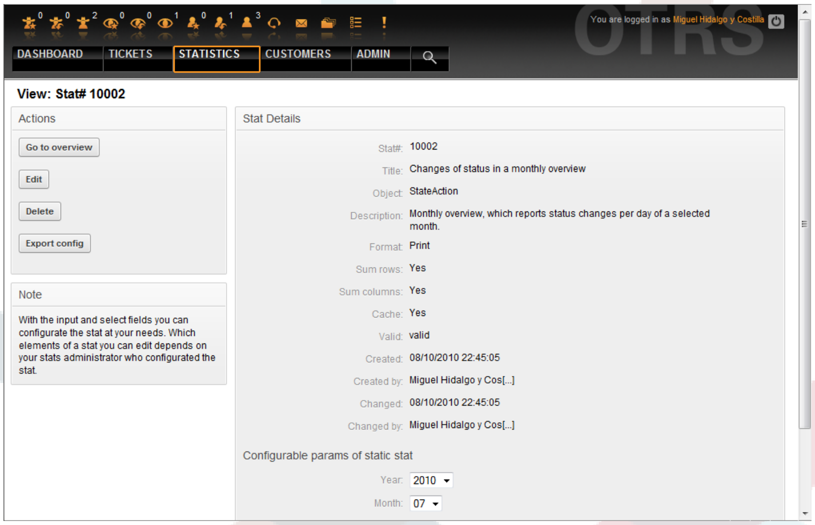
- Report details:
  - ✂✂✂✂✂. Number of the report.
  - ✂✂✂✂✂. Title of the report.
  - ✂✂✂✂✂✂. Object used for generating the report.
  - ✂✂✂✂✂✂✂✂✂✂✂. Description on the report's purpose.
  - ✂✂✂✂✂✂. Report output format which, depending on the configuration, can be any of the following output formats:
    - CSV.
    - Print.
    - Graph-lines.
    - Graph-bars.
    - Graph-hbars.
    - Graph-points.
    - Graph-lines-points.
    - Graph-area.
    - Graph-pie.
  - ✂✂✂✂✂✂✂✂✂. Size in pixels for the graphic / chart. This option is only given when the report configuration allows a chart. All generally usable graphic sizes are configured by the OTRS administrator in SysConfig. The agent can then pre-select all relevant formats, while configuring the report.
  - ✂✂✂ ✂✂✂✂. Indicates whether the report is amended by a column, whose cells state the sums of the respective rows.
  - ✂✂✂ ✂✂✂✂✂✂✂. Indicates whether the report is amended by a row, whose cells state the sum of the respective columns.
  - ✂✂✂✂✂. Indicates whether the generated report is cached in the filesystem.
  - ✂✂✂✂✂. This can be set to "invalid" if a report must not be run temporarily for any reason. The "Start" button in the bottom of the right panel is then no longer displayed. The report can no longer be generated.
  - ✂✂✂✂✂✂✂. Creation time of the report.
  - ✂✂✂✂✂✂✂ ✂✂. Name of the agent who created the report.
  - ✂✂✂✂✂✂✂. Time when the report was last modified.

- [illegible]. Name of the agent who altered the report last.

- [illegible]. Using this function, the agent can switch the x and y axes (only when activated by the OTRS administrator).

- The general information is followed by information about the report itself. There are two different report (or stat) views:

  - [illegible]. Static report generators can be integrated into the stats module (see Figure below).



View: Stat# 10002

With the input and select fields you can configurate the stat at your needs. Which elements of a stat you can edit depends on your stats administrator who configurated the stat.

Stat#: 10002
Title: Changes of status in a monthly overview
Object: StateAction
Description: Monthly overview, which reports status changes per day of a selected month.
Format: Print
Sum rows: Yes
Sum columns: Yes
Cache: Yes
Valid: valid
Created: 08/10/2010 22:45:05
Created by: Miguel Hidalgo y Cos[...]
Changed: 08/10/2010 22:45:05
Changed by: Miguel Hidalgo y Cos[...]

Configurable params of static stat

Year: 2010
Month: 07

[illegible]

  - [illegible] (see Figure above). They can be displayed in two different ways:

    - [illegible]. The originator of the report has no permission for modifying this fields.

    - [illegible]. The configuration settings of such reports can be changed by the agent.

Pressing the "Start" button (at the bottom of the screen) is the last step to generate the report. There are two possible reasons for this button to not be displayed:

1. The report was set to invalid and thus, deactivated.
2. The report was not configured cleanly and is, therefore, not yet executable. In this case, the necessary information can be found in the OTRS notification section (below the navigation bar).

If the settings on the View page are incorrect, this page is shown again after the "Start" button was pushed, and information about which input was incorrect is provided in the notification section.

9.1.3. Edit / New

Agents with write rights can edit an existing report configuration by calling up the edit user interface of the stats module. Alternately, they may create a new report. The associated screens can be reached in the following manner:

1. Edit: Via the "Edit" button in the stat view.
2. New: Via the "New" link in the Statistics menu from the navigation bar, or the "Add" button from the Overview page.

The stats are edited with a wizard in four steps:

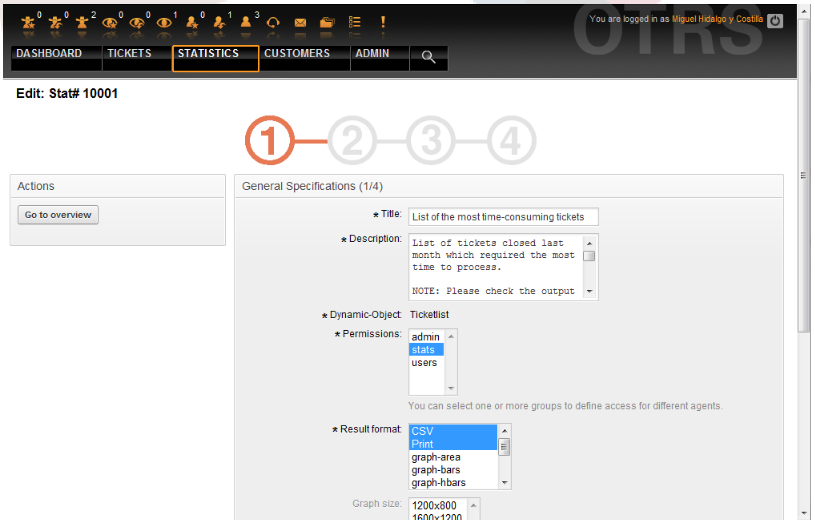
1. General specifications.
2. Definition of the element for the X-axis.
3. Specification of the value series.
4. Selecting the restrictions to limit the report.

Steps 2 through 4 are only needed for the generation of reports with dynamic stats. For a static stat, only the general information (point 1) is required.

Information about how to handle the page is provided on each of these screens, below the Actions panel in a Hints panel.

If incorrect inputs are entered, the previously processed user interface is displayed again and with information about the incorrect input. This information can be found in the OTRS notification section. The next input user interface is only displayed after the current form has been filled out correctly.

1. ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂. It is the first page of the Edit wizard (see Figure below).



Edit: Stat# 10001

General Specifications (1/4)

* Title: List of the most time-consuming tickets

* Description: List of tickets closed last month which required the most time to process.

NOTE: Please check the output

* Dynamic-Object: Ticketlist

* Permissions: admin, stats, users

You can select one or more groups to define access for different agents.

* Result format: CSV, Print, graph-area, graph-bars, graph-hbars

Graph size: 1200x800, 1600x1200

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂ ✂✂✂✂✂✂✂

In the screen showed in Figure, there are a great number of common specifications and settings that can be edited:

- [illegible]. Should reflect the stat's purpose in a concise manner.

- [illegible]. More descriptive information about the report definition, type of configuration parameters, etc.

- [illegible] [illegible]. If the OTRS installation provides various dynamic objects, one of them can be chosen. The objects meet the requirements of the particular modules.

- [illegible] [illegible]. Usually this selection is not shown, as only static files which are not yet assigned to any reports are displayed. If "Static file" is displayed, however, it is important to tick the option field and select a generation mode (dynamic with a dynamic object or static with a file). If a static file is selected, the input user interfaces 2 through 4 are not shown as the static file contains all required configuration settings.

- [illegible] [illegible]. Facilitate a restriction of the groups (and therefore, agents) who can later view and generate the preconfigured reports. Thus the various reports can be allocated to the different departments and work groups who need them. It is possible to allocate one report to various groups.

  Example 1: The "stats" group was selected. The report is viewable for all users having at least ro rights for the "stats" group. This access is available by default.

  Example 2: A group named "sales" was selected. All users with ro rights for the "sales" group can see the stat in the view mode and generate it. However, the report will not be available for viewing by other users.

- [illegible]. Output format of the stat: Depending on the configuration, one or more of the following formats can be chosen:

  - CSV.
  - Print.
  - graph-lines.
  - graph-bars.
  - graph-hbars.
  - graph-points.
  - graph-lines-points.
  - graph-area.
  - graph-pie.

- [illegible]. Select the chart size in pixels. This selection is only necessary if a graphical output format has been chosen under "Format". All graphic sizes that can generally be used are defined by the OTRS administrator in SysConfig. When configuring the report, the agent can pre-select all relevant formats.

- [illegible] [illegible]. Indicates whether the report is amended by a column, whose cells contain the sum of the respective row.

- ✂✂✂ ✂✂✂✂✂✂✂. Indicates whether the report is amended by a row, whose cells contain the sum of the respective column.

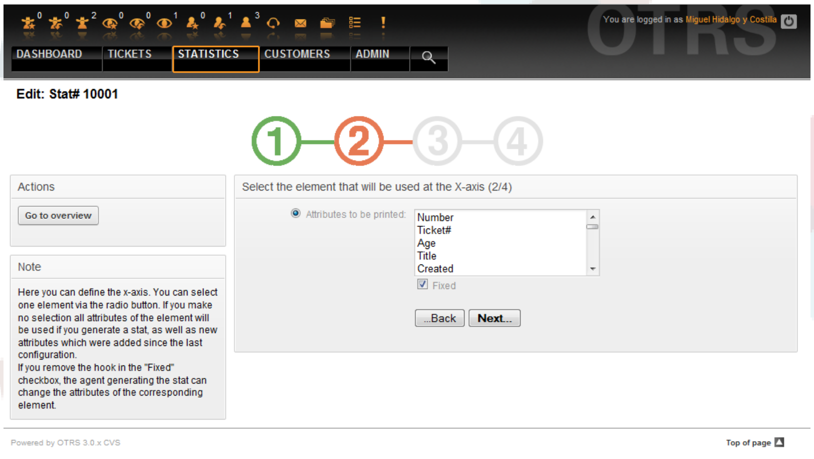
- ✂✂✂✂✂. Specifies if the generated report should be cached in the filesystem. This saves computing power and time if the report is called up again, but it should only be used if the report's content is no longer changing.

  Caching is automatically prevented if the report contains no time designation values, or if a time designation value points to the future.

  If a cached report is edited, all cached data is deleted.

- ✂✂✂✂✂. This can be set to "invalid" if a pre-configured report must not be run temporarily for any reason. The "Start" button in the bottom of the right panel is then no longer displayed. The report can no longer be generated.

2. ✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂ ✂✂✂✂✂✂. It is the configuration of the element used for the depiction of the X-axis or, if tables are used, of the column name applied to the X-axis (see Figure).


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
1
2
3
4
Actions
Go to overview
Note
Attributes to be printed:
Number
Ticket#
Age
Title
Created
Fixed
...Back
Next...
Powered by OTRS 3.0.x CVS
Top of page

Edit: Stat# 10001

Select the element that will be used at the X-axis (2/4)

Here you can define the x-axis. You can select one element via the radio button. If you make no selection all attributes of the element will be used if you generate a stat, as well as new attributes which were added since the last configuration.
If you remove the hook in the "Fixed" checkbox, the agent generating the stat can change the attributes of the corresponding element.

● Attributes to be printed: Number, Ticket#, Age, Title, Created

☑ Fixed

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂

First of all, an element is selected using the option field. Then two or more attributes of the element must be selected. If no attributes are selected, all attributes are used including those added after the configuration of the report.

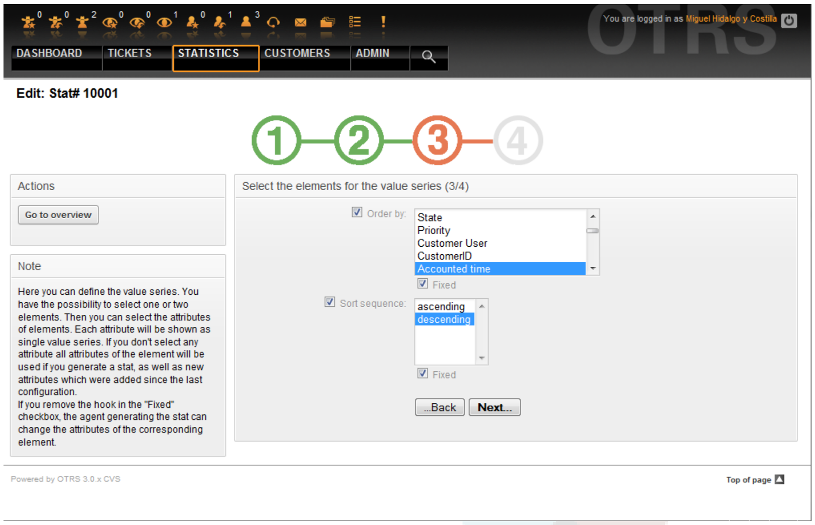
If the "Fixed" setting is disabled, the agent generating the report can change the attributes of the respective element in the "View" user interface.

Time elements are different as time period and scale have to be stated. Type and number of elements result from the used dynamic object and vary depending on it.

If all input is correct, the "Next" button leads to the "Value series" form. It is also possible to go back to editing earlier sections.

3. ✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂.

In the third step of the report configuration, the value series are defined (see Figure below). They will later form the individual graphs or the various series within a tabular view.



Edit: Stat# 10001

Select the elements for the value series (3/4)

☑ Order by: State, Priority, Customer User, CustomerID, **Accounted time** (selected)

☑ Fixed

☑ Sort sequence: ascending, **descending** (selected)

☑ Fixed

Note

Here you can define the value series. You have the possibility to select one or two elements. Then you can select the attributes of elements. Each attribute will be shown as single value series. If you don't select any attribute all attributes of the element will be used if you generate a stat, as well as new attributes which were added since the last configuration.
If you remove the hook in the "Fixed" checkbox, the agent generating the stat can change the attributes of the corresponding element.

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂

If an element is selected, each chosen attribute will correspond to a value series (see the Example 19-1 below).

Example 4.18. Definition of a value series - one element

Element Queue:

- Value series 1 = Raw
- Value series 2 = Junk
- ....

If two elements are selected, each selected attribute of the first element is combined with an attribute of the second element to form a value series (see Example 19-2 below).

Example 4.19. Definition of a value series - two elements

Element 1 queue, Element 2 status:

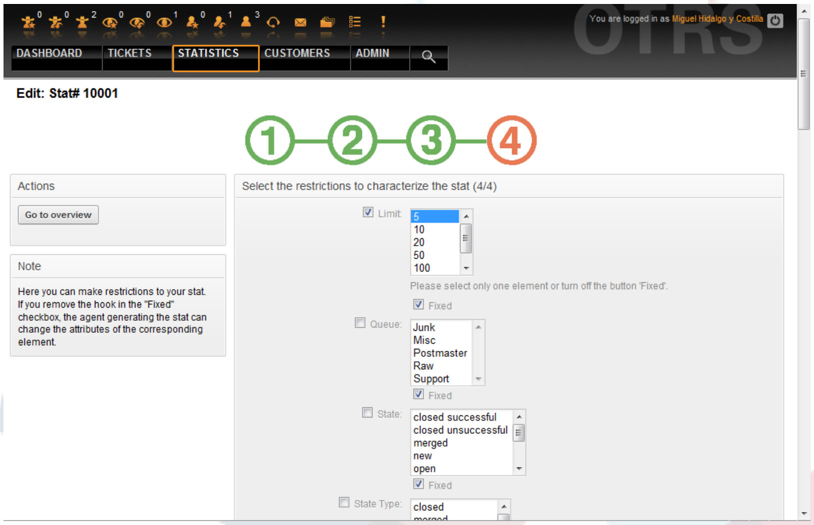
- Value chain 1 = Raw - open
- Value series 2 = Raw - successfully closed
- Value series 3 = Junk - open
- Value series 4 = Junk - successfully closed

Selection of three or more elements is not allowed.

Additionally the same conditions apply to the selection of the attributes and the "Fixed" checkbox as to the "X-axis" selection:

- If no attributes of the element are selected, all attributes are used, including those added after the configuration of the report.
- If the "Fixed" setting is disabled, the agent generating the report can change the attributes of the respective element.

4. ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂✂. This is the fourth and final step of the configuration (see Figure below). The restrictions serve to limit the results to the selected criteria. In many cases, no restrictions at all may be set up.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
1
2
3
4
Actions
Go to overview

Edit: Stat# 10001

Note

Here you can make restrictions to your stat. If you remove the hook in the "Fixed" checkbox, the agent generating the stat can change the attributes of the corresponding element.

Select the restrictions to characterize the stat (4/4)

☑ Limit: 5, 10, 20, 50, 100

Please select only one element or turn off the button 'Fixed'.

☑ Fixed

☐ Queue: Junk, Misc, Postmaster, Raw, Support

☑ Fixed

☐ State: closed successful, closed unsuccessful, merged, new, open

☑ Fixed

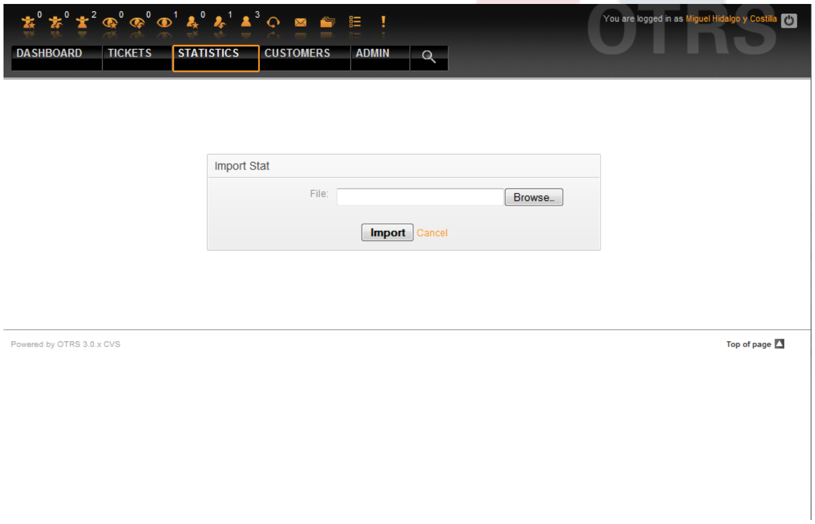
☐ State Type: closed, merged

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂

After all the restrictions are set up, the configuration of the report is completed by pressing the "Finish" button.

9.1.4. Import

The Import user interface (see Figure below) can be accessed by choosing from the navigation bar, the link "Statistics", then "Import". Alternately, pressing the Import button on the Overview screen achieves the same result. "rw" rights to the report are required.


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Miguel Hidalgo y Costilla
Browse...
Import
Cancel
Powered by OTRS 3.0.x CVS
Top of page

Import Stat

File:

✁✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁

Facilitates the import of reports and is, when combined with the export function of the module, a very handy functionality. Stats can be created and tested conveniently on test systems, then imported into the production system.

The import is effected by an easy file upload. The "View" user interface of the imported report is opened automatically afterwards.

9.2. Administration of the stats module by the OTRS administrator

This section provides information about the tasks and responsibilities of the OTRS administrator dealing with the statistics module.

9.2.1. Permission settings, Groups and Queues

No new queues and/or groups are created when the stats module is installed.

The default configuration of the module registration gives all agents with "stats" group permissions access to the stats module.

Access according to permission settings:

- ✁✁. Allows configuring statistics and reports.
- ✁✁. Permits generating pre-configured statistics and reports.

The OTRS administrator decides whether agents with the permission to generate pre-configured reports are allocated ro rights in the "stats" group, or if their respective groups are added in the module registration in SysConfig.

9.2.2. SysConfig

The SysConfig groups Framework:Core::Stats, Framework:Core::Stats::Graph and Framework:Frontend::Agent::Stats contain all configuration parameters for the basic set-up of the statistics module. Moreover, the configuration parameter $Self->{'Frontend::Module'}->{'AgentStats'} controls the arrangement and registration of the modules and icons within the statistics module.

9.3. Administration of the stats module by the system administrator

Generally, no system administrator is needed for the operation, configuration and maintenance of the statistics module. However, a little background information for the system administrator is given at this point.

> Note
>
> File paths refer to subdirectories of the OTRS home directory (in most cases`/opt/otrs`).

9.3.1. Data base table

All report configurations are implemented and administrated in XML, and therefore stored in the database table "xml_storage". Other modules whose content is presented in xml format use this table as well.

9.3.2. List of all files

The following files are necessary for the stats module to work accurately:

- `Kernel/System/Stats.pm`
- `Kernel/Modules/AgentStats.pm`
- `Kernel/System/CSV.pm`
- `Kernel/Output/HTML/Standard/AgentStatsOverview.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsDelete.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsEditSpecification.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsEditRestrictions.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsEditXaxis.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsEditValueSeries.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsImport.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsPrint.dtl`
- `Kernel/Output/HTML/Standard/AgentStatsView.dtl`
- `Kernel/System/Stats/Dynamic/Ticket.pm`
- `bin/otrs.GenerateStats.pl`

9.3.3. Caching

Whether the results of a statistic are to be cached or not can be setup in the configuration. Cached report results are stored as files in the `var/tmp`directory of the OTRS installation (in most cases `/opt/otrs/var/tmp`).

Cached stats can be recognized by the "Stats" prefix.

If the data is lost, no major damage is caused. The next time the report is called up, the stats module will not find the file any more and so will generate a new report. Of course this will probably take a little longer to run.

9.3.4. otrs.GenerateStats.pl

This file is saved in the `bin`directory. It facilitates the generation of report in the command line.

As an example, see the command line call in the following script.

```
bin> perl otrs.GenerateStats.pl -n 10004 -o /output/dir
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂ ✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂

A report from the stat configuration "Stat# 10004" is generated and saved as csv in the `/output/dir` directory.

The generated report can also be sent as an e-mail. More information can be called up with the command in the script below.

```
bin> perl otrs.GenerateStats.pl --help
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂
✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂

9.3.5. Automated stat generation - Cronjob

It usually does not make sense to generate reports manually via the command line, as the stats module has a convenient graphical user interface. However, generating reports manually does make sense when combined with a Cronjob.

Imagine the following scenario: On the first day of every month, the heads of department want to receive a report for the past month. By combining a cronjob and command line call the reports can be sent to them automatically by e-mail.

9.3.6. Static stats

The stats module facilitates the generation of static statistics. For every static stat a file exists in which its content is precisely defined.

This way, very complex stats can be generated. The disadvantage is that they are not particularly flexible.

The files are saved in the directory `Kernel/System/Stats/Static/`.

9.3.7. Using old static stats

Prior OTRS versions 1.3 and 2.0 already facilitated the generation of stats / reports. Various reports for OTRS versions 1.3 and 2.0 which have been specially developed to meet customers' requirements can be used in recent OTRS versions too.

The files must merely be moved from the `Kernel/System/Stats/` path to `Kernel/System/Stats/Static/`. Additionally the package name of the respective script must be amended by "::Static".

The following example shows how the first path is amended.

```
    package Kernel::System::Stats::AccountedTime;
```

```
    package Kernel::System::Stats::Static::AccountedTime;
```

9.3.8. Default stats

"It is not always necessary to reinvent the wheel..."

The stats module provides various default reports. Reports which are of interest for all OTRS users will in future be added to the default reports set of the stats module package. Default reports are saved in the stats module xml format in the`scripts/test/sample/` directory.

10. Dynamic Fields

10.1. Introduction

A dynamic field is a special kind of field in OTRS, created to extend the information stored on a ticket or article. These fields are not fixed in the system and they can appear only in specific screens, they can be mandatory or not, and their representation in the screens depends on the field type defined at their creation time according to the data to be held by the field. For example, there are fields to hold a text, a date, a selection of items, etc.

Dynamic fields are the evolution of TicketFreeText TicketFreeKey TicketFreeTime, ArticleFreeText and ArticleFreeKey fields that where commonly used in OTRS 3.0 and before. The limitation of these "Free Fields" was that they can be defined up to 16 (text or dropdown) fields and 6 time fields for a ticket and 3 (text or dropdown) fields for each article only, not more.

Now with dynamic fields the limitation in the number of fields per ticket or article is removed, you can create as many dynamic fields you like either for ticket or articles. And beyond that, the framework behind the dynamic fields is prepared to handle custom fields for other objects rather than just ticket and articles.

This new framework that handles the dynamic fields is build using a modular approach, where each kind of dynamic field can be seen as a plug-in module for the framework. This means that the variety of dynamic fields can be easily extended by public OTRS modules, OTRS Feature Add-ons, OTRS custom developments, and other custom developments.

The following dynamic field types are included with this release:

- Text (one line of text)
- Textarea (multiple lines of text)
- Checkbox
- Dropdown (single choice, multiple values)
- Multiselect (multiple choice, multiple values)
- Date
- Date / Time

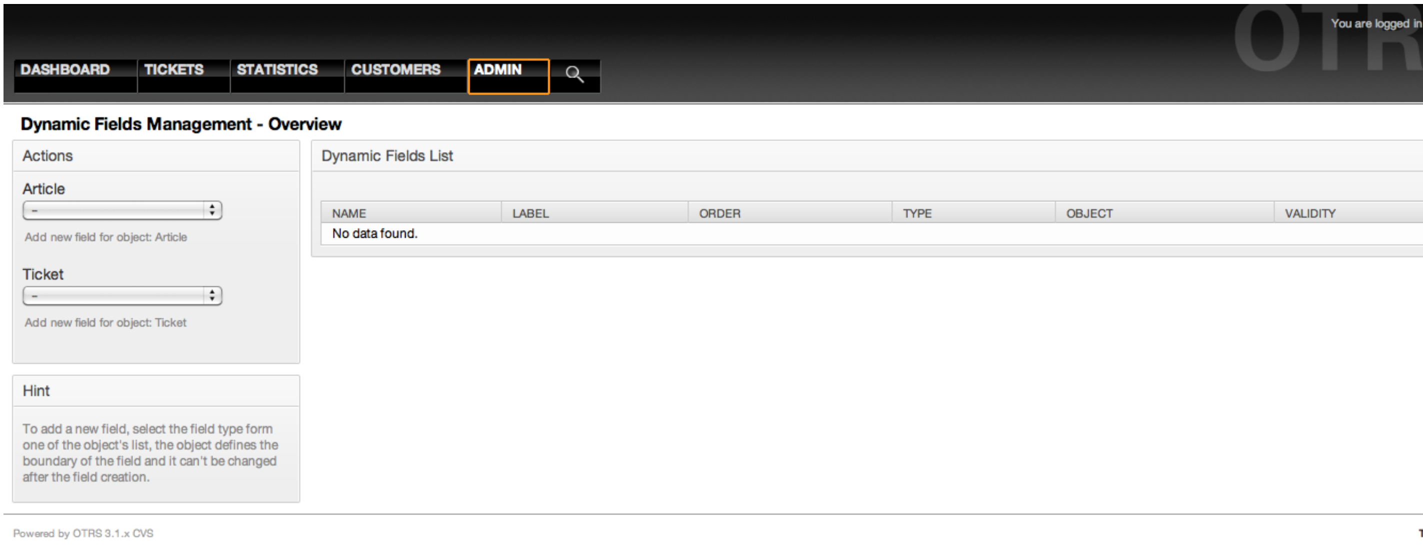
10.2. Configuration

By default, a clean installation of OTRS does not include any dynamic fields. If you plan to use such fields in tickets or articles you need to create dynamic fields.

The configuration of a dynamic field is split in two parts, to add a new dynamic field or manage an existing one you need to navigate into the "Admin" panel in the "Dynamic Fields" link. To show, show as mandatory or hide a dynamic field in one screen you need to change the OTRS settings in the "SysConfig" screen.

10.2.1. Adding a Dynamic Field

Click on the "Admin" button located in the navigation bar, then click on the "Dynamic Field" link inside "Ticket Settings" box located in the lower center of the screen. The dynamic fields overview will display as follows:



Dynamic Fields Management - Overview

Actions

Article

Add new field for object: Article

Ticket

Add new field for object: Ticket

Hint

To add a new field, select the field type form one of the object's list, the object defines the boundary of the field and it can't be changed after the field creation.

Dynamic Fields List

NAME	LABEL	ORDER	TYPE	OBJECT	VALIDITY
No data found.					

Powered by OTRS 3.1.x CVS

* * *

Notice that this screen will change as you add more dynamic fields to list all created dynamic fields. This screen might already have some fields if the installation was updated from an older version of OTRS.

The Actions in the side bar at the left of the screen describes two possibilities: Article and Ticket, each one has it's own dropdown selection of dynamic fields.

Note

The installation of an OTRS package could add more objects to the Action side bar.

The general procedure to create a dynamic field is:

- Click on the desired dynamic field object dropdown in the Action side bar.
- Click on the dynamic field type that you want to add from the list.
- Fill the configuration.
- Save.

The configuration dialogs for the dynamic fields are split in two parts, the upper section is common among all the fields and the lower part might be different from one type of dynamic field to another.

General dynamic field settings:

- Name: Mandatory, unique, only letters and numbers are allowed.

  This is the internal name of the field, used for example to show or hide a field in one screen. Any modification of a field name (not recommended) requires a manual update of the "SysConfig" settings where the field is referenciated.

- Label: Mandatory.

  This is the field name to be displayed on the screens, it supports translations.

  ### Note

  Label translations have to be added manually to language translations files.

- Field order: Mandatory.

  Defines the relative order in which the field will be displayed on the screen, by default each new field has the last position, a change in this setting will affect the order of the other created dynamic fields.

- Validity: Mandatory.

  An invalid dynamic field will not be displayed in any screen, no matter if is configured to displayed.

- Field type: Mandatory, Read only.

  Shows the current selected field type.

- Object type: Mandatory, Read only.

  Shows the scope of field.

Note

To illustrate each specific field type settings a few fields will be added in our example. These new fields will be referenciated in later sections.

For the following examples all the dynamic fields will be created for the Ticket object if you need to create a dynamic field for Article object, just choose the field from the Article dropdown list.

Table 4.5. The following fields will be added into the system:

Name	Label	Type
Field1	My Field 1	Text
Field2	My Field 2	Textarea
Field3	My Field 3	Checkbox
Field4	My Field 4	Dropdown
Field5	My Field 5	Multiselect
Field6	My Field 5	Date
Field7	My Field 6	Date / Time

10.2.2. Text Dynamic Field Configuration

Text dynamic field is used to store a single line string.

Text dynamic field settings:

- Default value: Optional.

  This is the value to be shown by default on the edit screens (like New Phone Ticket or Ticket Compose).

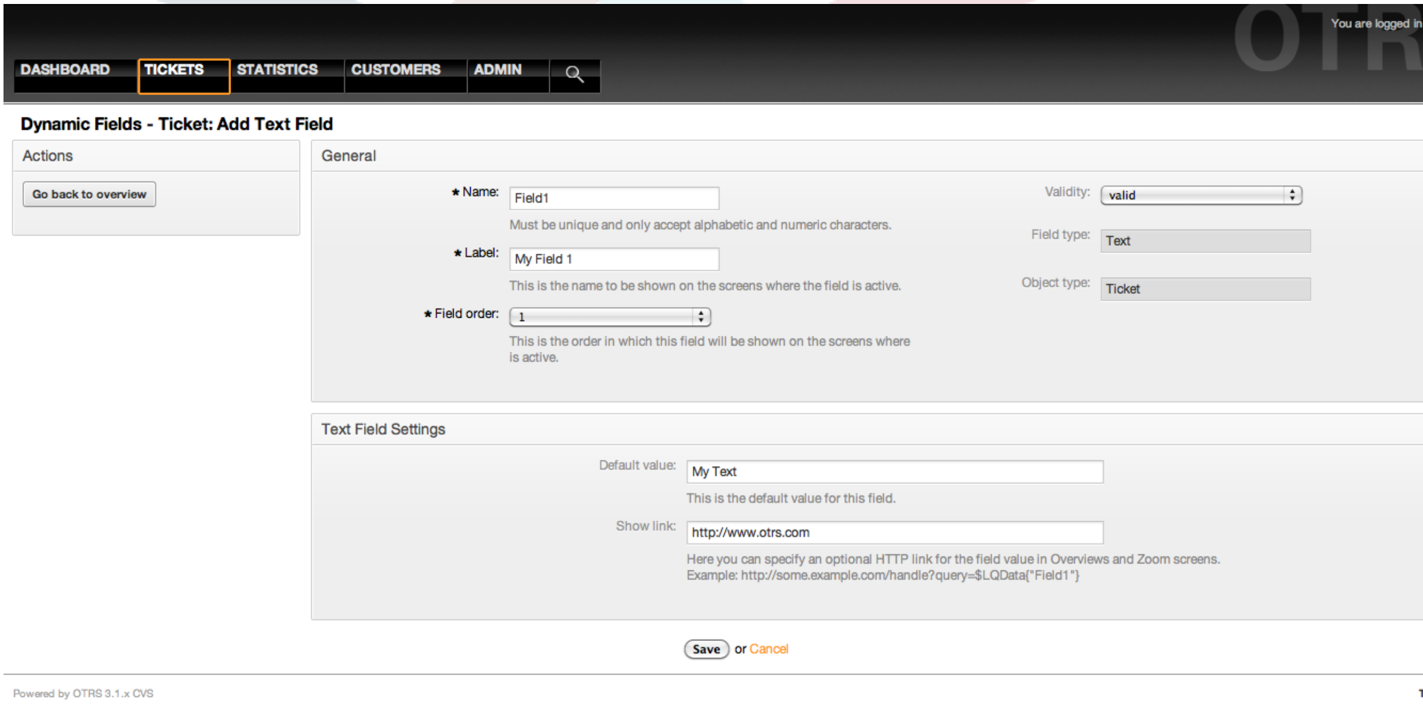
- Show link: Optional.

  If set, the field value will be converted into a clickable link for display screens (like ticket zoom or overviews).

  For example, if "Show link" is set to "http://www.otrs.com", clicking on the filled value will make your browser to open the OTRS web page.

  Note

  The use of $LQData{"NameX"} in the Set link value, where NameX is the name of the field will add the field value as part of the link reference.



Dynamic Fields - Ticket: Add Text Field

General

* Name: Field1
Must be unique and only accept alphabetic and numeric characters.
* Label: My Field 1
This is the name to be shown on the screens where the field is active.
* Field order: 1
This is the order in which this field will be shown on the screens where is active.
Validity: valid
Field type: Text
Object type: Ticket

Text Field Settings

Default value: My Text
This is the default value for this field.
Show link: http://www.otrs.com
Here you can specify an optional HTTP link for the field value in Overviews and Zoom screens.
Example: http://some.example.com/handle?query=$LQData{"Field1"}

10.2.3. Textarea Dynamic Field Configuration

Textarea dynamic field is used to store a multiple line string.

Textarea dynamic field settings:

- Number of rows: Optional, integer.

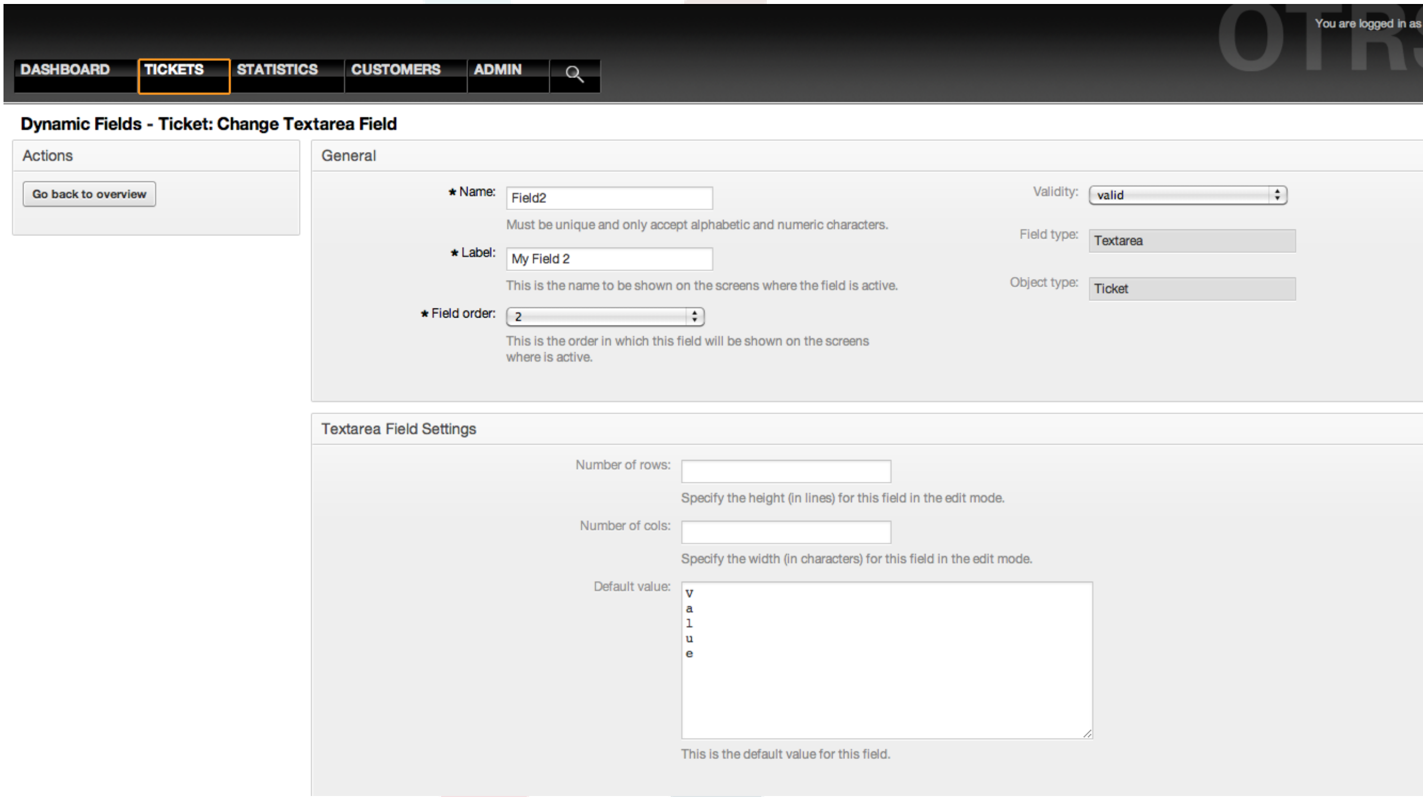
  Used to define the height of the field in the edit screens (like New Phone Ticket or Ticket Compose).

- Number of cols: Optional, Integer.

  This value is used to define the width of the field in the edit screens.

- Default value: Optional.

  This is the value to be shown by default in the edit screens (it can be a multiple line text).



Dynamic Fields - Ticket: Change Textarea Field

General

★ Name: Field2

Must be unique and only accept alphabetic and numeric characters.

★ Label: My Field 2

This is the name to be shown on the screens where the field is active.

★ Field order: 2

This is the order in which this field will be shown on the screens where is active.

Validity: valid

Field type: Textarea

Object type: Ticket

Textarea Field Settings

Number of rows:

Specify the height (in lines) for this field in the edit mode.

Number of cols:

Specify the width (in characters) for this field in the edit mode.

Default value:

```
v
a
l
u
e
```

This is the default value for this field.

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂
✂✂✂✂✂✂✂

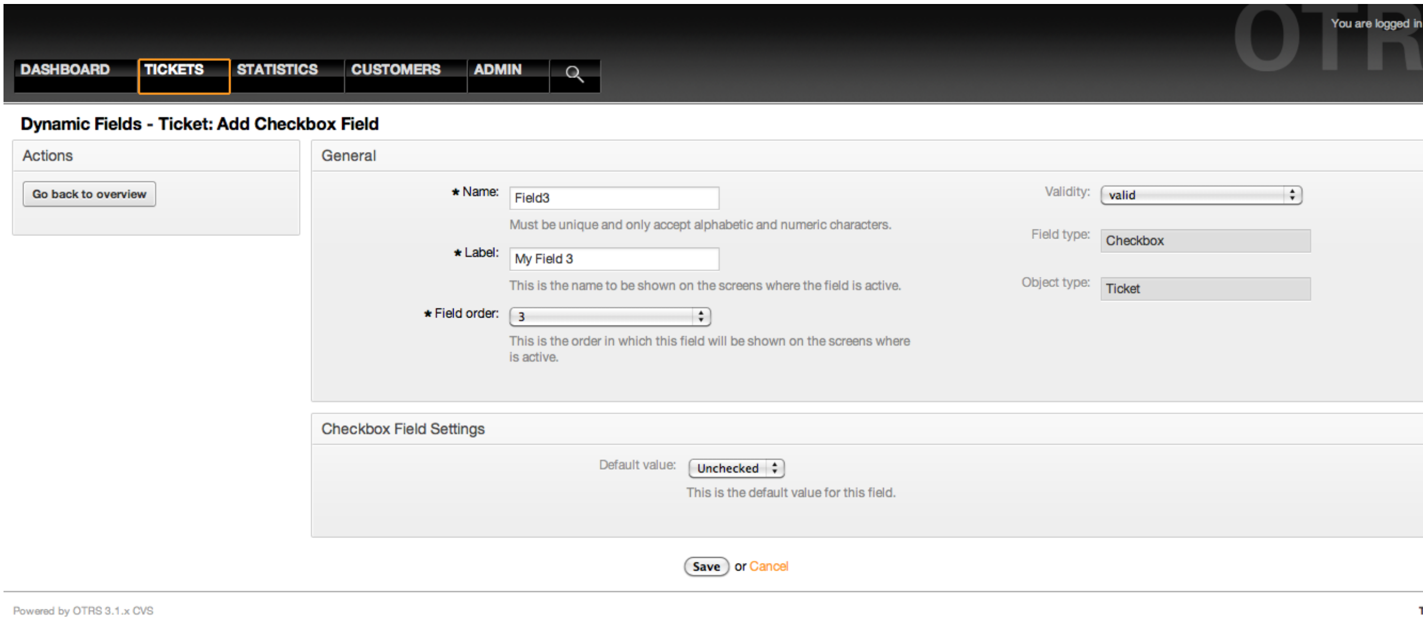
10.2.4. Checkbox Dynamic Field Configuration

Checkbox dynamic field is used to store true or false value, represented by a checked or unchecked check box.

Checkbox dynamic field settings:

- Default value: Mandatory.

  This is the value to be shown by default on the edit screens (like New Phone Ticket or Ticket Compose), the default value for this field is closed selection which can be Checked or Unchecked.



Dynamic Fields - Ticket: Add Checkbox Field

Actions

General

★ Name: Field3

Must be unique and only accept alphabetic and numeric characters.

★ Label: My Field 3

This is the name to be shown on the screens where the field is active.

★ Field order: 3

This is the order in which this field will be shown on the screens where is active.

Validity: valid

Field type: Checkbox

Object type: Ticket

Checkbox Field Settings

Default value: Unchecked

This is the default value for this field.

10.2.5. Dropdown Dynamic Field Configuration

Dropdown dynamic field is used to store a single value, from a closed list.

Dropdown dynamic field settings:

- Possible values: Mandatory.

  List of values to choose. If used, a new value is necessary to specify the Key (internal value) and the Value (display value).

- Default value: Optional.

  This is the value to be show by default on the edit screens (like New Phone Ticket or Ticket Compose), the default value for this field is closed selection defined by the Possible values.

- Add empty value: Mandatory, (Yes / No).

  If this option is activated an extra value is defined to show a "-" in the list of possible values, this special value is empty internally.

- Translatable values: Mandatory, (Yes / No).

  This setting is used mark the possible values of this field to be translated. Only the display values are translated, internal values are not affected, the translation of the values needs to be manually added to the language files.

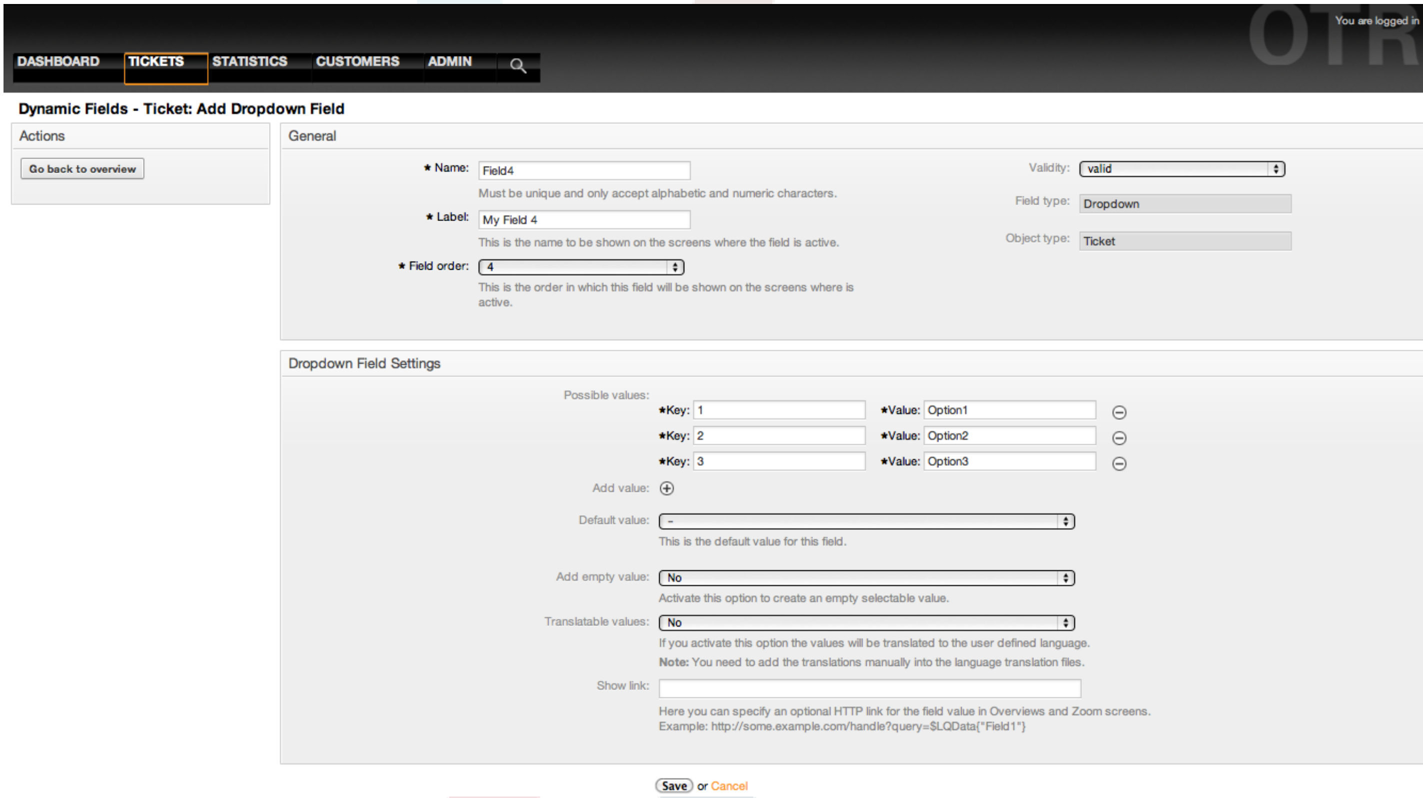
- Show link: Optional.

  If set, the field value will be converted into a clickable HTP link for display screens (like Zoom or overviews).

  For example, if Show link is set to "http://www.otrs.com", clicking on the filed value will make your browser to open the OTRS web page.

  ### Note

  The use of $LQData{"NameX"} in the Set link value, where NameX is the name of the field, will add the field value as part of the link reference.



Dynamic Fields - Ticket: Add Dropdown Field

Actions

General

★ Name: Field4

Must be unique and only accept alphabetic and numeric characters.

★ Label: My Field 4

This is the name to be shown on the screens where the field is active.

★ Field order: 4

This is the order in which this field will be shown on the screens where is active.

Validity: valid

Field type: Dropdown

Object type: Ticket

Dropdown Field Settings

Possible values:

★Key: 1 ★Value: Option1
★Key: 2 ★Value: Option2
★Key: 3 ★Value: Option3

Add value:

Default value: -

This is the default value for this field.

Add empty value: No

Activate this option to create an empty selectable value.

Translatable values: No

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.

Show link:

Here you can specify an optional HTTP link for the field value in Overviews and Zoom screens.
Example: http://some.example.com/handle?query=$LQData{"Field1"}

10.2.6. Multiselect Dynamic Field Configuration

Multiselect dynamic field is used to store multiple values, from a closed list.

Multiselect dynamic field settings:

- Possible values: Mandatory.

  List of values to choose from. When adding additional list items, it is necessary to specify the Key (internal value) and the Value (display value).

- Default value: Optional.

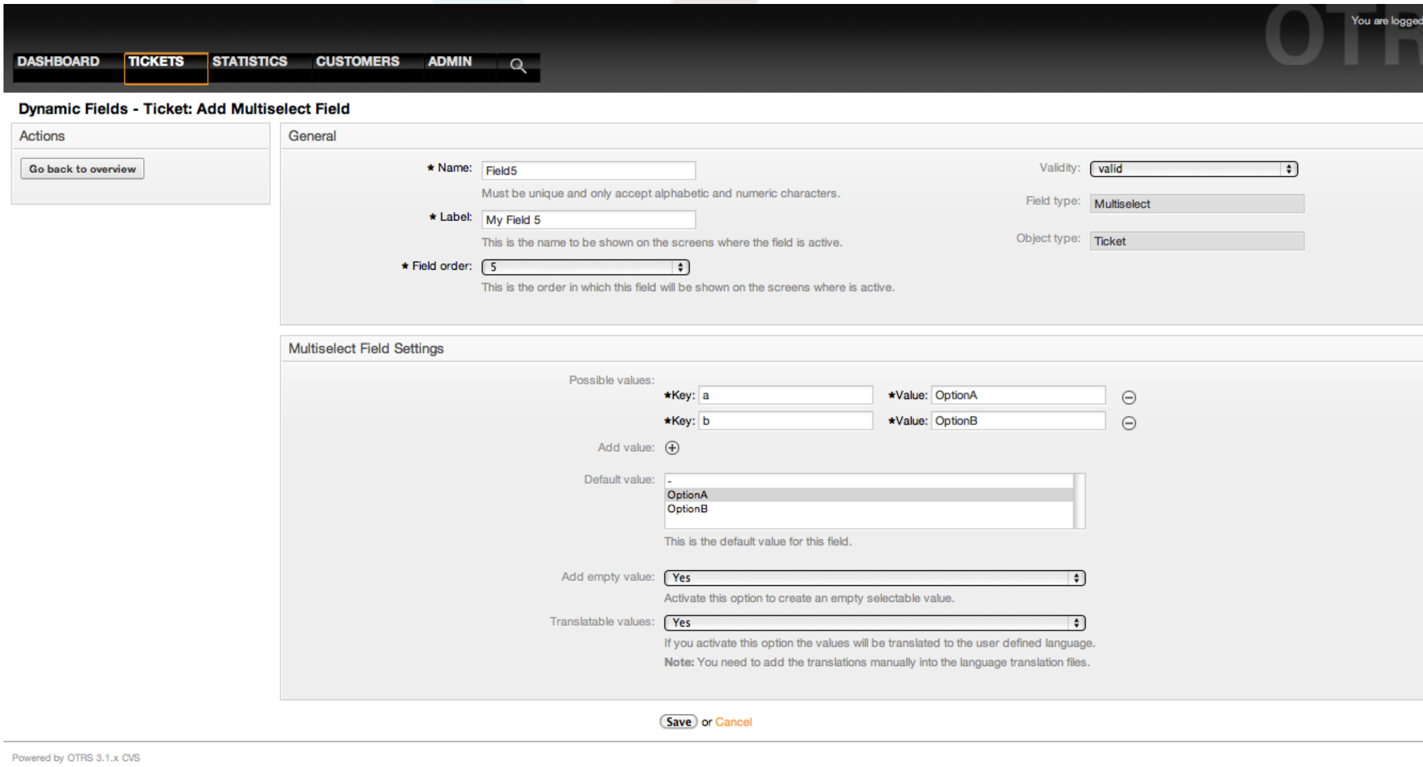
This is the value to be shown by default on the edit screens (like New Phone Ticket or Ticket Compose), the default value for this field is closed selection as defined by the Possible values.

- Add empty value: Mandatory, (Yes / No).

  If this option is activated an extra value is defined to show as a "-" in the list of possible values. This special value is empty internally.

- Translatable values: Mandatory, (Yes / No).

  This setting is used mark the possible values of this field to be translated. Only the display values are translated, internal values are not affected, the translation of the values needs to be manually added to the language files.



Dynamic Fields - Ticket: Add Multiselect Field

General

★ Name: Field5
Must be unique and only accept alphabetic and numeric characters.
★ Label: My Field 5
This is the name to be shown on the screens where the field is active.
★ Field order: 5
This is the order in which this field will be shown on the screens where is active.
Validity: valid
Field type: Multiselect
Object type: Ticket

Multiselect Field Settings

Possible values:
★Key: a ★Value: OptionA
★Key: b ★Value: OptionB
Add value:
Default value: - / OptionA / OptionB
This is the default value for this field.
Add empty value: Yes
Activate this option to create an empty selectable value.
Translatable values: Yes
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.

Powered by OTRS 3.1.x CVS

10.2.7. Date Dynamic Field Configuration

Date dynamic field is used to store a date value (Day, Month and Year).

Date dynamic field settings:

- Default date difference: Optional, Integer.

  Number of seconds (positive or negative) between the current date and the selected date to be show by default in the edit screens (like New Phone Ticket or Ticket Compose).

- Define years period: Mandatory (Yes / No).

  Used to set a defined number of years in the past and the future based on the current date of the year select for this field, If set to Yes the following options are available:

  - Years in the past: Optional, Positive integer.

    Define the number of years in the past from the current day to display in the year selection for this dined in edit screens.

  - Years in the future: Optional, Positive integer.

    Define the number of years in the future from the current day to display in the year selection for this dined in edit screens.

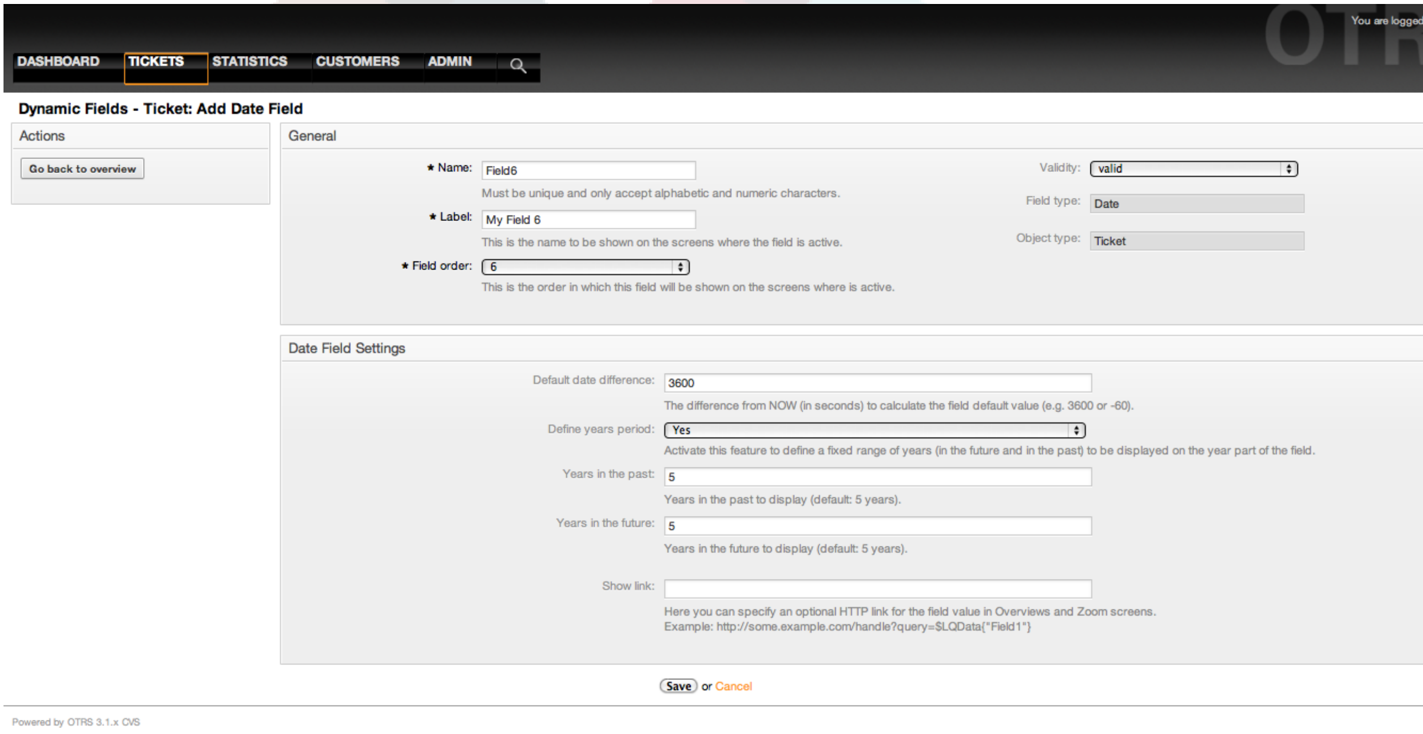
- Show link: Optional.

  If set, the field value will be converted into a clickable HTP link for display screens (like Zoom or overviews).

  For example, if Show link is set to "http://www.otrs.com", clicking on the filed value will make your browser to open the OTRS web page.

Note

The use of $LQData{"NameX"} in the Set link value, where NameX is the name of the field will add the field value as part of the link reference.



Dynamic Fields - Ticket: Add Date Field

General

* Name: Field6

Must be unique and only accept alphabetic and numeric characters.

* Label: My Field 6

This is the name to be shown on the screens where the field is active.

* Field order: 6

This is the order in which this field will be shown on the screens where is active.

Validity: valid

Field type: Date

Object type: Ticket

Date Field Settings

Default date difference: 3600

The difference from NOW (in seconds) to calculate the field default value (e.g. 3600 or -60).

Define years period: Yes

Activate this feature to define a fixed range of years (in the future and in the past) to be displayed on the year part of the field.

Years in the past: 5

Years in the past to display (default: 5 years).

Years in the future: 5

Years in the future to display (default: 5 years).

Show link:

Here you can specify an optional HTTP link for the field value in Overviews and Zoom screens.
Example: http://some.example.com/handle?query=$LQData{"Field1"}

Powered by OTRS 3.1.x CVS

* * *

10.2.8. Date / Time Dynamic Field Configuration

Date / Time dynamic field is used to store a date time value (Minute, Hour, Day, Month and Year).

Date / Time dynamic field settings:

- Default date difference: Optional, Integer.

  Number of seconds (positive or negative) between the current date and the selected date to be shown by default in the edit screens (like New Phone Ticket or Ticket Compose).

- Define years period: Mandatory (Yes / No).

  Used to set a defined number of years in the past and thr future from thee current date in the year select of this field, If set to Yes the following options are available:

  - Years in the past: Optional, Positive integer.

    Define the number of years in the past from the current day to display in the year selection for this dined in edit screens.

  - Years in the future: Optional, Positive integer.

    Define the number of years in the future from the current day to display in the year selection for this dined in edit screens.

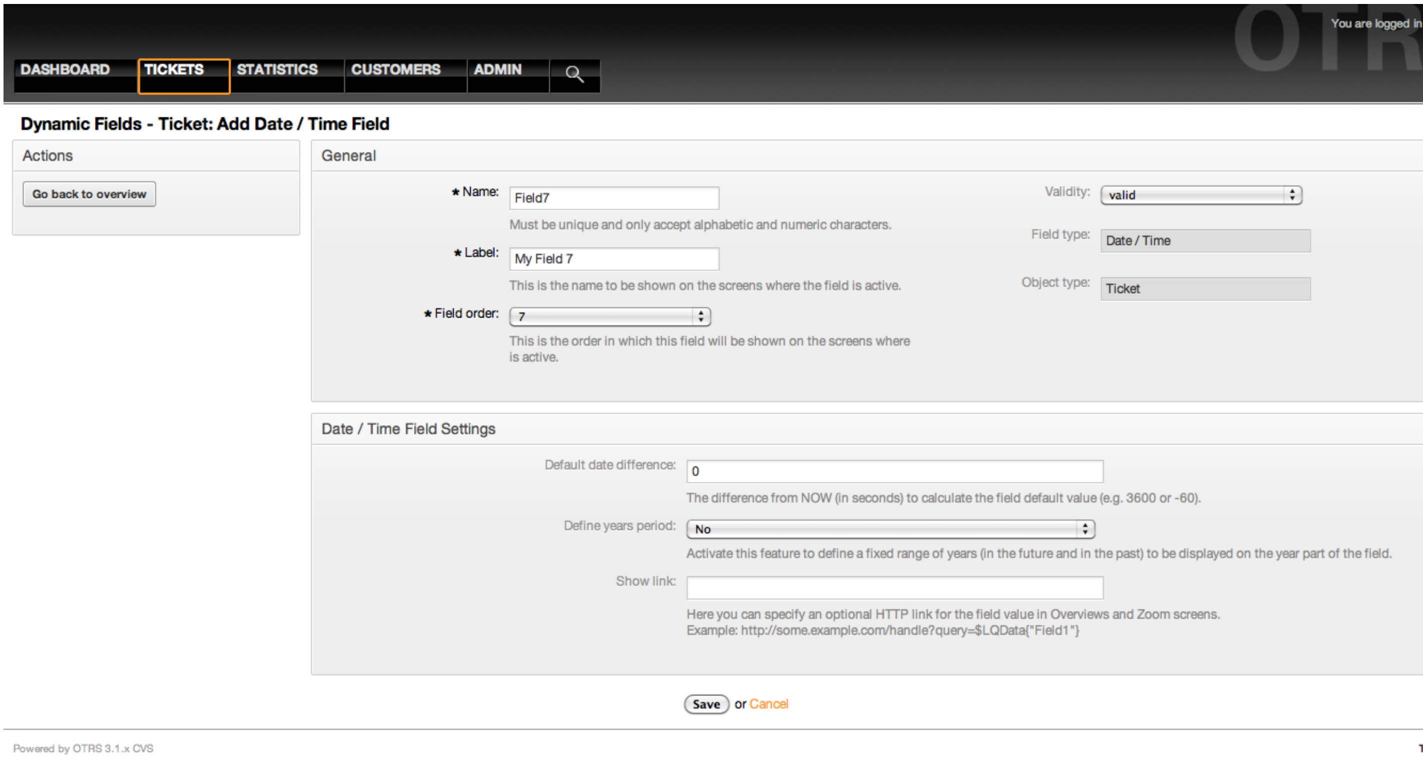
- Show link: Optional.

  If set, the field value will be converted into a clickable HTP link for display screens (like Zoom or overviews).

  For example, if Show link is set to "http://www.otrs.com", clicking on the filed value will make your browser to open the OTRS web page.

  #### Note

  The use of $LQData{"NameX"} in the Set link value, where NameX is the name of the field will add the field value as part of the link reference.


You are logged in
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Go back to overview
Save
or Cancel
Powered by OTRS 3.1.x CVS

Dynamic Fields - Ticket: Add Date / Time Field

Actions

General

* Name: Field7

Must be unique and only accept alphabetic and numeric characters.

* Label: My Field 7

This is the name to be shown on the screens where the field is active.

* Field order: 7

This is the order in which this field will be shown on the screens where is active.

Validity: valid

Field type: Date / Time

Object type: Ticket

Date / Time Field Settings

Default date difference: 0

The difference from NOW (in seconds) to calculate the field default value (e.g. 3600 or -60).

Define years period: No

Activate this feature to define a fixed range of years (in the future and in the past) to be displayed on the year part of the field.

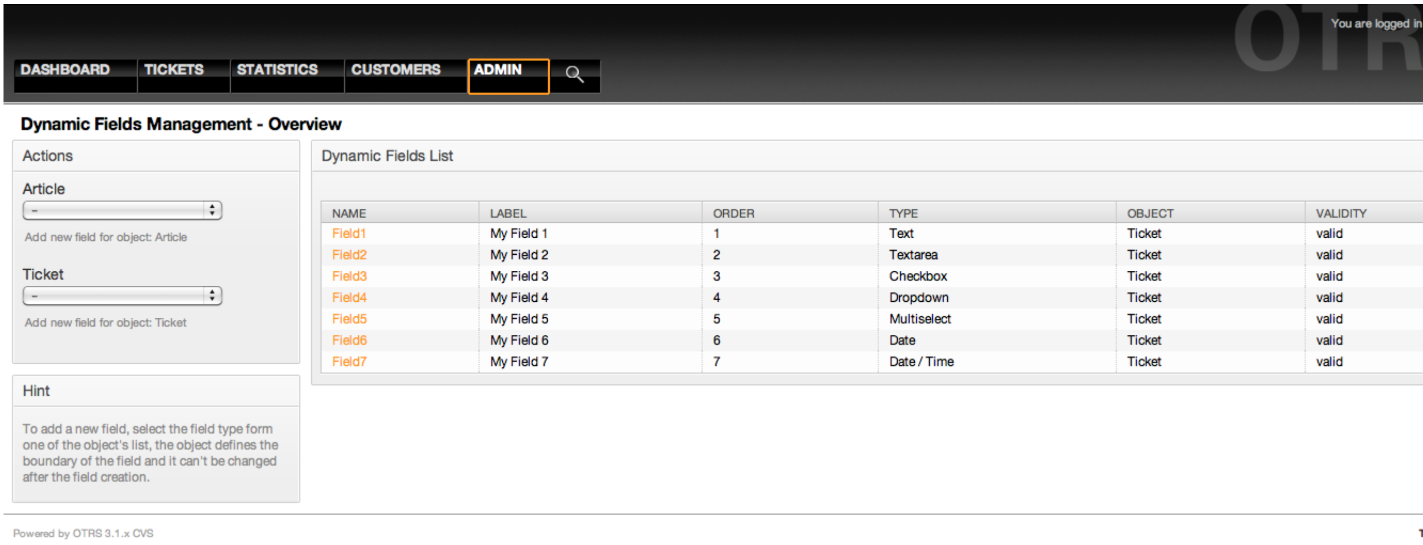
Show link:

Here you can specify an optional HTTP link for the field value in Overviews and Zoom screens.
Example: http://some.example.com/handle?query=$LQData{"Field1"}

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂ ✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂

10.2.9. Editing a Dynamic Field

A filled dynamic field overview screen (with the previous examples) should look like:



Dynamic Fields Management - Overview

Actions

Article

-

Add new field for object: Article

Ticket

-

Add new field for object: Ticket

Hint

To add a new field, select the field type form one of the object's list, the object defines the boundary of the field and it can't be changed after the field creation.

Dynamic Fields List

NAME	LABEL	ORDER	TYPE	OBJECT	VALIDITY
Field1	My Field 1	1	Text	Ticket	valid
Field2	My Field 2	2	Textarea	Ticket	valid
Field3	My Field 3	3	Checkbox	Ticket	valid
Field4	My Field 4	4	Dropdown	Ticket	valid
Field5	My Field 5	5	Multiselect	Ticket	valid
Field6	My Field 6	6	Date	Ticket	valid
Field7	My Field 7	7	Date / Time	Ticket	valid

Powered by OTRS 3.1.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂

To change or edit a dynamic field you must have at least one field defined, select an already added field from the dynamic fields overview screen and update its settings.

Note

Not all the dynamic field settings can be changed, the Field type and Object type are fixed from the selection of the field and they can't be changed.

It is not recommended to change the field internal name, but the label can be changed at any time. If internal name is changed all "SysConfig" settings that have a reference to that particular field needs to be updated as well as user preferences (if defined).

10.2.10. Showing a Dynamic Field on a Screen

To display a dynamic field on a particular screen there are two mandatory conditions:

1. The dynamic field must be valid.

2. The dynamic field must be set to 1 or 2 in the configuration of the screen.

Follow these steps to show a dynamic field in a screen

- Be sure that the dynamic field is set to valid, you can see the validity of the field from the dynamic field overview screen. Set to valid by editing the field if necessary.

- Open the "sysconfig" and select "Ticket" from the dropdown list in the Actions side bar located in the left part of the screen.

Note

You can also search for "DynamicField" in the search box above or the "sysconfig" key directly if you already know it.

- Locate the setting sub-group for the screen that you are looking for and click on it. For example "Frontend::Agent::Ticket::ViewPhoneNew".
- Search for the setting that ends with "###DynamicField". For example "Ticket::Frontend::AgentTicketPhone###DynamicField".
- If the setting is empty or does not have the required dynamic field name, click on the "+" button to add a new entry. For example Key: Field1, Content: 1.

  If the setting already has the dynamic field name listed be sure that is set to "1" to display the field or to "2" to display it as mandatory.
- Save the configuration by clicking in the "Update" button and the bottom of the screen and navigate to the screen where you want the field to be displayed.

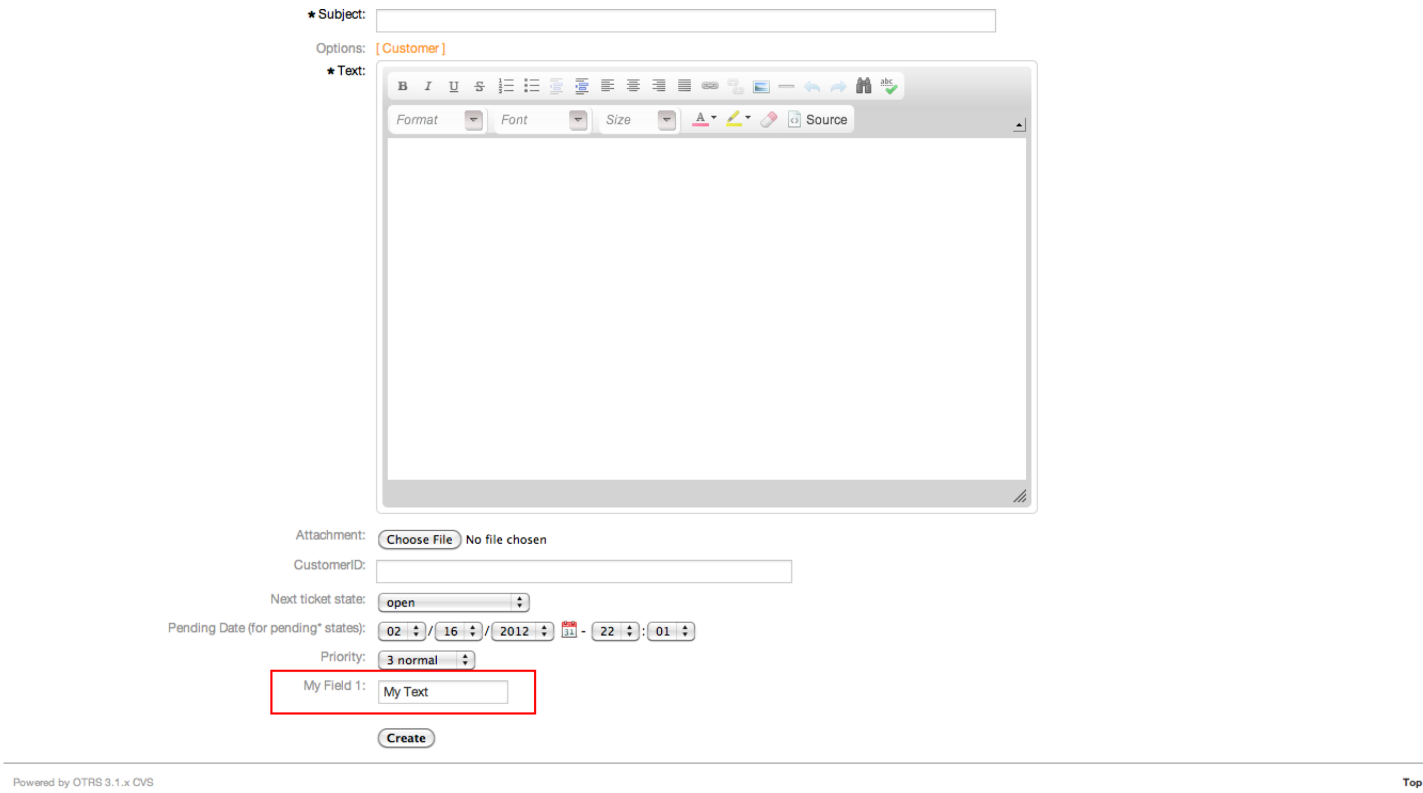
10.2.10.1. Show Examples

The following are "sysconfig" configurations examples to show or hide dynamic fields on different screens.

Example 4.20. Activate Field1 in New Phone Ticket Screen.

- ✂✂✂✂✂✂ Ticket
- ✂✂✂✂✂✂✂✂✂✂ Frontend::Agent::Ticket::ViewPhoneNew
- ✂✂✂✂✂✂✂✂ Ticket::Frontend::AgentTicketPhone###DynamicField
- ✂✂✂✂✂✂

Key	Content
Field1	1


★Subject:
Options: [Customer]
★Text:
Format
Font
Size
Source
Create
Top

Attachment: Choose File No file chosen

CustomerID:

Next ticket state: open

Pending Date (for pending* states): 02 / 16 / 2012 - 22 : 01

Priority: 3 normal

My Field 1: My Text

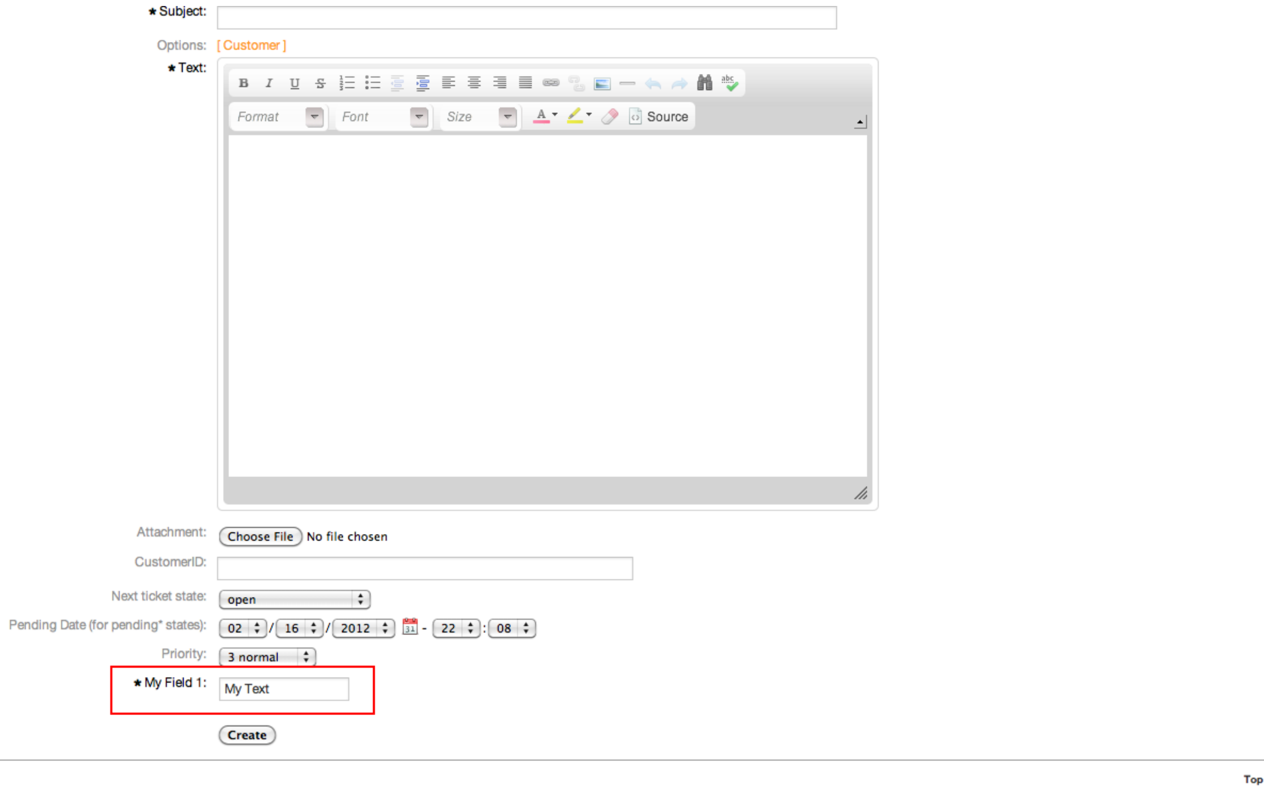
Powered by OTRS 3.1.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂✂

Example 4.21. Activate Field1 in New Phone Ticket Screen as mandatory.

- ✂✂✂✂✂✂ Ticket
- ✂✂✂✂✂✂✂✂✂✂ Frontend::Agent::Ticket::ViewPhoneNew
- ✂✂✂✂✂✂✂✂ Ticket::Frontend::AgentTicketPhone###DynamicField
- ✂✂✂✂✂✂

Key	Content
Field1	2



★ Subject:

Options: [Customer]

★ Text:

Attachment: No file chosen

CustomerID:

Next ticket state: open

Pending Date (for pending* states): 02 / 16 / 2012 - 22 : 08

Priority: 3 normal

★ My Field 1: My Text

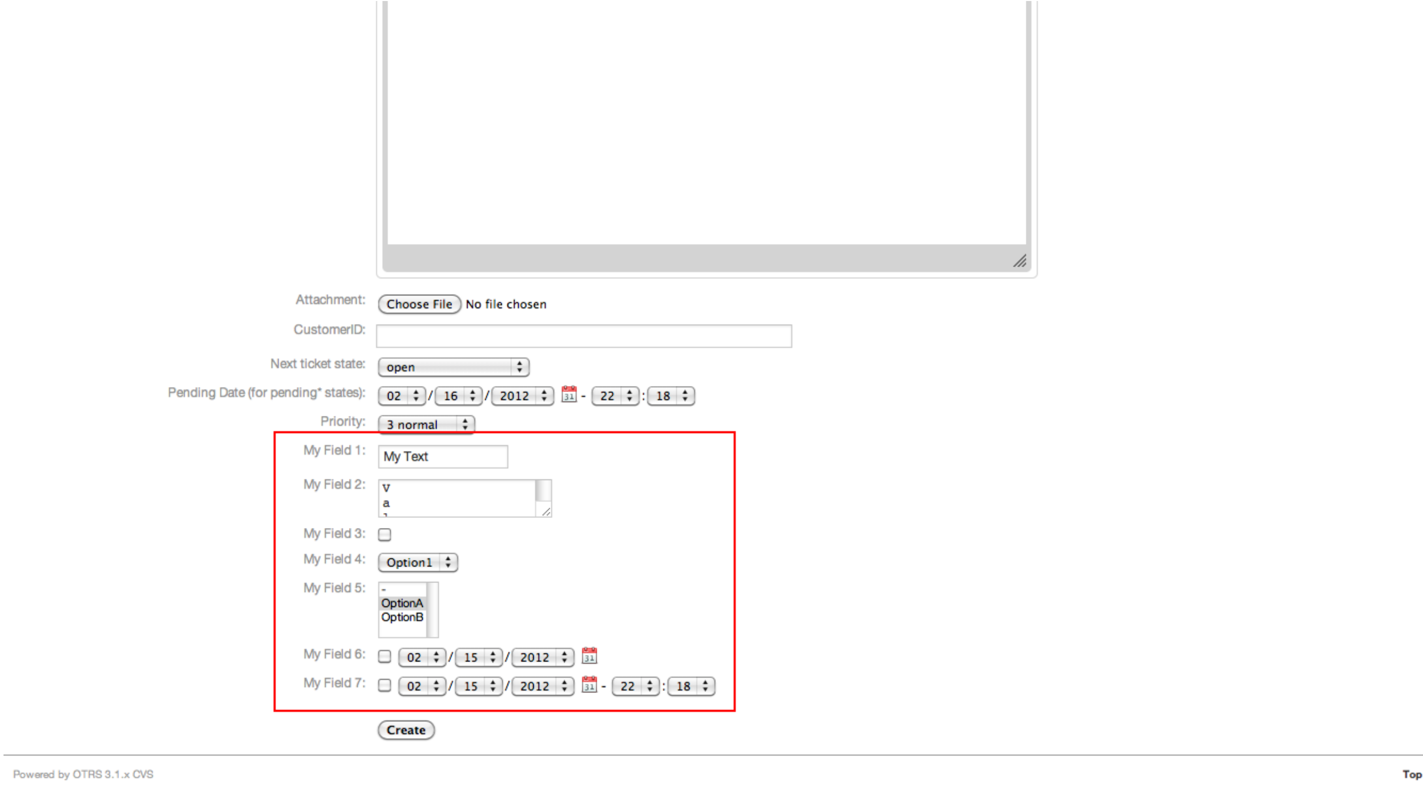
Powered by OTRS 3.1.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂✂✂✂

Example 4.22. Activate several fields in New Phone Ticket Screen.

- ✂✂✂✂✂✂ Ticket
- ✂✂✂✂✂✂✂✂✂✂ Frontend::Agent::Ticket::ViewPhoneNew
- ✂✂✂✂✂✂✂✂ Ticket::Frontend::AgentTicketPhone###DynamicField
- ✂✂✂✂✂✂

Key	Content
Field1	1
Field2	1
Field3	1
Field4	1
Field5	1
Field6	1
Field7	1



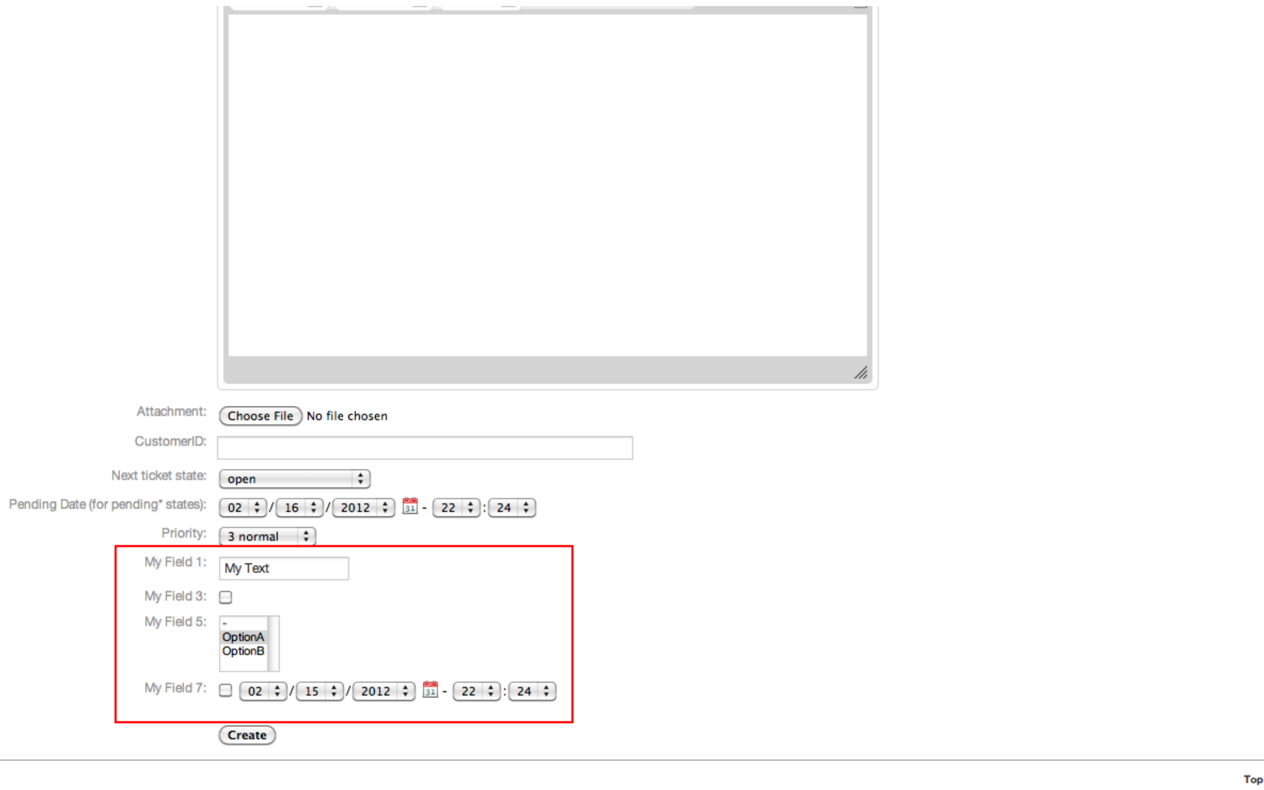
Attachment: Choose File No file chosen
CustomerID:
Next ticket state: open
Pending Date (for pending* states): 02 / 16 / 2012 - 22 : 18
Priority: 3 normal
My Field 1: My Text
My Field 2: v a l
My Field 3: ☐
My Field 4: Option1
My Field 5: - OptionA OptionB
My Field 6: ☐ 02 / 15 / 2012
My Field 7: ☐ 02 / 15 / 2012 - 22 : 18

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂✂✂✂

Example 4.23. Deactivate some fields in New Phone Ticket Screen.

- ✂✂✂✂✂✂ Ticket
- ✂✂✂✂✂✂✂✂✂✂ Frontend::Agent::Ticket::ViewPhoneNew
- ✂✂✂✂✂✂✂✂ Ticket::Frontend::AgentTicketPhone###DynamicField
- ✂✂✂✂✂✂

Key	Content
Field1	1
Field2	0
Field3	1
Field4	0
Field5	1
Field6	0
Field7	1


Attachment: Choose File No file chosen

CustomerID:

Next ticket state: open

Pending Date (for pending* states): 02 / 16 / 2012 - 22 : 24

Priority: 3 normal

My Field 1: My Text

My Field 3: ☐

My Field 5: - OptionA OptionB

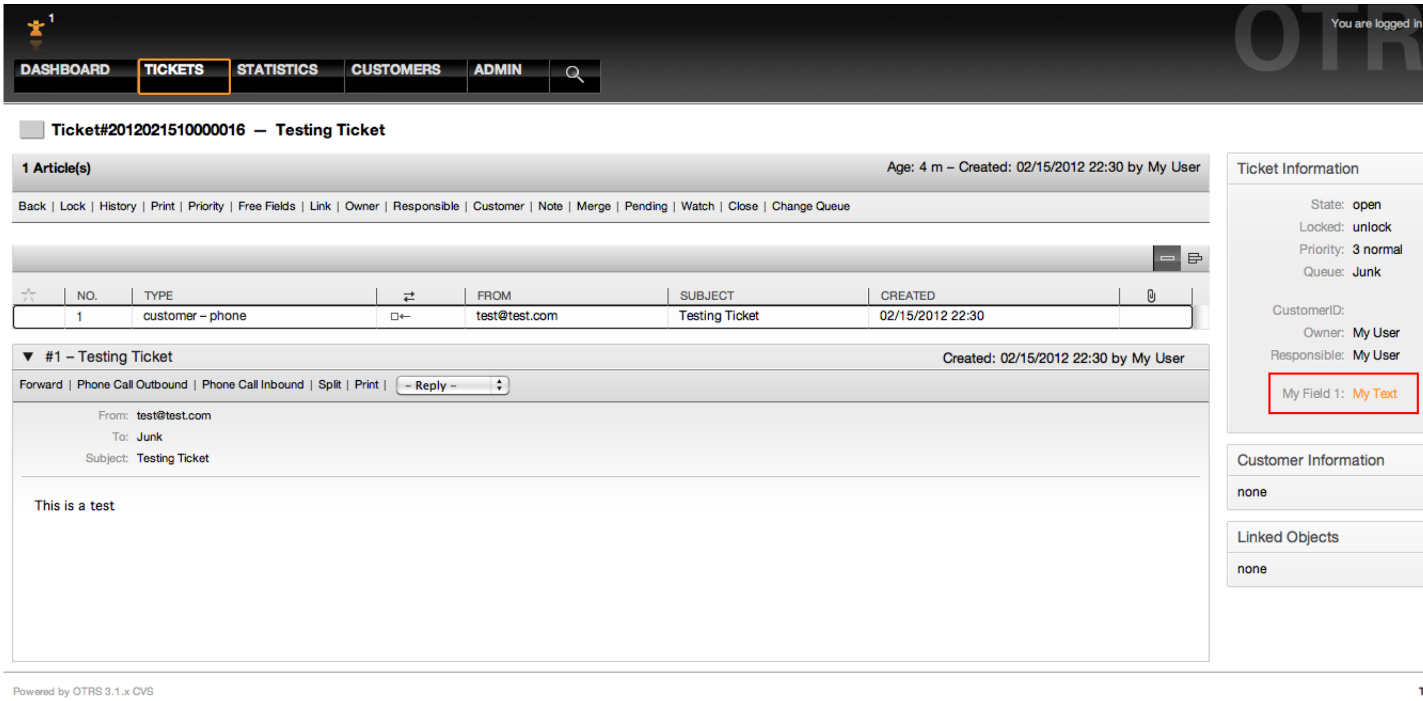
My Field 7: ☐ 02 / 15 / 2012 - 22 : 24

✁✁✁✁✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁ ✁✁✁ ✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁ ✁✁✁✁✁✁✁✁✁✁

Example 4.24. Activate Field1 in Ticket Zoom Screen.

- ✁✁✁✁✁✁ Ticket
- ✁✁✁✁✁✁✁✁✁✁ Frontend::Agent::Ticket::ViewZoom
- ✁✁✁✁✁✁✁✁ Ticket::Frontend::AgentTicketZoom###DynamicField
- ✁✁✁✁✁✁

Key	Content
Field1	1


DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in

Ticket#2012021510000016 — Testing Ticket

1 Article(s)

Age: 4 m – Created: 02/15/2012 22:30 by My User

Back | Lock | History | Print | Priority | Free Fields | Link | Owner | Responsible | Customer | Note | Merge | Pending | Watch | Close | Change Queue

NO.	TYPE		FROM	SUBJECT	CREATED
1	customer – phone		test@test.com	Testing Ticket	02/15/2012 22:30

#1 – Testing Ticket

Created: 02/15/2012 22:30 by My User

Forward | Phone Call Outbound | Phone Call Inbound | Split | Print | – Reply –

From: test@test.com
To: Junk
Subject: Testing Ticket

This is a test

Ticket Information

State: open
Locked: unlock
Priority: 3 normal
Queue: Junk
CustomerID:
Owner: My User
Responsible: My User
My Field 1: My Text

Customer Information

none

Linked Objects

none

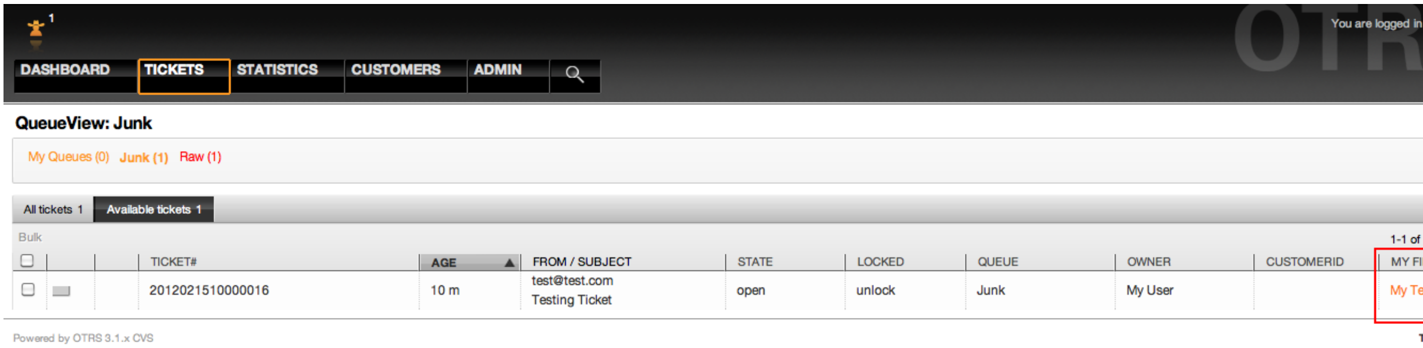
Powered by OTRS 3.1.x CVS

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁ ✁✁✁✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁✁

Example 4.25. Activate Field1 in Ticket Overview Small Screens.

- ✁✁✁✁✁✁ Ticket
- ✁✁✁✁✁✁✁✁✁✁ Frontend::Agent::TicketOverview
- ✁✁✁✁✁✁✁✁ Ticket::Frontend::OverviewSmall###DynamicField
- ✁✁✁✁✁✁

Key	Content
Field1	1



QueueView: Junk

My Queues (0) Junk (1) Raw (1)

All tickets 1 Available tickets 1

Bulk 1-1 of

☐			TICKET#	AGE	FROM / SUBJECT	STATE	LOCKED	QUEUE	OWNER	CUSTOMERID	MY FI
☐			2012021510000016	10 m	test@test.com Testing Ticket	open	unlock	Junk	My User		My Te

Powered by OTRS 3.1.x CVS

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁ ✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁ ✁✁✁✁✁ ✁✁✁✁✁✁✁

This setting affects: Escalation View, Locked View, Queue View, Responsible View, Status View and Watch View screens.

10.2.11. Setting a Default Value by a Ticket Event Module

A ticket event (e.g. TicketCreate) can trigger a value set for a certain field, if the field does not have a value yet.

Note

By using this method this default value, is not seen in the edit screen (e.g. New Phone Ticket) since the value is set after the creation of the ticket.

To activate this feature it is necessary to enable the following setting: "Ticket::EventModulePost###TicketDynamicFieldDefault".

Example 4.26. Activate Field1 in TicketCreate event.

- ✁✁✁✁✁✁ Ticket
- ✁✁✁✁✁✁✁✁✁✁ Core::TicketDynamicFieldDefault
- ✁✁✁✁✁✁✁✁ Ticket::TicketDynamicFieldDefault###Element1

Note

This configuration can be set in any of the 16 Ticket::TicketDynamicFieldDefault###Element settings.

If more that 16 fields needs to be set up a custom XML file must be places in $OTRS_HOME/Kernel/Config/files directory to extend this feature.

- ✂✂✂✂✂✂

Key	Content
Event	TicketCreate
Name	Field1
Value	a new value

10.2.12. Set a Default Value by User Preferences

The dynamic field default value can be overwritten with a user defined value stored in the user preferences.

Using this method, the default value of the field will be shown on any screen where the field is activated (if the field does not have already a different value).

The "sysconfig" setting "PreferencesGroups###DynamicField" located in the "Frontend::Agent::Preferences" Sub-group. This setting is an example of how to create an entry in the User Preferences screen to set an exclusive dynamic field default value for the selected user. The limitation of this setting is that it only permits the use of one dynamic field. If two or more fields will use this feature, it is necessary to create a custom XML configuration file to add more settings similar to this one.

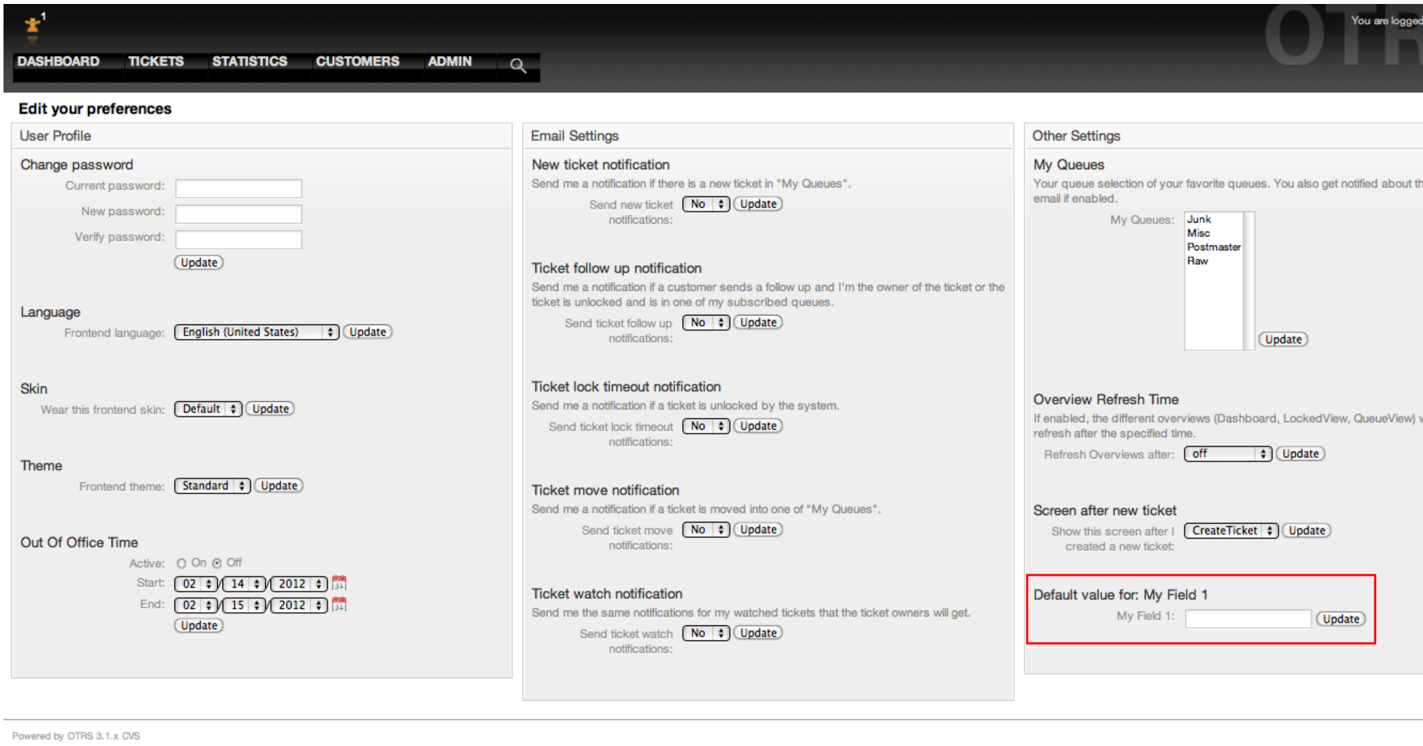
> ## Note
>
> Remember, if more settings are added in a new XML each setting name needs to be unique in the system and different than "PreferencesGroups###DynamicField". for example: PreferencesGroups###101-DynamicField-Field1, PreferencesGroups###102-DynamicField-Field2, PreferencesGroups###My-Field1, PreferencesGroups###My-Field2, etc.

Example 4.27. Activate Field1 in the User preferences.

- ✂✂✂✂✂✂ Ticket
- ✂✂✂✂✂✂✂✂✂✂ Frontend::Agent::Preferences
- ✂✂✂✂✂✂✂✂ PreferencesGroups###101-DynamicField-Field1
- ✂✂✂✂✂✂

Key	Content
Event	TicketCreate
Active	1
Block	Input
Column	Other Settings
Data:	$Env{"UserDynamicField_Field1"}
Key:	My Field 1
Label:	Default value for: My Field 1

Key	Content
Module:	Kernel::Output::HTML::PreferencesGeneric
PrefKey:	UserDynamicField_Field1
Prio:	7000



Edit your preferences

User Profile

Change password
Current password:
New password:
Verify password:
Update

Language
Frontend language: English (United States) Update

Skin
Wear this frontend skin: Default Update

Theme
Frontend theme: Standard Update

Out Of Office Time
Active: ○ On ● Off
Start: 02 / 14 / 2012
End: 02 / 15 / 2012
Update

Email Settings

New ticket notification
Send me a notification if there is a new ticket in "My Queues".
Send new ticket notifications: No Update

Ticket follow up notification
Send me a notification if a customer sends a follow up and I'm the owner of the ticket or the ticket is unlocked and is in one of my subscribed queues.
Send ticket follow up notifications: No Update

Ticket lock timeout notification
Send me a notification if a ticket is unlocked by the system.
Send ticket lock timeout notifications: No Update

Ticket move notification
Send me a notification if a ticket is moved into one of "My Queues".
Send ticket move notifications: No Update

Ticket watch notification
Send me the same notifications for my watched tickets that the ticket owners will get.
Send ticket watch notifications: No Update

Other Settings

My Queues
Your queue selection of your favorite queues. You also get notified about th
email if enabled.
My Queues: Junk, Misc, Postmaster, Raw
Update

Overview Refresh Time
If enabled, the different overviews (Dashboard, LockedView, QueueView) v
refresh after the specified time.
Refresh Overviews after: off Update

Screen after new ticket
Show this screen after I created a new ticket: CreateTicket Update

Default value for: My Field 1
My Field 1: Update

Powered by OTRS 3.1.x CVS

✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂

11. Generic Interface

The OTRS Generic Interface consists of a multiple layer framework that lets OTRS communicate with other systems via a web service. This communication could be bi-directional:

- ✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂✂✂ OTRS acts as a server listening to requests from the External System, processing the information, performing the requested action, and answering the request.
- ✂✂✂✂ ✂✂ ✂✂✂✂✂✂✂✂✂✂ OTRS acts as a client collecting information, sending the request to the Remote System, and waiting for the response.

11.1. Generic Interface Layers

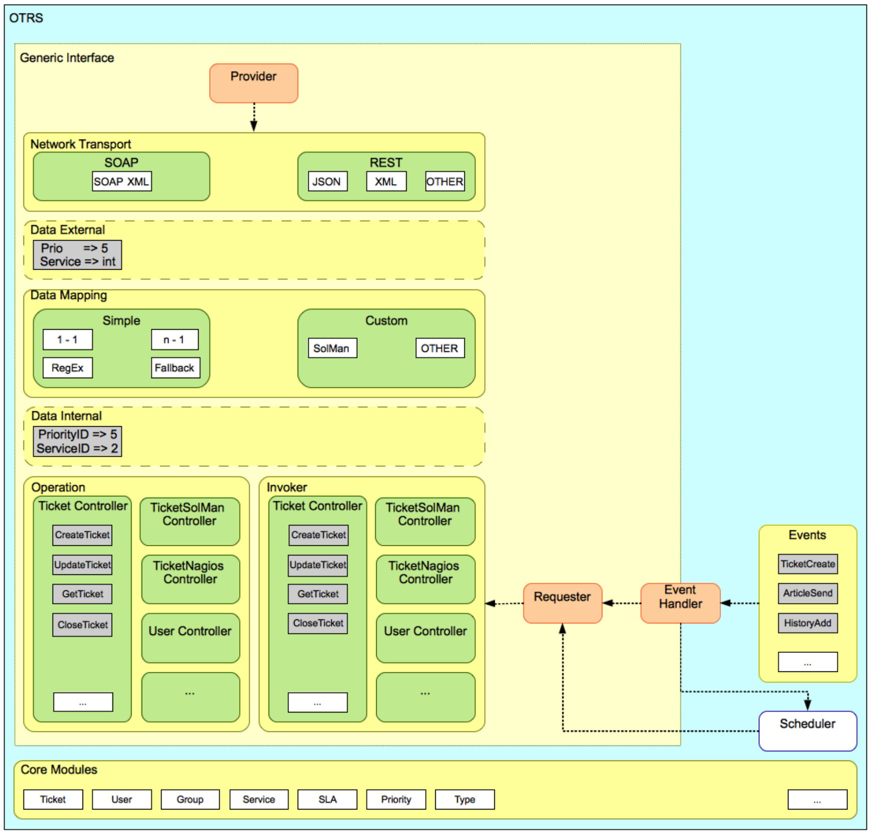
Generic Interface is build based on a layer model, to be flexible and easy to customize.

A layer is a set of files, which control how the Generic Interface performs different parts of a web service. Using the right configuration, one can build different web services for different External Systems without creating new modules.

Note

If the Remote System does not support the current bundled modules of the Generic Interface, special modules need to be developed for that specific web service.

The list of provided Generic Interface modules shipped with OTRS will be updated and increased over time.


OTRS
Generic Interface
Provider
Network Transport
SOAP
SOAP XML
REST
JSON
XML
OTHER
Data External
Prio => 5
Service => int
Data Mapping
Simple
1 - 1
n - 1
RegEx
Fallback
Custom
SolMan
OTHER
Data Internal
PriorityID => 5
ServiceID => 2
Operation
Ticket Controller
CreateTicket
UpdateTicket
GetTicket
CloseTicket
...
TicketSolMan Controller
TicketNagios Controller
User Controller
...
Invoker
Ticket Controller
CreateTicket
UpdateTicket
GetTicket
CloseTicket
...
TicketSolMan Controller
TicketNagios Controller
User Controller
...
Requester
Event Handler
Events
TicketCreate
ArticleSend
HistoryAdd
...
Scheduler
Core Modules
Ticket
User
Group
Service
SLA
Priority
Type
...

11.1.1. Network Transport

This layer is responsible for the correct communication with the Remote System. It receives requests and generates responses when acting as provider, and generates requests and receives responses when acting as requester.

Provider communication is handled by a new web server handle called "nph-genericinterface.pl".

Requester communication could be initiated during an event triggered by a Generic Interface module or any other OTRS module. This event is catched by the event handler and depending on the configuration the event will be processed directly by the requester object or delegated to the Scheduler (a separated deamon designed to process tasks asynchronously).

11.1.2. Data Mapping

This layer is responsible for translating data structures between OTRS and the Remote System (data internal and data external layers). Usually Remote Systems have different data structures than OTRS (including different values and names for those values), and here resides the importance of the layer to change the received information into something that OTRS can understand and on the opposite way send the information to each Remote System using their data dictionaries.

✃✃✃✃✃✃✃✃ "Priority" (OTRS) might be called "Prio" in a remote system and it could be that value "1 Low" (OTRS) should be mapped to "Information" on the remote system.

11.1.3. Controller

Controllers are collections of similar Operations or Invokers. For example, a Ticket controller might contain several standard ticket operations. Custom controllers can be implemented, for example a "TicketExternalCompany" controller which may contain similar functions as the standard Ticket controller, but with a different data interface, or function names (to adapt to the Remote System function names) or complete different code.

One application for Generic Interface could be to synchronize information with one Remote System that only can talk with another Remote System of the same kind. In this case new controllers needs to be developed and the Operations and Invokers has to emulate the Remote System behavior in such way that the interface that OTRS exposes is similar to the Remote System's interface.

11.1.4. Operation (OTRS as a provider)

An Operation is a single action that can be performed within OTRS. All operations have the same programming interface, they receive the data into one specific parameter, and return a data structure with a success status, potential error message and returning data.

Normally operations uses the already mapped data (internal) to call core modules and perform actions in OTRS like: Create a Ticket, Update a User, Invalidate a Queue, Send a Notification, etc. An operation has full access to the OTRS API to perform the action.

11.1.5. Invoker (OTRS as a requester)

An Invoker is an action that OTRS performs against a Remote System. Invokers use the OTRS Core modules to process and collect the needed information to create the request. When the information is ready it has to be mapped to the Remote System format in order to be sent to the Remote System, that will process the information execute the action and send the response back, to either process the success or handle errors.

11.2. Generic Interface Communication Flow

The Generic Interface has a defined flow to perform actions as a provider and as a requester.

These flows are described below:

11.2.1. OTRS as Provider

11.2.1.1. Remote Request:

1. HTTP request
   - OTRS receives HTTP request and passes it through the layers.
   - The provider module is in charge to execute and control these actions.
2. Network Transport
   - The network transport module decodes the data payload and separates the operation name from the rest of the data.

    - The operation name and the operation data are returned to the provider.

3. ✁✁✁✁ ✁✁✁✁✁✁✁✁
    - Data as sent from the remote system (This is not a module-based layer).
4. Mapping
    - The data is transformed from the External System format to the OTRS internal format as specified in the mapping configuration for this operation (Mapping for incoming request data).
    - The already transformed data is returned to the provider.
5. ✁✁✁✁ ✁✁✁✁✁✁✁✁
    - Data as transformed and prepared to be passed to the operation (This is not a module based layer).
6. Operation
    - Receives and validates data.
    - Performs user access control.
    - Executes the action.

11.2.1.2. OTRS Response:

1. Operation
    - Returns result data to the provider.
2. ✁✁✁✁ ✁✁✁✁✁✁✁✁
    - Data as returned from operation.
3. Mapping
    - The data is transformed back to the Remote system format as specified in the mapping configuration (Mapping for outgoing response data).
    - The already transformed data is returned to the provider.
4. ✁✁✁✁ ✁✁✁✁✁✁✁✁
    - Data as transformed and prepared to be passed to Network Transport as response.
5. Network Transport
    - Receives the data already in the Remote System format.
    - Constructs a valid response for this network transport type.
6. HTTP response
    - The response is sent back to the web service client.

- In the case of an error, an error response is sent to the remote system (e.g. SOAP fault, HTTP error, etc).

11.2.2. OTRS as Requester

11.2.2.1. OTRS Request:

1. Event Trigger Handler
   - Based on the web service configuration determines if the request will be synchronous or asynchronous.
     - Synchronous
       - A direct call to the Requester is made in order to create a new request and to pass it through the layers.
     - Asynchronous
       - Create a new Generic Interface (Requester) task for the OTRS Scheduler (by delegating the request execution to the Scheduler, the user experience could be highly improved, otherwise all the time needed to prepare the request and the remote execution will be added to the OTRS Events that trigger those requests).
       - In its next cycle the Scheduler process reads the new task and creates a call to the Requester that will create a new request and then passes it through the layers.
2. Invoker
   - Receives data from the event.
   - Validates received data (if needed).
   - Call core modules to complement the data (if needed).
   - Return the request data structure or send a Stop Communication signal to the requester, to gracefully cancel the request.
3. ✂✂✂✂ ✂✂✂✂✂✂✂✂
   - Data as passed from the invoker (This is not a module based layer).
4. Mapping
   - The data is transformed to the Remote system format as specified in the mapping configuration (Mapping for outgoing response data).
   - The already transformed data is returned to the requester.
5. ✂✂✂✂ ✂✂✂✂✂✂✂✂
   - Data as transformed and prepared for sending to the remote system.
6. Network Transport
   - Receives the remote operation name and the data already transformed to the Remote System format from the requester.

- Constructs a valid request for the network transport.
- Sends the request to the remote system and waits for the response

11.2.2.2. Remote Response:

1. Network transport
   - Receives the response and decodes the data payload.
   - Returns the data to the requester.
2. ✂✂✂✂ ✂✂✂✂✂✂✂✂
   - Data as received from the Remote System
3. Mapping
   - The data is transformed from the External System format to the OTRS internal format as specified in the mapping configuration for this operation (Mapping for incoming response data).
   - The already transformed data is returned to the requester.
4. ✂✂✂✂ ✂✂✂✂✂✂✂✂
   - Data as transformed and ready to be passed back to the requester.
5. Invoker
   - Receives return data.
   - Handles the data as needed specifically by each Invoker (included error handling if any).
   - Return the Invoker result and data to the Requester.
6. Event Handler or Scheduler
   - Receives the data from the Requester, in the case of the Scheduler this data might contain information to Re-Schedule the task immediately or in the future.

11.3. Web Services

A Web Service is a communication method between two systems, in our case OTRS and a Remote System.

The heart of the Web Service is its configuration, where it is defined what actions the web service can perform internally (Operation), what actions the OTRS request can perform Remote System (Invokers), how data is converted from one system to the other (Mapping), and over which protocol the communication will take place (Transport)

The Generic Interface is the framework that makes it possible to create Web Services for OTRS in a pre-defined way, using already made building blocks that are independent from each other and interchangeable.

11.4. Web Service Graphical Interface

The web service graphical user interface (GUI) is a tool that allows to construct complex web service configurations in a user friendly and convenient interface. It allows to:

- Create and Delete web services.
- Import and Export configurations (in YAML file format) for existing web services.
- View, Revert and Export old configurations for existing web services in the Web Service History screen.
- Track all communication logs for each web service in the Debugger screen.

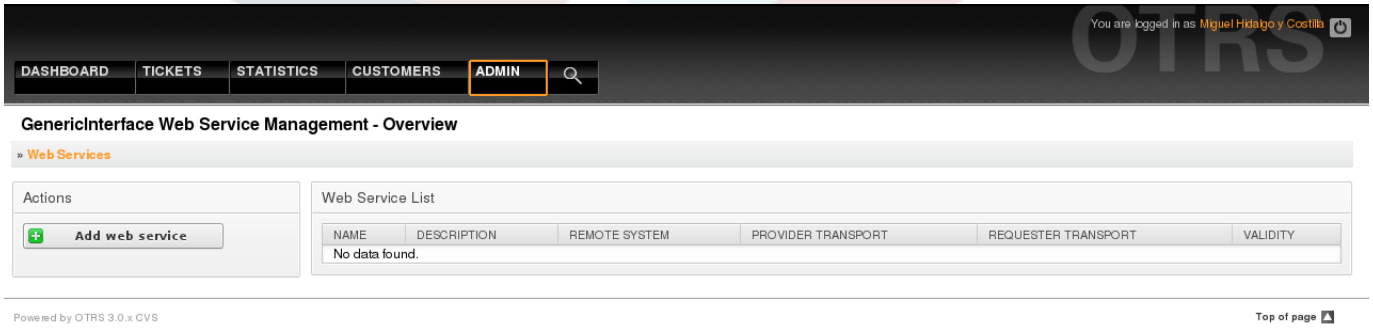
11.4.1. Web Service Overview

The "Web Services" link in the main screen of Admin Interface (in the System Administration box) leads to the web services overview screen, where you are able to manage your web service configurations. You can add new web services or change the configuration of the existing ones from this screen.

Every web service configuration screen has in the upper part of the screen a "bread crumbs" style navigation path. This navigation path is useful to know exactly in which part of the web service configuration we are, and also enables the user to jump back to any part of the configuration process at any time (this action will not save any changes).

Note

To create a new web service, press the button "Add web service", and provide the required information.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Add web service
Powered by OTRS 3.0.x CVS
Top of page

GenericInterface Web Service Management - Overview

» Web Services

Web Service List

NAME	DESCRIPTION	REMOTE SYSTEM	PROVIDER TRANSPORT	REQUESTER TRANSPORT	VALIDITY
No data found.					

✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂

11.4.2. Web Service Add

The only required field in this part is the web service "Name" that needs to be unique in the system and can not be left empty. Other fields are also necessary for the configuration like the "Debug Threshold" and "Validity" but these fields are already populated with the default value for each list.

The default value for "Debug Threshold" is "debug". When configured in this manner all communication logs are registered in the database. Each subsequent Debug Threshold value is more restrictive and discards communication logs of lower order than the one set in the system.

Debug Threshold levels (from lower to upper)

- Debug
- Info
- Notice
- Error

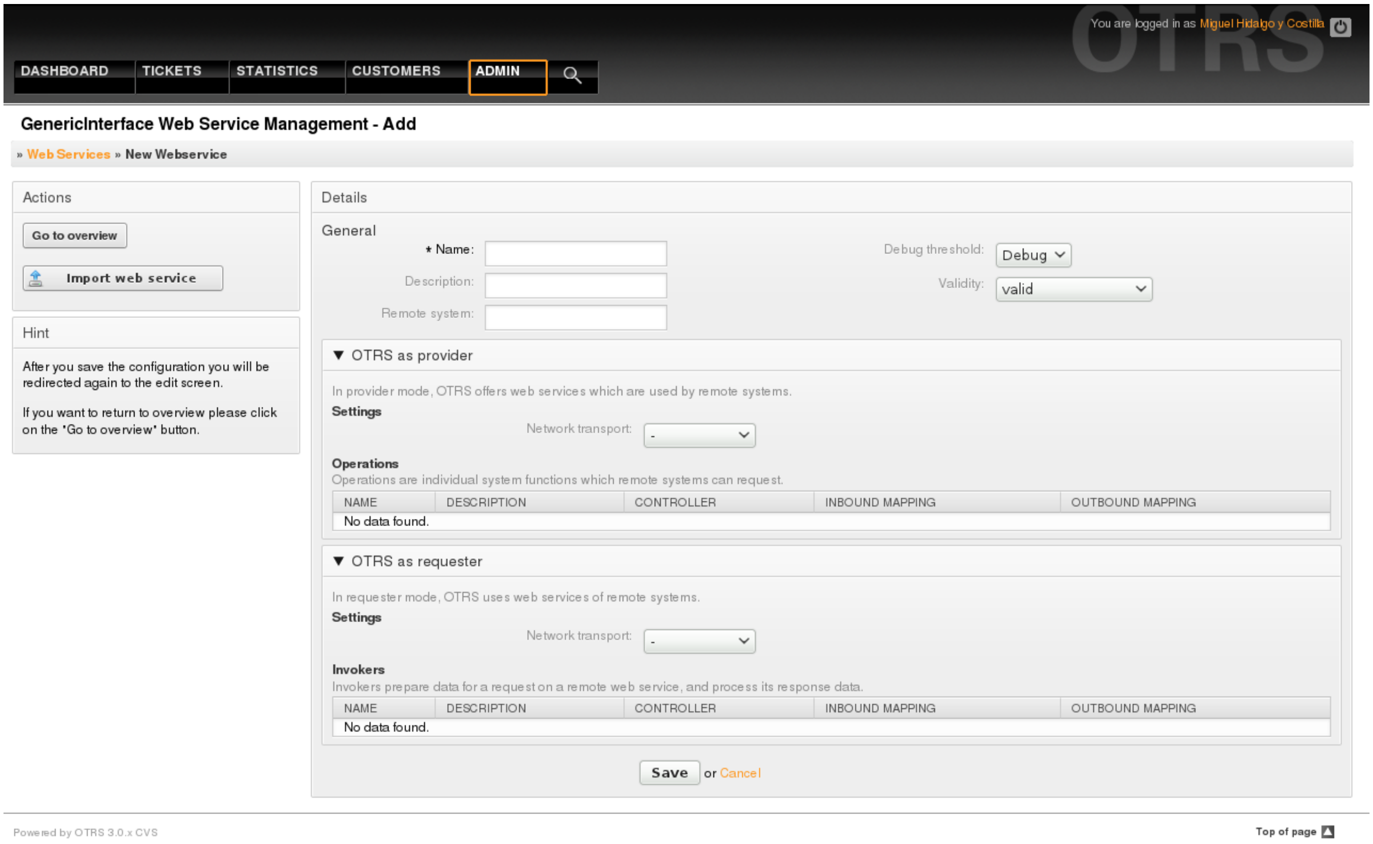
It is also possible to define the network transport protocol for "OTRS as Provider" and "OTRS as requester".

Click on the "Save" button to register the new web service in the database or click "Cancel" to discard this operation. You will now be returned to the web service overview screen.

If you already have a web service configuration file in YAML format you can click on the "Import web service" button on the left side of the screen. For more information on importing web services please check the next section "Web Service Change".

Note

To change or add more details to a web service, click on the web service name in the web service overview screen.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
GenericInterface Web Service Management - Add
» Web Services » New Webservice
Actions
Go to overview
Import web service
Hint
Details
General
* Name:
Description:
Remote system:
Debug threshold:
Debug
Validity:
valid
▼ OTRS as provider
Settings
Network transport:
-
Operations
NAME
DESCRIPTION
CONTROLLER
INBOUND MAPPING
OUTBOUND MAPPING
▼ OTRS as requester
Settings
Network transport:
-
Invokers
NAME
DESCRIPTION
CONTROLLER
INBOUND MAPPING
OUTBOUND MAPPING
Save
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

After you save the configuration you will be redirected again to the edit screen.

If you want to return to overview please click on the "Go to overview" button.

In provider mode, OTRS offers web services which are used by remote systems.

Operations are individual system functions which remote systems can request.

No data found.

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

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

No data found.

11.4.3. Web Service Change

On this screen you have a complete set of functions to handle every part of a web service. On the left side in the action column you can find some buttons that allows you to perform all possible actions on a web service:

- Clone web service.
- Export web service.
- Import web service.
- Configuration History.
- Delete web service.
- Debugger.

Note

"Configuration history" and "Debugger" will lead you to different screens.

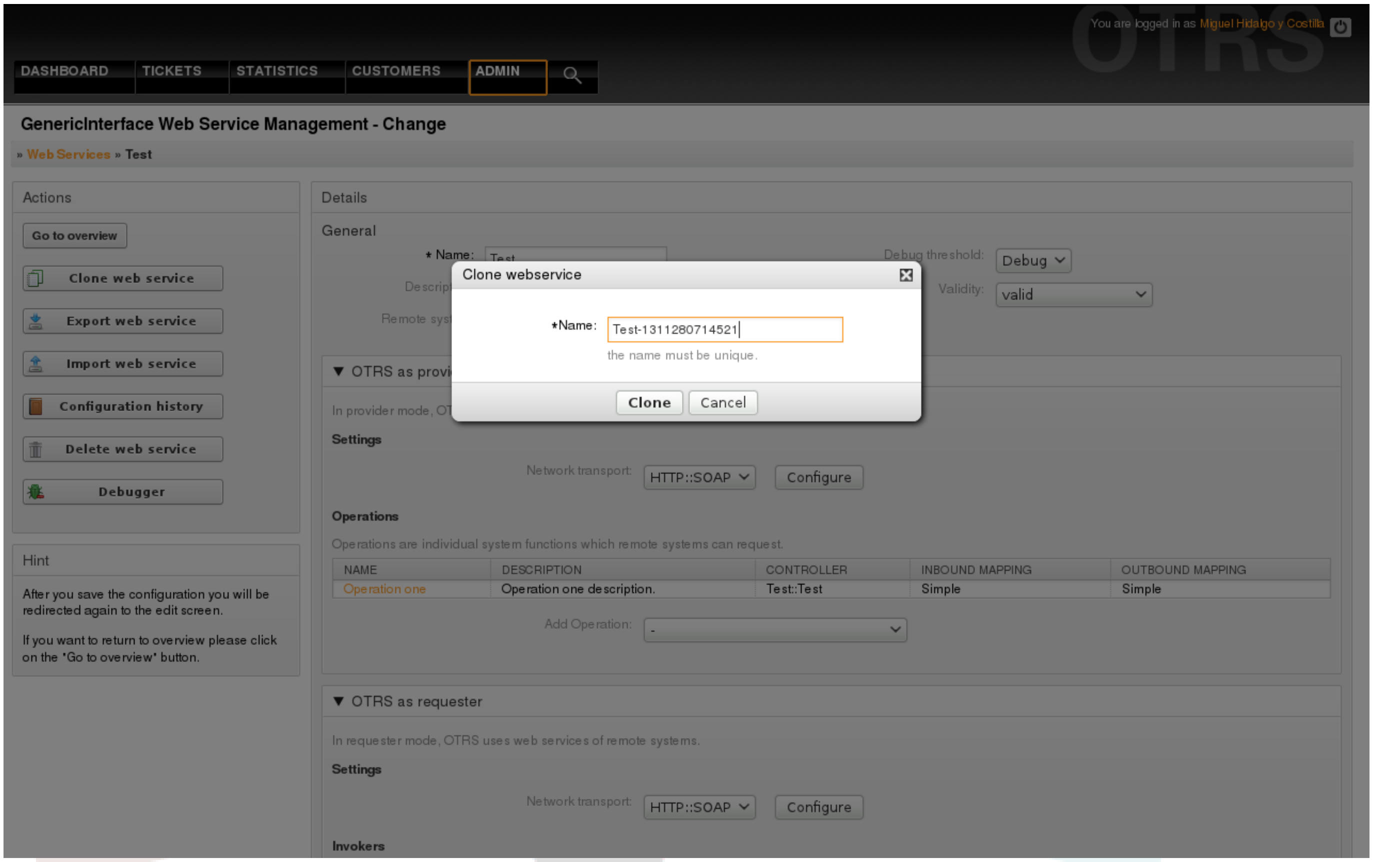
11.4.3.1. Web Service Clone

To clone a web service, you need to click on the "Clone web service" button. A dialog will be shown where you can use the default name or set a new name for the (cloned) web service.

Note

✁✁✁✁✁✁✁✁ that the name of the web service must be unique within the system.

Click on "Clone" button to create the web service clone or "Cancel" to close the dialog.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Go to overview
Clone web service
Export web service
Import web service
Configuration history
Delete web service
Debugger
Clone
Cancel
Configure

GenericInterface Web Service Management - Change

» Web Services » Test

Actions

Hint

After you save the configuration you will be redirected again to the edit screen.

If you want to return to overview please click on the "Go to overview" button.

Details

General

* Name: Test

Debug threshold: Debug

Validity: valid

Clone webservice

*Name: Test-1311280714521

the name must be unique.

▼ OTRS as provi

In provider mode, OT

Settings

Network transport: HTTP::SOAP

Operations

Operations are individual system functions which remote systems can request.

NAME	DESCRIPTION	CONTROLLER	INBOUND MAPPING	OUTBOUND MAPPING
Operation one	Operation one description.	Test::Test	Simple	Simple

Add Operation: -

▼ OTRS as requester

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

Settings

Network transport: HTTP::SOAP

Invokers

✁✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁✁ ✁✁✁✁✁✁

11.4.3.2. Web Service Export

The "Export web service" button gives you the opportunity to dump the configuration of the current web service into a YAML file, to download it and to store it on your file system. This can be specially useful if you want to migrate the web service from one server to another, for example from a testing environment to a production system.

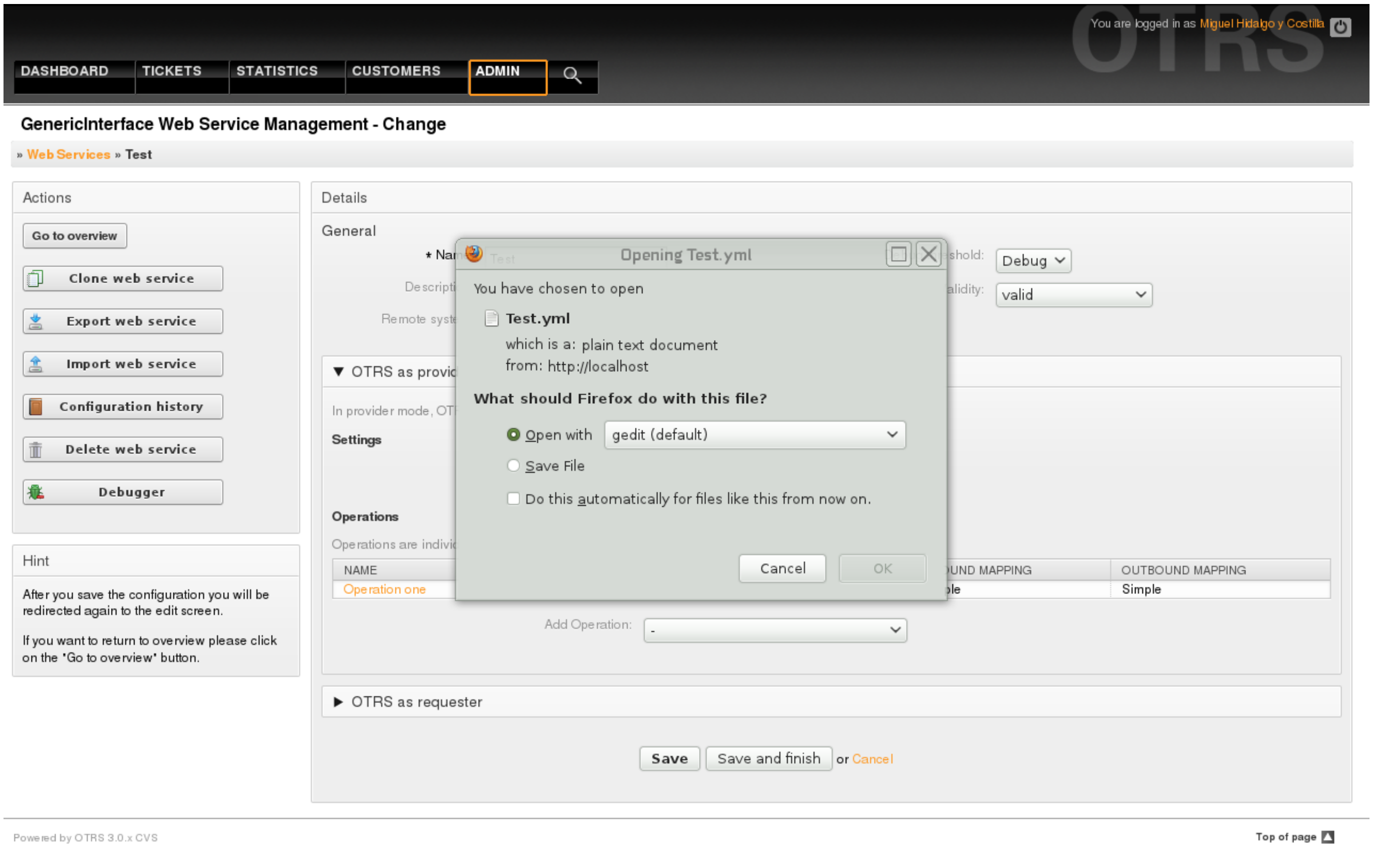
Warning

All stored passwords in the web service configuration will be exported in plain text format.

Right after clicking the "Export web service" button a save dialog of your browser will appear, just like when you click on a file download link on a web page.

Note

Each browser on each operating system has its own save dialog screen and style. Depending on the browser and its configuration it is possible that no dialog is shown and the file is saved to a default directory on your file system. Please check your browser documentation for more specific instructions if needed.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
GenericInterface Web Service Management - Change
» Web Services » Test
Actions
Go to overview
Clone web service
Export web service
Import web service
Configuration history
Delete web service
Debugger
Details
General
OTRS as provid
Settings
Operations
NAME
Operation one
UND MAPPING
OUTBOUND MAPPING
Simple
Add Operation:
OTRS as requester
Save
Save and finish
or Cancel
Opening Test.yml
Cancel
OK
Powered by OTRS 3.0.x CVS
Top of page

Hint

After you save the configuration you will be redirected again to the edit screen.

If you want to return to overview please click on the "Go to overview" button.

Debug

valid

You have chosen to open

Test.yml

which is a: plain text document
from: http://localhost

What should Firefox do with this file?

● Open with gedit (default)
○ Save File
☐ Do this automatically for files like this from now on.

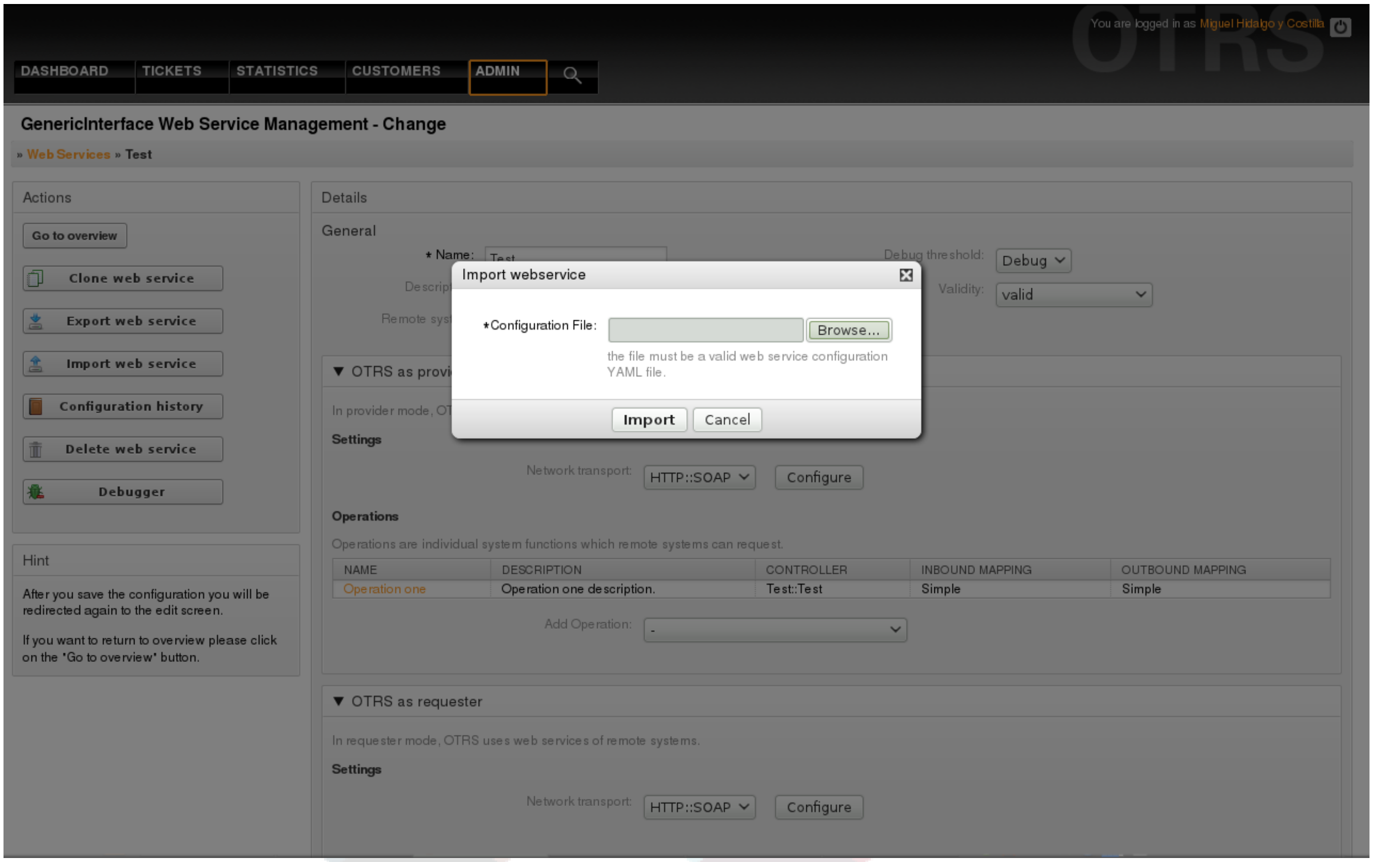
11.4.3.3. Web Service Import

A valid web service configuration YAML file is required to use the import web service feature. Click on the "Import web service" button, browse for the configuration file or provide the complete path in the input box.

Click "Import" button to create a new web service from a file or "Cancel" to close the dialog.

Note

The web service name will be taken from the configuration file name (e.g. if the file name is MyWebservice.yml the resulting web service will be named MyWebservice). If a web service is registered in the system with the same name as the web service that you want to import, the system will lead you to the web service change screen to let you change the name of the imported web service.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
GenericInterface Web Service Management - Change
» Web Services » Test
Actions
Go to overview
Clone web service
Export web service
Import web service
Configuration history
Delete web service
Debugger
Hint
After you save the configuration you will be redirected again to the edit screen.
If you want to return to overview please click on the "Go to overview" button.
Details
General
Name: Test
Debug threshold: Debug
Validity: valid
Import webservice
Configuration File:
Browse...
the file must be a valid web service configuration YAML file.
Import
Cancel
OTRS as provider
Settings
Network transport: HTTP::SOAP
Configure
Operations
Operations are individual system functions which remote systems can request.
NAME
DESCRIPTION
CONTROLLER
INBOUND MAPPING
OUTBOUND MAPPING
Operation one
Operation one description.
Test::Test
Simple
Simple
Add Operation: -
OTRS as requester
In requester mode, OTRS uses web services of remote systems.
Settings
Network transport: HTTP::SOAP
Configure

11.4.3.4. Web Service History

Every change to the web service configuration creates a new entry in the web service history (as a journal). The web service history screen displays a list of all configuration versions for a web service. Each row (version) in the "Configuration History List" represents a single revision in the web service history.

Click on one of the rows to show the whole configuration as it was on that particular date / time. The configuration will be shown in the "History details" section of this screen. Here you are also able to export the selected web service configuration version or to restore that version into the current web service configuration.

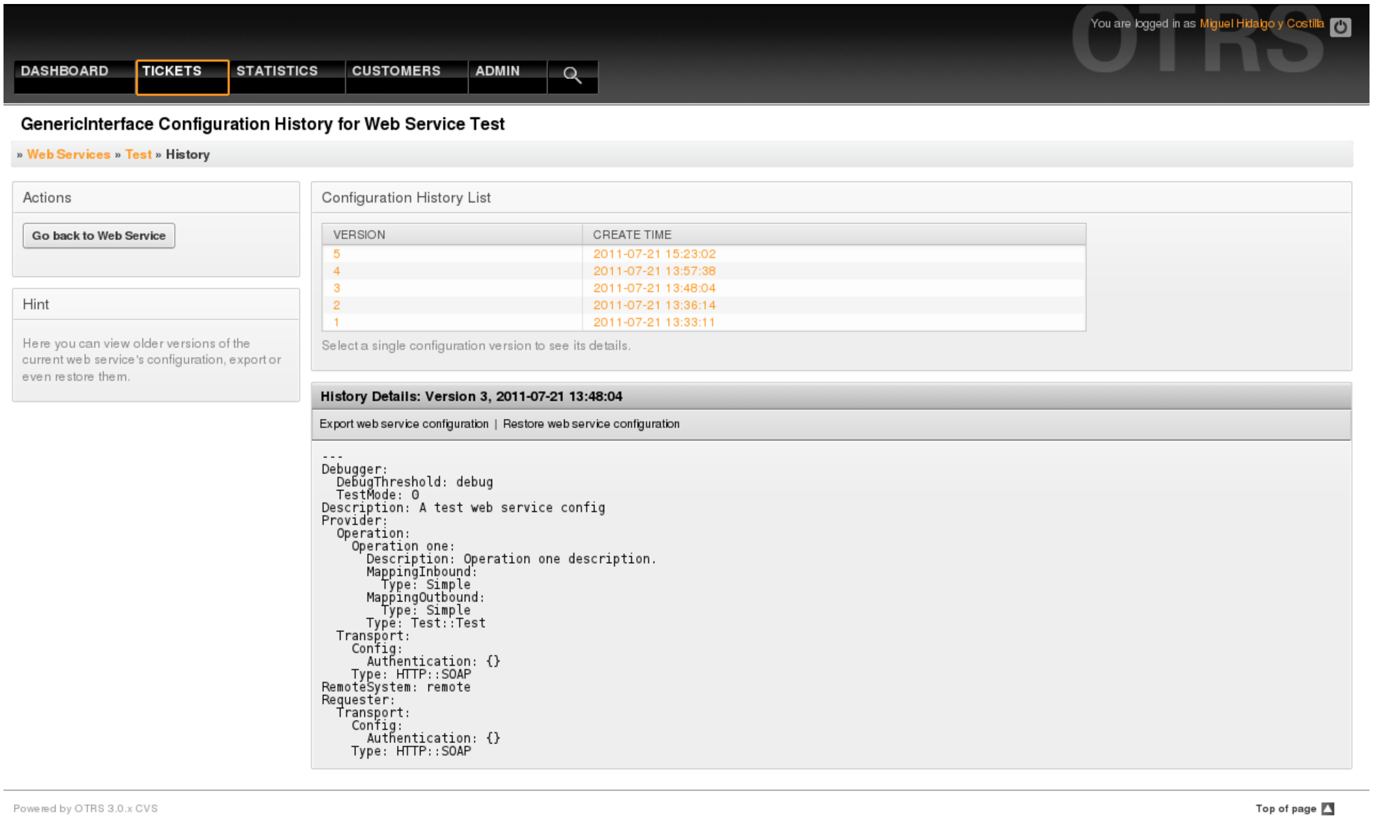
The "Export web service configuration" behaves exactly as the "Export web service" feature in the web service change screen. For more information refer to that section.

If changes to the current web service configuration do not work as expected and it is not easy to revert the changes manually, you can click on the "Revert web service configuration" button. This will open a dialog to ask you if you are sure to revert the web service configuration. Click "Revert web service configuration" in this dialog to replace the current configuration with the selected version, or click "Cancel" to close the dialog.

Warning

Remember that any passwords stored in the web service configuration will be exported in plain text format.

Please be careful when you restore a configuration because this the process is irreversible.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Go back to Web Service
Hint
Powered by OTRS 3.0.x CVS
Top of page

GenericInterface Configuration History for Web Service Test

» Web Services » Test » History

Here you can view older versions of the current web service's configuration, export or even restore them.

Configuration History List

VERSION	CREATE TIME
5	2011-07-21 15:23:02
4	2011-07-21 13:57:38
3	2011-07-21 13:48:04
2	2011-07-21 13:36:14
1	2011-07-21 13:33:11

Select a single configuration version to see its details.

History Details: Version 3, 2011-07-21 13:48:04

Export web service configuration | Restore web service configuration

```
---
Debugger:
  DebugThreshold: debug
  TestMode: 0
Description: A test web service config
Provider:
  Operation:
    Operation one:
      Description: Operation one description.
      MappingInbound:
        Type: Simple
      MappingOutbound:
        Type: Simple
      Type: Test::Test
  Transport:
    Config:
      Authentication: {}
    Type: HTTP::SOAP
RemoteSystem: remote
Requester:
  Transport:
    Config:
      Authentication: {}
    Type: HTTP::SOAP
```

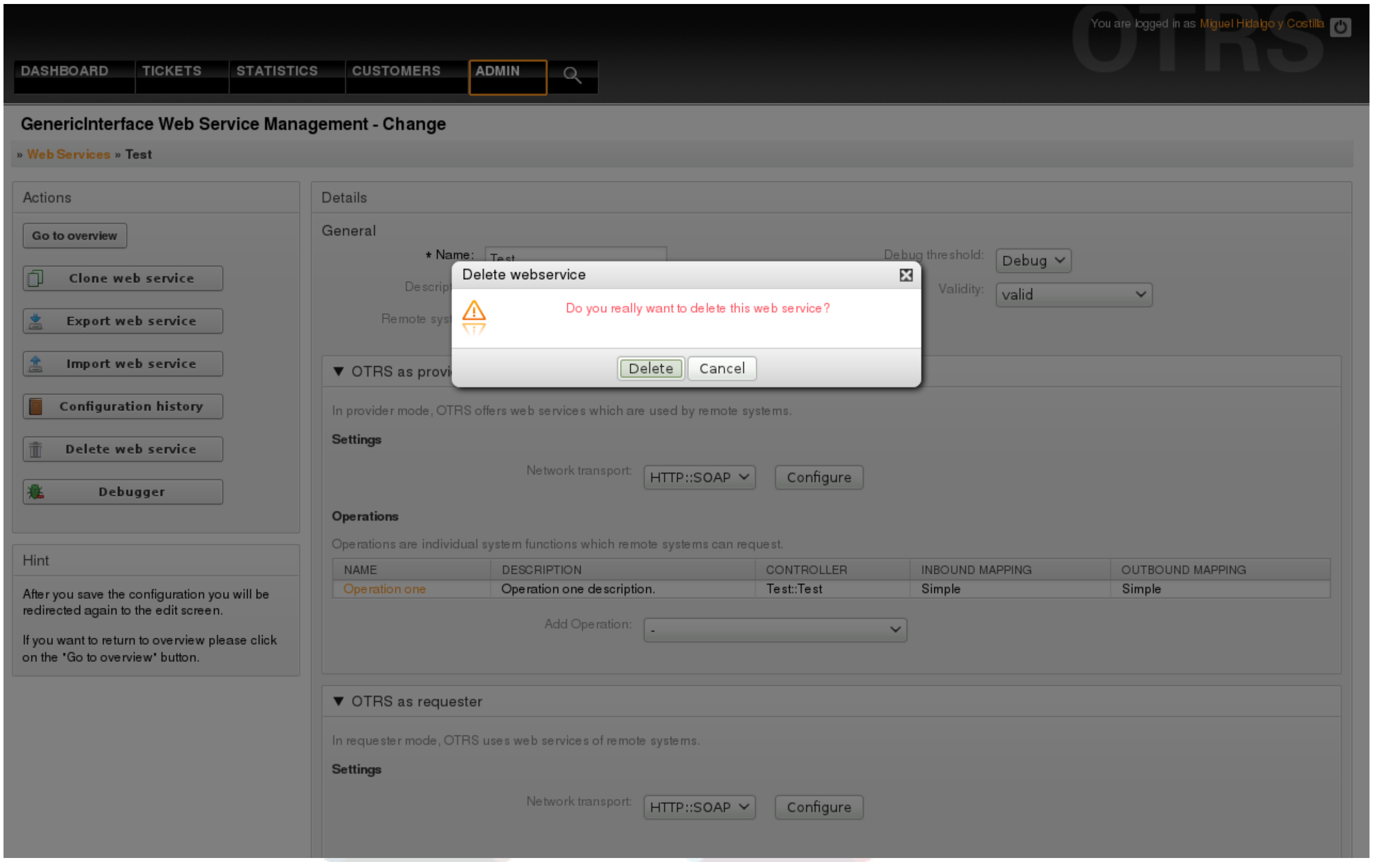
11.4.3.5. Web Service Delete

Sometimes it is necessary to delete a web service completely. To do this you can press on the "Delete web service" button and a new dialog will appear asking for confirmation.

Clink on "Delete" to confirm the removal of the web service or on "Cancel" to close the dialog.

Warning

Deleting a web service can't be undone, please be careful when deleting a web service.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
Actions
Go to overview
Clone web service
Export web service
Import web service
Configuration history
Delete web service
Debugger
Details
General
Delete webservice
Delete
Cancel
OTRS as provider
OTRS as requester
Configure
Configure

GenericInterface Web Service Management - Change

» Web Services » Test

Hint

After you save the configuration you will be redirected again to the edit screen.

If you want to return to overview please click on the "Go to overview" button.

* Name: Test

Debug threshold: Debug

Validity: valid

Do you really want to delete this web service?

In provider mode, OTRS offers web services which are used by remote systems.

Settings

Network transport: HTTP::SOAP

Operations

Operations are individual system functions which remote systems can request.

NAME	DESCRIPTION	CONTROLLER	INBOUND MAPPING	OUTBOUND MAPPING
Operation one	Operation one description.	Test::Test	Simple	Simple

Add Operation: -

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

Settings

Network transport: HTTP::SOAP

11.4.3.6. Web Service Debugger

The Debugger stores the log of a web service. In the debugger screen you can track all the web service communications for either provider or requester types.

When this screen is shown the request list starts to load. After the list is fully filled you can choose one of the rows (that means a communication sequence) to check its details. This details will appear in a box below.

You can narrow the communication list using the filter on the right part of the screen. You can filter by:

- Communication type (provider or requester)
- Date: before and / or after a particular date
- The remote IP Address
- A combination of all.

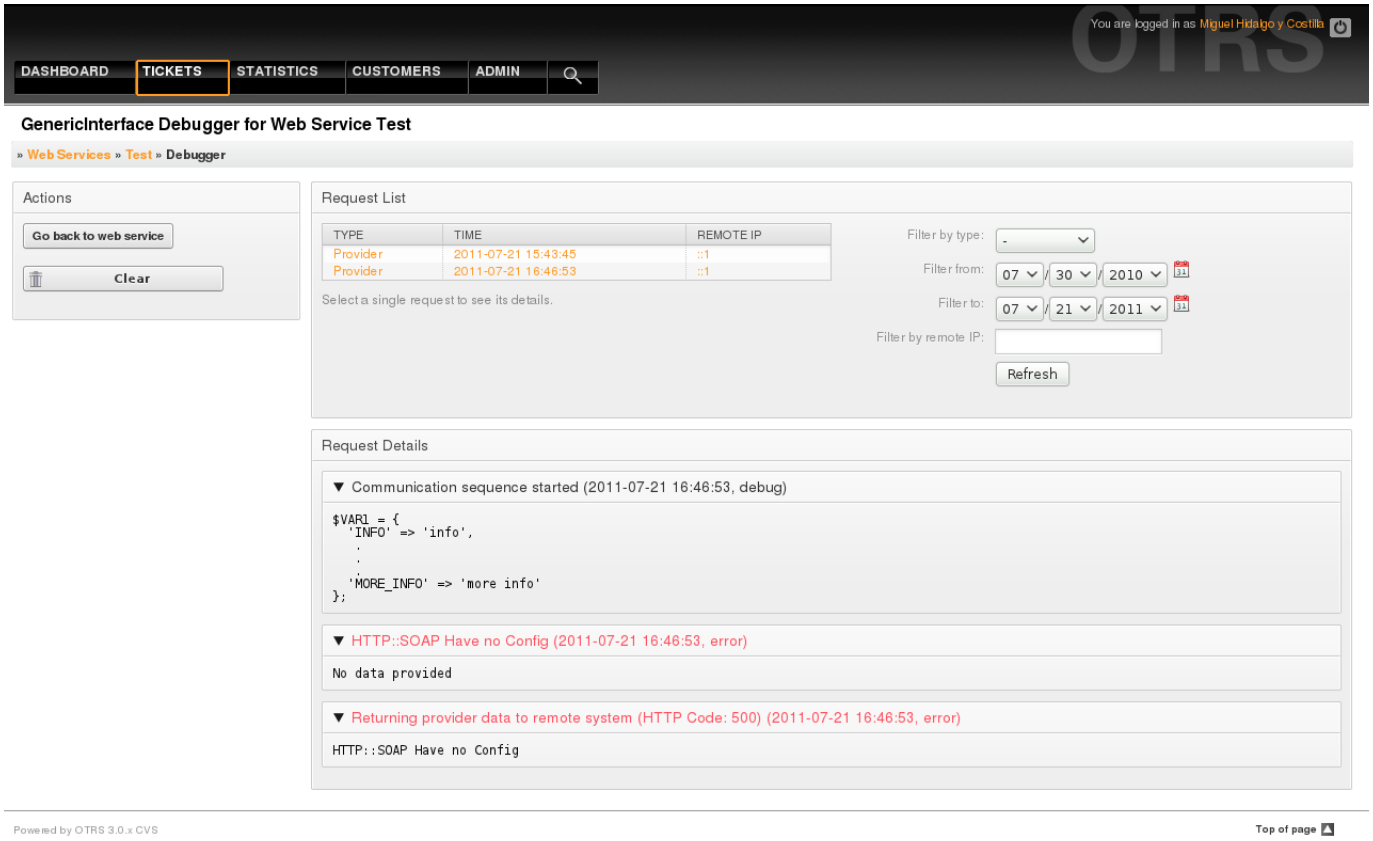
After filter settings are set, push the "Refresh" button and a new list will be displayed meeting your search criteria.

Note

Depending on the search criteria for the filters the new list could return no results.

On the left part of the screen under the action column you can select "Go back to the web service" or clear the debugger log by pushing the "Clear" button. This will open a dialog that ask you to confirm erasing of the log. Click "Clear" in the dialog button to perform the action or click on "Cancel" to close this dialog.

In the "Request details" section you can see all the details for the selected communication. Here you can track the complete flow and check for possible errors or confirm success responses.



GenericInterface Debugger for Web Service Test

» Web Services » Test » Debugger

TYPE	TIME	REMOTE IP
Provider	2011-07-21 15:43:45	::1
Provider	2011-07-21 16:46:53	::1

Select a single request to see its details.

Filter by type: -
Filter from: 07 / 30 / 2010
Filter to: 07 / 21 / 2011
Filter by remote IP:

▼ Communication sequence started (2011-07-21 16:46:53, debug)

```
$VAR1 = {
  'INFO' => 'info',
    .
    .
    .
  'MORE_INFO' => 'more info'
};
```

▼ HTTP::SOAP Have no Config (2011-07-21 16:46:53, error)

```
No data provided
```

▼ Returning provider data to remote system (HTTP Code: 500) (2011-07-21 16:46:53, error)

```
HTTP::SOAP Have no Config
```

11.4.3.7. Web Service Configuration Change

Returning to the web service change screen, now we are going to review the right side of it. Here we have the possibility to modify all the general data for a web service such as name, description, debug threshold, etc. Also there are two more sections below that allows us to modify specific parameters for communication types "OTRS as Provider" and "OTRS as Requester".

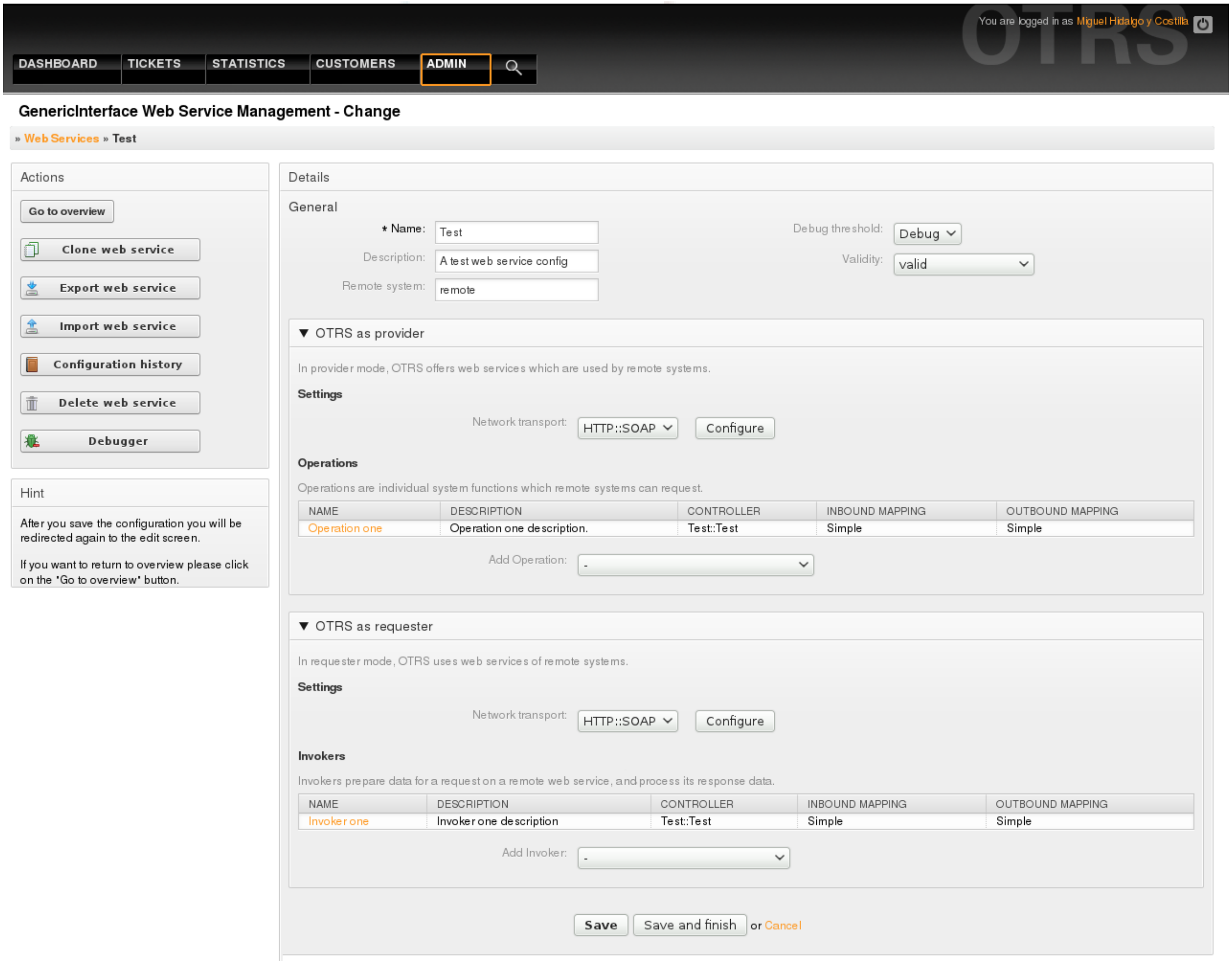
The web service configuration needs to be saved on each level. This means that if a setting is changed, links to other, deeper parts of the configuration will be disabled forcing you to save the current configuration level. After saving the disabled links will be re-enabled again allowing you to continue with the configuration.

On the "OTRS as provider" section it is possible to set or configure the network transport protocol. Only network transport back-ends that are registered are shown on the list. To configure the network transport click on the "Configure" button. It is also possible to add new operations in this box. To do this select one of the available operations from the "Add

Operation" list. This will lead you to the operation configuration screen. After saving the new operation it will be listed in the table above.

"OTRS as requester" is very similar to the previous one, but instead of "operations" you can add invokers here.

Click the "Save" button to save and continue configuring the web service, "Save and finish" to save and return to the web service overview screen, or "Cancel" to discard current configuration level changes and return to web service overview screen.


You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
OTRS
Actions
Go to overview
Clone web service
Export web service
Import web service
Configuration history
Delete web service
Debugger
Details
Save
Save and finish
or Cancel

GenericInterface Web Service Management - Change

» Web Services » Test

Hint

After you save the configuration you will be redirected again to the edit screen.

If you want to return to overview please click on the "Go to overview" button.

General

* Name: Test

Description: A test web service config

Remote system: remote

Debug threshold: Debug

Validity: valid

▼ OTRS as provider

In provider mode, OTRS offers web services which are used by remote systems.

Settings

Network transport: HTTP::SOAP Configure

Operations

Operations are individual system functions which remote systems can request.

NAME	DESCRIPTION	CONTROLLER	INBOUND MAPPING	OUTBOUND MAPPING
Operation one	Operation one description.	Test::Test	Simple	Simple

Add Operation: -

▼ OTRS as requester

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

Settings

Network transport: HTTP::SOAP Configure

Invokers

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

NAME	DESCRIPTION	CONTROLLER	INBOUND MAPPING	OUTBOUND MAPPING
Invoker one	Invoker one description	Test::Test	Simple	Simple

Add Invoker: -

Note

Like the other Generic Interface configuration screens such as Network Transport, Operation, Invoker and Mapping, the initial configuration (add) screen will only present two options: "Save" and "Cancel". If the configuration is re-visited, a new option "Save and Finish" will appear. The behavior of this feature is defined below.

"Save" will store the current configuration level in the database and it will return to the previous screen to review your changes or to configure deeper settings.

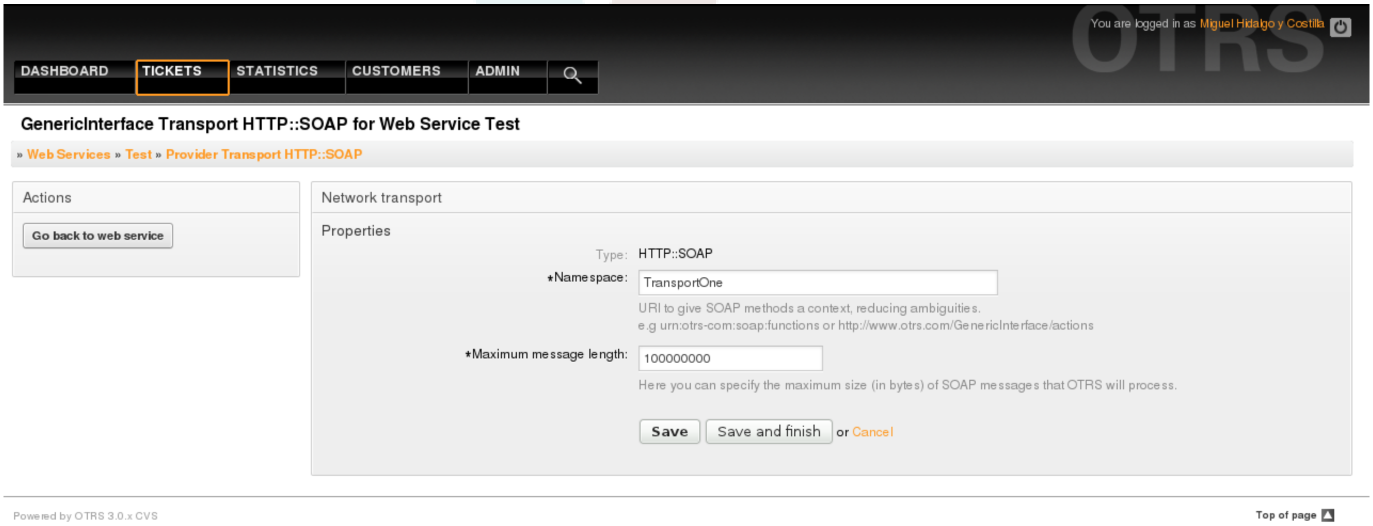
"Save and Finish" will store the current configuration level in the database and it will return to the previous screen in the configuration hierarchy (to the immediate upper configuration level).

"Cancel" will discard any configuration change to the current configuration level and will return to the previous screen in the configuration hierarchy.

11.4.3.7.1. Web Service Provider Network Transport

In future the list of available network transports will be increased. Currently only the "HTTP::SOAP" transport is available. Each transport has different configuration options to setup and they might use different frontend modules to configure it, but mostly they should look similar to the "HTTP::SOAP" transport configuration module.

It is quite simple to configure the "HTTP::SOAP" protocol as provider. There are only two settings: "Namespace" and "Maximum message length". These fields are required. The first one is a URI to give SOAP methods a context, reducing ambiguities, and the second one is a field where you can specify the maximum size (in bytes) for SOAP messages that OTRS will process.



You are logged in as Miguel Hidalgo y Costilla
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
» Web Services » Test » Provider Transport HTTP::SOAP
Actions
Go back to web service
Save
Save and finish
or Cancel
Powered by OTRS 3.0.x CVS
Top of page

GenericInterface Transport HTTP::SOAP for Web Service Test

Network transport

Properties

Type: HTTP::SOAP

*Namespace: TransportOne

URI to give SOAP methods a context, reducing ambiguities.
e.g urn:otrs-com:soap:functions or http://www.otrs.com/GenericInterface/actions

*Maximum message length: 100000000

Here you can specify the maximum size (in bytes) of SOAP messages that OTRS will process.

11.4.3.7.2. Web Service Operation

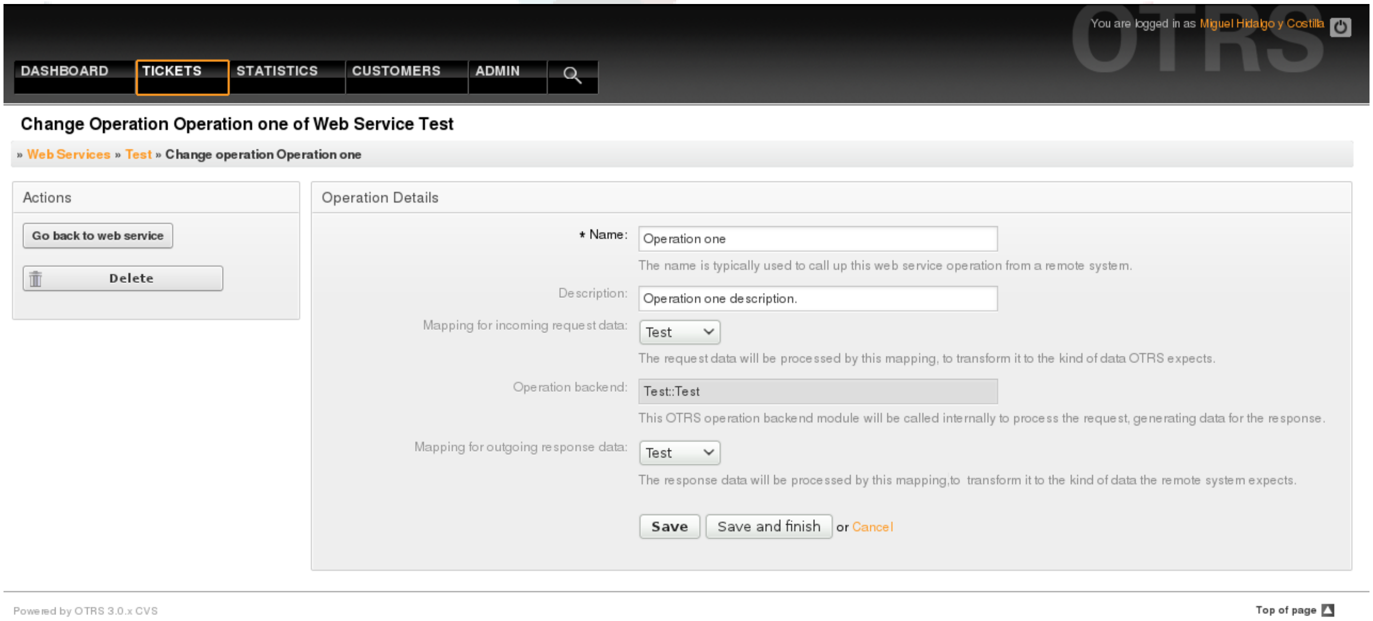
The actions that can be performed when you are using OTRS as a provider are called "Operations". Each operation belongs to a controller. Controllers are collections of operations or invokers, normally operations from the same controller need similar settings and share the same configuration dialog. But each operation can have independent configuration dialogues if needed.

Name, Description, Backend, and Mappings are fields that normally appear on every operation, other special fields can appear in non default configuration dialogues to fulfill specific needs of the operation.

Normally there are two mapping configuration sections on each operation, one for the incoming data and another one for the outgoing data. You can choose different mapping types (backends) for each mapping direction, since their configuration is independent from each other and also independent from the operation backend. The normal and most common practice is that the operation 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.

The operation backend is pre-populated and is not editable. You will see this parameter when you choose the operation on the web service edit screen. The field is only informative.

In the left part of the screen on the action column you have the options: "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 operation. Click on "Delete" button to confirm the removal of the operation and it configuration or "Cancel" to close the delete dialog.



Change Operation Operation one of Web Service Test

» Web Services » Test » Change operation Operation one

Operation Details

* Name: Operation one

The name is typically used to call up this web service operation from a remote system.

Description: Operation one description.

Mapping for incoming request data: Test

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

Operation backend: Test::Test

This OTRS operation backend module will be called internally to process the request, generating data for the response.

Mapping for outgoing response data: Test

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

11.4.3.7.3. Web Service Provider Transport

The network transport configuration for the requester is similar to the configuration for the provider. For the Requester "HTTP::SOAP" network transport there are more fields to be set.

Apart from the "Endpoint" (URI of the Remote System web service interface to accept requests) and "Namespace" which are required fields, you can also specify:

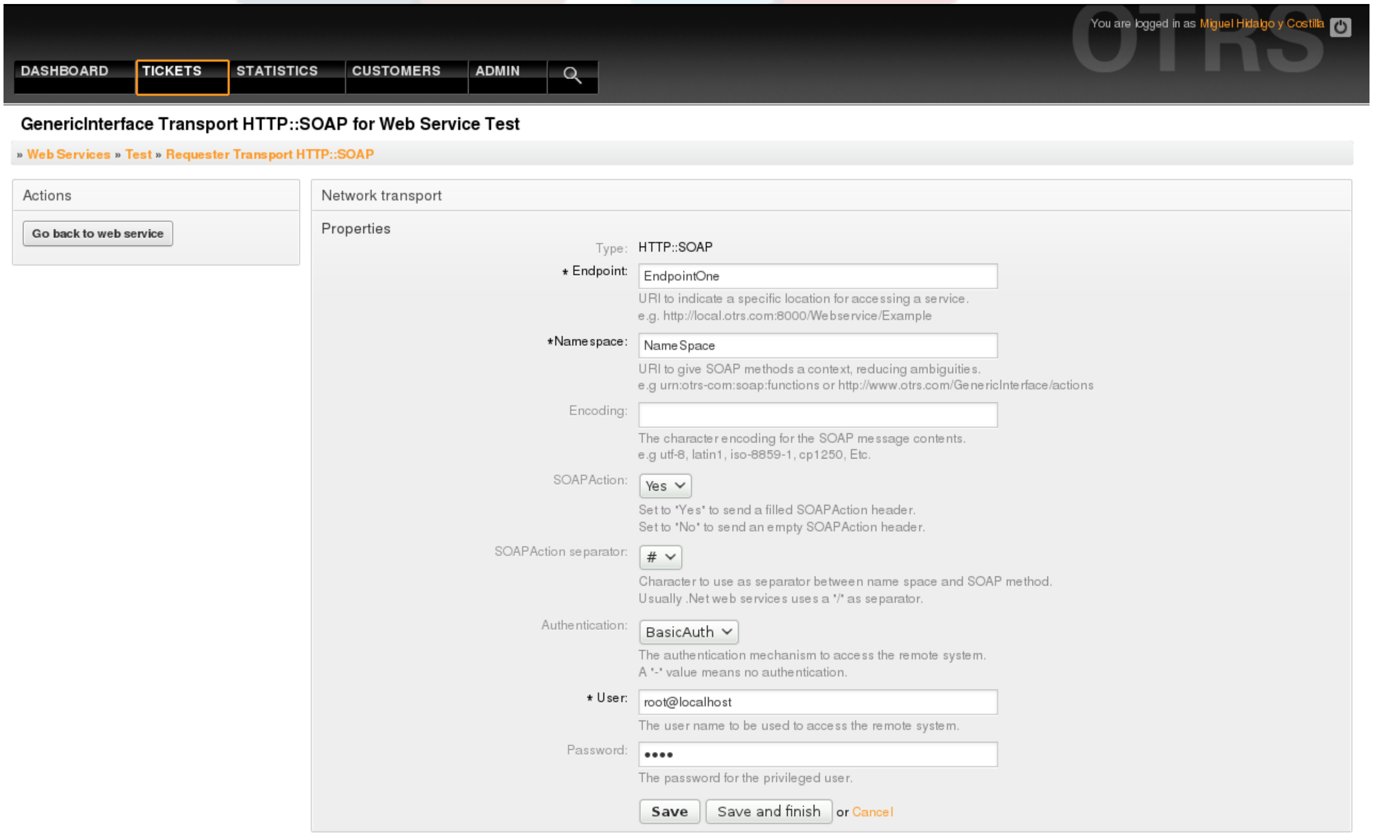
- Encoding (such as utf-8, latin1, iso-8859-1, cp1250, etc) for the SOAP message.
- SOAPAction Header: you can use this to send an empty or filled SOAPAction header. Set to "No" and the SOAPAction header on the SOAP message will be an empty string, or set to "Yes" to send the soap action in Namespace#Action format and define the separator (typically "/" for .Net web services and "#" for the rest).
- Authentication: to set the authentication mechanism, set to "-" to not use any authentication or select one from the list and the detail fields will appear.

Note

Currently only the "BasicAuth" (HTTP) authentication mechanism is implemented. You can decide whether or not to use it depending on the Remote System configuration. If used, you must provide the User Name and the Password to access the remote system.

Warning

If you supply a password for authentication and after you export the web service to a YAML file this password will be revealed and will be written into a plain text string inside the YAML file. Be aware of it and take precautions if needed.



GenericInterface Transport HTTP::SOAP for Web Service Test

» Web Services » Test » Requester Transport HTTP::SOAP

Actions

Go back to web service

Network transport

Properties

Type: HTTP::SOAP

* Endpoint: EndpointOne
URI to indicate a specific location for accessing a service.
e.g. http://local.otrs.com:8000/Webservice/Example

*Namespace: NameSpace
URI to give SOAP methods a context, reducing ambiguities.
e.g urn:otrs-com:soap:functions or http://www.otrs.com/GenericInterface/actions

Encoding:
The character encoding for the SOAP message contents.
e.g utf-8, latin1, iso-8859-1, cp1250, Etc.

SOAPAction: Yes
Set to "Yes" to send a filled SOAPAction header.
Set to "No" to send an empty SOAPAction header.

SOAPAction separator: #
Character to use as separator between name space and SOAP method.
Usually .Net web services uses a "/" as separator.

Authentication: BasicAuth
The authentication mechanism to access the remote system.
A "-" value means no authentication.

* User: root@localhost
The user name to be used to access the remote system.

Password: ••••
The password for the privileged user.

Save | Save and finish | or Cancel

* * *

11.4.3.7.4. Web Service Invoker

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 dialogues. Each invoker can have independent configuration dialogues if needed.

Name, Description, Backend, and Mappings are fields that normally appear on every invoker. Additionally the list of event triggers and other special fields can appear on non default configuration dialogues to fulfill special needs of the invoker.

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 (backends) for each mapping direction, since their configuration is independent from each other and also independent from the invoker backend. 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.

The invoker backend is pre-populated and can not be edited. You will see this parameter when you choose the invoker on the web service edit screen. The field is only informative.

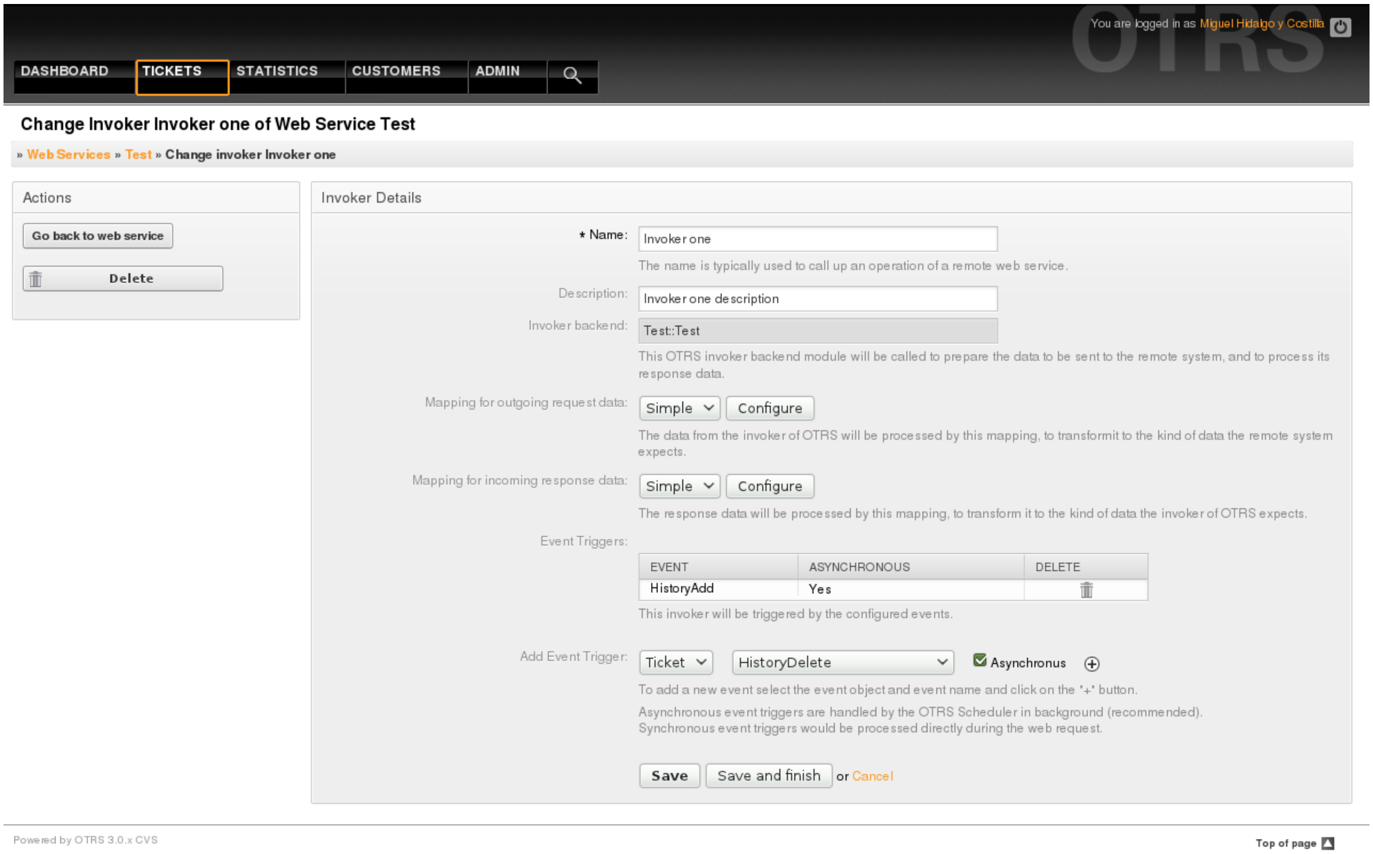
Event triggers are events within OTRS such as "TicketCreate", "ArticleSend", etc. These can act as triggers to execute the invoker. Each invoker needs to have at least one event trigger registered, or the invoker will be useless, because it will never be called. The asynchronous property of the event triggers define if the OTRS process will handle the invoker or if it will be delegated to the Scheduler.

> **Note**
>
> The OTRS Scheduler is a separate 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 don't use the scheduler using web services can make OTRS slow or non-responsive. Therefore it is highly recommend to use asynchronous event triggers as often as possible.

To add an Event trigger, first select the event family from the first list, then the event name from the second list, then set the asynchronous property (if unchecked means that the event trigger will not be asynchronous) and finally click on the plus button. A new event trigger will be created and it will be listed on the invoker "Event Triggers" list.

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 "Delete" to remove the event trigger from the list, or "Cancel" to close the dialog.

In the left part of the screen on the action column you have the options: "Go back to web service" (discarding all changes since the last save) and "Delete". If you click on the last one, a dialog will emerge 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 "Cancel" to close the delete dialog.



Change Invoker Invoker one of Web Service Test

» Web Services » Test » Change invoker Invoker one

* Name: Invoker one

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

Description: Invoker one description

Invoker backend: Test::Test

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: Simple

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: Simple

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

Event Triggers:

EVENT	ASYNCHRONOUS	DELETE
HistoryAdd	Yes	🗑

This invoker will be triggered by the configured events.

Add Event Trigger: Ticket HistoryDelete ☑ Asynchronus ⊕

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 in background (recommended). Synchronous event triggers would be processed directly during the web request.

11.4.3.7.5. Web Service Mapping

There are cases where you need to transform the data from one format to another (map or change data structure), because normally a web service is used to interact with a Remote System, that is highly probable that is not another OTRS system and / or could not understand the OTRS data structures and values. In these cases some or all values have to be changed, and sometimes even the names of the values (keys) or even the complete structure, in order to match with the expected data on the other end. To accomplish this task the Generic Interface Mapping Layer exists.

Each Remote System has it own data structures and it is possible to create new mapping modules for each case (e.g. there is a customized mapping module for SAP Solution Manager shipped with OTRS), but it is not always necessary. The module Mapping::Simple should cover most of the mapping needs.

> **Note**
>
> When Mapping::Simple does not cover all mapping needs for a web service, a new mapping module should be created. To learn more about how to create new mapping modules please consult the OTRS Development Manual.

This module gives you the opportunity to set default values to map for each key or value for the whole communication data.

At the beginning of the screen you will see a general section where you can set the default rules that will apply for all the unmapped keys and values. There are three options available, these options are listed below:

- Keep (leave unchanged): doesn't touch the keys or values in any way.
- Ignore (drop key/value pair): when this is applied to the key it deletes the key and value, because when a key is deleted then in consequence its associated value is deleted too. When this is applied to the value, only the value is deleted, keeping the key, that now will be associated to an empty value.
- MapTo (use provided key or value as default): all keys and / or values without a defined map rule, will use this as default, when you select this option a new text field will appear to set this default.

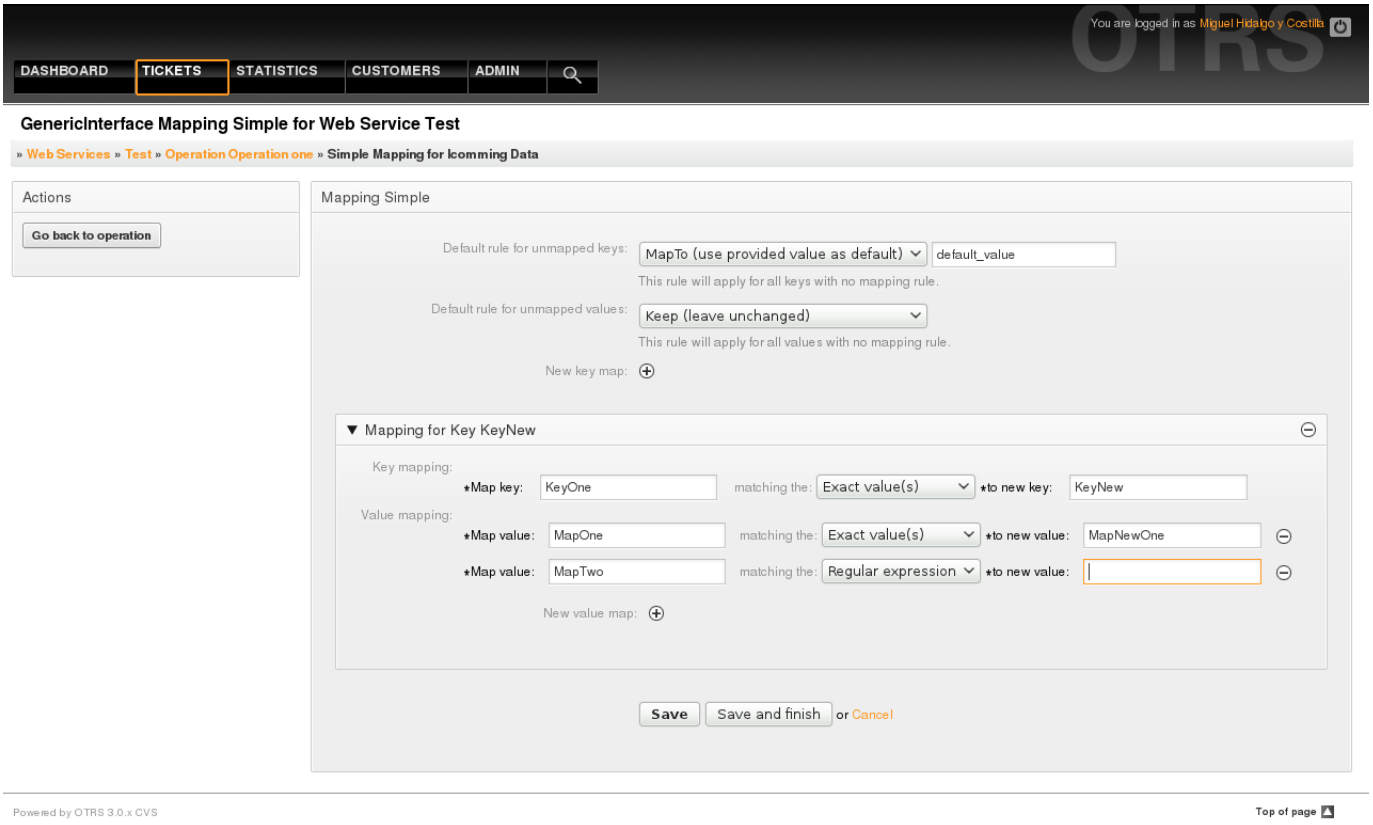
Clicking on the "+" button for new key map, will display a new box for a single mapping configuration. You can add as many key mappings as needed. Just click on the "+" button again and a new mapping box will appear below the existing one. From this mapping boxes you can define a map for a single key, with the next options:

- Exact value(s): the old key string will be changed to a new one if the old key matches exactly.
- Regular expression: The key string will be replaced following a regular expression rule.

Pressing the new value map "+" button will display a new row for a value map. Here it is also possible to define rules for each value to be mapped with the same options as for the key map (Exact value and Regular expression). You can add as many values to map as needed, and if you want to delete one of them, just click on the "-" button for each mapping value row.

Deleting the complete key mapping section (box) is possible, just push on the "-" button located on the up right corner of each box that you want to delete.

If you need to delete a complete mapping configuration: go back to the corresponding operation or invoker screen, look for the mapping direction that you select before and set its value to "-", and save the configuration to apply changes.



GenericInterface Mapping Simple for Web Service Test

» Web Services » Test » Operation Operation one » Simple Mapping for Icomming Data

Mapping Simple

Default rule for unmapped keys: MapTo (use provided value as default) default_value

This rule will apply for all keys with no mapping rule.

Default rule for unmapped values: Keep (leave unchanged)

This rule will apply for all values with no mapping rule.

New key map:

Mapping for Key KeyNew

Key mapping:

*Map key: KeyOne matching the: Exact value(s) *to new key: KeyNew

Value mapping:

*Map value: MapOne matching the: Exact value(s) *to new value: MapNewOne

*Map value: MapTwo matching the: Regular expression *to new value:

New value map:

11.5. Web Service Command Line Interface

The Command Line Interface (CLI) is a fast way to work with the web services. It consists of a set of tools that can be use to perform basic operations like:

- Create, Update, Read, List and Delete web services based on YAML files.
- Read the Debugger log, with filter options.

Note

You don't need to use the CLI to work with web services. Integrated into the Admin interface there is a complete set of screens to interact with every part of the web services. Please read the web service GUI section included in this manual.

11.5.1. Web Service Configuration

The "WebserviceConfig.pl" was developed in order to create basic, but fast and powerful tool to work with web service configurations. It gives you the ability to perform the following actions:

- Add: to create web services using a YAML file as the configuration source.
- Update: to change an existing web service, the configuration can be changed using a different or modified YAML file.
- Read: to get the current web service configuration displayed on the screen.

- List: to get a complete list of all the web services registered in system.
- Delete: to delete a web service from the system. Be careful when you use it, because this action can't be undone.

Warning

A web service READ operation will display all the configuration as plain text on the screen, including any stored passwords. Please be aware of this and take the needed precautions!

Example: Creating a new web service configuration:

```
                shell> OTRS_HOME/bin/otrs.WebserviceConfig.pl -a write -n <webservice_name> -
f /path/to/yaml/file
```

Also you can use 'otrs.WebserviceConfig.pl' with following options:

- -a read -i <webservice_id> - To read a stored configuration.
- -a write -n <webservice_name> -f /path/to/yaml/file - To create a new web service.
- -a write -i <webservice_id> -f /path/to/yaml/file - To update a web service.
- -a list - To list available web services.
- -a delete -i <webservice_id> - To delete a web service.

11.5.2. Web Service Debugger

Another available tool via the command line interface is the "otrs.GenericInterfaceDebugRead.pl" script, which is an interface to search for web service debugger log entries.

Example: Searching for debugger log entries:

```
                shell> bin/otrs.GenericInterfaceDebugRead.pl
```

Optional parameters can be used for the "otrs.GenericInterfaceDebugRead.pl" script:

- -c - to filter by Communication ID (md5sum format).
- -t - to filter by CommunicationType ('Provider' or 'Requester').
- -a - to filter by date (At or After a date).
- -b - to filter by date (At or Before a date).
- -i - to filter by IP Address (must be valid IPv4 or IPv6 address).
- -w - to filter by Web Service ID.
- -d - to include detailed communication data.

Example: Searching for debugger log entries with all parameters:

```
                shell> ./otrs.GenericInterfaceDebugRead.pl -c a7cc4d9f5c70387a9bfbe1351bc88966
-t Provider -a '2011-07-22 00:00:00' -b '2011-07-26 00:00:00' -i 127.0.0.1 -w 123 -d 1
```

Note

It is highly recommended to include at least one of the filter options listed above, and even more if the "-d" option is selected, because ✂ ✂✂✂ ✂✂ information can be retrieved from the data base and displayed on the screen, this could result in slow response times and much more information than what you really needed.

11.6. Web Service Configuration

From its design the web services were conceived to be portable from one OTRS system to another, e.g. from a test or development environment to a production system. Therefore it was needed to have an easy way to extract the web service configuration from the database, and import it to another. To accomplish this task the Generic Interface uses YAML files as the web services configuration basis.

Why YAML? YAML is a markup language designed to be human friendly to read and write (it is easier to understand than JSON), it does not have some of the limitations of XML like numeric tags, it is open, standardized, and is complete enough to store the whole web service configuration.

Note

To learn more about YAML please visit http://www.yaml.org/.

The following is a web service configuration file example in YAML format:

```
---
Debugger:
  DebugThreshold: debug
Description: This an example of a web service configuration
Provider:
  Operation:
    CloseIncident:
      Description: This is a test operation
      MappingInbound: {}
      MappingOutbound: {}
      RemoteSystemGuid: ''
      Type: Test::Test
    Test:
      Description: This is a test operation
      MappingInbound:
        Config:
          KeyMapDefault:
            MapTo: ''
            MapType: Keep
          KeyMapExact:
            Prio: Priority
          ValueMap:
            Priority:
              ValueMapExact:
                Critical: 5 Very High
                Information: 1 Very Low
                Warning: 3 Normal
          ValueMapDefault:
            MapTo: 3 Normal
            MapType: MapTo
        Type: Simple
      MappingOutbound:
        Config:
          KeyMapDefault:
            MapTo: ''
            MapType: Ignore
          KeyMapExact:
            Priority: Prio
          ValueMap:
            Prio:
              ValueMapExact:
                1 Very Low: Information
                3 Normal: Warning
                5 Very High: Critical
          ValueMapDefault:
            MapTo: ''
            MapType: Ignore
        Type: Simple
      Type: Test::Test
  Transport:
    Config:
      MaxLength: 10000000
      NameSpace: http://www.example.com/actions
    Type: HTTP::SOAP
RemoteSystem: remote.system.description.example.com
Requester:
  Invoker:
    Test:
      Description: This is a test invoker
      Events:
        - Asynchronous: 1
          Event: TicketCreate
        - Asynchronous: 0
          Event: ArticleUpdate
      MappingInbound:
        Type: Simple
      MappingOutbound:
        Type: Simple
      Type: Test::Test
  Transport:
    Config:
      Authentication:
        Password: '*******'
        Type: BasicAuth
        User: otrs
      Encoding: utf-8
      Endpoint: http://www.example.com:8080/endpoint
      NameSpace: http://www.example.com/actions
      SOAPAction: Yes
      SOAPActionSeparator: '#'
    Type: HTTP::SOAP
```

11.6.1. Configuration Details

11.6.1.1. General

- Description: a short text that describes the web service.
- RemoteSystem: a short description of the Remote System.
- Debugger: a container for the debugger settings.
- Provider: a container for the provider settings.
- Requester: a container for the requester settings.

11.6.1.2. Debugger

- DebugThreshold: the debugger level

  Possible Values

  - debug: all logs are stored in the database.
  - info: info, notice and error level logs are stored in the database.
  - notice: notice and error level logs are stored in the database.
  - error: only error level logs are stored in the database.

11.6.1.3. Provider

- Operation: a container for each operation settings.
- Transport: a container for provider network transport settings.

11.6.1.3.1. Operation

- <OperationName>: Unique name for the operation, container for its own operation settings (cardinality 0..n, but not duplicate).

11.6.1.3.1.1. <OperationName>

This section is based on operations from type "Test::Test" other operations might contain more or different settings.

- Description: a short text that describes the operation.
- MappingInbound: a container for the mapping settings for the incoming request data.
- MappingOutbound: a container for the mapping settings for the outgoing response data.
- Type: the operation backend, in Controller::Operation format.

11.6.1.3.1.1.1. MappingInbound

This section is based on mappings from type "Simple". Other mappings might contain more or different settings.

- Config: a container for this mapping settings.
- Type: the mapping backend.

11.6.1.3.1.1.1.1. Config

- KeyMapDefault: a container for all non mapped keys settings.
- ValueMapDefault: a container for all non mapped values settings.
- KeyMapExact: a container for all exact key mappings (cardinality 0 .. 1).
- KeyMapRegEx: a container for all regular expression key mappings (cardinality 0 .. 1).
- ValueMap: a container for all value mappings (cardinality 0 .. 1).

11.6.1.3.1.1.1.1.1. KeyMapDefault

- MapTo: the new value to be used (only applicable if MapType is set to MapTo).

- MapType: the rule for the mapping.

  Possible Values

  - Keep: leave unchanged.
  - Ignore: drop.
  - MapTo: change to the MapTo value.

11.6.1.3.1.1.1.1.2. ValueMapDefault

Similar to KeyMapDefault.

11.6.1.3.1.1.1.1.3. KeyMapExact

- <oldkey>: <newkey> (cardinality 0 .. n but not duplicate).

11.6.1.3.1.1.1.1.4. KeyMapRegEx

- <oldkey(RegEx)>: <newkey> (cardinality 0 .. n but no duplicates).

11.6.1.3.1.1.1.1.5. ValueMap

- <newkey>: a container for value mappings for this new key (cardinality depends on the new keys from KeyMapExact and KeyMapRegEx).

11.6.1.3.1.1.1.1.5.1. <newkey>

- ValueMapExact: a container for all exact value mappings (cardinality 0 .. 1).
- ValueMapRegEx: a container for all regular expression value mappings (cardinality 0 .. 1).

11.6.1.3.1.1.1.1.5.1.1. valueMapExact

- <oldvalue>: <newvalue> (cardinality 0 .. n but not duplicate).

11.6.1.3.1.1.1.1.5.1.2. ValueMapRegEx

- <oldvalue(RegEx)>: <newvalue> (cardinality 0 .. n but not duplicate).

11.6.1.3.1.1.2. MappingOutbound

Same as MappingInbound.

11.6.1.3.1.1.3. Transport

This section is based on the provider network transport HTTP::SOAP, other transports might contain more or different settings.

- Config: a container for the specific network transport configuration settings.
- Type: the provider network transport backend.

11.6.1.3.1.1.3.1. Config

- MaxLength: the maximum length in bytes to be read in a SOAP message by OTRS.
- NameSpace: an URI that gives a context to all operations that belongs to this web service.

11.6.1.4. Requester

- Invoker: a container for each invokers' settings.

- Transport: a container for requester network transport settings.

11.6.1.4.1. Invoker

- <InvokerName>: Unique name for the invoker, container for its own invoker settings (cardinality 0..n, but not duplicate).

11.6.1.4.1.1. <InvokerName>

This section is based on invokers from type "Test::Test" other invokers might contain more or different settings.

- Description: a short text that describes the invoker
- Events: a container for a unnamed list of event trigger settings.
- MappingInbound: a container for the mapping settings for the incoming response data.
- MappingOutbound: a container for the mapping settings for the outgoing request data.
- Type: the invoker backend, in Controller::Invoker format.

11.6.1.4.1.1.1. Events

- ✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂✂: (cardinality 0 .. n)
    - Asynchronous: to set if the invoker execution will be delegated to the Scheduler

      Possible Values

        - 0: not handled by the Scheduler.
        - 1: handled by the Scheduler.
    - Event: the name of the event trigger.

      Possible Values (for ticket events)

        - TicketCreate
        - TicketDelete
        - TicketTitleUpdate
        - TicketUnlockTimeoutUpdate
        - TicketQueueUpdate
        - TicketTypeUpdate
        - TicketServiceUpdate
        - TicketSLAUpdate
        - TicketCustomerUpdate
        - TicketFreeTextUpdate
        - TicketFreeTimeUpdate

- TicketPendingTimeUpdate
- TicketLockUpdate
- TicketArchiveFlagUpdate
- TicketStateUpdate
- TicketOwnerUpdate
- TicketResponsibleUpdate
- TicketPriorityUpdate
- HistoryAdd
- HistoryDelete
- TicketAccountTime
- TicketMerge
- TicketSubscribe
- TicketUnsubscribe
- TicketFlagSet
- TicketFlagDelete
- TicketSlaveLinkAdd
- TicketSlaveLinkDelete
- TicketMasterLinkDelete

Possible Values (for article events)

- Article Events
- ArticleCreate
- ArticleFreeTextUpdate
- ArticleUpdate
- ArticleSend
- ArticleBounce
- ArticleAgentNotification
- ArticleCustomerNotification
- ArticleAutoResponse
- ArticleFlagSet

  - ArticleFlagDelete
  - ArticleAgentNotification
  - ArticleCustomerNotification

11.6.1.4.1.1.2. MappingInbound

Same as Operation MappingInbound

11.6.1.4.1.1.3. MappingOutbound

Same as Operation MappingInbound.

11.6.1.4.1.1.4. Transport

This section is based on the requester network transport HTTP::SOAP, other transports might contain more or different settings.

- Config: a container for the specific network transport configuration settings.
- Type: the requester network transport backend.

11.6.1.4.1.1.4.1. Config

- Authentication: a container for authentication settings.
- Encoding: the SOAP Message request encoding
- Endpoint: the URI of the Remote Server web service to accept OTRS requests
- NameSpace: an URI that gives a context to all invokers that belongs to this web service.
- SOAPAction: to send an empty or filled SOAPAction header in the SOAP Message (in "<NameSpace> <Separator> <Action>" format).

  Possible Values

  - YES: to send a filled SOAPAction header.
  - No: to send an empty SOAPAction header.
- SOAPActionSeparator: to set the <Separator> of a filled SOAPAction header.

  Possible Values

  - '/': used for .net web services.
  - '#': used for all the rest web services.

11.6.1.4.1.1.4.1.1. Authentication

- User: the privileged user name that has access to the remote web service.
- Password: the password for privileged user in plain text.
- Type: the type of authentication.

11.7. Connectors

A Connector is in essence a set of actions that are either called Operations if OTRS acts as a web service provider or Invokers if OTRS acts as a web service requester. But it can also include special Mappings or Transports.

One Connector can either have only Operations, Only Invokers or both. A connector can even use parts of other connectors like the Mappings or Transports if they are not to specific for the Connector that is trying to implement them.

In other words a Connector is not limited to just the Controller layer but it can be extended to Data Mapping or Network Transport layers if needed.

Due to the modular design of the Generic Interface a Connector can be seen as a plug-in; this means that by adding Connectors the capabilities of the generic interface can be extended using: OTRS Feature add ons, OTRS Custom modules, 3rd Party modules, and so on.

11.7.1. Bundled Connectors

Included with this version of OTRS the following connectors are ready to be used.

- Session
- Ticket

11.7.1.1. Session Connector

This connector is capable to create a valid SessionID that can be used in any other operation.

Provides:

- Operations:
  - SessionCreate

11.7.1.1.1. Operations

11.7.1.1.1.1. SessionCreate

Creates a new new valid SessionID to be used in other operations from other connectors like TicketCreate.

> **Note**
>
> To use the SessionID in other operations from other connectors it is necessary that the operation implements authentication by SessionID. All the rest of the bundled operations are capable of accepting a valid SessionID as an authentication method.

Possible Attributes:

```
<SessionCreate>
   <!--You have a MANDATORY CHOICE of the next 2 items at this level-->
   <!--Optional:-->
   <UserLogin>?</UserLogin>
   <!--Optional:-->
   <CustomerUserLogin>?</CustomerUserLogin>
   <!--Optional:-->
   <Password>?</Password>
</SessionCreate>
```

11.7.1.2. Ticket Connector

This connector supplies the basic functionality to interact with tickets

Provides:

- Operations:
  - TicketCreate
  - TicketUpdate
  - TicketGet
  - TicketSearch

11.7.1.2.1. Operations

11.7.1.2.1.1. TicketCreate

Provides an interface to create a ticket in OTRS. A ticket must contain an Article and can contain several attachments, all defined Dynamic Fields can be also set on TicketCreate operation.

Possible Attributes:

```
<TicketCreate>
   <!--You have a MANDATORY CHOICE of the next 3 items at this level-->
   <!--Optional:-->
   <UserLogin>?</UserLogin>
   <!--Optional:-->
   <CustomerUserLogin>?</CustomerUserLogin>
   <!--Optional:-->
   <SessionID>?</SessionID>
   <!--Optional:-->
   <Password>?</Password>
   <Ticket>
      <Title>?</Title>
      <!--You have a MANDATORY CHOICE of the next 2 items at this level-->
      <!--Optional:-->
      <QueueID>?</QueueID>
      <!--Optional:-->
      <Queue>?</Queue>
      <!--You have a CHOICE of the next 2 items at this level-->
      <!--Optional:-->
      <TypeID>?</TypeID>
      <!--Optional:-->
      <Type>?</Type>
      <!--You have a CHOICE of the next 2 items at this level-->
      <!--Optional:-->
      <ServiceID>?</ServiceID>
      <!--Optional:-->
      <Service>?</Service>
      <!--You have a CHOICE of the next 2 items at this level-->
      <!--Optional:-->
      <SLAID>?</SLAID>
      <!--Optional:-->
      <SLA>?</SLA>
      <!--You have a MANDATORY CHOICE of the next 2 items at this level-->
      <!--Optional:-->
      <StateID>?</StateID>
```

```
            <!--Optional:-->
            <State>?</State>
            <!--You have a MANDATORY CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <PriorityID>?</PriorityID>
            <!--Optional:-->
            <Priority>?</Priority>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <OwnerID>?</OwnerID>
            <!--Optional:-->
            <Owner>?</Owner>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <ResponsibleID>?</ResponsibleID>
            <!--Optional:-->
            <Responsible>?</Responsible>
            <CustomerUser>?</CustomerUser>
            <!--Optional:-->
            <PendingTime>
               <Year>?</Year>
               <Month>?</Month>
               <Day>?</Day>
               <Hour>?</Hour>
               <Minute>?</Minute>
            </PendingTime>
         </Ticket>
         <Article>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <ArticleTypeID>?</ArticleTypeID>
            <!--Optional:-->
            <ArticleType>?</ArticleType>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <SenderTypeID>?</SenderTypeID>
            <!--Optional:-->
            <SenderType>?</SenderType>
            <!--Optional:-->
            <From>?</From>
            <Subject>?</Subject>
            <Body>?</Body>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <ContentType>?</ContentType>
            <Charset>?</Charset>
            <MimeType>?</MimeType>
            <!--Optional:-->
            <HistoryType>?</HistoryType>
            <!--Optional:-->
            <HistoryComment>?</HistoryComment>
            <!--Optional:-->
            <AutoResponseType>?</AutoResponseType>
            <!--Optional:-->
            <TimeUnit>?</TimeUnit>
            <!--Optional:-->
            <NoAgentNotify>?</NoAgentNotify>
            <!--Zero or more repetitions:-->
            <ForceNotificationToUserID>?</ForceNotificationToUserID>
            <!--Zero or more repetitions:-->
            <ExcludeNotificationToUserID>?</ExcludeNotificationToUserID>
            <!--Zero or more repetitions:-->
            <ExcludeMuteNotificationToUserID>?</ExcludeMuteNotificationToUserID>
         </Article>
         <!--Zero or more repetitions:-->
         <DynamicField>
            <Name>?</Name>
            <!--1 or more repetitions:-->
            <Value>?</Value>
         </DynamicField>
         <!--Zero or more repetitions:-->
         <Attachment>
            <Content>cid:61886944659</Content>
            <ContentType>?</ContentType>
            <Filename>?</Filename>
         </Attachment>
      </TicketCreate>
```

11.7.1.2.1.2. TicketUpdate

TicketUpdate operation adds the capability to modify attributes from an existing ticket or to add a new article, including attachments and all defined dynamic fields for the ticket and the new article.

Note

It is not necessary to create a new article to modify a ticket attribute.

Possible Attributes:

```
      <TicketUpdate>
         <!--You have a MANDATORY CHOICE of the next 3 items at this level-->
         <!--Optional:-->
         <UserLogin>?</UserLogin>
         <!--Optional:-->
         <CustomerUserLogin>?</CustomerUserLogin>
         <!--Optional:-->
         <SessionID>?</SessionID>
         <!--Optional:-->
         <Password>?</Password>
         <!--You have a CHOICE of the next 2 items at this level-->
         <TicketID>?</TicketID>
         <TicketNumber>?</TicketNumber>
         <!--Optional:-->
         <Ticket>
            <!--Optional:-->
            <Title>?</Title>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <QueueID>?</QueueID>
            <!--Optional:-->
            <Queue>?</Queue>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <TypeID>?</TypeID>
            <!--Optional:-->
            <Type>?</Type>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <ServiceID>?</ServiceID>
            <!--Optional:-->
            <Service>?</Service>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <SLAID>?</SLAID>
            <!--Optional:-->
            <SLA>?</SLA>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <StateID>?</StateID>
            <!--Optional:-->
            <State>?</State>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <PriorityID>?</PriorityID>
            <!--Optional:-->
            <Priority>?</Priority>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <OwnerID>?</OwnerID>
            <!--Optional:-->
            <Owner>?</Owner>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <ResponsibleID>?</ResponsibleID>
            <!--Optional:-->
            <Responsible>?</Responsible>
            <!--Optional:-->
            <CustomerUser>?</CustomerUser>
            <!--Optional:-->
            <PendingTime>
               <Year>?</Year>
               <Month>?</Month>
               <Day>?</Day>
               <Hour>?</Hour>
               <Minute>?</Minute>
            </PendingTime>
         </Ticket>
         <!--Optional:-->
         <Article>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <ArticleTypeID>?</ArticleTypeID>
            <!--Optional:-->
            <ArticleType>?</ArticleType>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <SenderTypeID>?</SenderTypeID>
            <!--Optional:-->
            <SenderType>?</SenderType>
            <!--Optional:-->
            <From>?</From>
            <Subject>?</Subject>
            <Body>?</Body>
            <!--You have a CHOICE of the next 2 items at this level-->
            <!--Optional:-->
            <ContentType>?</ContentType>
            <Charset>?</Charset>
            <MimeType>?</MimeType>
            <!--Optional:-->
            <HistoryType>?</HistoryType>
            <!--Optional:-->
            <HistoryComment>?</HistoryComment>
            <!--Optional:-->
            <AutoResponseType>?</AutoResponseType>
            <!--Optional:-->
            <TimeUnit>?</TimeUnit>
            <!--Optional:-->
            <NoAgentNotify>?</NoAgentNotify>
            <!--Zero or more repetitions:-->
            <ForceNotificationToUserID>?</ForceNotificationToUserID>
            <!--Zero or more repetitions:-->
            <ExcludeNotificationToUserID>?</ExcludeNotificationToUserID>
            <!--Zero or more repetitions:-->
            <ExcludeMuteNotificationToUserID>?</ExcludeMuteNotificationToUserID>
         </Article>
```

```
        <!--Zero or more repetitions:-->
        <DynamicField>
           <Name>?</Name>
           <!--1 or more repetitions:-->
           <Value>?</Value>
        </DynamicField>
        <!--Zero or more repetitions:-->
        <Attachment>
           <Content>cid:166861569966</Content>
           <ContentType>?</ContentType>
           <Filename>?</Filename>
        </Attachment>
     </TicketUpdate>
```

11.7.1.2.1.3. TicketGet

This operation is used to get all the attributes of a ticket including the dynamic fields, all articles and all of the attachments that belong to the ticket.

Possible Attributes:

```
     <TicketGet>
        <!--You have a MANDATORY CHOICE of the next 3 items at this level-->
        <!--Optional:-->
        <UserLogin>?</UserLogin>
        <!--Optional:-->
        <CustomerUserLogin>?</CustomerUserLogin>
        <!--Optional:-->
        <SessionID>?</SessionID>
        <!--Optional:-->
        <Password>?</Password>
        <!--1 or more repetitions:-->
        <TicketID>?</TicketID>
        <!--Optional:-->
        <DynamicFields>?</DynamicFields>
        <!--Optional:-->
        <Extended>?</Extended>
        <!--Optional:-->
        <AllArticles>?</AllArticles>
        <!--Optional:-->
        <ArticleSenderType>?</ArticleSenderType>
        <!--Optional:-->
        <ArticleOrder>?</ArticleOrder>
        <!--Optional:-->
        <ArticleLimit>?</ArticleLimit>
        <!--Optional:-->
        <Attachments>?</Attachments>
     </TicketGet>
```

11.7.1.2.1.4. TicketSearch

TicketSearch operation returns a list of Ticket IDs that matches a predefined criteria.

Possible Attributes:

```
     <TicketSearch>
        <!--You have a MANDATORY CHOICE of the next 3 items at this level-->
        <!--Optional:-->
        <UserLogin>?</UserLogin>
```

```
            <!--Optional:-->
            <CustomerUserLogin>?</CustomerUserLogin>
            <!--Optional:-->
            <SessionID>?</SessionID>
            <!--Optional:-->
            <Password>?</Password>
            <!--Optional:-->
            <Limit>?</Limit>
            <!--Zero or more repetitions:-->
            <TicketNumber>?</TicketNumber>
            <!--Zero or more repetitions:-->
            <Title>?</Title>
            <!--Zero or more repetitions:-->
            <Queues>?</Queues>
            <!--Zero or more repetitions:-->
            <QueueIDs>?</QueueIDs>
            <!--Optional:-->
            <UseSubQueues>?</UseSubQueues>
            <!--Zero or more repetitions:-->
            <Types>?</Types>
            <!--Zero or more repetitions:-->
            <TypeIDs>?</TypeIDs>
            <!--Zero or more repetitions:-->
            <States>?</States>
            <!--Zero or more repetitions:-->
            <StateIDs>?</StateIDs>
            <!--Zero or more repetitions:-->
            <StateType>?</StateType>
            <!--Zero or more repetitions:-->
            <StateTypeIDs>?</StateTypeIDs>
            <!--Zero or more repetitions:-->
            <Priorities>?</Priorities>
            <!--Zero or more repetitions:-->
            <PriorityIDs>?</PriorityIDs>
            <!--Zero or more repetitions:-->
            <Services>?</Services>
            <!--Zero or more repetitions:-->
            <ServiceIDs>?</ServiceIDs>
            <!--Zero or more repetitions:-->
            <SLAs>?</SLAs>
            <!--Zero or more repetitions:-->
            <SLAIDs>?</SLAIDs>
            <!--Zero or more repetitions:-->
            <Locks>?</Locks>
            <!--Zero or more repetitions:-->
            <LockIDs>?</LockIDs>
            <!--Zero or more repetitions:-->
            <OwnerIDs>?</OwnerIDs>
            <!--Zero or more repetitions:-->
            <ResponsibleIDs>?</ResponsibleIDs>
            <!--Zero or more repetitions:-->
            <WatchUserIDs>?</WatchUserIDs>
            <!--Zero or more repetitions:-->
            <CustomerID>?</CustomerID>
            <!--Zero or more repetitions:-->
            <CustomerUserLogin>?</CustomerUserLogin>
            <!--Zero or more repetitions:-->
            <CreatedUserIDs>?</CreatedUserIDs>
            <!--Zero or more repetitions:-->
            <CreatedTypes>?</CreatedTypes>
            <!--Zero or more repetitions:-->
            <CreatedTypeIDs>?</CreatedTypeIDs>
            <!--Zero or more repetitions:-->
            <CreatedPriorities>?</CreatedPriorities>
            <!--Zero or more repetitions:-->
            <CreatedPriorityIDs>?</CreatedPriorityIDs>
            <!--Zero or more repetitions:-->
```

```
        <CreatedStates>?</CreatedStates>
        <!--Zero or more repetitions:-->
        <CreatedStateIDs>?</CreatedStateIDs>
        <!--Zero or more repetitions:-->
        <CreatedQueues>?</CreatedQueues>
        <!--Zero or more repetitions:-->
        <CreatedQueueIDs>?</CreatedQueueIDs>
        <!--Zero or more repetitions:-->
        <DynamicFields>
           <!--You have a MANDATORY CHOICE of the next 6 items at this level-->
           <!--Optional:-->
           <Equals>?</Equals>
           <!--Optional:-->
           <Like>?</Like>
           <!--Optional:-->
           <GreaterThan>?</GreaterThan>
           <!--Optional:-->
           <GreaterThanEquals>?</GreaterThanEquals>
           <!--Optional:-->
           <SmallerThan>?</SmallerThan>
           <!--Optional:-->
           <SmallerThanEquals>?</SmallerThanEquals>
        </DynamicFields>
        <!--Optional:-->
        <Ticketflag>
           <!--Optional:-->
           <Seen>?</Seen>
        </Ticketflag>
        <!--Optional:-->
        <From>?</From>
        <!--Optional:-->
        <To>?</To>
        <!--Optional:-->
        <Cc>?</Cc>
        <!--Optional:-->
        <Subject>?</Subject>
        <!--Optional:-->
        <Body>?</Body>
        <!--Optional:-->
        <FullTextIndex>?</FullTextIndex>
        <!--Optional:-->
        <ContentSearch>?</ContentSearch>
        <!--Optional:-->
        <ConditionInline>?</ConditionInline>
        <!--Optional:-->
        <ArticleCreateTimeOlderMinutes>?</ArticleCreateTimeOlderMinutes>
        <!--Optional:-->
        <ArticleCreateTimeNewerMinutes>?</ArticleCreateTimeNewerMinutes>
        <!--Optional:-->
        <ArticleCreateTimeNewerDate>?</ArticleCreateTimeNewerDate>
        <!--Optional:-->
        <ArticleCreateTimeOlderDate>?</ArticleCreateTimeOlderDate>
        <!--Optional:-->
        <TicketCreateTimeOlderMinutes>?</TicketCreateTimeOlderMinutes>
        <!--Optional:-->
        <ATicketCreateTimeNewerMinutes>?</ATicketCreateTimeNewerMinutes>
        <!--Optional:-->
        <TicketCreateTimeNewerDate>?</TicketCreateTimeNewerDate>
        <!--Optional:-->
        <TicketCreateTimeOlderDate>?</TicketCreateTimeOlderDate>
        <!--Optional:-->
        <TicketChangeTimeOlderMinutes>?</TicketChangeTimeOlderMinutes>
        <!--Optional:-->
        <TicketChangeTimeNewerMinutes>?</TicketChangeTimeNewerMinutes>
        <!--Optional:-->
        <TicketChangeTimeNewerDate>?</TicketChangeTimeNewerDate>
        <!--Optional:-->
```

```
         <TicketChangeTimeOlderDate>?</TicketChangeTimeOlderDate>
         <!--Optional:-->
         <TicketCloseTimeOlderMinutes>?</TicketCloseTimeOlderMinutes>
         <!--Optional:-->
         <TicketCloseTimeNewerMinutes>?</TicketCloseTimeNewerMinutes>
         <!--Optional:-->
         <TicketCloseTimeNewerDate>?</TicketCloseTimeNewerDate>
         <!--Optional:-->
         <TicketCloseTimeOlderDate>?</TicketCloseTimeOlderDate>
         <!--Optional:-->
         <TicketPendingTimeOlderMinutes>?</TicketPendingTimeOlderMinutes>
         <!--Optional:-->
         <TicketPendingTimeNewerMinutes>?</TicketPendingTimeNewerMinutes>
         <!--Optional:-->
         <TicketPendingTimeNewerDate>?</TicketPendingTimeNewerDate>
         <!--Optional:-->
         <TicketPendingTimeOlderDate>?</TicketPendingTimeOlderDate>
         <!--Optional:-->
         <TicketEscalationTimeOlderMinutes>?</TicketEscalationTimeOlderMinutes>
         <!--Optional:-->
         <TTicketEscalationTimeNewerMinutes>?</TTicketEscalationTimeNewerMinutes>
         <!--Optional:-->
         <TicketEscalationTimeNewerDate>?</TicketEscalationTimeNewerDate>
         <!--Optional:-->
         <TicketEscalationTimeOlderDate>?</TicketEscalationTimeOlderDate>
         <!--Optional:-->
         <ArchiveFlags>?</ArchiveFlags>
         <!--Zero or more repetitions:-->
         <OrderBy>?</OrderBy>
         <!--Zero or more repetitions:-->
         <SortBy>?</SortBy>
         <!--Zero or more repetitions:-->
         <CustomerUserID>?</CustomerUserID>
      </TicketSearch>
```

11.7.2. Examples:

11.7.2.1. Web Service Configuration

The following is a basic but complete web service configuration file in YAML format to use all the Ticket Connector operations. In order to use it in OTRS you need to copy the content, save it into a file called GenericTicketConnector.yml, and import it into OTRS in the Web Services screen in the Admin panel by clicking in the "Add web service" action from the overview screen and then clicking in the "Import web service" action in the add screen.

```
---
Debugger:
  DebugThreshold: debug
  TestMode: 0
Description: Ticket Connector Sample
FrameworkVersion: 3.2.x CVS
Provider:
  Operation:
    SessionCreate:
      Description: Creates a Session
      MappingInbound: {}
      MappingOutbound: {}
      Type: Session::SessionCreate
    TicketCreate:
      Description: Creates a Ticket
      MappingInbound: {}
      MappingOutbound: {}
      Type: Ticket::TicketCreate
    TicketUpdate:
      Description: Updates a Ticket
      MappingInbound: {}
      MappingOutbound: {}
      Type: Ticket::TicketUpdate
    TicketGet:
      Description: Retrieve Ticket data
      MappingInbound: {}
      MappingOutbound: {}
      Type: Ticket::TicketGet
    TicketSearch:
      Description: Search for Tickets
      MappingInbound: {}
      MappingOutbound: {}
      Type: Ticket::TicketSearch
  Transport:
    Config:
      MaxLength: 100000000
      NameSpace: http://www.otrs.org/TicketConnector/
    Type: HTTP::SOAP
RemoteSystem: ''
Requester:
  Transport:
    Type: ''
```

11.7.2.2. Perl SOAP Requester

The following code is a Perl script that can connect to OTRS via the generic interface. In order to perform the operations provided by the Ticket Connector, it uses two Perl CPAN modules SOAP::Lite and Data::Dumper. Please make sure that your environment is capable to use these modules before you try to run the scipt.

```


#!/usr/bin/perl -w
# --
# otrs.SOAPRequest.pl - sample to send a SOAP request to OTRS Generic Interface Ticket
 Connector
# Copyright (C) 2001-2013 OTRS AG, http://otrs.com/
# --
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU AFFERO General Public License as published by
# the Free Software Foundation; either version 3 of the License, or
# any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU Affero General Public License
# along with this program; if not, write to the Free Software
# Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
# or see http://www.gnu.org/licenses/agpl.txt.
# --

use strict;
use warnings;

# use ../ as lib location
use File::Basename;
use FindBin qw($RealBin);
use lib dirname($RealBin);
use SOAP::Lite;
use Data::Dumper;

# ---
# Variables to be defined.

# this is the URL for the web service
# the format is
# <HTTP_TYPE>:://<OTRS_FQDN>/nph-genericinterface.pl/Webservice/<WEB_SERVICE_NAME>
# or
# <HTTP_TYPE>:://<OTRS_FQDN>/nph-genericinterface.pl/WebserviceID/<WEB_SERVICE_ID>
my $URL = 'http://localhost/otrs/nph-genericinterface.pl/Webservice/GenericTicketConnector';

# this name space should match the specified name space in the SOAP transport for the web
 service.
my $NameSpace = 'http://www.otrs.org/TicketConnector/';

# this is operation to execute, it could be TicketCreate, TicketUpdate, TicketGet, TicketSearch
# or SessionCreate. and they must to be defined in the web service.
my $Operation = 'TicketCreate';

# this variable is used to store all the parameters to be included on a request in XML format.
 Each
# operation has a determined set of mandatory and non mandatory parameters to work correctly.
 Please
# check the OTRS Admin Manual in order to get a complete list of parameters.
my $XMLData = '
<UserLogin>some user login</UserLogin>
<Password>some password</Password>
<Ticket>
    <Title>some title</Title>
    <CustomerUser>some customer user login</CustomerUser>
    <Queue>some queue</Queue>
    <State>some state</State>
    <Priority>some priority</Priority>
</Ticket>
<Article>
    <Subject>some subject</Subject>
    <Body>some body</Body>
    <ContentType>text/plain; charset=utf8</ContentType>
</Article>
';

# ---

# create a SOAP::Lite data structure from the provided XML data structure.
my $SOAPData = SOAP::Data
    ->type( 'xml' => $XMLData );

my $SOAPObject = SOAP::Lite
    ->uri($NameSpace)
    ->proxy($URL)
    ->$Operation($SOAPData);

# check for a fault in the soap code.
if ( $SOAPObject->fault ) {
    print $SOAPObject->faultcode, " ", $SOAPObject->faultstring, "\n";
}

# otherwise print the results.
else {

    # get the XML response part from the SOAP message.
    my $XMLResponse = $SOAPObject->context()->transport()->proxy()->http_response()->content();

    # deserialize response (convert it into a perl structure).
    my $Deserialized = eval {
        SOAP::Deserializer->deserialize($XMLResponse);
    };

    # remove all the headers and other not needed parts of the SOAP message.
    my $Body = $Deserialized->body();

    # just output relevant data and no the operation name key (like TicketCreateResponse).
    for my $ResponseKey ( keys %{$Body} ) {
        print Dumper( $Body->{$ResponseKey} );
    }
}
```

12. OTRS Scheduler

The OTRS Scheduler is an independent system process that executes tasks in background. These kind of processes are know as ✁✁✁✁✁✁✁ in Unix / Linux systems or as ✁✁✁✁✁✁✁✁ on Windows environments. It is independent but that doesn't mean that the Scheduler does everything alone, it is fully integrated into OTRS and can use any OTRS module as needed to complete each task.

Currently the OTRS Scheduler is only able to handle Generic Interface tasks. These kind of tasks execute invokers that send requests to remote systems. Other handlers for different tasks will be added in future OTRS versions.

For sanity reasons the Scheduler process needs to be restarted from time to time. This is done automatically by the scheduler process itself once a day, but it can be adjusted as needed using the SysConfig by editing the "Scheduler::RestartAfterSeconds" setting.

The OTRS Scheduler is a fully automated process, the only needed human interaction is to check its status periodically and start or stop it as needed.

> Note
>
> If the Scheduler is stopped for any reason, all pending tasks and new tasks registered when the Scheduler is stopped will be executed as soon as the Scheduler starts again (unless the tasks are set to be executed in the future).

12.1. Scheduler Graphical Interface

The Scheduler is not visible in the OTRS Graphical User Interface unless it stops running.

12.1.1. Scheduler Not Running Notification

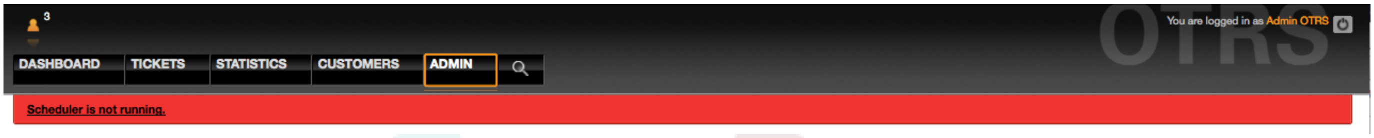
There are two different types of notifications if the system detects that the scheduler is not running. This detection is based on the update frequency of the Scheduler process. If the difference between current time and the last process update time is 2 times the process update frequency a warning message will be displayed in the OTRS notification area. If it is over 4 times the process frequency then an alert will be displayed instead.

The Scheduler process update time can be configured via the SysConfig in the "Scheduler::PIDUpdateTime" setting.

If you see a warning message it is not always necessary to take an action, but it is highly recommended to check if the scheduler process is running. If you see an alert, then there is a high chance that the scheduler is in fact not running and should be started.

By default the Scheduler not running notification is enabled, if there is a valid web service registered in the database, and is only displayed to the users in the "admin" group.

To disable the notification (not recommended) or to change or add the notification groups, please edit the "Frontend::NotifyModule###800-Scheduler-Check" setting in the SysConfig.


3
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Admin OTRS

Scheduler is not running.

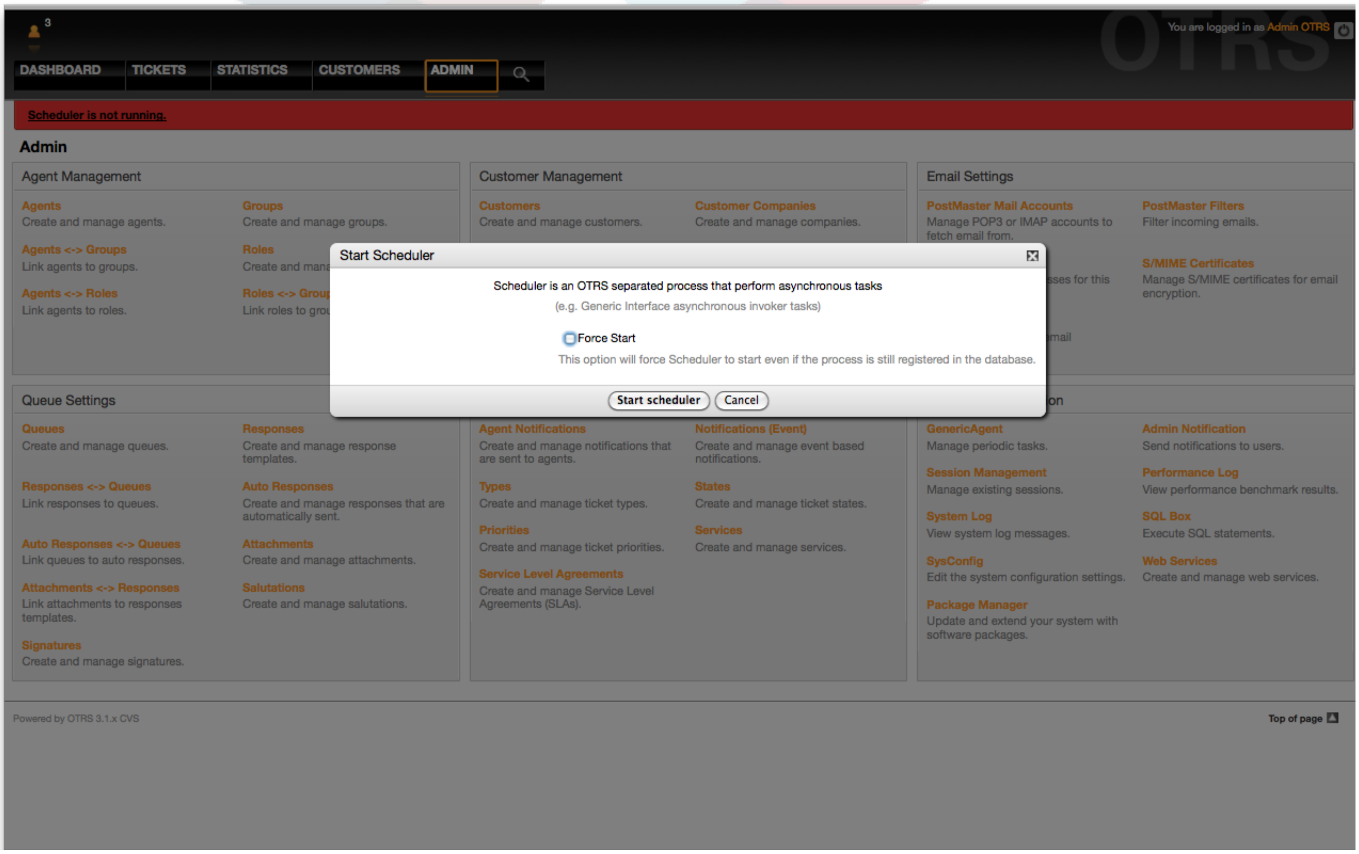
12.1.2. Start Scheduler

By clicking on the Scheduler not running notification link (either warning or alert) a dialog box will open to let you start the Scheduler process again. The Scheduler can be started normally or forced to start, by clicking on the appropriate check box in the dialog.

Note

A forced Scheduler start is only necessary if previous Scheduler process was terminated abnormally and the Process ID is still registered in the database.

To have full control of the Scheduler process and to check it real status please use the command line tools described below.


3
DASHBOARD
TICKETS
STATISTICS
CUSTOMERS
ADMIN
You are logged in as Admin OTRS
Admin
Agent Management
Customer Management
Email Settings
Queue Settings
Agents
Groups
Agents <-> Groups
Roles
Agents <-> Roles
Roles <-> Groups
Customers
Customer Companies
PostMaster Mail Accounts
PostMaster Filters
S/MIME Certificates
Queues
Responses
Responses <-> Queues
Auto Responses
Auto Responses <-> Queues
Attachments
Attachments <-> Responses
Salutations
Signatures
Agent Notifications
Notifications (Event)
Types
States
Priorities
Services
Service Level Agreements
GenericAgent
Admin Notification
Session Management
Performance Log
System Log
SQL Box
SysConfig
Web Services
Package Manager
Powered by OTRS 3.1.x CVS
Top of page
Start Scheduler
Start scheduler
Cancel

Scheduler is not running.

Scheduler is an OTRS separated process that perform asynchronous tasks
(e.g. Generic Interface asynchronous invoker tasks)

☐ Force Start

This option will force Scheduler to start even if the process is still registered in the database.

* * *

12.2. Scheduler Command Line Interface

The Scheduler command line tools let you control the Scheduler process (Start / Stop) or query it status. There are also tools to register the process to be controlled by the operating system.

Included with OTRS there are two set of CLI tools, one for Unix / Linux OS and another for MS Windows OS.

12.2.1. Unix / Linux

12.2.1.1. Scheduler Init.d Files

Init.d files are special scripts that are called by the operating system at startup and shutdown (or restart) times.

OTRS provide init.d scripts to start / stop the OTRS Scheduler process automatically by the operating system, these scripts are located under OTRS_HOME/scripts.

Init.d scripts need to be copied to the correct location for your operating system. They need to have the proper permissions and some internal variables need to be set to work properly.

Init.d Script Internal Variables

- OTRS_HOME - the path of your OTRS installation.
- User - the apache process user name.
- Group - the apache process user's group name.

Note

Currently there are only init.d scripts for Linux platforms.

Table 4.6. List of Init Scripts And Supported Operating Systems

Init Script	Supported OS
otrs-scheduler-linux	Red Hat, Fedora, CentOS, SUSE, openSUSE, Debian, Ubuntu
otrs-scheduler-gentoo-init.d, otrs-scheduler-gentoo-conf.d	Gentoo

Example 4.28. Example To Start The OTRS Scheduler Form An Init.d Script

```
shell> /etc/init.d/otrs-scheduler-linux start
```

Available Actions

- start to start the OTRS Scheduler process.
- stop to stop the OTRS Scheduler process.
- restart to restart the OTRS Scheduler process.

- status to query the OTRS Scheduler process status.

The Scheduler needs the database to be available to register its Process ID, for this reason is necessary to:

- Execute the Scheduler init.d script to start the Scheduler process after the database process is up and running.
- Execute the Scheduler init.d script to stop the Scheduler before the database process shuts down.

> ### Note
>
> If you want the Scheduler to run at system startup, please read the documentation of the operating system to find out the right location to place the init.d scripts, how to configure them to run automatically and how to set the run order.

12.2.1.2. Scheduler Daemon File

This is the part of the Scheduler that stays running in the background checking for tasks to execute. It also provides the main functions to control the process.

All Unix / Linux uses the file OTRS_HOME/bin/otrs.Scheduler.pl.

Example 4.29. Example To Start The OTRS Scheduler

```
shell> OTRS_HOME/bin/otrs.Scheduler.pl -a start
```

Available Options

- -a action.

  Possible Values

  - start- to start the Scheduler process.
  - stop- to stop the Scheduler process.
  - status- to query Scheduler process status.
- -f to force the start or stop of the Scheduler process.

Example 4.30. Example To Force Stop The OTRS Scheduler

```
shell> OTRS_HOME/bin/otrs.Scheduler.pl -a stop -f 1
```

> ### Note
>
> Force stop the Scheduler is used remove the process ID from the database when the scheduler is not running and the process is still registered.
>
> Force start the Scheduler is used to start the Scheduler process if the scheduler is not running and the process is registered.

Force start or stop are only necessary if the start of the process is needed to be done before the process update time expires. Otherwise an expired entry in the database is discarded by normal start.

12.2.2. Windows

12.2.2.1. Scheduler Service Installer

The integration of the services into the MS Windows Operating System is done via the Windows Service Control Manager (SCM). In order to make the OTRS Scheduler process to be controlled by the SCM is necessary to register this service

OTRS provides the script OTRS_HOME/bin/otrs.Scheduler4WinInstaller.pl to register or unregister the OTRS Scheduler into the SCM.

Example 4.31. Example To Register The OTRS Scheduler Into the Widows SCM

```
shell> OTRS_HOME/bin/otrs.Scheduler4WinInstaller.pl -a install
```

Available Options

- -a action.

Possible Values

- install- to install the Scheduler process into the Windows SCM.
- remove- to remove the Scheduler process from the Windows SCM.

After installing into the Widows SCM the OTRS Scheduler process can be used like any other service in Windows. It can be started, stopped and restarted and can be configured to be started manually or automatic.

Note

To learn more about Windows Services and the Windows SCM please read the Windows documentation, and Microsoft online help.

12.2.2.2. Scheduler Service File

This is the part of the Scheduler that stays running in the background checking for tasks to execute. It also provides the main functions to control the process.

Windows Operating System uses the file OTRS_HOME/bin/otrs.Scheduler4Win.pl.

Example 4.32. Example To Start The OTRS Scheduler

```
shell> OTRS_HOME/bin/otrs.Scheduler4Win.pl -a start
```

Available Options

- -a action.

Possible Values

  - start- to start the Scheduler process.
  - stop- to stop the Scheduler process.
  - status- to query Scheduler process status.
- -f to force the start or stop of the Scheduler process.

Example 4.33. Example To Force Stop The OTRS Scheduler

```
                    shell> OTRS_HOME/bin/otrs.Scheduler4Win.pl -a stop -f 1
```

Note

Force stopping the Scheduler is used to remove the process ID from the database when the scheduler is not running and the process is still registered.

Force starting the Scheduler is used to start the Scheduler process if the scheduler is not running and the process is still registered.

Force start or stop are only necessary if starting the process is needed to be done before the process update time expires. Otherwise an expired entry in the database would be discarded by a normal start.

Chapter 5. Customization

1. Access Control Lists (ACLs)

1.1. Introduction

From OTRS 2.0 on, Access Control Lists (ACLs) can be used to control access to tickets, modules, queues, etc., or to influence actions on tickets (closing, moving, etc.) in certain situations. ACLs can be used to supplement the existing permission system of roles and groups. Using ACLs, rudimental workflows within the system can be mapped, based on ticket attributes.

As yet, ACLs cannot be created using the SysConfig interface. They must be directly entered into the `Kernel/Config.pm` file. This chapter has some ACL examples which will walk you trough the process of defining ACL definitions, and a reference of all possible important ACL settings.

1.2. Examples

Example 5.1. ACL allowing movement into a queue of only those tickets with ticket priority 5.

This example shows you the basic structure of an ACL. First, it needs to have a name. In this case, it is "ACL-Name-2". Note that the ACLs will be numerically sorted before execution, so you should use the names carefully.

Secondly, you have a "Properties" section which is a filter for your tickets. All the criteria defined here will be applied to a ticket to determine if the ACL must be applied or not. In our example, a ticket will match if it is in the queue "Raw" and has priority "5 very high". This is also affected by changes in the form (e.g. if the ticket is the queue "raw" and had a priority "3 normal", but then priority drop-down is selected and the priority is changed now to "5 very high" will also match).

Lastly, a section "Possible" defines modifications to the screens. In this case, from the available queues, only the queue "Alert" can be selected in a ticket screen.

```
# ticket acl
$Self->{TicketAcl}->{'100-Example-ACL'} = {
    # match properties
    Properties => {
        # current ticket match properties
        Ticket => {
            Queue => ['Raw'],
            Priority => ['5 very high'],
        }
    },
    # return possible options (white list)
    Possible => {
        # possible ticket options (white list)
        Ticket => {
            Queue => ['Alert'],
        },
    },
};
```

Example 5.2. ACL allowing movement into a queue of only those tickets with ticket priority 5 stored in the database.

This example is very similar to the last one, but in this case only tickets in the queue "Raw" and with a priority "5 very high", both stored in the database will match. This kind of ACLs does not consider changes in the form before the ticket is really updated in the database.

```
# ticket acl
$Self->{TicketAcl}->{'100-Example-ACL'} = {
    # match properties
    PropertiesDatabase => {
        # current ticket match properties
        Ticket => {
            Queue => ['Raw'],
            Priority => ['5 very high'],
        }
    },
    # return possible options (white list)
    Possible => {
        # possible ticket options (white list)
        Ticket => {
            Queue => ['Alert'],
        },
    },
};
```

Example 5.3. ACL disabling the closing of tickets in the raw queue, and hiding the close button.

Here you can see how a ticket field (state) can be filtered with more than one possible value to select from. It is also possible to limit the actions that can be executed for a certain ticket. In this case, the ticket cannot be closed.

```
$Self->{TicketAcl}->{'101-Second-Example-ACL'} = {
    # match properties
    Properties => {
        # current ticket match properties
        Ticket => {
            Queue => ['Raw'],
        }
    },
    # return possible options (white list)
    Possible => {
        # possible ticket options (white list)
        Ticket => {
            State => ['new', 'open', 'pending reminder'],
        },
        # possible action options
        Action => {
            AgentTicketBounce        => 1,
            AgentTicketClose         => 0,
            AgentTicketCompose       => 1,
            AgentTicketCustomer      => 1,
            AgentTicketForward       => 1,
            AgentTicketFreeText      => 1,
            AgentTicketHistory       => 1,
            AgentTicketLink          => 1,
            AgentTicketLock          => 1,
            AgentTicketMerge         => 1,
            AgentTicketMove          => 1,
            AgentTicketNote          => 1,
            AgentTicketOwner         => 1,
            AgentTicketPending       => 1,
            AgentTicketPhone         => 1, # only used to hide the Split action
            AgentTicketPhoneInbound  => 1,
            AgentTicketPhoneOutbound => 1,
            AgentTicketPrint         => 1,
            AgentTicketPriority      => 1,
            AgentTicketResponsible   => 1,
            AgentTicketWatcher       => 1,
            AgentTicketZoom          => 1,
            AgentLinkObject          => 1, # only used to hide the Link action
        },
    },
};
```

Example 5.4. ACL removing always state closed successful.

This example shows how it is possible to define negative filters (the state "closed successful" will be removed). You can also see that not defining match properties for a ticket will match any ticket, i. e. the ACL will always be applied. This may be useful if you want to hide certain values by default, and only enable them in special circumstances (e. g. if the agent is in a specific group).

```
$Self->{TicketAcl}->{'102-Third-ACL-Example'} = {
    # match properties
    Properties => {
        # current ticket match properties (match always)
    },
    # return possible options
    PossibleNot => {
        # possible ticket options
        Ticket => {
            State => ['closed successful'],
        },
    },
};
```

Example 5.5. ACL only showing Hardware services for tickets that are created in queues that start with "HW".

This example also shows you how you can use regular expressions for matching tickets and for filtering the available options.

```
$Self->{TicketAcl}->{'Only-Hardware-Services-for-HW-Queues'} = {
    # match properties
    # note we don't have "Ticket => {" because there's no ticket yet
    Properties => {
    Queue => {
        Name => ['[RegExp]HW'],
        }
    },
    # return possible options
    Possible => {
        # possible ticket options
        Ticket => {
            Service => ['[RegExp]^(Hardware)'],
        },
    },
};
```

1.3. Reference

In the example below there is a list of all parameters which can be used for ACLs.

Please see the section on ACLs in the ProcessManagement documentation for a detailed descripton of how to use ACLs for process tickets.

Example 5.6. Reference showing all possible important ACL settings.

```
# ticket acl
$Self->{TicketAcl}->{'200-ACL-Reference'} = {
    # match properties (current values from the form)
    Properties => {

        # the used frontend module
        Frontend => {
            Action => ['AgentTicketPhone', 'AgentTicketEmail'],
        },

        # the logged in agent
        User => {
            UserLogin => ['some login'],
            Group_rw => [
                'hotline',
            ],
            Role => [
                'admin',
            ],
            # ...
        },

        # the logged in customer
        CustomerUser => {
            UserLogin => ['some login'],
            Group_rw => [
                'hotline',
            ],
            Role => [
                'admin',
            ],
            # ...
        },

        # process properties
        Process => {
            ProcessEntityID        => ['P1'],   # the Process that the current ticket is part
 of
            ActivityEntityID       => ['A1'],   # the current Activity of the ticket
            ActivityDialogEntityID => ['AD1'],  # the current ActivityDialog that the Agent/
Customer is using
        },

        # ticket properties
        Queue => {
            Name     => ['Raw'],
            QueueID  => ['some id'],
            GroupID  => ['some id'],
            Email    => ['some email'],
            RealName => ['OTRS System'],
            # ...
        },
        Service => {
            ServiceID => ['some id'],
            Name      => ['some name'],
            ParentID  => ['some id'],
            # ...
        },
        Type => {
            ID   => ['some id'],
            Name => ['some name'],
            # ...
        },
        Priority = {
            ID   => ['some id'],
            Name => ['some name'],
            # ...
        },
        SLA = {
            SLAID    => ['some id'],
            Name     => ['some name'],
            Calendar => ['some calendar'],
            # ...
        },
        State = {
            ID       => ['some id'],
            Name     => ['some name'],
            TypeName => ['some state type name'],,
            TypeID   => ['some state type id'],
            # ...
        },
        Owner => {
            UserLogin => ['some login'],
            Group_rw => [
                'some group',
            ],
            Role => [
                'admin',
            ],
            # ...
        },
        Responsible => {
            UserLogin => ['some login'],
            Group_rw => [
                'some group',
            ],
            Role => [
                'admin',
            ],
            # ...
        },
        DynamicField => {
            # Names must be in DynamicField_<field_name> format.
            # Values in [ ... ] must always be the untranslated internal data keys
            #   specified in the dynamic field definition and
            #   not the data values shown to the user.
            DynamicField_Field1          => ['some value'],
            DynamicField_OtherField      => ['some value'],
            DynamicField_TicketFreeText2 => ['some value'],
            # ...
        },
        # alternatively, ticket properties can be specified in the ticket hash
        Ticket => {
            Queue                => ['Raw'],
            State                => ['new', 'open'],
            Priority             => ['some priority'],
            Lock                 => ['lock'],
            CustomerID           => ['some id'],
            CustomerUserID       => ['some id'],
            Owner                => ['some owner'],
            DynamicField_Field1  => ['some value'],
            DynamicField_MyField => ['some value'],
            # ...
        },
    },

    # match properties (existing values from the database)
    PropertiesDatabase => {
        # See section "Properties", the same config can be used here.
        # ...
    }

    # return possible options (white list)
    Possible => {
        # possible ticket options (white list)
        Ticket => {
            Queue => ['Hotline', 'Coordination'],
            State => ['some state'],
            Priority => ['5 very high'],
            DynamicField_Field1  => ['some value'],
            DynamicField_MyField => ['some value'],
            # ...
            NewOwner => ['some owner'],
            OldOwner => ['some owner'],
            # ...
        },

        # Limit the number of possible ActivityDialogs the Agent/Customer
        #   can use in a process ticket.
        ActivityDialog => ['AD1', 'AD3'],

        # possible action options (white list)
        Action => {
            AgentTicketBounce        => 1,
            AgentTicketClose         => 1,
            AgentTicketCompose       => 0,
            AgentTicketCustomer      => 0,
            AgentTicketForward       => 0,
            AgentTicketFreeText      => 1,
            AgentTicketHistory       => 1,
            AgentTicketLink          => 0,
            AgentTicketLock          => 1,
            AgentTicketMerge         => 0,
            AgentTicketMove          => 1,
            AgentTicketNote          => 1,
            AgentTicketOwner         => 1,
            AgentTicketPending       => 1,
            AgentTicketPhone         => 1, # only used to hide the Split action
            AgentTicketPhoneInbound  => 0,
            AgentTicketPhoneOutbound => 1,
            AgentTicketPrint         => 1,
            AgentTicketPriority      => 0,
            AgentTicketResponsible   => 1,
            AgentTicketWatcher       => 1,
            AgentTicketZoom          => 1,
            AgentLinkObject          => 1, # only used to hide the Link action
        },
    },
    # remove options (black list)
    PossibleNot => {
        # See section "Possible"
        # ...
    },
};
```

Note

While matching ACLs if CustomerUserID parameter sent, the ACL mechanism will compare the defined ACLs using the supplied CustomerUserID to gather the CustomerUser details to fill the CustomerUser hash and it also overrides the Customer information in the Ticket hash for the Properties match. On the other hand this calculations are also made for the PropertiesDatabase part, but using the Ticket Customer as the basis to gather the data.

Notice that in Customer Interface, the CustomerUserID is always sent with the current logged Customer User.

Be aware that in ticket search screens (AgentTicketSearch and CustomerTicketSearch) the only affected attributes by ACLs are the Dynamic Fields. This means that this screens you can not restrict any other attribute like ticket type, state, queue, etc.

2. Process Management

2.1. Introduction

This feature of OTRS allows you to model processes (work-flows) in the ticket system. The basic idea is to be able to define recurring processes, and to delegate work items to different people, as well as leading the progress of a process in different directions based on certain criteria.

2.2. Example process

Let's see an example to make it more demonstrative. We will define a book order process:

2.2.1. Recording the demand

Before an order will be placed, the demand for literature by an employee will be recorded. The following book is needed in our example:

```
Title: Prozessmanagement für Dummies
Autor: Thilo Knuppertz
ISBN: 3527703713
```

2.2.2. Approval by manager

The head of the employee's department needs to decide on the order. In case of a denial, a reason should be recorded by the manager. In case of approval, the order is passed to the purchasing department.

2.2.3. Processing by purchasing department

Purchasing now has the task to find out where the book can be ordered with the best conditions. If it is out of stock, this can be recorded in the order. In case of a successful order purchasing will record the supplier, the price and the delivery date.

2.2.4. Processing by the mail room

The shipment will arrive at the company. The incoming goods department checks the shipment and records the date of receipt. Now the employee will be informed that their order has arrived and is ready to be collected.

2.3. Implementing the example

If we assume that a ticket acts in this work-flow like an accompanying document that can receive change notes, we already have a clear picture of process tickets.

From the analysis of the example process we can identify the following necessary items:

- Possibilities to record data, let's call them ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂,
- Checks which can react to changed data automatically, let's call them ✂✂✂✂✂✂✂✂✂✂✂,
- changes which can be applied to a process ticket after successful transitions of a process ticket, let's call them ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂.

We also need an additional item which might not be as obvious:

- A possibility to offer more than just one Activity Dialog to be available. In our example this is needed when the manager must have the choice between "Approve" and "Deny". Let's call this ✂✂✂✂✂✂✂✂.

Now, with Activities, Activity Dialogs, Transitions and Transition Actions we have the necessary tools to model the individual steps of our example. What's still missing is an area where for each work-flow the order of the steps can be specified. Let's call this ✂✂✂✂✂✂✂. To be able to refer to all these entities later, we will assign to them an abbreviation in parentheses. This abbreviation is based on an internal identification mechanism called EntityIDs.

The EntityIDs are conformed with one or two letters (depending on the process part or entity) and then a consecutive number, examples:

- Process: 'P1', 'P2' ... 'Pn'.
- Activity: 'A1', 'A2' ... 'An'.
- Activity Dialog: 'AD1', 'AD2' ... 'ADn'.
- Transition: 'T1', 'T2' ... 'Tn'.
- Transition Action: 'TA1', 'TA2' ... 'TAn'.

Before the creation of the process and its parts is necessary to prepare the system, we will need to define some Queues, Users and Dynamic Fields as well as set some SysConfig options.

Create the following Queues:

- Management
- Employees
- Purchasing
- Post office

Create the following Users:

- Manager
- Employee

Create the following Dynamic Fields:

- Title

Label	Title
Type	Text
Object	Ticket

- Author

Label	Author
Type	Text
Object	Ticket

- ISBN

Label	ISBN
Type	Text
Object	Ticket

- Status

Label	Status
Type	Dropdown
Object	Ticket
Possible Values	• Approval • Approval denied • Approved • Order denied • Order placed • Shipment received

Note: Please use this exactly this possible values for "Key" and "Value" in the Dynamic Field setup.

- Supplier

Label	Supplier
Type	Text
Object	Ticket

- Price

Label	Price
Type	Text
Object	Ticket

- DeliveryDate

Label	Delivery date
Type	Date
Object	Ticket

- DateOfReceipt

Label	Date Of Receipt
Type	Date
Object	Ticket

Set the the following SysConfig settings:

- 'Ticket::Responsible': Yes
- 'Ticket::Frontend::AgentTicketZoom###ProcessWidgetDynamicFieldGroups':

Key:	***Content:***
Book	Title, Author, ISBN
General	Status
Order	Price, Supplier, DeliveryDate
Shipment	DateOfReceipt

- 'Ticket::Frontend::AgentTicketZoom###ProcessWidgetDynamicField':

Key:	***Content:***
Author	1
DateOfReceipt	1
DeliveryDate	1
ISBN	1
Price	1
Status	1
Supplier	1
Title	1

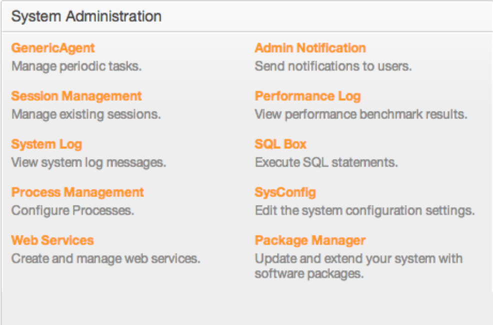
Now lets start with the real Process Management stuff. In the next step, we will define the individual entities that we need.

2.3.1. Process (as a container)

To create a new process is necessary to click on the "Process Management" link in the System Administration box in the Admin panel, this will lead to the Process Management Overview screen. After the creation of the process we can create all other entities (or process parts).

Note

Activities, Activity Dialogs, Transitions and Transition Actions defined in one process will be available for all the processes in the system.


System Administration

GenericAgent
Manage periodic tasks.

Admin Notification
Send notifications to users.

Session Management
Manage existing sessions.

Performance Log
View performance benchmark results.

System Log
View system log messages.

SQL Box
Execute SQL statements.

Process Management
Configure Processes.

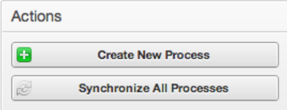
SysConfig
Edit the system configuration settings.

Web Services
Create and manage web services.

Package Manager
Update and extend your system with software packages.

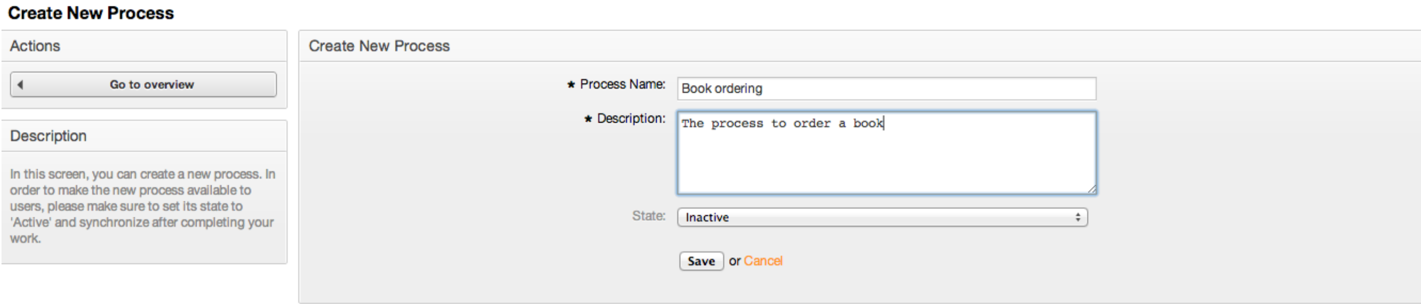
* * *

Click on the "Create New Process" action from the Actions box.


Actions
Create New Process
Synchronize All Processes

* * *

Fill the process information,set Process Name and the Description, we will leave the process State as "inactive", until we finish all the tasks. Save the process.


Create New Process
Actions
Go to overview
Description
Create New Process
Save
or Cancel

In this screen, you can create a new process. In order to make the new process available to users, please make sure to set its state to 'Active' and synchronize after completing your work.

* Process Name: Book ordering

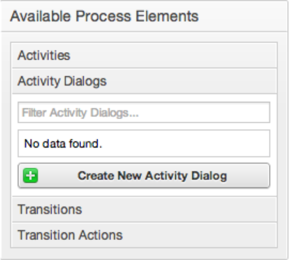
* Description: The process to order a book

State: Inactive

* * *

2.3.2. Activity Dialogs

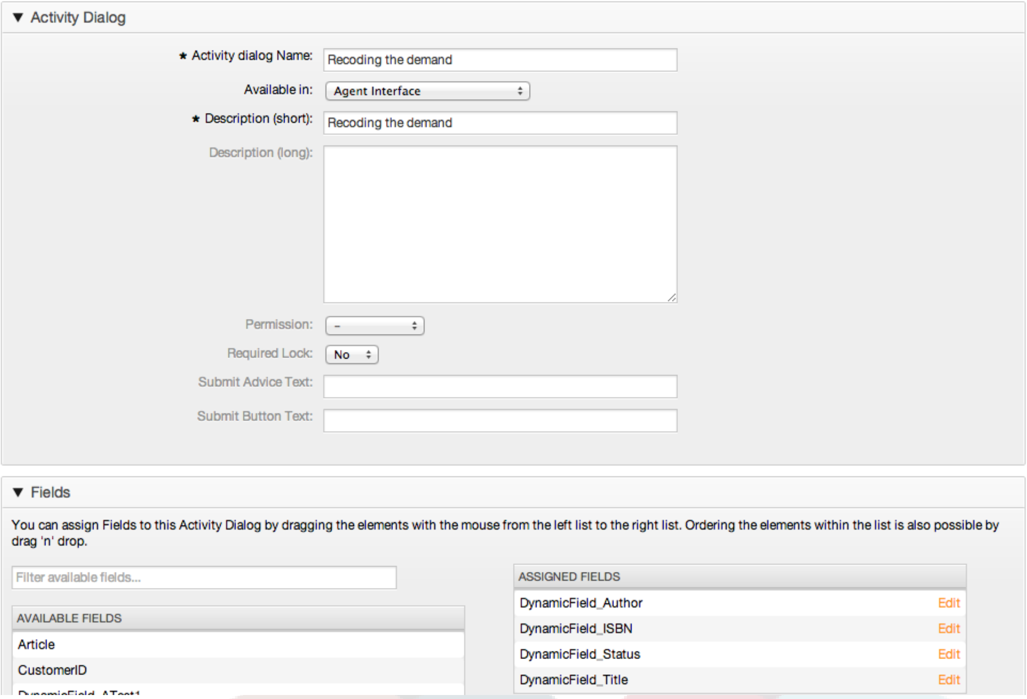
Click on the new process name in the Process Management Overview Screen, then in the "Available Process Elements" click in "Activity Dialogs" (this action will expand the activity dialog options and will collapse all others doing and accordion like effect), then click on "Create New Activity Dialog".


Available Process Elements
Activities
Activity Dialogs
Filter Activity Dialogs...
Create New Activity Dialog
Transitions
Transition Actions

No data found.

* * *

In the opened popup screen fill the "Activity dialog Name" as well as the "Description (short)" fields, for this example we will leave all other fields as the default, to assign fields to the Activity Dialog simple drag the required field from the "Available Fields" pool and drop into the "Assigned Fields" pool. The order in the "Assigned Fields" pool is the order as the fields will have in the screen, to modify the order simply drag and drop the filed within the pool to rearrange it in the correct place.



- ★ Activity dialog Name: Recoding the demand
- Available in: Agent Interface
- ★ Description (short): Recoding the demand
- Description (long):
- Permission: -
- Required Lock: No
- Submit Advice Text:
- Submit Button Text:

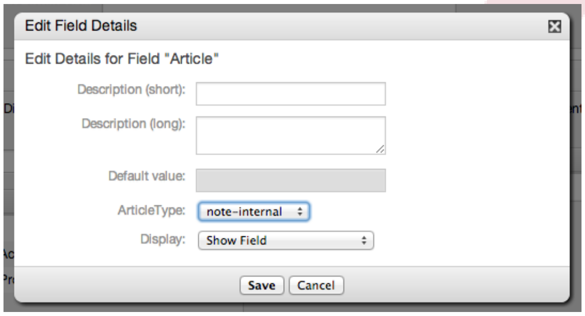
You can assign Fields to this Activity Dialog by dragging the elements with the mouse from the left list to the right list. Ordering the elements within the list is also possible by drag 'n' drop.

AVAILABLE FIELDS
Article
CustomerID

ASSIGNED FIELDS	
DynamicField_Author	Edit
DynamicField_ISBN	Edit
DynamicField_Status	Edit
DynamicField_Title	Edit

✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂

As soon as the fields are dropped into the "Assigned Fields" pool another popup screen is shown with some details about the field, we will leave the default options and only for Article fields we should make sure that the ArticleType field is set to "note-internal".



Edit Details for Field "Article"

- Description (short):
- Description (long):
- Default value:
- ArticleType: note-internal
- Display: Show Field

✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂✂

After all fields are assigned click on the submit button in the main popup screen to save the changes.

In this example we will use Article field for comments, but another option could be to create a TextArea type Dynamic Field, the rest of the mentioned fields in the lines below are the Dynamic Fields that we define before.

Please be aware that in this screen all the Dynamic Fields has the prefix "DynamicField_" as in "DynamicField_Title", Do not confuse with the field "Title" that is the Ticket Title.

Create the following Activity Dialogs:

- "Recoding the demand" (AD1)

  An Activity Dialog that contains all the required fields for the data to be collected for the order (Title, Author and ISBN), and a Status field with the possibility to choose "Approval".

- "Approval denied" (AD2)

  An Activity Dialog with a comment field (Article) and a Status field with the option "Approval denied".

- "Approved" (AD3)

  Here we just need the Status field with the option "Approved".

- "Order denied" (AD4)

  An activity dialog which makes it possible for purchasing to reject an impossible order (book out of stock). Here we also need a comment field and the Status field with the option "Order denied".

- "Order placed" (AD5)

  An activity dialog with the fields Supplier, Price and Delivery date for purchasing and the Status field with the option "Order placed".

- "Shipment received" (AD6)

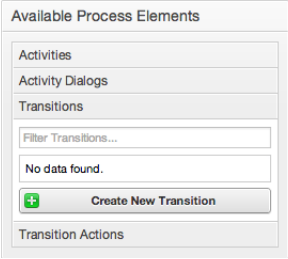
  An activity for the mail room with a field for the Date of receipt and the Status field with the option "Shipment received".

To restrict the Status field for each activity dialog we need to add some ACLs in the Kernel/Config.pm or to a new perl file located in Kernel/Config/Files.

```
$Self->{TicketAcl}->{'P1-AD1-1'} = {
    Properties => {
        Process => {
            ActivityDialogEntityID => ['AD1'],
        },
    },
    Possible => {
        Ticket => {
            DynamicField_Status => ['Approval'],
        },
    },
};

$Self->{TicketAcl}->{'P1-AD2-1'} = {
    Properties => {
        Process => {
            ActivityDialogEntityID => ['AD2'],
        },
    },
    Possible => {
        Ticket => {
                DynamicField_Status => ['Approval denied'],
            },
        },
    };

    $Self->{TicketAcl}->{'P1-AD3-1'} = {
        Properties => {
            Process => {
                ActivityDialogEntityID => ['AD3'],
            },
        },
        Possible => {
            Ticket => {
                DynamicField_Status => ['Approved'],
            },
        },
    };

    $Self->{TicketAcl}->{'P1-AD4-1'} = {
        Properties => {
            Process => {
                ActivityDialogEntityID => ['AD4'],
            },
        },
        Possible => {
            Ticket => {
                DynamicField_Status => ['Order denied'],
            },
        },
    };

    $Self->{TicketAcl}->{'P1-AD5-1'} = {
        Properties => {
            Process => {
                ActivityDialogEntityID => ['AD5'],
            },
        },
        Possible => {
            Ticket => {
                DynamicField_Status => ['Order placed'],
            },
        },
    };

    $Self->{TicketAcl}->{'P1-AD6-1'} = {
        Properties => {
            Process => {
                ActivityDialogEntityID => ['AD6'],
            },
        },
        Possible => {
            Ticket => {
                DynamicField_Status => ['Shipment received'],
            },
        },
    };
```

2.3.3. Transitions

In the "Available Process Elements" click in "Transitions", then click on "Create New Transition".

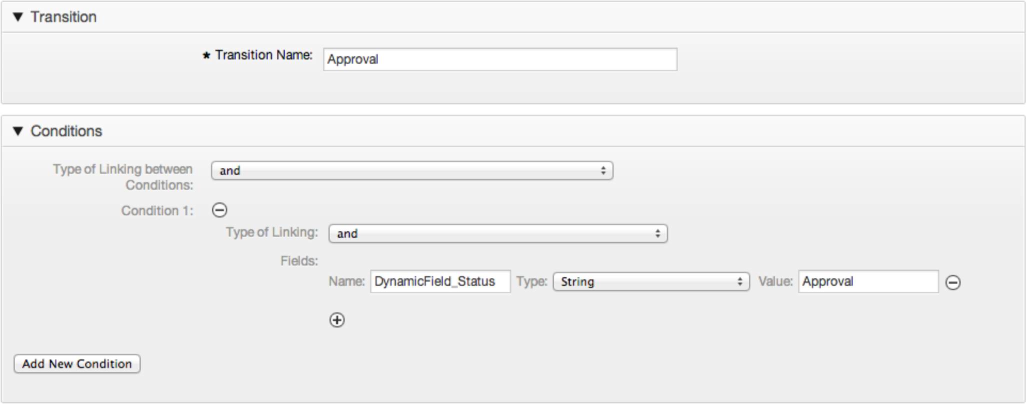


Available Process Elements

- Activities
- Activity Dialogs
- Transitions
  - Filter Transitions...
  - No data found.
  - Create New Transition
- Transition Actions

✁✁✁✁✁✁✁ ✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁

In the opened popup screen fill the "Transition Name", then in the conditions, for this examples we will use just one condition and just one field, for both we can leave the Type of Linking as "and" and we will use the filed match type value as "String".



▼ Transition

★ Transition Name: Approval

▼ Conditions

Type of Linking between Conditions: and

Condition 1:

Type of Linking: and

Fields:

Name: DynamicField_Status Type: String Value: Approval

✁✁✁✁✁✁✁ ✁✁✁ ✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁

After all conditions are set click on the submit button to save the changes.

Create the following Transitions:

- "Approval" (T1)

  A transition which checks if the Status field is set to "Approval".

- "Approval denied" (T2)

  A transition which checks if the Status field is set to "Approval denied".

- "Approved" (T3)

  A transition which checks if the Status field is set to "Approved".

- "Order denied" (T4)

  A transition which checks if the Status field is set to "Order denied".

- "Order placed" (T5)

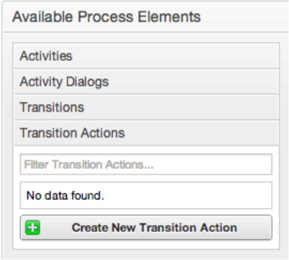
  A transition which checks if the Status field is set to "Order placed".

- "Shipment received" (T6)

  A transition which checks if the Status field is set to "Shipment received".

2.3.4. Transition Actions

Click on "Transition Actions" in the "Available Process Elements", then click on "Create New Transition Action".


Available Process Elements
Activities
Activity Dialogs
Transitions
Transition Actions
Filter Transition Actions...
No data found.
Create New Transition Action

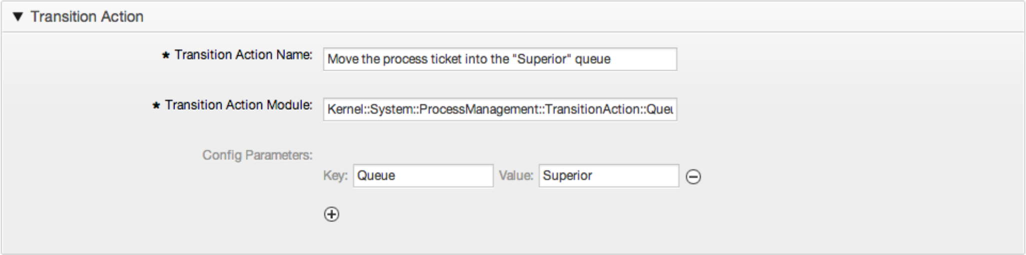
✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂✂

In the opened popup screen fill the "Transition Name", and the "Transition Action module" then add the required and optional parameter names and values.

All the Transition Action Modules are located in Kernel/System/ProcessManagement/TransitionAction and the following is the list of bundled Transition Actions included in this release

- DynamicFieldSet
- TicketArticleCreate
- TicketCustomerSet
- TicketLockSet
- TicketOwnerSet
- TicketQueueSet
- TicketResponsibleSet
- TicketServiceSet
- TicketSLASet
- TicketStateSet
- TicketTitleSet
- TicketTypeSet

Each module has its own and different parameters. Please review the module documentation to learn all require and optional parameters.


Transition Action
Config Parameters:

* Transition Action Name: Move the process ticket into the "Superior" queue

* Transition Action Module: Kernel::System::ProcessManagement::TransitionAction::Queu

Key: Queue Value: Superior

✂✂✂✂✂✂✂ ✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂

After all parameters and values are set click on the submit button to save the changes.

Create the following Transitions Actions:

- "Move the process ticket into the 'Management' queue" (TA1)

  This action is supposed to be executed when the Transition "Approval" (T1) applied.

- "Change ticket responsible to 'manager'" (TA2)

  To be executed when the Transition "Approval" (T1) applied.

- "Move the process ticket into the 'Employees' queue"" (TA3)

  To be executed when:

  - The Transition "Approval denied" (T2) applied
  - The Transition "Order denied" (T4) applied
  - The Transition "Shipment received" (T6) applied

- "Change ticket responsible to 'Employee'" (TA4)

  To be executed when:

  - The transition "Approval denied" (T2) applied
  - The transition "Order denied" (T4) applied
  - The transition "Shipment received" (T6) applied

- "Move process ticket into the 'Purchasing' queue" (TA5)

  To be executed when the transition "Approved" (T3) applied.

- "Move process ticket into the 'Post office' queue" (TA6)

  To be executed when the transition "Order placed" (T5) applied.

- "Close ticket successfully" (TA7)

  To be executed when:

  - The transition "Shipment received" (T6) applied

- "Close ticket unsuccessfully" (TA8)

  To be executed when:

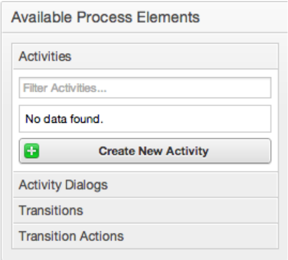
  - The transition "Approval denied" (T2) applied
  - The transition "Order denied" (T4) applied

As you can see, there are places where the same Transition Actions should be executed. Therefore it is reasonable to make it possible to link Transition Actions freely with Transitions to be able to reuse them.

2.3.5. Activities

We chose the approach to see Activities as a basket which can contain one or more Activity Dialogs.

Click on "Activities" in the "Available Process Elements", then click on "Create New Activity".



✂✂✂✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂

In the opened popup screen fill the "Activity Name", Then drap the required Activity Dialogs from the "Available Activity Dialogs" pool, and drop then into to the "Assigned Activity Dialogs" pool. This dialogs will be presented (in the ticket zoom screen) in the same order as it is defined on this screen translating from top to bottom, to left to right.

This order is specially important in the first Activity, since the first Activity Dialog for this activity is the only one that is presented when the process starts

Create the following Activities:

- "Recording the demand" (A1)

  Contains the Activity Dialog "Recording the demand" (AD1)

- "Approval" (A2)

  Contains the Activity Dialogs "Approval denied" (AD2) as well as "Approved" (AD3)

- "Order" (A3)

  Contains the Activity Dialogs "Order rejected" (AD4) as well as "Order placed" (AD5)

- "Incoming" (A4)

  Contains the Activity Dialog "Shipment received" (AD6)

- "Process complete" (A5): This is an Activity without possible Activity Dialogs. It will be set after "Approval denied", "Order denied" or "Shipment received" and represents the end of the process.

Now we can clearly see that Activities are precisely defined states of a process ticket. After a successful Transition a process ticket moves from one Activity to another.

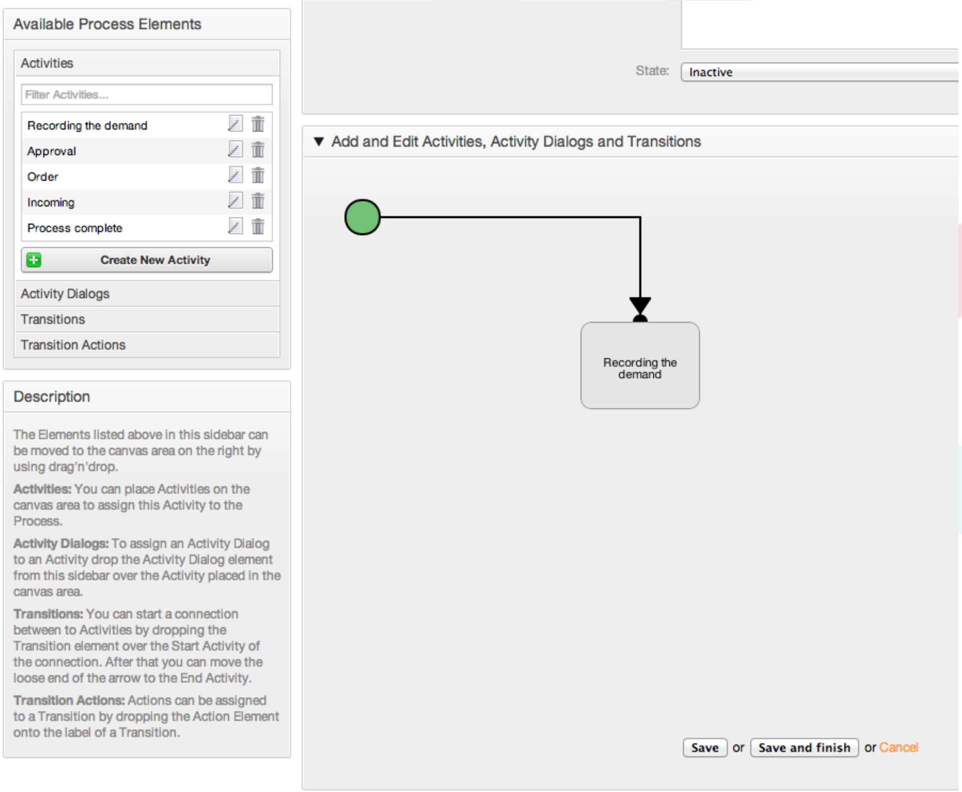
2.3.6. Book ordering process Path

Let us conclude our example with the last missing piece in the puzzle, the Process as the a flow describer. In our case this is the whole ordering work-flow. Other processes could be office supply ordering or completely different processes.

The process has a starting point which consists of the start Activity and the start Activity Dialog. For any new book order, the start Activity Dialog (first Activity Dialog for the first Activity) is the first screen that is displayed. If this is completed and saved, the Process ticket will be created and can follow the configured work-flow.

The process also contains the directions for how the process ticket can move through the Process. Let's call this the "Path". It consists of the start Activity, one or more Transitions (possibly with Transition Actions), and other Activities.

Assuming that the Activities has already assigned their Activity Dialogs drag an Activity from the accordion (in the left part of the screen) and drop it into the canvas area (below process information). Notice that an arrow from the process start (green circle) to the Activity is placed automatically. (This is the first Activity and its first Activity Dialog is the first screen that will be shown when the process starts).


Available Process Elements
Activities
Filter Activities...
Recording the demand
Approval
Order
Incoming
Process complete
Create New Activity
Activity Dialogs
Transitions
Transition Actions
Description
State:
Inactive
Add and Edit Activities, Activity Dialogs and Transitions
Recording the demand
Save
or
Save and finish
or Cancel

The Elements listed above in this sidebar can be moved to the canvas area on the right by using drag'n'drop.

Activities: You can place Activities on the canvas area to assign this Activity to the Process.

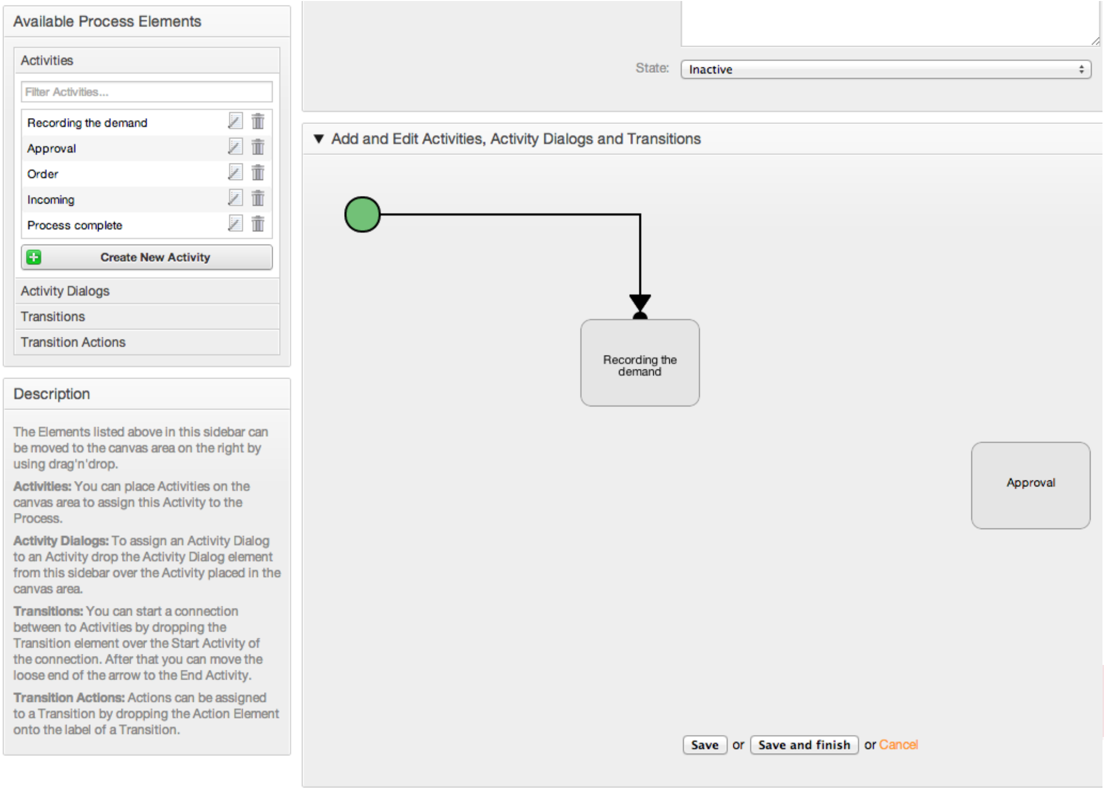
Activity Dialogs: To assign an Activity Dialog to an Activity drop the Activity Dialog element from this sidebar over the Activity placed in the canvas area.

Transitions: You can start a connection between to Activities by dropping the Transition element over the Start Activity of the connection. After that you can move the loose end of the arrow to the End Activity.

Transition Actions: Actions can be assigned to a Transition by dropping the Action Element onto the label of a Transition.

* * *

Next, drag another Activity into the canvas too. now we will have two Activities in the canvas the first one is connected to the start point and the second has no connections, you can hover the mouse over each activity to reveal their own Activity Dialogs.


Available Process Elements
Activities
Filter Activities...
Recording the demand
Approval
Order
Incoming
Process complete
Create New Activity
Activity Dialogs
Transitions
Transition Actions
State:
Inactive
Add and Edit Activities, Activity Dialogs and Transitions
Recording the demand
Approval
Save
or
Save and finish
or Cancel

Description

The Elements listed above in this sidebar can be moved to the canvas area on the right by using drag'n'drop.

Activities: You can place Activities on the canvas area to assign this Activity to the Process.

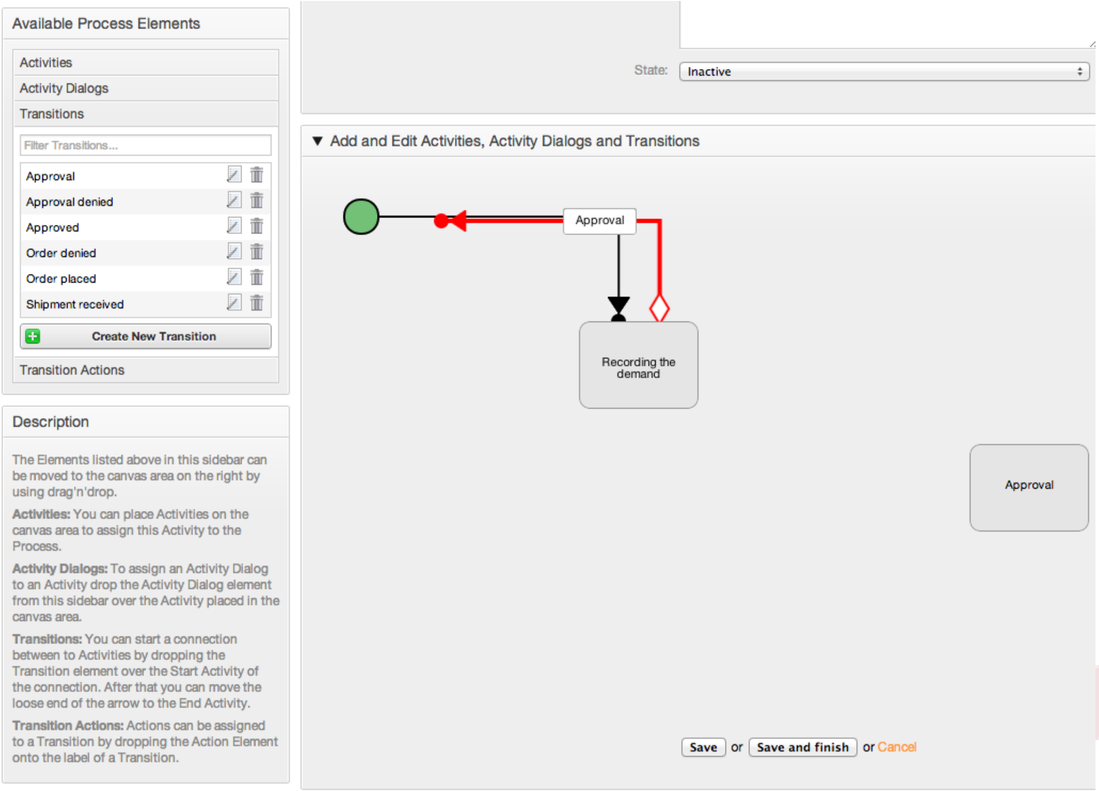
Activity Dialogs: To assign an Activity Dialog to an Activity drop the Activity Dialog element from this sidebar over the Activity placed in the canvas area.

Transitions: You can start a connection between to Activities by dropping the Transition element over the Start Activity of the connection. After that you can move the loose end of the arrow to the End Activity.

Transition Actions: Actions can be assigned to a Transition by dropping the Action Element onto the label of a Transition.

✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂

Then let's create the "Path" (connection) between this two Activities, for this we will use the Transitions, Click on Transitions in the accordion drag a Transition and drop it inside the first Activity, notice that the Activity change its color indicating that the Transition is attached, as soon as the Transition is dropped the end point of the Transition arrow will be placed next to the process start point. Drag the Transition arrow end point and drop it inside the other Activity to create the connection between the Activities.



Available Process Elements
Activities
Activity Dialogs
Transitions
Filter Transitions...
Approval
Approval denied
Approved
Order denied
Order placed
Shipment received
Create New Transition
Transition Actions
State:
Inactive
Add and Edit Activities, Activity Dialogs and Transitions
Approval
Recording the demand
Approval
Save
or
Save and finish
or Cancel

Description

The Elements listed above in this sidebar can be moved to the canvas area on the right by using drag'n'drop.

Activities: You can place Activities on the canvas area to assign this Activity to the Process.

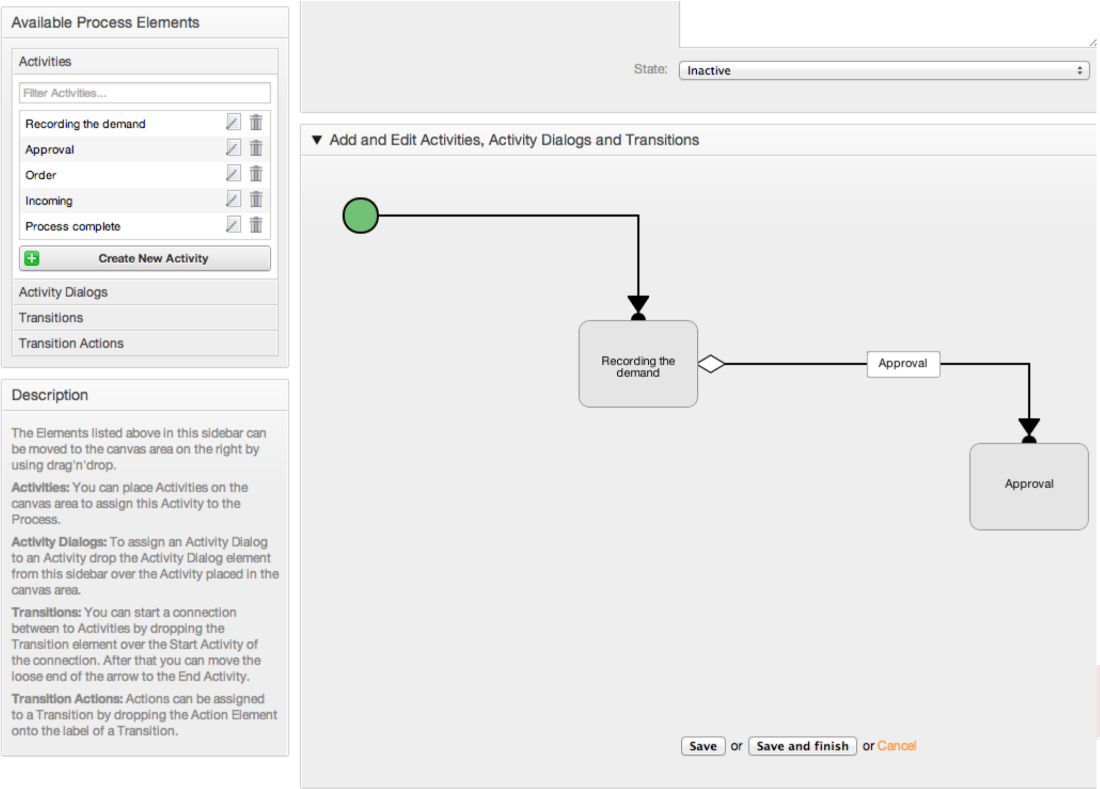
Activity Dialogs: To assign an Activity Dialog to an Activity drop the Activity Dialog element from this sidebar over the Activity placed in the canvas area.

Transitions: You can start a connection between to Activities by dropping the Transition element over the Start Activity of the connection. After that you can move the loose end of the arrow to the End Activity.

Transition Actions: Actions can be assigned to a Transition by dropping the Action Element onto the label of a Transition.

✂✂✂✂✂✂✂ ✂✂✂✂ ✂ ✂✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂ ✂✂✂✂✂✂✂

Now that the "Path" between the Actions is defined, then we need to assign the Transition Actions to the Transition, double click the Transition label (in the canvas), this will open a new popup window.


Available Process Elements
Activities
Filter Activities...
Recording the demand
Approval
Order
Incoming
Process complete
Create New Activity
Activity Dialogs
Transitions
Transition Actions
State:
Inactive
Add and Edit Activities, Activity Dialogs and Transitions
Recording the demand
Approval
Approval
Save
or
Save and finish
or Cancel

Description

The Elements listed above in this sidebar can be moved to the canvas area on the right by using drag'n'drop.

Activities: You can place Activities on the canvas area to assign this Activity to the Process.

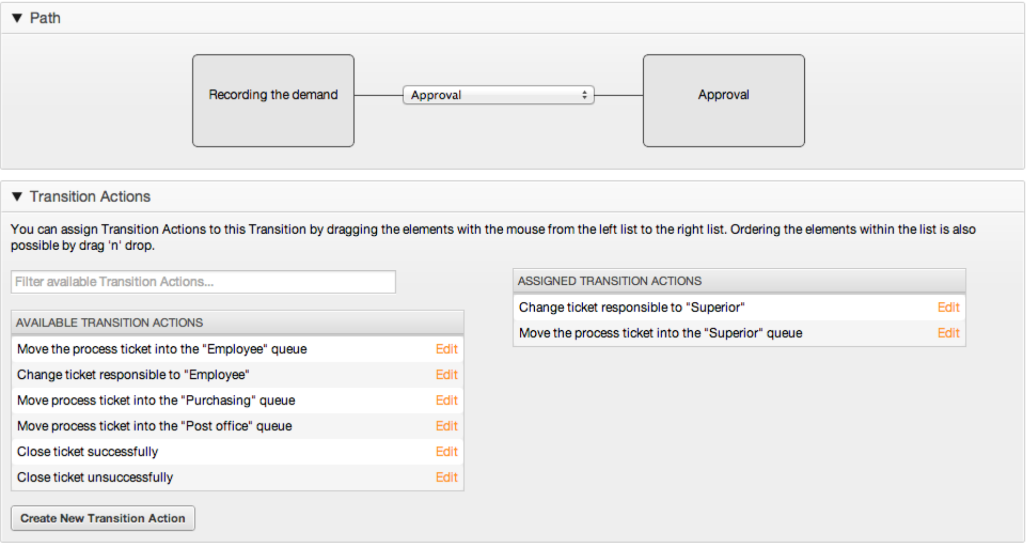
Activity Dialogs: To assign an Activity Dialog to an Activity drop the Activity Dialog element from this sidebar over the Activity placed in the canvas area.

Transitions: You can start a connection between to Activities by dropping the Transition element over the Start Activity of the connection. After that you can move the loose end of the arrow to the End Activity.

Transition Actions: Actions can be assigned to a Transition by dropping the Action Element onto the label of a Transition.

* * *

Drag the needed Transition Actions from Available Transition Actions pool and drop them into the Assigned Transition Actions pool and click on submit button.


Path
Recording the demand
Approval
Approval
Transition Actions
Filter available Transition Actions...
Create New Transition Action

You can assign Transition Actions to this Transition by dragging the elements with the mouse from the left list to the right list. Ordering the elements within the list is also possible by drag 'n' drop.

AVAILABLE TRANSITION ACTIONS	
Move the process ticket into the "Employee" queue	Edit
Change ticket responsible to "Employee"	Edit
Move process ticket into the "Purchasing" queue	Edit
Move process ticket into the "Post office" queue	Edit
Close ticket successfully	Edit
Close ticket unsuccessfully	Edit

ASSIGNED TRANSITION ACTIONS	
Change ticket responsible to "Superior"	Edit
Move the process ticket into the "Superior" queue	Edit

* * *

Then back in the main process edit screen click on save button below the canvas to save all other changes.

Complete the "path"" adding the following Activities, Transitions and Transition Actions:

Recording the demand until "Approval"

- Starting point: Activity: Recording the demand (A1)
- Possible Transition: Approval (T1)
  - If the condition of this activity is fulfilled, the ticket will move to Activity: Approval (A2)
  - Additionally, the following TransitionActions are executed:
    - "Moving the ticket into the 'Management' Queue" (TA1)
    - "Set ticket responsible to 'manager'" (TA2)

The Activity: "Recording the demand" (A1) is a defined step of the process ticket, where there is the possibility for the Transition: "Approval" (T1). If this applies, the ticket will move to the next Activity: "Approval" (A2), and the Transition Actions: "Moving the ticket into the 'Management' Queue" (TA1) and "Set ticket responsible to 'manager'" (TA2) are executed. In the Activity: "Approval" (A2), the Activity Dialogs: "Approval denied" (AD2) and "Approved" (AD3) are available.

Approval

- Starting Point: Activity "Approval" (A2)
- Possible Transitions:
  - "Approval denied" (T2)
    - If this matches, the process ticket will move to Activity:"Process complete" (A5).
    - Additionally, the following Transition Actions are executed:
      - "Move process ticket to the 'Employees' Queue" (TA3)
      - "Set ticket responsible to 'Employee'" (TA4)
      - "Close ticket unsuccessfully" (TA8)
  - "Approved" (T3)
    - If this matches, the process ticket will move to Activity: "Order" (A3).
    - Additionally, the following Transition Action is executed:
      - "Move process ticket to 'Purchasing' Queue" (TA5)

We can see that from the current Activity, which defines a step of the process ticket, there are one or more possibilities for Transition which have exactly one target Activity (and possibly one or more Transition Actions).

Order

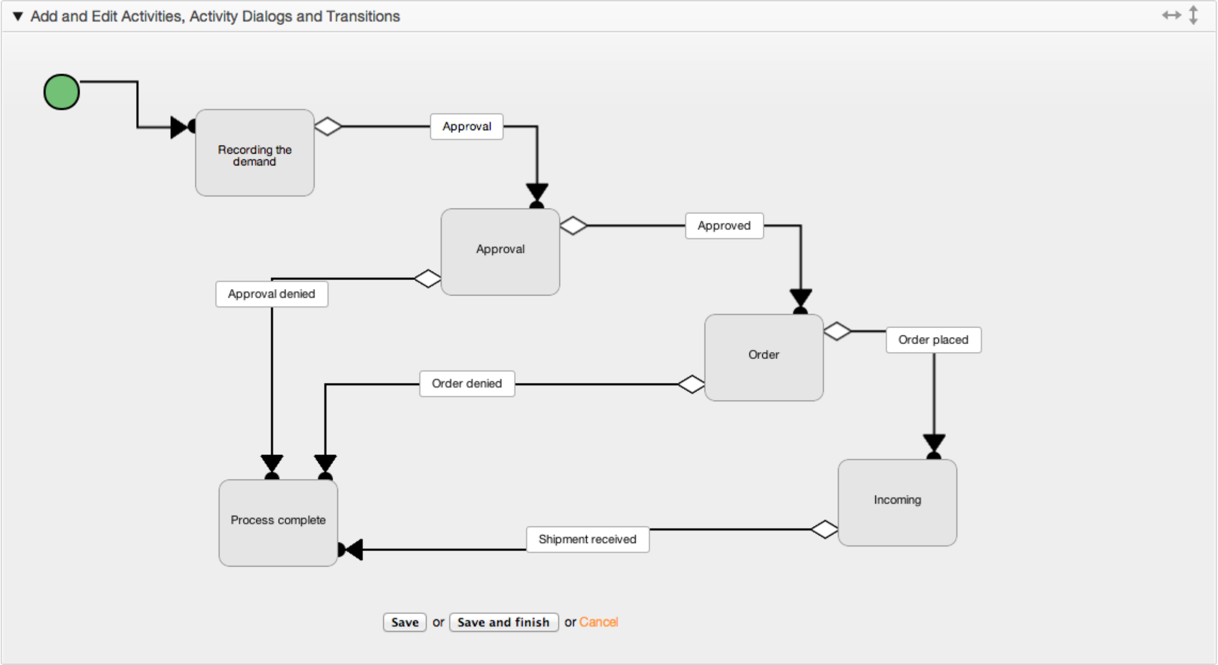
- Starting Point: Activity "Order" (A3)
- Possible Transitions:
  - "Order denied" (T4)

    - If this matches, the process ticket will move to Activity:"Process complete" (A5).
    - Additionally, the following Transition Actions are executed:
        - "Move process ticket to the 'Employees' Queue" (TA3)
        - "Set ticket responsible to 'Employee'" (TA4)
        - "Close ticket unsuccessfully" (TA8)
  - "Order placed" (T5)
    - If this matches, the process ticket will move to Activity: "Incoming" (A5).
    - Additionally, the following Transition Action is executed:
        - "Move process ticket to 'Post office' Queue" (TA6)

Incoming

- Starting Point: Activity "Incoming" (A4)
- Possible Transitions:
  - "Shipment received" (T6)
    - If this matches, the process ticket will move to Activity:"Process complete" (A5).
    - Additionally, the following Transition Actions are executed:
      - "Move process ticket to the 'Employees' Queue" (TA3)
      - "Set ticket responsible to 'Employee'" (TA4)
      - "Close ticket successfully" (TA7)

The complete Path for the book ordering Process will then look like this:


▼ Add and Edit Activities, Activity Dialogs and Transitions
Recording the demand
Approval
Approval
Approved
Approval denied
Order
Order placed
Order denied
Process complete
Incoming
Shipment received
Save
or
Save and finish
or Cancel

* * *

After you finish the process path please click on "Save" button in the lower part of the canvas and then click on "Synchronize All Processes" button. This will gather all processes information form the Database and create a cache file (in Perl language). This cache file is acctually the processes configuration that the system will use to create or use process tickets.

Any change that is made of the process (in the GUI) will require to re-synchronize the cache file in order to get the change reflected in the system.

It is also possible to import the whole process from a YAML file, but it is still necessary to create all Dynamic Fields, Users, Queues, etc that are needed by each process before the import.

Notice that if the process requires the use of ACLs those are also needed to be set manually.

The follwing is the complete YAML file for the book ordering process example:

```
---
Activities:
  A1:
    ActivityDialogs:
      - AD1
    ChangeTime: 2012-11-23 14:49:22
    Config:
      ActivityDialog:
        1: AD1
    CreateTime: 2012-11-23 11:49:38
    EntityID: A1
    ID: 151
    Name: Recording the demand
  A2:
    ActivityDialogs:
      - AD2
      - AD3
    ChangeTime: 2012-12-13 00:55:12
    Config:
      ActivityDialog:
        1: AD2
        2: AD3
    CreateTime: 2012-11-23 11:50:11
    EntityID: A2
    ID: 152
    Name: Approval
  A3:
    ActivityDialogs:
      - AD4
      - AD5
    ChangeTime: 2012-11-23 18:12:14
    Config:
      ActivityDialog:
        1: AD4
        2: AD5
    CreateTime: 2012-11-23 11:50:35
    EntityID: A3
    ID: 153
    Name: Order
  A4:
    ActivityDialogs:
      - AD6
    ChangeTime: 2012-11-23 18:12:35
    Config:
      ActivityDialog:
        1: AD6
    CreateTime: 2012-11-23 11:51:00
    EntityID: A4
    ID: 154
    Name: Incoming
  A5:
    ActivityDialogs: []
    ChangeTime: 2012-11-23 11:51:33
    Config: {}
    CreateTime: 2012-11-23 11:51:33
    EntityID: A5
    ID: 155
    Name: Process complete
ActivityDialogs:
  AD1:
    ChangeTime: 2012-12-06 02:16:21
    Config:
      DescriptionLong: ''
      DescriptionShort: Recoding the demand
      FieldOrder:
        - DynamicField_Author
        - DynamicField_ISBN
        - DynamicField_Title
        - DynamicField_Status
      Fields:
        DynamicField_Author:
          DefaultValue: ''
          DescriptionLong: ''
          DescriptionShort: ''
          Display: 1
        DynamicField_ISBN:
          DefaultValue: ''
          DescriptionLong: ''
          DescriptionShort: ''
          Display: 1
        DynamicField_Status:
          DefaultValue: ''
          DescriptionLong: ''
          DescriptionShort: ''
          Display: 1
        DynamicField_Title:
          DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
    Interface:
      - AgentInterface
    Permission: ''
    RequiredLock: 0
    SubmitAdviceText: ''
    SubmitButtonText: ''
  CreateTime: 2012-11-23 14:34:43
  EntityID: AD1
  ID: 154
  Name: Recording the demand
AD2:
  ChangeTime: 2012-11-23 14:57:41
  Config:
    DescriptionLong: ''
    DescriptionShort: Approval denied
    FieldOrder:
      - Article
      - DynamicField_Status
    Fields:
      Article:
        Config:
          ArticleType: note-internal
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
      DynamicField_Status:
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
    Interface:
      - AgentInterface
    Permission: ''
    RequiredLock: 0
    SubmitAdviceText: ''
    SubmitButtonText: Deny Request
  CreateTime: 2012-11-23 14:36:39
  EntityID: AD2
  ID: 155
  Name: Approval denied
AD3:
  ChangeTime: 2012-12-14 03:14:23
  Config:
    DescriptionLong: ''
    DescriptionShort: Approved
    FieldOrder:
      - DynamicField_Status
    Fields:
      DynamicField_Status:
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
    Interface:
      - AgentInterface
    Permission: ''
    RequiredLock: 0
    SubmitAdviceText: ''
    SubmitButtonText: Approve Request
  CreateTime: 2012-11-23 14:37:35
  EntityID: AD3
  ID: 156
  Name: Approved
```

```
AD4:
  ChangeTime: 2012-11-23 14:58:52
  Config:
    DescriptionLong: ''
    DescriptionShort: Order rejected
    FieldOrder:
      - Article
      - DynamicField_Status
    Fields:
      Article:
        Config:
          ArticleType: note-internal
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
      DynamicField_Status:
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
    Interface:
      - AgentInterface
    Permission: ''
    RequiredLock: 0
    SubmitAdviceText: ''
    SubmitButtonText: Reject Order
  CreateTime: 2012-11-23 14:38:48
  EntityID: AD4
  ID: 157
  Name: Order rejected
AD5:
  ChangeTime: 2012-12-06 02:20:12
  Config:
    DescriptionLong: ''
    DescriptionShort: Order placed
    FieldOrder:
      - DynamicField_DeliveryDate
      - DynamicField_Price
      - DynamicField_Supplier
      - DynamicField_Status
    Fields:
      DynamicField_DeliveryDate:
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
      DynamicField_Price:
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
      DynamicField_Status:
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
      DynamicField_Supplier:
        DefaultValue: ''
        DescriptionLong: ''
        DescriptionShort: ''
        Display: 1
    Interface:
      - AgentInterface
    Permission: ''
    RequiredLock: 0
    SubmitAdviceText: ''
```

```
      SubmitButtonText: Place Order
    CreateTime: 2012-11-23 14:41:28
    EntityID: AD5
    ID: 158
    Name: Order placed
  AD6:
    ChangeTime: 2012-11-23 14:42:43
    Config:
      DescriptionLong: ''
      DescriptionShort: Shipment received
      FieldOrder:
        - DynamicField_DateOfReceipt
        - DynamicField_Status
      Fields:
        DynamicField_DateOfReceipt:
          DefaultValue: ''
          DescriptionLong: ''
          DescriptionShort: ''
          Display: 1
        DynamicField_Status:
          DefaultValue: ''
          DescriptionLong: ''
          DescriptionShort: ''
          Display: 1
      Interface:
        - AgentInterface
      Permission: ''
      RequiredLock: 0
      SubmitAdviceText: ''
      SubmitButtonText: ''
    CreateTime: 2012-11-23 14:42:43
    EntityID: AD6
    ID: 159
    Name: Shipment received
Process:
  Activities:
    - A1
    - A2
    - A3
    - A4
    - A5
  ChangeTime: 2012-12-06 02:31:59
  Config:
    Description: The process to order a book
    Path:
      A1:
        T1:
          ActivityEntityID: A2
          TransitionAction:
            - TA2
            - TA1
      A2:
        T2:
          ActivityEntityID: A5
          TransitionAction:
            - TA3
            - TA4
            - TA8
        T3:
          ActivityEntityID: A3
          TransitionAction:
            - TA5
      A3:
        T4:
          ActivityEntityID: A5
          TransitionAction:
            - TA3
            - TA4
            - TA8
        T5:
          ActivityEntityID: A4
          TransitionAction:
            - TA6
      A4:
        T6:
          ActivityEntityID: A5
          TransitionAction:
            - TA3
            - TA4
            - TA7
      A5: {}
    StartActivity: A1
    StartActivityDialog: AD1
  CreateTime: 2012-11-23 11:45:12
  EntityID: P1
  ID: 94
  Layout:
    A1:
      left: 172
      top: 63
    A2:
      left: 402
      top: 156
    A3:
      left: 649
      top: 255
    A4:
      left: 774
      top: 391
    A5:
      left: 194
      top: 410
  Name: Book ordering
  State: Active
  StateEntityID: S1
  TransitionActions:
    - TA1
    - TA2
    - TA3
    - TA4
    - TA8
    - TA5
    - TA3
    - TA4
    - TA8
    - TA6
    - TA3
    - TA4
    - TA7
  Transitions:
    - T1
    - T2
    - T3
    - T4
    - T5
    - T6
TransitionActions:
  TA1:
    ChangeTime: 2012-11-23 16:01:37
    Config:
      Config:
        Queue: Management
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet
    CreateTime: 2012-11-23 15:50:59
    EntityID: TA1
    ID: 61
    Name: Move the process ticket into the "Management" queue
  TA2:
    ChangeTime: 2012-11-23 16:02:12
    Config:
      Config:
        Responsible: manager
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketResponsibleSet
    CreateTime: 2012-11-23 15:58:22
    EntityID: TA2
    ID: 62
    Name: Change ticket responsible to "manager"
  TA3:
    ChangeTime: 2012-11-24 14:27:02
    Config:
      Config:
        Queue: Employees
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet
    CreateTime: 2012-11-23 16:02:54
    EntityID: TA3
    ID: 63
    Name: Move the process ticket into the "Employees" queue
  TA4:
    ChangeTime: 2012-11-23 16:04:06
    Config:
      Config:
        Responsible: Employee
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketResponsibleSet
    CreateTime: 2012-11-23 16:04:06
    EntityID: TA4
    ID: 64
    Name: Change ticket responsible to "Employee"
  TA5:
    ChangeTime: 2012-12-06 02:18:34
    Config:
      Config:
        Queue: Purchasing
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet
    CreateTime: 2012-11-23 16:04:54
    EntityID: TA5
    ID: 65
    Name: Move process ticket into the "Purchasing" queue
  TA6:
    ChangeTime: 2012-12-06 02:18:48
    Config:
      Config:
        Queue: Post office
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet
    CreateTime: 2012-11-23 16:06:20
    EntityID: TA6
    ID: 66
    Name: Move process ticket into the "Post office" queue
  TA7:
    ChangeTime: 2012-12-06 02:29:55
    Config:
      Config:
        State: closed successful
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketStateSet
    CreateTime: 2012-12-06 02:29:27
    EntityID: TA7
    ID: 67
    Name: Close ticket successfully
  TA8:
    ChangeTime: 2012-12-06 02:31:12
    Config:
      Config:
        State: closed unsuccessful
      Module: Kernel::System::ProcessManagement::TransitionAction::TicketStateSet
    CreateTime: 2012-12-06 02:31:12
    EntityID: TA8
    ID: 68
    Name: Close ticket unsuccessfully
Transitions:
  T1:
    ChangeTime: 2012-11-23 15:12:20
    Config:
      Condition:
        1:
          Fields:
            DynamicField_Status:
              Match: Approval
              Type: String
          Type: and
      ConditionLinking: and
    CreateTime: 2012-11-23 11:53:52
    EntityID: T1
    ID: 94
    Name: Approval
  T2:
    ChangeTime: 2012-11-23 15:12:50
    Config:
      Condition:
        1:
          Fields:
            DynamicField_Status:
              Match: Approval denied
              Type: String
          Type: and
      ConditionLinking: and
    CreateTime: 2012-11-23 11:54:26
    EntityID: T2
    ID: 95
    Name: Approval denied
  T3:
    ChangeTime: 2012-11-23 15:13:29
    Config:
      Condition:
        1:
          Fields:
            DynamicField_Status:
              Match: Approved
              Type: String
          Type: and
      ConditionLinking: and
    CreateTime: 2012-11-23 11:54:54
    EntityID: T3
    ID: 96
    Name: Approved
  T4:
    ChangeTime: 2012-11-23 15:14:08
    Config:
      Condition:
        1:
          Fields:
            DynamicField_Status:
              Match: Order denied
              Type: String
          Type: and
      ConditionLinking: and
    CreateTime: 2012-11-23 11:55:25
    EntityID: T4
    ID: 97
    Name: Order denied
  T5:
    ChangeTime: 2012-11-23 18:30:33
    Config:
      Condition:
        1:
          Fields:
            DynamicField_Status:
              Match: Order placed
              Type: String
          Type: and
      ConditionLinking: and
    CreateTime: 2012-11-23 11:56:15
    EntityID: T5
    ID: 98
    Name: Order placed
  T6:
    ChangeTime: 2012-11-23 15:15:30
    Config:
      Condition:
        1:
          Fields:
            DynamicField_Status:
              Match: Shipment received
              Type: String
          Type: and
      ConditionLinking: and
    CreateTime: 2012-11-23 11:56:48
    EntityID: T6
    ID: 99
    Name: Shipment received
```

2.4. Process configuration reference

2.4.1. Process

A Process models the path of a workflow/process. The waypoints on this path can be Activities or Tranistions, we'll talk about these later.

2.4.1.1. Process configuration

The Process configuration can be done in the file `Kernel/Config.pm` but it is strongly recommended to create new files like `Kernel/Config/Files/MyProcess.pm`. notice that the GUI generates the file `Kernel/Config/File/ZZZProcessManagement` please avoid to use that filename, otherwise it will be overwritten when you sync processes. Let's see an example process configuration (from process cache file):

```
$Self->{'Process'} = {
    'P1' => {
        Name                => 'Book order',
        CreateTime          => '16-02-2012 13:37:00',
        CreateBy            => '1',
        ChangeTime          => '17-02-2012 13:37:00',
        ChangeBy            => '1',
        State               => 'Active',
        StartActivity       => 'A1',
        StartActivityDialog => 'AD1',
        Path => {
            'A1' => {
                'T1' => {
                    ActivityEntityID => 'A2',
                },
            },
            'A2' => {
                'T2' =>  {
                    ActivityEntityID => 'A3',
                },
            },
        },
    },
    'P2' => {
        Name                => 'IT order',
        CreateTime          => '26-02-2012 13:37:00',
        CreateBy            => '1',
        ChangeTime          => '27-02-2012 13:37:00',
        ChangeBy            => '1',
        State               => 'Active',
        StartActivity       => 'A2',
        StartActivityDialog => 'AD2',
        Path => {
            'A2' => {
                'T3' => {
                    ActivityEntityID => 'A4',
                },
            },
        },
    }
};
```

2.4.1.2. Name

The name of the process, this can be selected by the agent when creating a new process ticket.

2.4.1.3. CreateTime

The time when the process was created.

2.4.1.4. CreateBy

The UID of the user creating the process.

2.4.1.5. ChangeTime

The time when the process was changed.

2.4.1.6. ChangeBy

The UID of the user who made the last change to the process.

2.4.1.7. State

Defines the state of a process. Possible values:

- 'Active' are all processes which can be used in new process tickets.
- 'FadeAway' are processes which cannot be selected any more for new tickets, but existing tickets still can use the process.
- 'Inactive' processes are deactivated and cannot be used for new or existing tickets.

2.4.1.8. StartActivity

When creating a new process ticket, a StartActivity must be defined. As soon as the ticket is created, this Activity will be set and used as the base for the first transition checks.

2.4.1.9. StartActivityDialog

For new process tickets, a StartActivityDialog must be defined. This will be shown when creating a new process ticket (after the process was selected). At this point, the ticket does not exist yet, it will be created after submitting the StartActivityDialog.

2.4.1.10. Path

The Path contains the structure of the Activities, and the possible Transitions between them, for the current process. And also the Transition Actions that happens when transitioning . This controls the way that a process ticket can take. Example:

```
'A1' => {
    'T1' => {
        ActivityEntityID => 'A2',
    },
    'T2' => {
        ActivityEntityID => 'A3',
    },
    'T3' => {
        ActivityEntityID => 'A4',
        TransitionAction => ['TA1', 'TA2'],
    },
},
```

If a process ticket is in Activity 'A1', it has three possible ways to get to another Activity. In the Transitions 'T1' to 'T3', conditions are defined, that a process ticket must fulfill to move (transit) to another Activity.

If in this case all the values of the process ticket and its dynamic fields that are needed for the Transition 'T2' are correct, the ticket will be moved from Activity 'A1' to 'A3'. After an ActivityDialog is submitted, or any other change is made to a ticket, it will be checked for possible Transitions from the current Activity. If multiple Transitions are possible, the first one will be used (based on nummerical sorting of the TransitionIDs).

Additionally, it is possible to assign Transition Actions to Transitions in the Path configuration. These are modules which are executed after a successful Transition. They have to be specified in array form as in the example, we'll talk about the details later.

2.4.2. Activity

An Activity contains one or more Activity Dialogs and models a 'step' in the process. All Activity Dialogs of the current Activity are displayed in the ticket zoom and can be used until the conditions of a Transition are fulfilled.

2.4.2.1. Activity configuration

Let's see an example activity configuration:

```
$Self->{'Process::Activity'} =
{
    'A1' => {
        Name       => 'Activity 1 optional',
        CreateTime => '16-02-2012 13:37:00',
        CreateBy   => '1',
        ChangeTime => '17-02-2012 13:37:00',
        ChangeBy   => '1',
        ActivityDialog => {
            1 => 'AD1',
        },
    },
    'A2' => {
        Name       => 'Activity 2 optional',
        CreateTime => '16-02-2012 13:37:00',
        CreateBy   => '1',
        ChangeTime => '17-02-2012 13:37:00',
        ChangeBy   => '1',
        ActivityDialog => {
            1 => 'AD5',
            2 => 'AD6',
            3 => 'AD1',
        },
    },
};
```

2.4.2.2. Name

The name of the activity.

2.4.2.3. CreateTime

The time when it was created.

2.4.2.4. CreateBy

UID of the user who created the Activity.

2.4.2.5. ChangeTime

The last time when it was changed.

2.4.2.6. ChangeBy

UID of the last user who changed the Activity.

2.4.2.7. ActivityDialog

Activity Dialog contains the list of Activity Dialogs which are available in this Activity. All Activity Dialogs of the current Activity are displayed in the ticket zoom. Their order is set by the order in the configuration, here 'AD5' is shown before 'AD6' and 'AD1'.

2.4.3. ActivityDialog

An Activity Dialog is a particular screen and can be used in different Activities.

2.4.3.1. ActivityDialog configuration

Let's see an example config

```
$Self->{'Process::ActivityDialog'} = {
    'AD1' => {
        Name             => 'ActivityDialog 1 optional',
        DescriptionShort => 'Basic info',
        DescriptionLong  => 'Please insert the necessesary basic information for IT orders',
        CreateTime       => '28-02-2012 13:37:00',
        CreateBy         => '1',
        ChangeTime       => '29-02-2012 13:37:00',
        ChangeBy         => '1',
        Fields => {
            PriorityID => {
                DescriptionShort => 'Priority ID',
                DescriptionLong  => 'Enter the priority here',
                Display          => 2,
            },
        },
        FieldOrder       => [ 'PriorityID' ],
        SubmitAdviceText => 'Note: If you submit the form...',
        SubmitButtonText => 'Send request',
    },
    'AD2' => {
        Name => 'ActivityDialog 2 optional',
        DescriptionShort => 'Basic info',
        DescriptionLong  => 'Please insert the necessesary basic information for Book orders',
        CreateTime       => '28-02-2012 13:37:00',
        CreateBy         => '1',
        ChangeTime       => '29-02-2012 13:37:00',
        ChangeBy         => '1',
        Fields => {
            StateID => {
                DescriptionShort => 'State ID',
                DescriptionLong  => 'Enter the state here',
                Display          => 2,
                DefaultValue     => '2',
            },
            Queue => {
                DescriptionShort => 'Queue ID',
                DescriptionLong  => 'Enter the queue here',
                Display          => 2,
                DefaultValue     => 'Raw',
            },
            Title => {
                DescriptionShort => 'Title',
                DescriptionLong  => 'Enter the title here',
                Display          => 1,
                DefaultValue     => 'Default Title',
            },
            DynamicField_Anzahl => {
                DescriptionShort    => 'Amount',
                DescriptionLong     => 'Enter the amount here',
                Display             => 2,
                DefaultValue        => '4',
            },
        },
        FieldOrder       => [ 'DynamicField_Anzahl', 'StateID', 'Queue', 'Title' ],
        SubmitAdviceText => 'Note: If you submit the form...',
        SubmitButtonText => 'Send request',
    },
};
```

2.4.3.2. Name

Name of the Activity Dialog.

2.4.3.3. CreateTime

Time when it was created.

2.4.3.4. CreateBy

UID of the user who created this Activity Dialog.

2.4.3.5. ChangeTime

Last time when it was changed.

2.4.3.6. ChangeBy

UID of the last user who changed this Activity Dialog.

2.4.3.7. Fields

Contains all fields which can be displayed in this Activity Dialog. The following fields can currently be used:

```
    Title
    State
    StateID
    Priority
    PriorityID
    Lock
    LockID
    Queue
    QueueID
    Customer
    CustomerID
    CustomerNo
    CustomerUserID
    Owner
    OwnerID
    Type
    TypeID
    SLA
    SLAID
    Service
    ServiceID
    Responsible
    ResponsibleID
    PendingTime
    DynamicField_$FieldName  # for all dynamic fields
```

Example of a single field configuration:

```
StateID => {
    DescriptionShort => 'State ID',
    DescriptionLong  => 'Enter the state here',
    Display          => 2,
    DefaultValue     => '2',
},
```

The field "Article" is a special case. If it is present in a "Fields" configuration, the Activity Dialog will contain a complete Richtext editor with subject field and attachment handling. The entered text will then be added to the ticket as an article and sent by email. Let's see an example Article field configuration:

```
Article => {
    DescriptionShort => 'Please insert your comment here.',
    DescriptionLong => '',
    Display         => 1,
    Config          => {
        ArticleType  => 'note-internal',
        LabelSubject => '',
        LabelBody    => '',
    },
},
```

Let's look at the field configuration options:

2.4.3.7.1. DescriptionShort

Optional short description that is shown with the field title.

2.4.3.7.2. DescriptionLong

Optional longer field description that is shown then the mouse is over the field, for example advice on how to fill out the field.

2.4.3.7.3. Display

Controls if the field is shown and/or mandatory. Possible values:

- '0': field is invisible. This can be helpful if field values should automatically be set. The configured DefaultValue will be stored in this case.
- '1': field is visible, but optional.
- '2': field is visible and mandatory. The following fields can only be invisible or mandatory:

```
    QueueID
    Queue
    State
    StateID
    Lock
    LockID
    Priority
    PriorityID
    Type
    TypeID
```

If fields are configured as optional, and no value is submitted by the user, the Default Value will be saved when the Activity Dialog is submitted by the user.

2.4.3.7.4. DefaultValue

For fields with 'ID' (like QueueID, OwnerID), this refers to the database ID of the value. For other fields without 'ID' (like Queue, Owner), the DefaultValue must contain the value itself. Example:

```
Queue => {
    DescriptionShort => 'Queue',
    DescriptionLong  => 'Enter the queue here',
    Display          => 2,
    DefaultValue     => 'Raw',
},
```

2.4.3.8. FieldOrder

Here the display order of the fields is configured. IMPORTANT: Invisible fields also must be configured here, because only configured fields will be considered when saving. Fields which are not configured will not be saved.

2.4.3.9. SubmitAdviceText

Optional text to be shown right above the submit button for additional help or advice text.

2.4.3.10. SubmitButtonText

Optional custom text for the submit button.

2.4.4. Transition

A Transition decides - based on configurable conditions - which path in the Process is taken, i. e. to which Activity a Process ticket can be moved.

2.4.4.1. Transition configuration

Let's see an example:

```
$Self->{'Process::Transition'} = {
    'T1' => {
        Name => 'Transition 1',
        CreateTime => '14-03-2012 13:37:00', # optional
        CreateBy   => '1',                   # optional
        ChangeTime => '15-03-2012 13:37:00', # optional
        ChangeBy   => '15-03-2012 13:37:00', # optional
        Condition  => {
            Cond1  => {
                Fields => {
                    StateID => {
                        Type  => 'String',
                        Match => '1',
                    },
                },
            },
        },
    },
    'T2' => {
        Name       => 'Transition 2 optional',
        CreateTime => 'DATE',   # optional
        CreateBy   => 'USERID', # optional
        ChangeTime => 'DATE',   # optional
        ChangeBy => 'USERID',   # optional
        Condition => {
            Cond1 => {
                Queue               => 'Raw',
                DynamicField_Farbe  => '2',
                DynamicField_Anzahl => '1',
            },
        },
    },
};
```

2.4.4.2. Name

Name of the transition.

2.4.4.3. CreateTime

Time when it was created.

2.4.4.4. CreateBy

UID of the user who created this Transition.

2.4.4.5. ChangeTime

Last time when it was changed.

2.4.4.6. ChangeBy

UID of the last user who changed this Transition.

2.4.4.7. Condition

Contains all conditions that are neccessary for this Transition to take effect. Example:

```
Condition => {
    Type  => 'and',
    Cond1 => {
        Type   => 'and',
        Fields => {
            StateID => {
                Type  => 'String',
                Match => '1',
            },
            DynamicField_Marke => {
                Type  => 'String',
                Match => 'VW',
        },
    },
    Cond2 => {
        Type => 'and',
        Fields => {
            Queue => {
                Type  => 'String',
                Match => 'Raw',
            },
        },
    },
},
```

Let's look at the condition configuration in detail.

2.4.4.7.1. Type (Condition)

Specifies the way the different condition elements are connected to each other. Possible values:

- 'and': This is the default. All conditions must be met for the transition to take effect.
- 'or': At least one condition must match.
- 'xor': Exactly one condition must match, not more.

2.4.4.7.2. Cond1

This is the name of an example condition. It can be freely chosen. Conditions are evaluated in sorted order.

2.4.4.7.3. Type (Cond)

Specifies the way how the individual field tests of this condition are connected to each other. Possible values:

- 'and': This is the default. All field tests must match for this condition to match.
- 'or': At least one field test must match.
- 'xor': Exactly one field test must match, not more.

2.4.4.7.4. Fields

Specifies the particular fields whose values should be tested. From our example:

```
Fields => {
    StateID => {
        Type  => 'String',
        Match => '1',
    },
```

2.4.4.7.5. StateID

Example of a field name. The following ticket fields can be used:

```
    Title
    State
    StateID
    Priority
    PriorityID
    Lock
    LockID
    Queue
    QueueID
    Customer
    CustomerID
    CustomerNo
    CustomerUserID
    Owner
    OwnerID
    Type
    TypeID
    SLA
    SLAID
    Service
    ServiceID
    Responsible
    ResponsibleID
    PendingTime
    DynamicField_$FieldName # for all DynamicFields
```

When testing a field with 'ID' (like SLAID), the database ID of the field will be used for testing, for other fields (like SLA) the actual value is used for testing.

2.4.4.7.6. Type

Determines the kind of field testing. Possible values:

- 'String': Compares the field value with the string specified in 'Match'. Matches if they are exactly the same.
- 'Hash': Compares the field value (hash) with the hash specified in 'Match'. All hash values must be the same.
- 'Array': Compares the field value (array) with the array specified in 'Match'. Both lists must be the same.
- 'Regex': The field value can be tested with a regular expression. It is important that 'Match' contains ✁✁✁✁✁✁✁ as a base condition. Between the braces the actual regular expression can be noted.

- 'Module': Allows you to use a perl module for condition checking. If it returns 1, the check was positive. You can find an example module in `Kernel/System/ProcessManagement/TransitionValidation/ValidateDemo.pm`.

2.4.5. Transition Actions

Transition Actinons are actions which can be triggered after successfully applied transitions (when a process ticket moves from one activity to another). These Transition Actions can be used to perform different changes on the ticket, e. g. change the Queue or the Owner of the ticket, and you can also create your own Transition Actions to perform other complex changes.

2.4.5.1. Transition Action configuration

Let's see an example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Queue Move',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet',
        Config => {
            Queue  => 'Junk',
            UserID => 123,
        },
    },
};
```

2.4.5.2. Name

The name of the Transition Action.

2.4.5.3. Module

Specifies the Perl module to be used.

2.4.5.4. Config

This parameter contains all settings which are required for the module. Its content depends on the particular Transition Action module which is used. Please see the documentation of the individual modules for details. In our example, only the Queue must be specified, Nevertheless we are also sending UserID parameter, by using the UserID parameter, the transition action will be executed impersonating the user with the given UserID.

The use of UserID inside the "Config" parameter of a Transition Action is accepted by all Transition Actions (since OTRS 3.2.4), in this example it could be particularly important if the user that triggers the Transition does not have permissions to move the ticket to the queue 'Junk', while the user with the UserID 123 might have.

2.4.5.5. Reusing Transition Action modules

To use Transition Action modules multiple times, just specify several TransitionActions in your configuration. Example:

```
    $Self->{'Process::TransitionAction'} = {
        'TA1' => {
            Name   => 'Queue Move Junk',
            Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet',
            Config => {
                Queue => 'Junk',
            },
        },
        'TA2' => {
            Name   => 'Queue Move Raw',
            Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet',
            Config => {
                Queue => 'Raw',
            },
        },
    };
```

Here the same module is used to move a process ticket into the 'Raw' queue, and another time to move it into the junk queue. The Transition Action which must be used for a particular Transition is determined from the 'Path' setting of the Process configuration.

2.4.5.6. Available Transition Actions

OTRS comes with several Transition Actions that can be used in your processes. Here you can find their documentation and how they need to be configured.

2.4.5.6.1. DynamicFieldSet

Sets one or more dynamic fields at a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Set DynamicField Master to Master and Approved to 1',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::DynamicFieldSet',
        Config => {
            MasterSlave => 'Master',
            Approved    => '1',

        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'MasterSlave' and 'Approved' are given as examples of DynamicField names. The values of the fields ('Master' and '1') will be set by this TransitionAction.

2.4.5.6.2. TicketArticleCreate

Creates an article and can be used to create notes or email replies. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Article Create Note Internal',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketArticleCreate',
        Config => {
            ArticleType      => 'note-internal',                                        # note-
external|phone|fax|sms|...
                                                                                        #
 excluding any email type
            SenderType       => 'agent',                                                #
 agent|system|customer
            ContentType      => 'text/plain; charset=ISO-8859-15',                      # or
 optional Charset & MimeType
```

```
            Subject          => 'some short description',                               #
 required
            Body             => 'the message text',                                     #
 required
            HistoryType      => 'OwnerUpdate',                                          #
 EmailCustomer|Move|AddNote|PriorityUpdate|WebRequestCustomer|...
            HistoryComment   => 'Some free text!',
            From             => 'Some Agent <email@example.com>',                       # not
 required but useful
            To               => 'Some Customer A <customer-a@example.com>',             # not
 required but useful
            Cc               => 'Some Customer B <customer-b@example.com>',             # not
 required but useful
            ReplyTo          => 'Some Customer B <customer-b@example.com>',             # not
 required
            InReplyTo        => '<asdasdasd.12@example.com>',                           # not
 required but useful
            References       => '<asdasdasd.1@example.com> <asdasdasd.12@example.com>', # not
 required but useful
            NoAgentNotify    => 0,                                                      # if
 you don't want to send agent notifications
            AutoResponseType => 'auto reply',                                           # auto
 reject|auto follow up|auto reply/new ticket|auto remove

            ForceNotificationToUserID   => [ 1, 43, 56 ],                               # if
 you want to force somebody
            ExcludeNotificationToUserID => [ 43, 56 ],
                 # if you want full exclude somebody from notfications,
                 # will also be removed in To: line of article,
                 # higher prio as ForceNotificationToUserID
            ExcludeMuteNotificationToUserID => [ 43, 56 ],
                 # the same as ExcludeNotificationToUserID but only the
                 # sending gets muted, agent will still shown in To:
                 # line of article
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction. It can be freely chosen, but should reflect the purpose of the configured action.

'ArticleType' defines the type of the article to be created. Possible values: phone, fax, sms, webrequest, note-internal, note-external and note-report.

SenderType defines the sender type of the article. Possible values: agent, system, customer.

'ContentType' defines the content type of the article. Possible values: 'text/plain; charset=ISO-8859-15' or any other valid charset and mime type.

'Subject' defines the article title. Mandatory.

'Body' defines the article content. Mandatory.

HistoryType defines the type of the history entry. Possible values: AddNote, ArchiveFlagUpdate, Bounce, CustomerUpdate, EmailAgent, EmailCustomer, EscalationResponseTimeNotifyBefore, EscalationResponseTimeStart, EscalationResponseTimeStop, EscalationSolutionTimeNotifyBefore, EscalationSolutionTimeStart, EscalationSolutionTimeStop, EscalationUpdateTimeNotifyBefore, EscalationUpdateTimeStart, EscalationUpdateTimeStop, FollowUp, Forward, Lock, LoopProtection, Merged, Misc, Move, NewTicket, OwnerUpdate, PhoneCallAgent, PhoneCallCustomer, PriorityUpdate, Remove, ResponsibleUpdate, SendAgentNotification, SendAnswer, SendAutoFollowUp, SendAutoReject, SendAutoReply, SendCustomerNotification, ServiceUpdate, SetPendingTime, SLAUpdate, StateUpdate, Subscribe, SystemRequest, TicketDynamicFieldUpdate, TicketLinkAdd, TicketLinkDelete, TimeAccounting, TypeUpdate, Unlock, Unsubscribe, WebRequestCustomer.

'HistoryComment' defines the content of the history entry.

'From', 'To', 'Cc' and 'ReplyTo' take email addresses in the notation specified above.

'InReplyTo' and 'References' take email message IDs.

'NoAgentNotify' - if set to 1, the email notification of the Agent will not be sent.

'AutoResponseType' can take the following values: auto follow up, auto reject, auto remove, auto reply, auto reply/new ticket.

'ForceNotificationToUserID', 'ExcludeNotificationToUserID', 'ExcludeMuteNotificationToUserID' can take a list of UserIDs that are either always notified, not notified or listed as notified but not actually sent a notification email.

2.4.5.6.3. TicketCustomerSet

Sets the customer of a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Customer Set Customer to test',
        Module => 'Kernel::System::Process::TransitionAction::TicketCustomerSet',
        Config => {
            No      => 'test',
            User    => 'client-user-123',
            # or in other words
            # CustomerID     => 'client123',
            # CustomerUserID => 'client-user-123',

        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

No or CustomerID set the Customer ID of the customer.

User or CustomerUserID set the Username of the customer.

2.4.5.6.4. TicketLockSet

Changes the lock of a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Set Lock to lock',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketLockSet',
        Config => {
            Lock   => 'lock',
            # or
            LockID => 2,
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'Lock' defines the new lock of the process ticket.

'LockID' defines the internal ID of the new lock.

2.4.5.6.5. TicketOwnerSet

Changes the owner of a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Owner Set root@localhost',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketOwnerSet',
        Config => {
            Owner => 'root@localhost',
            # or
            OwnerID => 1,
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'Owner' specifies the login name of the new owner.

'OwnerID' specifies the internal ID of the new owner.

2.4.5.6.6. TicketQueueSet

Moves the ticket into a target queue. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Queue Move Raw',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketQueueSet',
        Config => {
            Queue => 'Raw',
            # or
            # QueueID => '2',
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'Queue' specifies the name of the target queue.

'QueueID' specifies the internal ID of the target queue.

2.4.5.6.7. TicketResponsibleSet

Changes the responsible of a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Responsible Set root@localhost',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketResponsibleSet',
        Config => {
            Responsible => 'root@localhost',
            # or
            ResponsibleID => 1,
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'Responsible' specifies the login name of the new responsible.

'ResponsibleID' specifies the internal ID of the new responsible.

2.4.5.6.8. TicketServiceSet

Assigns a service to a process ticket. The ticket requires to have a customer and the service must be assigned to that customer. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Set MyService service',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketServiceSet',
        Config => {
            Service   => 'MyService',
            # or
            ServiceID => 123,
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'Service' defines the new service of the process ticket. The full name is required (e.g. GramdFatherService::FatherService::SonService).

'ServiceID' defines the internal ID of the new service.

2.4.5.6.9. TicketSLASet

Assigns a service level agreement to a process ticket. The ticket requires to have a service and the SLA must be assigned to that service. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Set MySLA SLA',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketSLASet',
        Config => {
            SLA   => 'MyService',
            # or
            SLAID => 123,
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'SLA' defines the new service level agreement of the process ticket.

'SLAID' defines the internal ID of the new SLA.

2.4.5.6.10. TicketStateSet

Changes the state of a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Set State to open',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketStateSet',
        Config => {
            State   => 'open',
            # or
            StateID => 4,

            PendingTimeDiff => 123,
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'State' defines the new state of the process ticket.

'StateID' defines the internal ID of the new state.

'PendingTimeDiff' used only for pending type states, defines the time difference in seconds relative (relative to the Transition Action execution time) to set ticket pending time (e.g. 3600 means that the pending time is 1hr after the Transition Action is executed).

2.4.5.6.11. TicketTitleSet

Sets the ticket title of a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Set Ticket Title to Ticket-title',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketTitleSet',
        Config => {
            Title => 'Ticket-title',
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'Title' specifies the new title of the ticket.

2.4.5.6.12. TicketTypeSet

Sets the ticket type of a process ticket. Example:

```
$Self->{'Process::TransitionAction'} = {
    'TA1' => {
        Name   => 'Set Ticket Type to default',
        Module => 'Kernel::System::ProcessManagement::TransitionAction::TicketTypeSet',
        Config => {
            Type     => 'default',
            # or
            # TypeID => '1',
        },
    },
};
```

'Name' specifies the name of the configured TransitionAction.

'Type' specifies the name of the ticket type.

'TypeID' specifies the internal ID of the ticket type.

2.4.6. Access Control Lists (ACLs)

With the help of ACLs, you can limit selectable values in process tickets. Please also see the ACL reference for a description of the full ticket ACL syntax.

2.4.6.1. ACL configuration

ACLs can only be defined in `Kernel/Config.pm`. Example:

```
$Self->{TicketAcl}->{'001-ACL-ProcessProperties'} = {
    Properties => {
        Process => {
            ProcessEntityID        => ['P1'],
            ActivityEntityID       => ['A1'],
            ActivityDialogEntityID => ['AD1'],
        }
    },
    Possible => {
        ActivityDialog => ['AD1', 'AD3'],
    },
    PossibleNot => {
        ActivityDialog => ['AD3'],
    },
};
```

2.4.6.2. 001-ACL-ProcessProperties

Name of the ACL rule. For further information on ACL rules in general, please consult: the manual .

2.4.6.3. Process

This is the section that is used to check if an ACL must be applied. If it has the specified values, the rule is applied. The following values can be used:

2.4.6.3.1. ProcessEntityID

The ID of a process that the process. Matches if the ticket is assigned to this process.

2.4.6.3.2. ActivityEntityID

The ID of the Activity that the process ticket currently is assigned to.

2.4.6.3.3. ActivityDialogEntityID

The ID of the Activity Dialog that is currently open for a process ticket.

2.4.6.4. Possible/PossibleNot Activity Dialog

Here you can specify a list of Activity Dialog IDs. This list will limit the possible Activity Dialogs that are offered to the user in the ticket zoom mask.

'Possible' lists the Activity Dialogs that are allowed. The setting above will only allow 'AD1' and 'AD3' of the list of configured Activity Dialogs.

'PossibleNot' lists the Activity Dialogs that are not allowed. In the example above, the setting will remove 'AD3' from the list of configured Activity Dialogs.

If both 'Possible' and 'PossibleNot' are specified, the list of configured Activity Dialogs will first be filtered by 'Possible', leaving only 'AD1' and 'AD3' in our example. Then 'PossibleNot' will be applied and filter out 'AD3', so that only 'AD1' remains and is shown as a possible Activity Dialog that the user can use.

If multiple ACL rules match, the intersection of all matching rules will be calculated to determine the possible Activity Dialogs. Example:

Configured Activity Dialogs: 'AD1', 'AD2', 'AD3', 'AD4', 'AD5', 'AD6', 'AD7'.

```
$Self->{TicketAcl}->{'001-ACL-Status'} = {
    Properties => {
        Ticket => {
            Status => 'new',
        }
    },
    Possible => {
        ActivityDialog => ['AD1', 'AD2', 'AD3', 'AD6', 'AD7'],
    },
};
$Self->{TicketAcl}->{'002-ACL-Queue'} = {
    Properties => {
        Ticket => {
            Queue => ['Raw']
        }
    },
    Possible => {
        ActivityDialog => ['AD2', 'AD3', 'AD4', 'AD7'],
    },
};
$Self->{TicketAcl}->{'003-ACL-Priority'} = {
    Properties => {
        Ticket => {
            Priority => ['3 normal']
        }
    },
    PossibleNot => {
        ActivityDialog => ['AD3', 'AD4'],
    },
};
```

If a process ticket has the state 'new', is in the 'Raw' queue and has a priority '3 normal', then all three ACL rules will match.

The first rule reduces the Activity Dialogs from 'AD1', 'AD2', 'AD3', 'AD4', 'AD5', 'AD6', 'AD7' to 'AD1', 'AD2', 'AD3', 'AD6', 'AD7' and forbids 'AD4' and 'AD5'.

The second rule will now further reduce the remaining Activity Dialogs. In our example, 'AD2', 'AD3', 'AD7' will remain.

Now the third rule will further reduce the list by 'PossibleNot'. 'AD3' is removed from the list. 'AD4' is not removed, since it was not on the list in the first place. At the end, 'AD2' and 'AD7' remain as possible Activity Dialogs that the user can utilize.

3. Creating your own themes

You can create your own themes so as to use the layout you like in the OTRS web frontend. To create own themes, you should customize the output templates to your needs.

More information on the syntax and structure of output templates can be found in the Developer Manual at ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂, especialy in the chapter on ✂✂✂✂✂✂✂✂✂.

As an example, perform the following steps to create a new theme called "Company":

1. Create a directory called `Kernel/Output/HTML/Company` and copy all files that you like to change, from `Kernel/Output/HTML/Standard` into the new folder.

   ### Important

   Only copy over the files you actually change. OTRS will automatically get the missing files from the Standard theme. This will make upgrading at a later stage much easier.

2. Customize the files in the directory `Kernel/Output/HTML/Company`, and change the layout to your needs.

3. To activate the new theme, add them in SysConfig under Frontend::Themes.

Now the new theme should be useable. You can select it via your personal preferences page.

Warning

Do not change the theme files shipped with OTRS, since these changes will be lost after an update. Create your own themes only by performing the steps described above.

4. Localization of the OTRS frontend

OTRS offers multi-language support for its web interface.

Procedures for localization for the OTRS framework, steps to be followed to create a new language translation, as well as procedures for translation customizations, can be found in the "Language Translations" chapter from the developer manual on http://doc.otrs.org.

Chapter 6. Performance Tuning

Presented below is a list of performance enhancing techniques for your OTRS installation, including configuration, coding, memory use, and more.

1. OTRS

There are several options for improving OTRS performance.

1.1. TicketIndexModule

There are two backend modules for the ticket index:

- Using Kernel::System::Ticket::IndexAccelerator::RuntimeDB (default), generate each queue view on the fly from the ticket table. You will not have performance trouble until you have about 60,000 open tickets in your system.

- Kernel::System::Ticket::IndexAccelerator::StaticDB, the most powerful module, should be used when you have above 80,000 open tickets. It uses an extra ticket_index table, which works like a view. Use `bin/otrs.RebuildTicketIndex.pl` for generating an initial index build after switching backends.

You can change the IndexAccelerator via SysConfig.

1.2. TicketStorageModule

There are two different backend modules for the ticket/article storage:

- Configure Kernel::System::Ticket::ArticleStorageDB (default) to store attachments, etc., in the database. Note: Don't use it with large set ups.

  Pro: If your webserver user isn't the 'otrs' user, use this module to avoid file permission problems.

  Con: It is not advisable to store attachments in your database. Take care that your database is able to store large objects. E.g. Configure MySQL with "set-variable = max_allowed_packet=8M" to store 8 MB objects (the default is 2M).

- Configure Kernel::System::Ticket::ArticleStorageFS to store attachments etc. on the local file system. Note: Recommended for large setups.

  Pro: It is fast!

  Con: Your web server user should be the 'otrs' user. Also, if you have multiple front-end servers, you should make sure the filesystem is shared between the servers. Place it on an NFS share or preferably a SAN or similar solution.

Note: you can switch from one back-end to the other on the fly. You can switch the backend in the SysConfig, and then run the command line utility `otrs.ArticleStorageSwitch.pl` to put the articles from the database onto the filesystem or the other way around. You can use the ✃✃ and ✃✃ options to specify the source and destination back-ends. Please note that the entire process can take considerable time to run, depending on the number of articles you have and the available CPU power and/or network capacity.

```
shell> bin/otrs.ArticleStorageSwitch.pl -s ArticleStorageDB -d ArticleStorageFS
```

✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂ ✂✂✂✂✂✂✂✂✂ ✂✂✂✂ ✂✂✂✂✂✂✂✂ ✂✂
✂✂✂✂✂✂✂✂✂✂✂

1.3. Archiving Tickets

As OTRS can be used as an audit-proof system, deleting closed tickets may not be a good idea. Therefore we implemented a feature that allows you to archive tickets.

Tickets that match certain criteria can be marked as "archived" These tickets are not accessed if you do a regular ticket search or run a Generic Agent job. The system itself does not have to deal with a huge amount of tickets any longer as only the "latest" tickets are taken into consideration when using OTRS. This can result in a huge performance gain on large systems.

To use the archive feature simply follow these steps:

1. Activate the archive system in SysConfig

   In the Admin page, go to SysConfig and select the group `Ticket`. In `Core::Ticket` you find the option `Ticket::ArchiveSystem` which is set to "no" by default. Change this setting to "yes" and save this change.

2. Define a GenericAgent job

   On the Admin page, select GenericAgent and add a new job there.

   a. Job Settings

      Provide a name for the archiving job, and select proper options to schedule this job.

   b. Ticket Filter

      The ticket filter is searches for tickets that match the selected criteria. It might be a good idea to only archive those tickets in a closed state that have been closed a few months before.

   c. Ticket Action

      In this section, set the field labeled "Archive selected tickets" to "archive tickets".

   d. Save the job

      At the end of the page you will find an option to save the job.

   e. Affected tickets

      The system will display all tickets which will be archived when executing the Generic Agent job.

3. Ticket Search

   When you search for tickets, the system default is to search tickets which are not archived. If you want to search through archived tickets also, simply add "archive search" while defining search criteria.

2. Database

DB issues vary by the database being used. Study the documentation for your database or check with your database administrator.

2.1. MySQL

If you use the MySQL table type MyISAM (which is the default), and have deleted a large part of a table or if you have made many changes to a table with variable-length rows (tables that have VARCHAR, BLOB or TEXT columns), you must defragment the datafile (tables) with the "optimize" command.

You should try this if the mysqld daemon needs a lot of your CPU time. Optimize the tables - ticket, ticket_history and article (see Script below).

```
shell$ mysql -u user -p database
mysql$ optimize table ticket;
mysql$ optimize table ticket_history;
mysql$ optimize table article;
```

✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁✁

2.2. PostgreSQL

PostgreSQL is best tuned by modifying the postgresql.conf file in your PostgreSQL data directory. For advice on how to do this, reference the following articles:

- ✁✁ ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁
- ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁
- ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁

If performance is still not satisfactory, we suggest that you join the PostgreSQL Performance mailing list (http://www.postgresql.org/community/lists/), and ask questions there. The folks on the PostgreSQL list are very friendly and can probably help.

3. Webserver

Of course you should use mod_perl 2.0 (✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁). It's much faster (~ * 100) than pure cgi. But it needs more RAM.

3.1. Pre-established database connections

You can have the database connections pre-established on startup of the web server. This saves time (see README.webserver).

3.2. Preloaded modules - startup.pl

Use the startup script `scripts/apache2-perl-startup.pl` for preloaded/precompiled Perl modules on your mod_perl webserver to be faster, with a smaller memory footprint (see README.webserver).

3.3. Reload Perl modules when updated on disk

By default Apache::Reload is used in scripts/apache2-httpd.include.conf. Disable it and you will get 8% more speed. But remember to restart the web server if you install any modules via the OTRS Package Manager, or any values in your SysConfig or in Kernel/Config.pm. Important: this would also mean you can't use the OTRS Package Manager via the web interface, you need to use the command line variant - `bin/otrs.PackageManager.pl`.

3.4. Choosing the Right Strategy

If you have a larger installation, e.g. over 1,000 new tickets per day and over 40 agents, it is a good idea to read the chapters on Performance of the mod_perl User's Guide (✂✂✂✂✂✂✂ ✂✂).

3.5. mod_gzip/mod_deflate

If your bandwidth is small, use mod_deflate for Apache2. If you have an html page with 45k, mod_gzip/mod_deflate compresses it to about 7k. The drawback is that this increases the load on the server side.

Appendix A. Additional Resources

We try to support you with the very latest information about OTRS. We also give you an opportunity to provide us with your feedback.

1. Website OTRS Group

You can find the website of the OTRS Group, the company behind OTRS, at ✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂✂. It contains a lot of product-related information, such as white papers, release notes, case studies and so on.

2. Mailing lists

The Table A-1 displays our various community mailing lists.

Table A.1. Mailinglists

Name	Description	Homepage
announce@otrs.org	Low traffic list, in English, for announcements of new OTRS releases and security issues.	✂✂✂
otrs@otrs.org	Medium to high traffic list, in English, where you can find all sorts of relevant questions and support for the product.	✂✂✂
otrs-de@otrs.org	Medium to high traffic list, in German, where you can find all sorts of relevant questions and support for the product.	✂✂
dev@otrs.org	Medium traffic list, in English, where the OTRS developers discuss various design and implementation issues.	✂✂
doc-de@otrs.org	Low traffic list, in German, with all sorts of questions on the documentation of the product.	✂✂✂
i18n@otrs.org	Low traffic list, in English, for internationalization and localization questions. If you are or want to become a translator of the OTRS project or have any problems with one of our applications in an international environment, this is where you should connect.	✂✂✂

Name	Description	Homepage
cvs-log@otrs.org	Very high traffic list of CVS commit notifications.	✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁✁✁ ✁✁✁

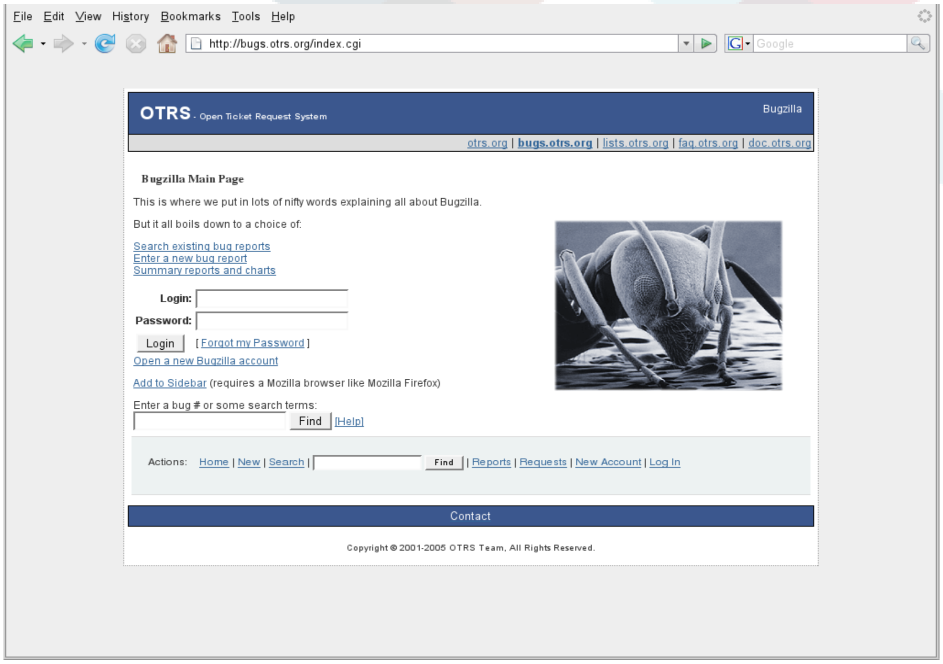
To subscribe to any of these lists, visit the following link: ✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁.

3. User Forums

You can find community user forums at ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁. It allows you to get in contact with users all around the world and exchange experiences regarding the use of OTRS.

4. Bug tracking

To submit bugs visit ✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁✁ (see Figure below). Please take note of the difference between a bug and a configuration issue. Configuration issues are problems that you encounter when setting a system, or general questions regarding the use of OTRS. Bug reports should only be used for issues with the source code of OTRS itself, or to file enhancements for OTRS. All your bug reports and enhancement requests are very welcome in the bug tracker.


File Edit View History Bookmarks Tools Help
http://bugs.otrs.org/index.cgi
Google
OTRS - Open Ticket Request System
Bugzilla
otrs.org | bugs.otrs.org | lists.otrs.org | faq.otrs.org | doc.otrs.org

Bugzilla Main Page

This is where we put in lots of nifty words explaining all about Bugzilla.

But it all boils down to a choice of:

Search existing bug reports
Enter a new bug report
Summary reports and charts

Login:
Password:

Login
[Forgot my Password]
Open a new Bugzilla account

Add to Sidebar (requires a Mozilla browser like Mozilla Firefox)

Enter a bug # or some search terms:

Find
[Help]
Actions: Home | New | Search | Find | Reports | Requests | New Account | Log In
Contact

Copyright © 2001-2005 OTRS Team, All Rights Reserved.

✁✁✁✁✁✁✁ ✁✁✁ ✁✁✁✁✁✁✁✁

For configuration issues, you should either use the ✁✁✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁ ✁✁✁✁✁✁✁✁✁ ✁✁✁✁ ✁✁✁✁✁✁✁✁, or the public mailing lists.

You help us improve the product by reporting bugs. We appreciate your input!

5. Commercial Support

For services (support, consulting, development, and training) you can contact the company behind OTRS, OTRS AG. Our offices are located in Germany, USA, Mexico, the Netherlands, and in other countries. Visit our website for contact information: ✂✂

Appendix B. Configuration Options Reference

1. DynamicFields

1.1. DynamicFields::Backend::Registration

1.1.1. DynamicFields::Backend###Text

Description:	DynamicField backend registration.
Group:	DynamicFields
SubGroup:	DynamicFields::Backend::Registration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DynamicFields::Backend'}->{'Text'} =  {
  'ConfigDialog' => 'AdminDynamicFieldText',
  'DisplayName' => 'Text',
  'Module' => 'Kernel::System::DynamicField::Backend::Text'
};
```

1.1.2. DynamicFields::Backend###TextArea

Description:	DynamicField backend registration.
Group:	DynamicFields
SubGroup:	DynamicFields::Backend::Registration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DynamicFields::Backend'}->{'TextArea'} =  {
  'ConfigDialog' => 'AdminDynamicFieldText',
  'DisplayName' => 'Textarea',
  'Module' => 'Kernel::System::DynamicField::Backend::TextArea'
};
```

1.1.3. DynamicFields::Backend###Checkbox

Description:	DynamicField backend registration.
Group:	DynamicFields
SubGroup:	DynamicFields::Backend::Registration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DynamicFields::Backend'}->{'Checkbox'} =  {
  'ConfigDialog' => 'AdminDynamicFieldCheckbox',
  'DisplayName' => 'Checkbox',
  'Module' => 'Kernel::System::DynamicField::Backend::Checkbox'
};
```

1.1.4. DynamicFields::Backend###Dropdown

Description:	DynamicField backend registration.
Group:	DynamicFields
SubGroup:	DynamicFields::Backend::Registration
Valid:	1

Required:	0
Config-Setting:	

```
 $Self->{'DynamicFields::Backend'}->{'Dropdown'} =  {
  'ConfigDialog' => 'AdminDynamicFieldDropdown',
  'DisplayName' => 'Dropdown',
  'Module' => 'Kernel::System::DynamicField::Backend::Dropdown'
};
```

1.1.5. DynamicFields::Backend###DateTime

Description:	DynamicField backend registration.
Group:	DynamicFields
SubGroup:	DynamicFields::Backend::Registration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DynamicFields::Backend'}->{'DateTime'} =  {
  'ConfigDialog' => 'AdminDynamicFieldDateTime',
  'DisplayName' => 'Date / Time',
  'Module' => 'Kernel::System::DynamicField::Backend::DateTime'
};
```

1.1.6. DynamicFields::Backend###Date

Description:	DynamicField backend registration.
Group:	DynamicFields
SubGroup:	DynamicFields::Backend::Registration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DynamicFields::Backend'}->{'Date'} =  {
  'ConfigDialog' => 'AdminDynamicFieldDateTime',
  'DisplayName' => 'Date',
  'Module' => 'Kernel::System::DynamicField::Backend::Date'
};
```

1.1.7. DynamicFields::Backend###Multiselect

Description:	DynamicField backend registration.
Group:	DynamicFields
SubGroup:	DynamicFields::Backend::Registration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DynamicFields::Backend'}->{'Multiselect'} =  {
  'ConfigDialog' => 'AdminDynamicFieldMultiselect',
  'DisplayName' => 'Multiselect',
  'ItemSeparator' => ', ',
  'Module' => 'Kernel::System::DynamicField::Backend::Multiselect'
};
```

1.2. DynamicFields::ObjectType::Registration

1.2.1. DynamicFields::ObjectType###Ticket

Description:	DynamicField object registration.

Group:	DynamicFields
SubGroup:	DynamicFields::ObjectType::Registration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'DynamicFields::ObjectType'}->{'Ticket'} =  {
  'DisplayName' => 'Ticket',
  'Module' => 'Kernel::System::DynamicField::ObjectType::Ticket'
};
```

1.2.2. DynamicFields::ObjectType###Article

Description:	DynamicField object registration.
Group:	DynamicFields
SubGroup:	DynamicFields::ObjectType::Registration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'DynamicFields::ObjectType'}->{'Article'} =  {
  'DisplayName' => 'Article',
  'Module' => 'Kernel::System::DynamicField::ObjectType::Article'
};
```

1.3. Frontend::Admin::ModuleRegistration

1.3.1. Frontend::Module###AdminDynamicField

Description:	Frontend module registration for the agent interface.
Group:	DynamicFields
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminDynamicField'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.DynamicField.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.DynamicField.js'
    ]
  },
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Create and manage dynamic fields.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Dynamic Fields',
    'Prio' => '1000'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Dynamic Fields GUI'
};
```

1.3.2. Frontend::Module###AdminDynamicFieldText

Description:	Frontend module registration for the agent interface.
Group:	DynamicFields
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminDynamicFieldText'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.Admin.DynamicField.js'
    ]
  },
  'Title' => 'Dynamic Fields Text Backend GUI'
};
```

1.3.3. Frontend::Module###AdminDynamicFieldCheckbox

Description:	Frontend module registration for the agent interface.
Group:	DynamicFields
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminDynamicFieldCheckbox'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.Admin.DynamicField.js'
    ]
  },
  'Title' => 'Dynamic Fields Checkbox Backend GUI'
};
```

1.3.4. Frontend::Module###AdminDynamicFieldDropdown

Description:	Frontend module registration for the agent interface.
Group:	DynamicFields
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminDynamicFieldDropdown'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.DynamicField.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.DynamicField.js',
      'Core.Agent.Admin.DynamicFieldDropdown.js'
    ]
  },
  'Title' => 'Dynamic Fields Drop-down Backend GUI'
};
```

1.3.5. Frontend::Module###AdminDynamicFieldDateTime

Description:	Frontend module registration for the agent interface.
Group:	DynamicFields
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminDynamicFieldDateTime'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.DynamicField.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.DynamicField.js',
      'Core.Agent.Admin.DynamicFieldDateTime.js'
    ]
  },
  'Title' => 'Dynamic Fields Date Time Backend GUI'
};
```

1.3.6. Frontend::Module###AdminDynamicFieldMultiselect

Description:	Frontend module registration for the agent interface.
Group:	DynamicFields
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminDynamicFieldMultiselect'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.DynamicField.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.DynamicField.js',
      'Core.Agent.Admin.DynamicFieldMultiselect.js'
    ]
  },
  'Title' => 'Dynamic Fields Multiselect Backend GUI'
};
```

1.4. Frontend::Agent::Preferences

1.4.1. PreferencesGroups###DynamicField

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	DynamicFields
SubGroup:	Frontend::Agent::Preferences
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'PreferencesGroups'}->{'DynamicField'} =  {
  'Active' => '1',
  'Block' => 'Input',
  'Column' => 'Other Settings',
  'Data' => '$Env{"UserDynamicField_NameX"}',
  'Key' => 'Default value for NameX',
  'Label' => 'NameX',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserDynamicField_NameX',
  'Prio' => '7000'
};
```

1.4.2. PreferencesGroups###DynamicFieldsOverviewPageShown

Description:	Parameters for the pages (in which the dynamic fields are shown) of the dynamic fields overview.
Group:	DynamicFields
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'PreferencesGroups'}->{'DynamicFieldsOverviewPageShown'} =  {
  'Active' => '0',
  'Column' => 'Other Settings',
  'Data' => {
    '10' => '10',
    '15' => '15',
    '20' => '20',
    '25' => '25',
    '30' => '30',
    '35' => '35'
  },
  'DataSelected' => '25',
  'Key' => 'Dynamic fields limit per page for Dynamic Fields
 Overview',
  'Label' => 'Dynamic Fields Overview Limit',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'AdminDynamicFieldsOverviewPageShown',
  'Prio' => '8000'
};
```

2. Framework

2.1. Core

2.1.1. SecureMode

Description:	Disables the web installer (http://yourhost.example.com/otrs/installer.pl), to prevent the system from being hijacked. If set to "No",

	the system can be reinstalled and the current basic configuration will be used to pre-populate the questions within the installer script. If not active, it also disables the GenericAgent, PackageManager and SQL Box.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SecureMode'} =  '0';`

2.1.2. Frontend::DebugMode

Description:	Enables or disable the debug mode over frontend interface.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Frontend::DebugMode'} =  '0';`

2.1.3. ConfigLevel

Description:	Sets the configuration level of the administrator. Depending on the config level, some sysconfig options will be not shown. The config levels are in in ascending order: Expert, Advanced, Beginner. The higher the config level is (e.g. Beginner is the highest), the less likely is it that the user can accidentally configure the system in a way that it is not usable any more.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'ConfigLevel'} =  '100';`

2.1.4. ProductName

Description:	Defines the name of the application, shown in the web interface, tabs and title bar of the web browser.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'ProductName'} =  'OTRS';`

2.1.5. SystemID

Description:	Defines the system identifier. Every ticket number and http session string contain this ID. This ensures that only tickets which belong

	to your system will be processed as follow-ups (useful when communicating between two instances of OTRS).
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SystemID'} =  '10';`

2.1.6. FQDN

Description:	Defines the fully qualified domain name of the system. This setting is used as a variable, OTRS_CONFIG_FQDN which is found in all forms of messaging used by the application, to build links to the tickets within your system.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'FQDN'} =  'yourhost.example.com';`

2.1.7. HttpType

Description:	Defines the type of protocol, used by ther web server, to serve the application. If https protocol will be used instead of plain http, it must be specified it here. Since this has no affect on the web server's settings or behavior, it will not change the method of access to the application and, if it is wrong, it will not prevent you from logging into the application. This setting is used as a variable, OTRS_CONFIG_HttpType which is found in all forms of messaging used by the application, to build links to the tickets within your system.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'HttpType'} =  'http';`

2.1.8. ScriptAlias

Description:	Sets the prefix to the scripts folder on the server, as configured on the web server. This setting is used as a variable, OTRS_CONFIG_ScriptAlias which is found in all forms of messaging used by the application, to build links to the tickets within the system.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1

Config-Setting:	`$Self->{'ScriptAlias'} =  'otrs/';`

2.1.9. AdminEmail

Description:	Defines the system administrator's email address. It will be displayed in the error screens of the application.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'AdminEmail'} =  'admin@example.com';`

2.1.10. Organization

Description:	Company name for the customer web interface. Will also be included in emails as an X-Header.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Organization'} =  'Example Company';`

2.1.11. DefaultLanguage

Description:	Defines the default front-end language. All the possible values are determined by the available language files on the system (see the next setting).
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'DefaultLanguage'} =  'en';`

2.1.12. DefaultUsedLanguages

Description:	Defines all the languages that are available to the application. The Key/Content pair links the front-end display name to the appropriate language PM file. The "Key" value should be the base-name of the PM file (i.e. de.pm is the file, then de is the "Key" value). The "Content" value should be the display name for the front-end. Specify any own-defined language here (see the developer documentation http://doc.otrs.org/ for more infomation). Please remember to use the HTML equivalents for non-ASCII characters (i.e. for the German oe = o umlaut, it is necessary to use the &ouml; symbol).
Group:	Framework
SubGroup:	Core
Valid:	1

Required:	1

Config-Setting:

```
 $Self->{'DefaultUsedLanguages'} =  {
  'ar_SA' => 'Arabic (Saudi Arabia)',
  'bg' => 'Bulgarian
 (&#x0411;&#x044a;&#x043b;&#x0433;&#x0430;&#x0440;&#x0441;&#x043a;&#x0438;)',
  'ca' => 'Catal&agrave;',
  'cs' => 'Czech (&#x010c;esky)',
  'da' => 'Dansk',
  'de' => 'Deutsch',
  'el' => 'Greek
 (&#x0395;&#x03bb;&#x03bb;&#x03b7;&#x03bd;&#x03b9;&#x03ba;&#x03ac;)',
  'en' => 'English (United States)',
  'en_CA' => 'English (Canada)',
  'en_GB' => 'English (United Kingdom)',
  'es' => 'Espa&ntilde;ol',
  'es_CO' => 'Espa&ntilde;ol (Colombia)',
  'es_MX' => 'Espa&ntilde;ol (M&eacute;xico)',
  'et' => 'Eesti',
  'fa' => 'Persian (&#x0641;&#x0627;&#x0631;&#x0633;&#x0649;)',
  'fi' => 'Suomi',
  'fr' => 'Fran&ccedil;ais',
  'fr_CA' => 'Fran&ccedil;ais (Canada)',
  'hi' => 'Hindi',
  'hr' => 'Hrvatski',
  'hu' => 'Magyar',
  'it' => 'Italiano',
  'ja' => 'Japanese (&#x65e5;&#x672c;&#x8a9e)',
  'lt' => "Lietuvi\x{173} kalba",
  'lv' => 'Latvijas',
  'ms' => 'Malay',
  'nb_NO' => 'Norsk bokm&aring;l',
  'nl' => 'Nederlands',
  'pl' => 'Polski',
  'pt' => 'Portugu&ecirc;s',
  'pt_BR' => 'Portugu&ecirc;s Brasileiro',
  'ru' => 'Russian
 (&#x0420;&#x0443;&#x0441;&#x0441;&#x043a;&#x0438;&#x0439;)',
  'sk_SK' => 'Slovak (Sloven&#x010d;ina)',
  'sl' => "Slovenian (Sloven\x{161}\x{10d}ina)",
  'sr_Cyrl' => "Serbian Cyrillic
 (\x{441}\x{440}\x{43f}\x{441}\x{43a}\x{438})",
  'sr_Latn' => 'Serbian Latin (Srpski)',
  'sv' => 'Svenska',
  'tr' => 'T&uuml;rk&ccedil;e',
  'uk' => 'Ukrainian
 (&#x0423;&#x043a;&#x0440;&#x0430;&#x0457;&#x043d;&#x0441;&#x044c;&#x043a;&#x0430;;)',
  'vi_VN' => 'Vietnam (Vi&#x0246;t Nam)',
  'zh_CN' => 'Chinese (Sim.) (&#x7b80;&#x4f53;&#x4e2d;&#x6587;)',
  'zh_TW' => 'Chinese (Tradi.) (&#x6b63;&#x9ad4;&#x4e2d;&#x6587;)'
};
```

2.1.13. DefaultTheme

Description:	Defines the default front-end (HTML) theme to be used by the agents and customers. The default themes are Standard and Lite. If you like, you can add your own theme. Please refer the administrator manual located at http://doc.otrs.org/.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1

Config-Setting:	`$Self->{'DefaultTheme'} =  'Standard';`

2.1.14. DefaultTheme::HostBased

Description:	It is possible to configure different themes, for example to distinguish between agents and customers, to be used on a per-domain basis within the application. Using a regular expression (regex), you can configure a Key/Content pair to match a domain. The value in "Key" should match the domain, and the value in "Content" should be a valid theme on your system. Please see the example entries for the proper form of the regex.
Group:	Framework
SubGroup:	Core
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'DefaultTheme::HostBased'} =  {
  'host1\\.example\\.com' => 'SomeTheme1',
  'host2\\.example\\.com' => 'SomeTheme2'
};
```

2.1.15. CheckMXRecord

Description:	Makes the application check the MX record of email addresses before sending an email or submitting a telephone or email ticket.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CheckMXRecord'} =  '1';`

2.1.16. CheckMXRecord::Nameserver

Description:	Defines the address of a dedicated DNS server, if necessary, for the "CheckMXRecord" look-ups.
Group:	Framework
SubGroup:	Core
Valid:	0
Required:	0
Config-Setting:	`$Self->{'CheckMXRecord::Nameserver'} =  'ns.example.com';`

2.1.17. CheckEmailAddresses

Description:	Makes the application check the syntax of email addresses.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1

Config-Setting:	`$Self->{'CheckEmailAddresses'} =  '1';`

2.1.18. CheckEmailValidAddress

Description:	Defines a regular expression that excludes some addresses from the syntax check (if "CheckEmailAddresses" is set to "Yes"). Please enter a regex in this field for email addresses, that aren't syntactically valid, but are necessary for the system (i.e. "root@localhost").
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CheckEmailValidAddress'} =  '^(root@localhost\|admin@localhost)$';`

2.1.19. CheckEmailInvalidAddress

Description:	Defines a regular expression that filters all email addresses that should not be used in the application.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CheckEmailInvalidAddress'} =  '@(example)\\.(..\|...)$';`

2.1.20. CGILogPrefix

Description:	Specifies the text that should appear in the log file to denote a CGI script entry.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CGILogPrefix'} =  'OTRS-CGI';`

2.1.21. DemoSystem

Description:	Runs the system in "Demo" mode. If set to "Yes", agents can change preferences, such as selection of language and theme via the agent web interface. These changes are only valid for the current session. It will not be possible for agents to change their passwords.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'DemoSystem'} =  '0';`

2.1.22. SwitchToUser

Description:	Allows the administrators to login as other users, via the users administration panel.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SwitchToUser'} =  '0';`

2.1.23. SwitchToCustomer

Description:	Allows the administrators to login as other customers, via the customer user administration panel.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SwitchToCustomer'} =  '0';`

2.1.24. SwitchToCustomer::PermissionGroup

Description:	Specifies the group where the user needs rw permissions so that he can access the "SwitchToCustomer" feature.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SwitchToCustomer::PermissionGroup'} =  'admin';`

2.1.25. NotificationSenderName

Description:	Specifies the name that should be used by the application when sending notifications. The sender name is used to build the complete display name for the notification master (i.e. "OTRS Notification Master" otrs@your.example.com). Notifications are messages such as en::Customer::QueueUpdate or en::Agent::Move.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'NotificationSenderName'} =  'OTRS Notification Master';`

2.1.26. NotificationSenderEmail

Description:	Specifies the email address that should be used by the application when sending notifications. The email address is used to build

	the complete display name for the notification master (i.e. "OTRS Notification Master" otrs@your.example.com). You can use the OTRS_CONFIG_FQDN variable as set in your configuation, or choose another email address. Notifications are messages such as en::Customer::QueueUpdate or en::Agent::Move.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'NotificationSenderEmail'} =  'otrs@<OTRS_CONFIG_FQDN>';`

2.1.27. System::Customer::Permission

Description:	Defines the standard permissions available for customers within the application. If more permissions are needed, you can enter them here. Permissions must be hard coded to be effective. Please ensure, when adding any of the afore mentioned permissions, that the "rw" permission remains the last entry.
Group:	Framework
SubGroup:	Core
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'System::Customer::Permission'} =  [
  'ro',
  'rw'
];
```

2.1.28. LanguageDebug

Description:	Debugs the translation set. If this is set to "Yes" all strings (text) without translations are written to STDERR. This can be helpful when you are creating a new translation file. Otherwise, this option should remain set to "No".
Group:	Framework
SubGroup:	Core
Valid:	0
Required:	0
Config-Setting:	`$Self->{'LanguageDebug'} =  '0';`

2.1.29. Secure::DisableBanner

Description:	If enabled, the OTRS version tag will be removed from the HTTP headers.
Group:	Framework
SubGroup:	Core
Valid:	0
Required:	0

Config-Setting:	`$Self->{'Secure::DisableBanner'} =  '0';`

2.1.30. StandardResponse2QueueByCreating

Description:	List of default StandardResponses which are assigned automatically to new Queues upon creation.
Group:	Framework
SubGroup:	Core
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'StandardResponse2QueueByCreating'} =  [
  ''
];
```

2.2. Core::Cache

2.2.1. Cache::Module

Description:	Selects the cache backend to use.
Group:	Framework
SubGroup:	Core::Cache
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Cache::Module'} =  'Kernel::System::Cache::FileStorable';`

2.2.2. Cache::SubdirLevels

Description:	Specify how many sub directory levels to use when creating cache files. This should prevent too many cache files being in one directory.
Group:	Framework
SubGroup:	Core::Cache
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Cache::SubdirLevels'} =  '2';`

2.3. Core::LinkObject

2.3.1. LinkObject::ViewMode

Description:	Determines the way the linked objects are displayed in each zoom mask.
Group:	Framework
SubGroup:	Core::LinkObject
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LinkObject::ViewMode'} =  'Simple';`

2.3.2. LinkObject::Type###Normal

Description:	Defines the link type 'Normal'. If the source name and the target name contain the same value, the resulting link is a non-directional one; otherwise, the result is a directional link.
Group:	Framework
SubGroup:	Core::LinkObject
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'LinkObject::Type'}->{'Normal'} =  {
  'SourceName' => 'Normal',
  'TargetName' => 'Normal'
};
```

2.3.3. LinkObject::Type###ParentChild

Description:	Defines the link type 'ParentChild'. If the source name and the target name contain the same value, the resulting link is a non-directional one; otherwise, the result is a directional link.
Group:	Framework
SubGroup:	Core::LinkObject
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'LinkObject::Type'}->{'ParentChild'} =  {
  'SourceName' => 'Parent',
  'TargetName' => 'Child'
};
```

2.3.4. LinkObject::TypeGroup###0001

Description:	Defines the link type groups. The link types of the same group cancel one another. Example: If ticket A is linked per a 'Normal' link with ticket B, then these tickets could not be additionally linked with link of a 'ParentChild' relationship.
Group:	Framework
SubGroup:	Core::LinkObject
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'LinkObject::TypeGroup'}->{'0001'} =  [
  'Normal',
  'ParentChild'
];
```

2.4. Core::Log

2.4.1. LogModule

Description:	Defines the log module for the system. "File" writes all messages in a given logfile, "SysLog" uses the syslog daemon of the system, e.g. syslogd.

Group:	Framework
SubGroup:	Core::Log
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LogModule'} =  'Kernel::System::Log::SysLog';`

2.4.2. LogModule::SysLog::Facility

Description:	If "SysLog" was selected for LogModule, a special log facility can be specified.
Group:	Framework
SubGroup:	Core::Log
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LogModule::SysLog::Facility'} =  'user';`

2.4.3. LogModule::SysLog::LogSock

Description:	If "SysLog" was selected for LogModule, a special log sock can be specified (on solaris you may need to use 'stream').
Group:	Framework
SubGroup:	Core::Log
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LogModule::SysLog::LogSock'} =  'unix';`

2.4.4. LogModule::SysLog::Charset

Description:	If "SysLog" was selected for LogModule, the charset that should be used for logging can be specified.
Group:	Framework
SubGroup:	Core::Log
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LogModule::SysLog::Charset'} =  'utf-8';`

2.4.5. LogModule::LogFile

Description:	If "file" was selected for LogModule, a logfile must be specified. If the file doesn't exist, it will be created by the system.
Group:	Framework
SubGroup:	Core::Log
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LogModule::LogFile'} =  '/tmp/otrs.log';`

2.4.6. LogModule::LogFile::Date

Description:	Adds a suffix with the actual year and month to the OTRS log file. A logfile for every month will be created.
Group:	Framework
SubGroup:	Core::Log
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LogModule::LogFile::Date'} =  '0';`

2.5. Core::MIME-Viewer

2.5.1. MIME-Viewer###application/excel

Description:	Specifies the path to the converter that allows the view of Microsoft Excel files, in the web interface.
Group:	Framework
SubGroup:	Core::MIME-Viewer
Valid:	0
Required:	0
Config-Setting:	`$Self->{'MIME-Viewer'}->{'application/excel'} =  'xlhtml';`

2.5.2. MIME-Viewer###application/msword

Description:	Specifies the path to the converter that allows the view of Microsoft Word files, in the web interface.
Group:	Framework
SubGroup:	Core::MIME-Viewer
Valid:	0
Required:	0
Config-Setting:	`$Self->{'MIME-Viewer'}->{'application/msword'} =  'wvWare';`

2.5.3. MIME-Viewer###application/pdf

Description:	Specifies the path to the converter that allows the view of PDF documents, in the web interface.
Group:	Framework
SubGroup:	Core::MIME-Viewer
Valid:	0
Required:	0
Config-Setting:	`$Self->{'MIME-Viewer'}->{'application/pdf'} =  'pdftohtml -stdout -i';`

2.5.4. MIME-Viewer###text/xml

Description:	Specifies the path to the converter that allows the view of XML files, in the web interface.

Group:	Framework
SubGroup:	Core::MIME-Viewer
Valid:	0
Required:	0
Config-Setting:	`$Self->{'MIME-Viewer'}->{'text/xml'} =  '<OTRS_CONFIG_Home>/scripts/tools/xml2html.pl';`

2.6. Core::MirrorDB

2.6.1. Core::MirrorDB::DSN

Description:	If you want to use a mirror database for agent ticket fulltext search or to generate stats, specify the DSN to this database.
Group:	Framework
SubGroup:	Core::MirrorDB
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Core::MirrorDB::DSN'} = 'DBI:mysql:database=mirrordb;host=mirrordbhost';`

2.6.2. Core::MirrorDB::User

Description:	If you want to use a mirror database for agent ticket fulltext search or to generate stats, the user to authenticate to this database can be specified.
Group:	Framework
SubGroup:	Core::MirrorDB
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Core::MirrorDB::User'} =  'some_user';`

2.6.3. Core::MirrorDB::Password

Description:	If you want to use a mirror database for agent ticket fulltext search or to generate stats, the password to authenticate to this database can be specified.
Group:	Framework
SubGroup:	Core::MirrorDB
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Core::MirrorDB::Password'} =  'some_password';`

2.7. Core::PDF

2.7.1. PDF

Description:	Enables PDF output. The CPAN module PDF::API2 is required, if not installed, PDF output will be disabled.

Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF'} =  '1';`

2.7.2. PDF::LogoFile

Description:	Specifies the path of the file for the logo in the page header (gif|jpg|png, 700 x 100 pixel).
Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF::LogoFile'} =  '<OTRS_CONFIG_Home>/var/logo-otrs.png';`

2.7.3. PDF::PageSize

Description:	Defines the standard size of PDF pages.
Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF::PageSize'} =  'a4';`

2.7.4. PDF::MaxPages

Description:	Defines the maximum number of pages per PDF file.
Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF::MaxPages'} =  '100';`

2.7.5. PDF::TTFontFile###Proportional

Description:	Defines the path and TTF-File to handle proportional font in PDF documents.
Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'Proportional'} =  'DejaVuSans.ttf';`

2.7.6. PDF::TTFontFile###ProportionalBold

Description:	Defines the path and TTF-File to handle bold proportional font in PDF documents.
Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'ProportionalBold'} =  'DejaVuSans-Bold.ttf';`

2.7.7. PDF::TTFontFile###ProportionalItalic

Description:	Defines the path and TTF-File to handle italic proportional font in PDF documents.
Group:	Framework
SubGroup:	Core::PDF
Valid:	0
Required:	0
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'ProportionalItalic'} =  'DejaVuSans-Oblique.ttf';`

2.7.8. PDF::TTFontFile###ProportionalBoldItalic

Description:	Defines the path and TTF-File to handle bold italic proportional font in PDF documents.
Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'ProportionalBoldItalic'} = 'DejaVuSans-BoldOblique.ttf';`

2.7.9. PDF::TTFontFile###Monospaced

Description:	Defines the path and TTF-File to handle monospaced font in PDF documents.
Group:	Framework
SubGroup:	Core::PDF
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'Monospaced'} =  'DejaVuSansMono.ttf';`

2.7.10. PDF::TTFontFile###MonospacedBold

Description:	Defines the path and TTF-File to handle bold monospaced font in PDF documents.

Group:	Framework
SubGroup:	Core::PDF
Valid:	0
Required:	0
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'MonospacedBold'} =  'DejaVuSansMono-Bold.ttf';`

2.7.11. PDF::TTFontFile###MonospacedItalic

Description:	Defines the path and TTF-File to handle italic monospaced font in PDF documents.
Group:	Framework
SubGroup:	Core::PDF
Valid:	0
Required:	0
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'MonospacedItalic'} =  'DejaVuSansMono-Oblique.ttf';`

2.7.12. PDF::TTFontFile###MonospacedBoldItalic

Description:	Defines the path and TTF-File to handle bold italic monospaced font in PDF documents.
Group:	Framework
SubGroup:	Core::PDF
Valid:	0
Required:	0
Config-Setting:	`$Self->{'PDF::TTFontFile'}->{'MonospacedBoldItalic'} = 'DejaVuSansMono-BoldOblique.ttf';`

2.8. Core::Package

2.8.1. Package::FileUpload

Description:	Enables file upload in the package manager frontend.
Group:	Framework
SubGroup:	Core::Package
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Package::FileUpload'} =  '1';`

2.8.2. Package::RepositoryRoot

Description:	Defines the location to get online repository list for additional packages. The first available result will be used.
Group:	Framework
SubGroup:	Core::Package

Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Package::RepositoryRoot'} =  [
  'http://ftp.otrs.org/pub/otrs/misc/packages/repository.xml'
];
```

2.8.3. Package::RepositoryList

Description:	Defines the list of online repositories. Another installations can be used as repository, for example: Key="http://example.com/otrs/public.pl?Action=PublicRepository;File=" and Content="Some Name".
Group:	Framework
SubGroup:	Core::Package
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Package::RepositoryList'} =  {
  'ftp://ftp.example.com/pub/otrs/misc/packages/' => '[Example] ftp://
ftp.example.com/'
};
```

2.8.4. Package::RepositoryAccessRegExp

Description:	Defines the IP regular expression for accessing the local repository. You need to enable this to have access to your local repository and the package::RepositoryList is required on the remote host.
Group:	Framework
SubGroup:	Core::Package
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Package::RepositoryAccessRegExp'} =  '127\\.0\\.0\\.1';`

2.8.5. Package::Timeout

Description:	Sets the timeout (in seconds) for package downloads. Overwrites "WebUserAgent::Timeout".
Group:	Framework
SubGroup:	Core::Package
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Package::Timeout'} =  '15';`

2.8.6. Package::Proxy

Description:	Fetches packages via proxy. Overwrites "WebUserAgent::Proxy".
Group:	Framework

SubGroup:	Core::Package
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Package::Proxy'} =  'http://proxy.sn.no:8001/';`

2.8.7. Package::ShowFeatureAddons

Description:	Toggles display of OTRS FeatureAddons list in PackageManager.
Group:	Framework
SubGroup:	Core::Package
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Package::ShowFeatureAddons'} =  '1';`

2.9. Core::PerformanceLog

2.9.1. PerformanceLog

Description:	Enables performance log (to log the page response time). It will affect the system performance. Frontend::Module###AdminPerformanceLog must be enabled.
Group:	Framework
SubGroup:	Core::PerformanceLog
Valid:	1
Required:	0
Config-Setting:	`$Self->{'PerformanceLog'} =  '0';`

2.9.2. PerformanceLog::File

Description:	Specifies the path of the file for the performance log.
Group:	Framework
SubGroup:	Core::PerformanceLog
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PerformanceLog::File'} =  '<OTRS_CONFIG_Home>/var/log/Performance.log';`

2.9.3. PerformanceLog::FileMax

Description:	Defines the maximum size (in MB) of the log file.
Group:	Framework
SubGroup:	Core::PerformanceLog
Valid:	1
Required:	1

Config-Setting:	`$Self->{'PerformanceLog::FileMax'} =  '25';`

2.10. Core::ReferenceData

2.10.1. ReferenceData::OwnCountryList

Description:	This setting allows you to override the built-in country list with your own list of countries. This is particularly handy if you just want to use a small select group of countries.
Group:	Framework
SubGroup:	Core::ReferenceData
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'ReferenceData::OwnCountryList'} =  {
  'AT' => 'Austria',
  'CH' => 'Switzerland',
  'DE' => 'Germany'
};
```

2.11. Core::SOAP

2.11.1. SOAP::User

Description:	Defines the username to access the SOAP handle (bin/cgi-bin/rpc.pl).
Group:	Framework
SubGroup:	Core::SOAP
Valid:	0
Required:	0
Config-Setting:	`$Self->{'SOAP::User'} =  'some_user';`

2.11.2. SOAP::Password

Description:	Defines the password to access the SOAP handle (bin/cgi-bin/rpc.pl).
Group:	Framework
SubGroup:	Core::SOAP
Valid:	0
Required:	0
Config-Setting:	`$Self->{'SOAP::Password'} =  'some_pass';`

2.11.3. SOAP::Keep-Alive

Description:	Enable keep-alive connection header for SOAP responses.
Group:	Framework
SubGroup:	Core::SOAP
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SOAP::Keep-Alive'} =  '0';`

2.12. Core::Sendmail

2.12.1. SendmailModule

Description:	Defines the module to send emails. "Sendmail" directly uses the sendmail binary of your operating system. Any of the "SMTP" mechanisms use a specified (external) mailserver. "DoNotSendEmail" doesn't send emails and it is useful for test systems.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SendmailModule'} =  'Kernel::System::Email::Sendmail';`

2.12.2. SendmailModule::CMD

Description:	If "Sendmail" was selected as SendmailModule, the location of the sendmail binary and the needed options must be specified.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SendmailModule::CMD'} =  '/usr/sbin/sendmail -i -f';`

2.12.3. SendmailModule::Host

Description:	If any of the "SMTP" mechanisms was selected as SendmailModule, the mailhost that sends out the mails must be specified.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SendmailModule::Host'} =  'mail.example.com';`

2.12.4. SendmailModule::Port

Description:	If any of the "SMTP" mechanisms was selected as SendmailModule, the port where your mailserver is listening for incoming connections must be specified.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	0
Required:	0
Config-Setting:	`$Self->{'SendmailModule::Port'} =  '25';`

2.12.5. SendmailModule::AuthUser

Description:	If any of the "SMTP" mechanisms was selected as SendmailModule, and authentication to the mail server is needed, an username must be specified.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	0
Required:	0
Config-Setting:	`$Self->{'SendmailModule::AuthUser'} =  'MailserverLogin';`

2.12.6. SendmailModule::AuthPassword

Description:	If any of the "SMTP" mechanisms was selected as SendmailModule, and authentication to the mail server is needed, a password must be specified.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	0
Required:	0
Config-Setting:	`$Self->{'SendmailModule::AuthPassword'} =  'MailserverPassword';`

2.12.7. SendmailBcc

Description:	Sends all outgoing email via bcc to the specified address. Please use this only for backup reasons.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	1
Required:	0
Config-Setting:	`$Self->{'SendmailBcc'} =  '';`

2.12.8. SendmailNotificationEnvelopeFrom

Description:	If set, this address is used as envelope sender header in outgoing notifications. If no address is specified, the envelope sender header is empty.
Group:	Framework
SubGroup:	Core::Sendmail
Valid:	0
Required:	0
Config-Setting:	`$Self->{'SendmailNotificationEnvelopeFrom'} =  '';`

2.12.9. SendmailEncodingForce

Description:	Forces encoding of outgoing emails (7bit|8bit|quoted-printable|base64).

Group:	Framework
SubGroup:	Core::Sendmail
Valid:	0
Required:	0
Config-Setting:	`$Self->{'SendmailEncodingForce'} =  'base64';`

2.13. Core::Session

2.13.1. SessionModule

Description:	Defines the module used to store the session data. With "DB" the frontend server can be splitted from the db server. "FS" is faster.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionModule'} =  'Kernel::System::AuthSession::DB';`

2.13.2. SessionName

Description:	Defines the name of the session key. E.g. Session, SessionID or OTRS.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionName'} =  'OTRSAgentInterface';`

2.13.3. CustomerPanelSessionName

Description:	Defines the name of the key for customer sessions.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelSessionName'} =  'OTRSCustomerInterface';`

2.13.4. SessionCheckRemoteIP

Description:	Turns on the remote ip address check. It should be set to "No" if the application is used, for example, via a proxy farm or a dialup connection, because the remote ip address is mostly different for the requests.
Group:	Framework
SubGroup:	Core::Session
Valid:	1

Required:	1
Config-Setting:	`$Self->{'SessionCheckRemoteIP'} =  '1';`

2.13.5. SessionDeleteIfNotRemoteID

Description:	Deletes a session if the session id is used with an invalid remote IP address.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionDeleteIfNotRemoteID'} =  '1';`

2.13.6. SessionMaxTime

Description:	Defines the maximal valid time (in seconds) for a session id.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionMaxTime'} =  '57600';`

2.13.7. SessionMaxIdleTime

Description:	Sets the inactivity time (in seconds) to pass before a session is killed and a user is loged out.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionMaxIdleTime'} =  '21600';`

2.13.8. SessionActiveTime

Description:	Sets the time (in seconds) a user is marked as active.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionActiveTime'} =  '600';`

2.13.9. SessionDeleteIfTimeToOld

Description:	Deletes requested sessions if they have timed out.
Group:	Framework

SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionDeleteIfTimeToOld'} =  '1';`

2.13.10. SessionUseCookie

Description:	Makes the session management use html cookies. If html cookies are disabled or if the client browser disabled html cookies, then the system will work as usual and append the session id to the links.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionUseCookie'} =  '1';`

2.13.11. SessionUseCookieAfterBrowserClose

Description:	Stores cookies after the browser has been closed.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionUseCookieAfterBrowserClose'} =  '0';`

2.13.12. SessionCSRFProtection

Description:	Protection against CSRF (Cross Site Request Forgery) exploits (for more info see http://en.wikipedia.org/wiki/Cross-site_request_forgery).
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionCSRFProtection'} =  '1';`

2.13.13. AgentSessionLimit

Description:	Sets the maximum number of active agents within the timespan defined in SessionActiveTime.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	0
Config-Setting:	`$Self->{'AgentSessionLimit'} =  '100';`

2.13.14. CustomerSessionLimit

Description:	Sets the maximum number of active customers within the timespan defined in SessionActiveTime.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	0
Config-Setting:	`$Self->{'CustomerSessionLimit'} =  '100';`

2.13.15. SessionDir

Description:	If "FS" was selected for SessionModule, a directory where the session data will be stored must be specified.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionDir'} =  '<OTRS_CONFIG_Home>/var/sessions';`

2.13.16. SessionTable

Description:	If "DB" was selected for SessionModule, a table in database where session data will be stored must be specified.
Group:	Framework
SubGroup:	Core::Session
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SessionTable'} =  'sessions';`

2.14. Core::SpellChecker

2.14.1. SpellChecker

Description:	Enables spell checker support.
Group:	Framework
SubGroup:	Core::SpellChecker
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SpellChecker'} =  '0';`

2.14.2. SpellCheckerBin

Description:	Install ispell or aspell on the system, if you want to use a spell checker. Please specify the path to the aspell or ispell binary on your operating system.

Group:	Framework
SubGroup:	Core::SpellChecker
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SpellCheckerBin'} =  '/usr/bin/ispell';`

2.14.3. SpellCheckerDictDefault

Description:	Defines the default spell checker dictionary.
Group:	Framework
SubGroup:	Core::SpellChecker
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SpellCheckerDictDefault'} =  'english';`

2.14.4. SpellCheckerIgnore

Description:	Defines a default list of words, that are ignored by the spell checker.
Group:	Framework
SubGroup:	Core::SpellChecker
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'SpellCheckerIgnore'} =  [
  'www',
  'webmail',
  'https',
  'http',
  'html',
  'rfc'
];
```

2.15. Core::Stats

2.15.1. Stats::StatsHook

Description:	Sets the stats hook.
Group:	Framework
SubGroup:	Core::Stats
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::StatsHook'} =  'Stat#';`

2.15.2. Stats::StatsStartNumber

Description:	Start number for statistics counting. Every new stat increments this number.
Group:	Framework
SubGroup:	Core::Stats

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::StatsStartNumber'} =  '10000';`

2.15.3. Stats::MaxXaxisAttributes

Description:	Defines the default maximum number of X-axis attributes for the time scale.
Group:	Framework
SubGroup:	Core::Stats
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Stats::MaxXaxisAttributes'} =  '1000';`

2.16. Core::Stats::Graph

2.16.1. Stats::Graph::t_margin

Description:	Specifies the top margin of the chart.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::t_margin'} =  '10';`

2.16.2. Stats::Graph::l_margin

Description:	Specifies the left margin of the chart.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::l_margin'} =  '10';`

2.16.3. Stats::Graph::b_margin

Description:	Specifies the bottom margin of the chart.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::b_margin'} =  '10';`

2.16.4. Stats::Graph::r_margin

Description:	Specifies the right margin of the chart.

Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::r_margin'} =  '20';`

2.16.5. Stats::Graph::bgclr

Description:	Specifies the background color of the picture.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::bgclr'} =  'white';`

2.16.6. Stats::Graph::transparent

Description:	Makes the picture transparent.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::transparent'} =  '0';`

2.16.7. Stats::Graph::fgclr

Description:	Specifies the border color of the chart.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::fgclr'} =  'black';`

2.16.8. Stats::Graph::boxclr

Description:	Specifies the background color of the chart.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::boxclr'} =  'white';`

2.16.9. Stats::Graph::accentclr

Description:	Specifies the border color of the legend.

Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::accentclr'} =  'black';`

2.16.10. Stats::Graph::legendclr

Description:	Specifies the text color of the legend.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::legendclr'} =  'black';`

2.16.11. Stats::Graph::textclr

Description:	Specifies the text color of the chart (e. g. caption).
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::textclr'} =  'black';`

2.16.12. Stats::Graph::dclrs

Description:	Defines the colors for the graphs.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	

Config-Setting:

```
$Self->{'Stats::Graph::dclrs'} =  [
  'red',
  'green',
  'blue',
  'yellow',
  'black',
  'purple',
  'orange',
  'pink',
  'marine',
  'cyan',
  'lgray',
  'lblue',
  'lyellow',
  'lgreen',
  'lred',
  'lpurple',
  'lorange',
   'lbrown'
];
```

2.16.13. Stats::Graph::line_width

Description:	Defines the boldness of the line drawed by the graph.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::line_width'} =  '1';`

2.16.14. Stats::Graph::legend_placement

Description:	Defines the placement of the legend. This should be a two letter key of the form: 'B[LCR]|R[TCB]'. The first letter indicates the placement (Bottom or Right), and the second letter the alignment (Left, Right, Center, Top, or Bottom).
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::legend_placement'} =  'BC';`

2.16.15. Stats::Graph::legend_spacing

Description:	Defines the spacing of the legends.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::legend_spacing'} =  '4';`

2.16.16. Stats::Graph::legend_marker_width

Description:	Defines the width of the legend.
Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::legend_marker_width'} =  '12';`

2.16.17. Stats::Graph::legend_marker_height

Description:	Defines the height of the legend.

Group:	Framework
SubGroup:	Core::Stats::Graph
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::Graph::legend_marker_height'} =  '8';`

2.17. Core::Time

2.17.1. TimeInputFormat

Description:	Defines the date input format used in forms (option or input fields).
Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeInputFormat'} =  'Option';`

2.17.2. TimeShowAlwaysLong

Description:	Shows time in long format (days, hours, minutes), if set to "Yes"; or in short format (days, hours), if set to "No".
Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeShowAlwaysLong'} =  '0';`

2.17.3. TimeZone

Description:	Sets the system time zone (required a system with UTC as system time). Otherwise this is a diff time to the local time.
Group:	Framework
SubGroup:	Core::Time
Valid:	0
Required:	0
Config-Setting:	`$Self->{'TimeZone'} =  '+0';`

2.17.4. TimeZoneUser

Description:	Sets the user time zone per user (required a system with UTC as system time and UTC under TimeZone). Otherwise this is a diff time to the local time.
Group:	Framework
SubGroup:	Core::Time
Valid:	1

Required:	0
Config-Setting:	`$Self->{'TimeZoneUser'} =  '0';`

2.17.5. TimeZoneUserBrowserAutoOffset

Description:	Sets the user time zone per user based on java script / browser time zone offset feature at login time.
Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	0
Config-Setting:	`$Self->{'TimeZoneUserBrowserAutoOffset'} =  '1';`

2.17.6. CalendarWeekDayStart

Description:	Define the start day of the week for the date picker.
Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CalendarWeekDayStart'} =  '1';`

2.17.7. TimeVacationDays

Description:	Adds the permanent vacation days. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDays'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.17.8. TimeVacationDaysOneTime

Description:	Adds the one time vacation days. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).

Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'TimeVacationDaysOneTime'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.17.9. TimeWorkingHours

Description:	Defines the hours and week days to count the working time.
Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'TimeWorkingHours'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.17.10. TimeShowCompleteDescription

Description:	Shows time use complete description (days, hours, minutes), if set to "Yes"; or just first letter (d, h, m), if set to "No".
Group:	Framework
SubGroup:	Core::Time
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeShowCompleteDescription'} =  '0';`

2.18. Core::Time::Calendar1

2.18.1. TimeZone::Calendar1Name

Description:	Defines the name of the indicated calendar.
Group:	Framework
SubGroup:	Core::Time::Calendar1
Valid:	1
Required:	1

Config-Setting:	`$Self->{'TimeZone::Calendar1Name'} =  'Calendar Name 1';`

2.18.2. TimeZone::Calendar1

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar1
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar1'} =  '0';`

2.18.3. TimeVacationDays::Calendar1

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar1
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'TimeVacationDays::Calendar1'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.18.4. TimeVacationDaysOneTime::Calendar1

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar1
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'TimeVacationDaysOneTime::Calendar1'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.18.5. TimeWorkingHours::Calendar1

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar1
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'TimeWorkingHours::Calendar1'} =  {
 'Fri' => [
   '8',
   '9',
   '10',
   '11',
   '12',
   '13',
   '14',
   '15',
   '16',
   '17',
   '18',
   '19',
   '20'
 ],
 'Mon' => [
   '8',
   '9',
   '10',
   '11',
   '12',
   '13',
   '14',
   '15',
   '16',
   '17',
   '18',
   '19',
   '20'
 ],
 'Sat' => [],
 'Sun' => [],
 'Thu' => [
   '8',
   '9',
   '10',
   '11',
   '12',
   '13',
   '14',
   '15',
   '16',
   '17',
   '18',
   '19',
   '20'
 ],
 'Tue' => [
   '8',
   '9',
   '10',
   '11',
   '12',
   '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.19. Core::Time::Calendar2

2.19.1. TimeZone::Calendar2Name

Description:	Defines the name of the indicated calendar.
Group:	Framework
SubGroup:	Core::Time::Calendar2
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar2Name'} =  'Calendar Name 2';`

2.19.2. TimeZone::Calendar2

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar2
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar2'} =  '0';`

2.19.3. TimeVacationDays::Calendar2

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar2
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeVacationDays::Calendar2'} =  {`

```
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.19.4. TimeVacationDaysOneTime::Calendar2

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar2
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDaysOneTime::Calendar2'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.19.5. TimeWorkingHours::Calendar2

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar2
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeWorkingHours::Calendar2'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.20. Core::Time::Calendar3

2.20.1. TimeZone::Calendar3Name

Description:	Defines the name of the indicated calendar.
Group:	Framework

SubGroup:	Core::Time::Calendar3
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar3Name'} =  'Calendar Name 3';`

2.20.2. TimeZone::Calendar3

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar3
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar3'} =  '0';`

2.20.3. TimeVacationDays::Calendar3

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar3
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'TimeVacationDays::Calendar3'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.20.4. TimeVacationDaysOneTime::Calendar3

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar3
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'TimeVacationDaysOneTime::Calendar3'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.20.5. TimeWorkingHours::Calendar3

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar3
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'TimeWorkingHours::Calendar3'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.21. Core::Time::Calendar4

2.21.1. TimeZone::Calendar4Name

Description:	Defines the name of the indicated calendar.
Group:	Framework
SubGroup:	Core::Time::Calendar4
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar4Name'} =  'Calendar Name 4';`

2.21.2. TimeZone::Calendar4

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar4
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar4'} =  '0';`

2.21.3. TimeVacationDays::Calendar4

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar4
Valid:	1

Required:	1

Config-Setting:

```
$Self->{'TimeVacationDays::Calendar4'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.21.4. TimeVacationDaysOneTime::Calendar4

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar4
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDaysOneTime::Calendar4'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.21.5. TimeWorkingHours::Calendar4

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar4
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeWorkingHours::Calendar4'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.22. Core::Time::Calendar5

2.22.1. TimeZone::Calendar5Name

Description:	Defines the name of the indicated calendar.

Group:	Framework
SubGroup:	Core::Time::Calendar5
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar5Name'} =  'Calendar Name 5';`

2.22.2. TimeZone::Calendar5

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar5
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar5'} =  '0';`

2.22.3. TimeVacationDays::Calendar5

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar5
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDays::Calendar5'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.22.4. TimeVacationDaysOneTime::Calendar5

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar5
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDaysOneTime::Calendar5'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.22.5. TimeWorkingHours::Calendar5

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar5
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'TimeWorkingHours::Calendar5'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.23. Core::Time::Calendar6

2.23.1. TimeZone::Calendar6Name

Description:	Defines the name of the indicated calendar.
Group:	Framework
SubGroup:	Core::Time::Calendar6
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar6Name'} =  'Calendar Name 6';`

2.23.2. TimeZone::Calendar6

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar6
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar6'} =  '0';`

2.23.3. TimeVacationDays::Calendar6

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar6
Valid:	1

Required:	1

Config-Setting:

```
$Self->{'TimeVacationDays::Calendar6'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.23.4. TimeVacationDaysOneTime::Calendar6

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar6
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDaysOneTime::Calendar6'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.23.5. TimeWorkingHours::Calendar6

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar6
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeWorkingHours::Calendar6'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.24. Core::Time::Calendar7

2.24.1. TimeZone::Calendar7Name

Description:	Defines the name of the indicated calendar.

Group:	Framework
SubGroup:	Core::Time::Calendar7
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar7Name'} =  'Calendar Name 7';`

2.24.2. TimeZone::Calendar7

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar7
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar7'} =  '0';`

2.24.3. TimeVacationDays::Calendar7

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar7
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDays::Calendar7'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.24.4. TimeVacationDaysOneTime::Calendar7

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar7
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDaysOneTime::Calendar7'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.24.5. TimeWorkingHours::Calendar7

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar7
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'TimeWorkingHours::Calendar7'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.25. Core::Time::Calendar8

2.25.1. TimeZone::Calendar8Name

Description:	Defines the name of the indicated calendar.
Group:	Framework
SubGroup:	Core::Time::Calendar8
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar8Name'} =  'Calendar Name 8';`

2.25.2. TimeZone::Calendar8

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar8
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar8'} =  '0';`

2.25.3. TimeVacationDays::Calendar8

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar8
Valid:	1

Required:	1

Config-Setting:

```
$Self->{'TimeVacationDays::Calendar8'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.25.4. TimeVacationDaysOneTime::Calendar8

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar8
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDaysOneTime::Calendar8'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.25.5. TimeWorkingHours::Calendar8

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar8
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeWorkingHours::Calendar8'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.26. Core::Time::Calendar9

2.26.1. TimeZone::Calendar9Name

Description:	Defines the name of the indicated calendar.

Group:	Framework
SubGroup:	Core::Time::Calendar9
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar9Name'} =  'Calendar Name 9';`

2.26.2. TimeZone::Calendar9

Description:	Defines the time zone of the indicated calendar, which can be assigned later to a specific queue.
Group:	Framework
SubGroup:	Core::Time::Calendar9
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TimeZone::Calendar9'} =  '0';`

2.26.3. TimeVacationDays::Calendar9

Description:	Adds the permanent vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar9
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDays::Calendar9'} =  {
  '1' => {
    '1' => 'New Year\'s Day'
  },
  '12' => {
    '24' => 'Christmas Eve',
    '25' => 'First Christmas Day',
    '26' => 'Second Christmas Day',
    '31' => 'New Year\'s Eve'
  },
  '5' => {
    '1' => 'International Workers\' Day'
  }
};
```

2.26.4. TimeVacationDaysOneTime::Calendar9

Description:	Adds the one time vacation days for the indicated calendar. Please use single digit pattern for numbers from 1 to 9 (instead of 01 - 09).
Group:	Framework
SubGroup:	Core::Time::Calendar9
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'TimeVacationDaysOneTime::Calendar9'} =  {
  '2004' => {
    '1' => {
      '1' => 'test'
    }
  }
};
```

2.26.5. TimeWorkingHours::Calendar9

Description:	Defines the hours and week days of the indicated calendar, to count the working time.
Group:	Framework
SubGroup:	Core::Time::Calendar9
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'TimeWorkingHours::Calendar9'} =  {
  'Fri' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Mon' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Sat' => [],
  'Sun' => [],
  'Thu' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Tue' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ],
  'Wed' => [
    '8',
    '9',
    '10',
    '11',
    '12',
    '13',
    '14',
    '15',
    '16',
    '17',
    '18',
    '19',
    '20'
  ]
};
```

2.27. Core::Web

2.27.1. Frontend::WebPath

Description:	Defines the URL base path of icons, CSS and Java Script.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::WebPath'} =  '/otrs-web/';`

2.27.2. Frontend::ImagePath

Description:	Defines the URL image path of icons for navigation.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::ImagePath'} = '<OTRS_CONFIG_Frontend::WebPath>skins/Agent/default/img/';`

2.27.3. Frontend::CSSPath

Description:	Defines the URL CSS path.
Group:	Framework

SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::CSSPath'} = '<OTRS_CONFIG_Frontend::WebPath>css/';`

2.27.4. Frontend::JavaScriptPath

Description:	Defines the URL java script path.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::JavaScriptPath'} = '<OTRS_CONFIG_Frontend::WebPath>js/';`

2.27.5. Frontend::RichText

Description:	Uses richtext for viewing and editing: articles, salutations, signatures, standard responses, auto responses and notifications.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::RichText'} =  '1';`

2.27.6. Frontend::RichTextPath

Description:	Defines the URL rich text editor path.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::RichTextPath'} = '<OTRS_CONFIG_Frontend::WebPath>js/thirdparty/ckeditor-4.0/';`

2.27.7. Frontend::RichTextWidth

Description:	Defines the width for the rich text editor component. Enter number (pixels) or percent value (relative).
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::RichTextWidth'} =  '620';`

2.27.8. Frontend::RichTextHeight

Description:	Defines the height for the rich text editor component. Enter number (pixels) or percent value (relative).
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::RichTextHeight'} =  '320';`

2.27.9. Frontend::RichText::DefaultCSS

Description:	Defines the default CSS used in rich text editors.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::RichText::DefaultCSS'} =  'font-family:Geneva,Helvetica,Arial,sans-serif; font-size: 12px;';`

2.27.10. Frontend::RichText::EnhancedMode

Description:	Defines if the enhanced mode should be used (enables use of table, replace, subscript, superscript, paste from word, etc.).
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::RichText::EnhancedMode'} =  '0';`

2.27.11. DefaultViewNewLine

Description:	Automated line break in text messages after x number of chars.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'DefaultViewNewLine'} =  '90';`

2.27.12. DefaultViewLines

Description:	Sets the number of lines that are displayed in text messages (e.g. ticket lines in the QueueZoom).
Group:	Framework
SubGroup:	Core::Web

Valid:	1
Required:	1
Config-Setting:	`$Self->{'DefaultViewLines'} =  '6000';`

2.27.13. Frontend::AnimationEnabled

Description:	Turns on the animations used in the GUI. If you have problems with these animations (e.g. performance issues), you can turn them off here.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::AnimationEnabled'} =  '1';`

2.27.14. AttachmentDownloadType

Description:	Allows choosing between showing the attachments of a ticket in the browser (inline) or just make them downloadable (attachment).
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'AttachmentDownloadType'} =  'attachment';`

2.27.15. WebMaxFileUpload

Description:	Defines the maximal size (in bytes) for file uploads via the browser.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'WebMaxFileUpload'} =  '24000000';`

2.27.16. WebUploadCacheModule

Description:	Selects the module to handle uploads via the web interface. "DB" stores all uploads in the database, "FS" uses the file system.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'WebUploadCacheModule'} =` `'Kernel::System::Web::UploadCache::DB';`

2.27.17. Frontend::Output::FilterText###AAAURL

Description:	Defines the filter that processes the text in the articles, in order to highlight URLs.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Output::FilterText'}->{'AAAURL'} =  {
  'Module' => 'Kernel::Output::HTML::OutputFilterTextURL',
  'Templates' => {
    'AgentTicketZoom' => '1'
  }
};
```

2.27.18. Frontend::Themes

Description:	Activates the available themes on the system. Value 1 means active, 0 means inactive.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Themes'} =  {
  'Lite' => '0',
  'Standard' => '1'
};
```

2.27.19. Frontend::Output::FilterText###OutputFilterTextAutoLink

Description:	Defines a filter to process the text in the articles, in order to highlight predefined keywords.
Group:	Framework
SubGroup:	Core::Web
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Output::FilterText'}->{'OutputFilterTextAutoLink'}
 =  {
  'Module' => 'Kernel::Output::HTML::OutputFilterTextAutoLink',
  'Templates' => {
    'AgentTicketZoom' => '1'
  }
};
```

2.27.20. Frontend::Output::OutputFilterTextAutoLink###CVE

Description:	Defines a filter for html output to add links behind CVE numbers. The element Image allows two input kinds. At once the name of an image (e.g. faq.png). In this case the OTRS image path will be used. The second possiblity is to insert the link to the image.
Group:	Framework

SubGroup:	Core::Web
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::Output::OutputFilterTextAutoLink'}->{'CVE'} =  {
  'RegExp' => [
    '(CVE|CAN)\\-(\\d{3,4})\\-(\\d{2,6})'
  ],
  'Templates' => {
    'AgentTicketZoom' => '1'
  },
  'URL1' => {
    'Description' => 'Mitre',
    'Image' => 'http://cve.mitre.org/favicon.ico',
    'Target' => '_blank',
    'URL' => 'http://cve.mitre.org/cgi-bin/cvename.cgi?name=<MATCH1>-
<MATCH2>-<MATCH3>'
  },
  'URL2' => {
    'Description' => 'Google',
    'Image' => 'http://www.google.de/favicon.ico',
    'Target' => '_blank',
    'URL' => 'http://google.com/search?q=<MATCH1>-<MATCH2>-<MATCH3>'
  },
  'URL3' => {
    'Description' => 'US-CERT NVD',
    'Image' => 'http://nvd.nist.gov/favicon.ico',
    'Target' => '_blank',
    'URL' => 'http://nvd.nist.gov/nvd.cfm?cvename=<MATCH1>-<MATCH2>-
<MATCH3>'
  }
};
```

2.27.21. Frontend::Output::OutputFilterTextAutoLink###Bugtraq

Description:	Defines a filter for html output to add links behind bugtraq numbers. The element Image allows two input kinds. At once the name of an image (e.g. faq.png). In this case the OTRS image path will be used. The second possiblity is to insert the link to the image.
Group:	Framework
SubGroup:	Core::Web
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::Output::OutputFilterTextAutoLink'}->{'Bugtraq'} =
{
  'RegExp' => [
    'Bugtraq[\\s\\w\\t]*?ID[\\s\\w\\t]*?:[\\s\\w\\t]*?(\\d{2,8})',
    'Bugtraq[\\s\\w\\t]*?ID[\\s\\w\\t]*?(\\d{2,8})',
    'Bugtraq[\\s\\w\\t]*?:[\\s\\w\\t]*?(\\d{2,8})',
    'Bugtraq[\\s\\w\\t]*?(\\d{2,8})',
    'BID[\\s\\w\\t]*?:[\\s\\w\\t]*?(\\d{2,8})',
    'BID[\\s\\w\\t]*?(\\d{2,8})'
  ],
  'Templates' => {
    'AgentTicketZoom' => '1'
  },
  'URL1' => {
    'Description' => 'Security Focus',
    'Image' => 'http://www.securityfocus.com/favicon.ico',
    'Target' => '_blank',
    'URL' => 'http://www.securityfocus.com/bid/<MATCH1>/info'
  },
  'URL2' => {
    'Description' => 'Google',
    'Image' => 'http://www.google.de/favicon.ico',
    'Target' => '_blank',
    'URL' => 'http://google.com/search?q=<MATCH>'
  }
};
```

2.27.22. Frontend::Output::OutputFilterTextAutoLink###MSBulletins

Description:	Defines a filter for html output to add links behind MSBulletin numbers. The element Image allows two input kinds. At once the name of an image (e.g. faq.png). In this case the OTRS image path will be used. The second possiblity is to insert the link to the image.
Group:	Framework
SubGroup:	Core::Web
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Output::OutputFilterTextAutoLink'}-
>{'MSBulletins'} =  {
  'RegExp' => [
    'MS[^A-Za-z]{0,5}(\\d\\d).?(\\d{2,4})'
  ],
  'Templates' => {
    'AgentTicketZoom' => '1'
  },
  'URL1' => {
    'Description' => 'Microsoft Technet',
    'Image' => 'http://www.microsoft.com/favicon.ico',
    'Target' => '_blank',
    'URL' => 'http://www.microsoft.com/technet/security/bulletin/
MS<MATCH1>-<MATCH2>.mspx'
  },
  'URL2' => {
    'Description' => 'Google',
    'Image' => 'http://www.google.de/favicon.ico',
    'Target' => '_blank',
    'URL' => 'http://google.com/search?q=MS<MATCH1>-<MATCH2>'
  }
};
```

2.27.23. Frontend::Output::OutputFilterTextAutoLink###Setting1

Description:	Define a filter for html output to add links behind a defined string. The element Image allows two input kinds. At once the name of an image (e.g. faq.png). In this case the OTRS image path will be used. The second possiblity is to insert the link to the image.
Group:	Framework
SubGroup:	Core::Web
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Output::OutputFilterTextAutoLink'}->{'Setting1'} =
  {
  'RegExp' => [
    'RegExp'
  ],
  'Templates' => {
    'AgentTicketZoom' => '1'
  },
  'URL1' => {
    'Description' => 'Description',
    'Image' => 'right-small.png',
    'Target' => '_blank',
    'URL' => 'URL'
  },
  'URL2' => {
    'Description' => 'Description',
    'Image' => 'Image',
    'Target' => '_blank',
    'URL' => 'URL'
  }
};
```

2.27.24. Frontend::Output::OutputFilterTextAutoLink###Setting2

Description:	Defines a filter for html output to add links behind a defined string. The element Image allows two input kinds. At once the name of an image (e.g. faq.png). In this case the OTRS image path will be used. The second possiblity is to insert the link to the image.
Group:	Framework
SubGroup:	Core::Web
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Output::OutputFilterTextAutoLink'}->{'Setting2'} =
  {
  'RegExp' => [
    'RegExp'
  ],
  'Templates' => {
    'AgentTicketZoom' => '1'
  },
  'URL1' => {
    'Description' => 'Description',
    'Image' => 'right-small.png',
    'Target' => '_blank',
    'URL' => 'URL'
  },
  'URL2' => {
    'Description' => 'Description',
    'Image' => 'Image',
    'Target' => '_blank',
    'URL' => 'URL'
  },
  'URL3' => {
    'Description' => 'Description',
    'Image' => 'Image',
    'Target' => '_blank',
    'URL' => 'URL'
  }
};
```

2.27.25. Loader::Enabled::CSS

Description:	If enabled, OTRS will deliver all CSS files in minified form. WARNING: If you turn this off, there will likely be problems in IE 7, because it cannot load more than 32 CSS files.

Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Loader::Enabled::CSS'} =  '1';`

2.27.26. Loader::Enabled::JS

Description:	If enabled, OTRS will deliver all JavaScript files in minified form.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Loader::Enabled::JS'} =  '1';`

2.27.27. Loader::Agent::CommonCSS###000-Framework

Description:	List of CSS files to always be loaded for the agent interface.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Loader::Agent::CommonCSS'}->{'000-Framework'} =  [
  'Core.Reset.css',
  'Core.Default.css',
  'Core.Header.css',
  'Core.OverviewControl.css',
  'Core.OverviewSmall.css',
  'Core.OverviewMedium.css',
  'Core.OverviewLarge.css',
  'Core.Footer.css',
  'Core.PageLayout.css',
  'Core.Form.css',
  'Core.Table.css',
  'Core.Widget.css',
  'Core.WidgetMenu.css',
  'Core.TicketDetail.css',
  'Core.Tooltip.css',
  'Core.Dialog.css',
  'Core.Print.css'
];
```

2.27.28. Loader::Agent::CommonCSS::IE8###000-Framework

Description:	List of IE8-specific CSS files to always be loaded for the agent interface.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Loader::Agent::CommonCSS::IE8'}->{'000-Framework'} =  [
  'Core.OverviewSmall.IE8.css'
];
```

2.27.29. Loader::Agent::CommonJS###000-Framework

Description:	List of JS files to always be loaded for the agent interface.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Loader::Agent::CommonJS'}->{'000-Framework'} =  [
  'thirdparty/json/json2.js',
  'thirdparty/jquery-1.6.4/jquery.js',
  'thirdparty/jquery-ui-1.8.21/jquery-ui.js',
  'thirdparty/jquery-validate-1.10/jquery.validate.js',
  'thirdparty/stacktrace-0.4/stacktrace.js',
  'thirdparty/jquery-pubsub/pubsub.js',
  'Core.JavaScriptEnhancements.js',
  'Core.Debug.js',
  'Core.Data.js',
  'Core.Config.js',
  'Core.Exception.js',
  'Core.JSON.js',
  'Core.AJAX.js',
  'Core.App.js',
  'Core.UI.js',
  'Core.UI.IE7Fixes.js',
  'Core.UI.Accordion.js',
  'Core.UI.Datepicker.js',
  'Core.UI.Resizable.js',
  'Core.UI.Table.js',
  'Core.UI.Accessibility.js',
  'Core.UI.RichTextEditor.js',
  'Core.UI.Dialog.js',
  'Core.UI.ActionRow.js',
  'Core.UI.Popup.js',
  'Core.Form.js',
  'Core.Form.ErrorTooltips.js',
  'Core.Form.Validate.js',
  'Core.Agent.js',
  'Core.Agent.Search.js',
  'Core.Agent.CustomerInformationCenterSearch.js'
];
```

2.27.30. Loader::Customer::CommonCSS###000-Framework

Description:	List of CSS files to always be loaded for the customer interface.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Loader::Customer::CommonCSS'}->{'000-Framework'} =  [
  'Core.Reset.css',
  'Core.Default.css',
  'Core.Form.css',
  'Core.Dialog.css',
  'Core.Tooltip.css',
  'Core.Login.css',
  'Core.Control.css',
  'Core.Table.css',
  'Core.TicketZoom.css',
  'Core.Print.css'
];
```

2.27.31. Loader::Customer::CommonCSS::IE7###000-Framework

Description:	List of IE7-specific CSS files to always be loaded for the customer interface.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Loader::Customer::CommonCSS::IE7'}->{'000-Framework'} =  [
  'Core.IE7.css',
  'Core.Tooltip.IE7.css',
  'Core.Dialog.IE7.css'
];
```

2.27.32. Loader::Customer::CommonCSS::IE8###000-Framework

Description:	List of IE8-specific CSS files to always be loaded for the customer interface.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Loader::Customer::CommonCSS::IE8'}->{'000-Framework'} =  [];
```

2.27.33. Loader::Customer::CommonJS###000-Framework

Description:	List of JS files to always be loaded for the customer interface.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Loader::Customer::CommonJS'}->{'000-Framework'} =  [
  'thirdparty/jquery-1.6.4/jquery.js',
  'thirdparty/jquery-validate-1.10/jquery.validate.js',
  'thirdparty/jquery-ui-1.8.21/jquery-ui.js',
  'thirdparty/stacktrace-0.4/stacktrace.js',
  'thirdparty/jquery-pubsub/pubsub.js',
  'Core.Debug.js',
  'Core.Data.js',
  'Core.Exception.js',
  'Core.JavaScriptEnhancements.js',
  'Core.Config.js',
  'Core.AJAX.js',
  'Core.App.js',
  'Core.UI.js',
  'Core.UI.IE7Fixes.js',
  'Core.UI.Accessibility.js',
  'Core.UI.Dialog.js',
  'Core.UI.RichTextEditor.js',
  'Core.UI.Datepicker.js',
  'Core.UI.Popup.js',
  'Core.Form.js',
  'Core.Form.ErrorTooltips.js',
  'Core.Form.Validate.js',
  'Core.Customer.js'
];
```

2.27.34. Loader::Agent::DefaultSelectedSkin

Description:	The agent skin's InternalName which should be used in the agent interface. Please check the available skins in Frontend::Agent::Skins.
Group:	Framework
SubGroup:	Core::Web
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Loader::Agent::DefaultSelectedSkin'} =  'default';`

2.27.35. Loader::Customer::SelectedSkin::HostBased

Description:	It is possible to configure different skins, for example to distinguish between diferent customers, to be used on a per-domain basis within the application. Using a regular expression (regex), you can configure a Key/Content pair to match a domain. The value in "Key" should match the domain, and the value in "Content" should be a valid skin on your system. Please see the example entries for the proper form of the regex.
Group:	Framework
SubGroup:	Core::Web
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Loader::Customer::SelectedSkin::HostBased'} =  {
  'host1\\.example\\.com' => 'Someskin1',
  'host2\\.example\\.com' => 'Someskin2'
};
```

2.28. Core::WebUserAgent

2.28.1. WebUserAgent::Timeout

Description:	Sets the timeout (in seconds) for http/ftp downloads.
Group:	Framework
SubGroup:	Core::WebUserAgent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'WebUserAgent::Timeout'} =  '15';`

2.28.2. WebUserAgent::Proxy

Description:	Defines the connections for http/ftp, via a proxy.
Group:	Framework
SubGroup:	Core::WebUserAgent
Valid:	0
Required:	0
Config-Setting:	`$Self->{'WebUserAgent::Proxy'} =  'http://proxy.sn.no:8001/';`

2.29. Crypt::PGP

2.29.1. PGP

Description:	Enables PGP support. When PGP support is enabled for signing and securing mail, it is HIGHLY recommended that the web server be run as the OTRS user. Otherwise, there will be problems with the privileges when accessing .gnupg folder.
Group:	Framework
SubGroup:	Crypt::PGP
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PGP'} =  '0';`

2.29.2. PGP::Bin

Description:	Defines the path to PGP binary.
Group:	Framework
SubGroup:	Crypt::PGP
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PGP::Bin'} =  '/usr/bin/gpg';`

2.29.3. PGP::Options

Description:	Sets the options for PGP binary.
Group:	Framework
SubGroup:	Crypt::PGP
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PGP::Options'} =  '--homedir /opt/otrs/.gnupg/ --batch --no-tty --yes';`

2.29.4. PGP::Key::Password

Description:	Sets the password for private PGP key.
Group:	Framework

SubGroup:	Crypt::PGP
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'PGP::Key::Password'} =  {
  '488A0B8F' => 'SomePassword',
  'D2DF79FA' => 'SomePassword'
};
```

2.29.5. PGP::TrustedNetwork

Description:	Set this to yes if you trust in all your public and private pgp keys, even if they are not certified with a trusted signature.
Group:	Framework
SubGroup:	Crypt::PGP
Valid:	1
Required:	0
Config-Setting:	`$Self->{'PGP::TrustedNetwork'} =  '0';`

2.29.6. PGP::Log

Description:	Configure your own log text for PGP.
Group:	Framework
SubGroup:	Crypt::PGP
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'PGP::Log'} =  {
  'BADSIG' => 'The PGP signature with the keyid has not been verified
successfully.',
  'ERRSIG' => 'It was not possible to check the PGP signature, this
may be caused by a missing public key or an unsupported algorithm.',
  'EXPKEYSIG' => 'The PGP signature was made by an expired key.',
  'GOODSIG' => 'Good PGP signature.',
  'KEYREVOKED' => 'The PGP signature was made by a revoked key, this
could mean that the signature is forged.',
  'NODATA' => 'No valid OpenPGP data found.',
  'NO_PUBKEY' => 'No public key found.',
  'REVKEYSIG' => 'The PGP signature was made by a revoked key, this
could mean that the signature is forged.',
  'SIGEXPIRED' => 'The PGP signature is expired.',
  'SIG_ID' => 'Signature data.',
  'TRUST_UNDEFINED' => 'This key is not certified with a trusted
signature!.',
  'VALIDSIG' => 'The PGP signature with the keyid is good.'
};
```

2.30. Crypt::SMIME

2.30.1. SMIME

Description:	Enables S/MIME support.
Group:	Framework
SubGroup:	Crypt::SMIME
Valid:	1

Required:	1
Config-Setting:	`$Self->{'SMIME'} =  '0';`

2.30.2. SMIME::Bin

Description:	Defines the path to open ssl binary. It may need a HOME env ($ENV{HOME} = '/var/lib/wwwrun';).
Group:	Framework
SubGroup:	Crypt::SMIME
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SMIME::Bin'} =  '/usr/bin/openssl';`

2.30.3. SMIME::CertPath

Description:	Specifies the directory where SSL certificates are stored.
Group:	Framework
SubGroup:	Crypt::SMIME
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SMIME::CertPath'} =  '/etc/ssl/certs';`

2.30.4. SMIME::PrivatePath

Description:	Specifies the directory where private SSL certificates are stored.
Group:	Framework
SubGroup:	Crypt::SMIME
Valid:	1
Required:	1
Config-Setting:	`$Self->{'SMIME::PrivatePath'} =  '/etc/ssl/private';`

2.31. CustomerInformationCenter

2.31.1. AgentCustomerInformationCenter::MainMenu###010-EditCustomerID

Description:	Main menu registration.
Group:	Framework
SubGroup:	CustomerInformationCenter
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'AgentCustomerInformationCenter::MainMenu'}->{'010-
EditCustomerID'} =  {
  'Link' =>
 '$Env{"Baselink"}Action=AdminCustomerCompany;Subaction=Change;CustomerID=
$LQData{"CustomerID"};Nav=0',
  'Name' => 'Edit customer company'
};
```

2.32. Frontend::Admin::AdminCustomerUser

2.32.1. AdminCustomerUser::RunInitialWildcardSearch

Description:	Runs an initial wildcard search of the existing customer users when accessing the AdminCustomerUser module.
Group:	Framework
SubGroup:	Frontend::Admin::AdminCustomerUser
Valid:	1
Required:	1
Config-Setting:	`$Self->{'AdminCustomerUser::RunInitialWildcardSearch'} =  '1';`

2.33. Frontend::Admin::ModuleRegistration

2.33.1. Frontend::Module###Admin

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'Admin'} =  {
  'Description' => 'Admin-Area',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.SysConfig.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'a',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=Admin',
      'LinkOption' => '',
      'Name' => 'Admin',
      'NavBar' => 'Admin',
      'Prio' => '10000',
      'Type' => 'Menu'
    }
  ],
  'NavBarModule' => {
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin'
  },
  'NavBarName' => 'Admin',
  'Title' => ''
};
```

2.33.2. Frontend::Module###AdminInit

Description:	Frontend module registration for the agent interface.

Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminInit'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarName' => '',
  'Title' => 'Init'
};
```

2.33.3. Frontend::Module###AdminUser

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminUser'} =  {
  'Description' => 'Create and manage agents.',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Agent',
    'Description' => 'Create and manage agents.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Agents',
    'Prio' => '100'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Agents'
};
```

2.33.4. Frontend::Module###AdminGroup

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminGroup'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Agent',
    'Description' => 'Create and manage groups.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Groups',
    'Prio' => '150'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Groups'
};
```

2.33.5. Frontend::Module###AdminUserGroup

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminUserGroup'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Agent',
    'Description' => 'Link agents to groups.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Agents <-> Groups',
    'Prio' => '200'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Agents <-> Groups'
};
```

2.33.6. Frontend::Module###AdminCustomerUser

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminCustomerUser'} =  {
  'Description' => 'Edit Customers',
  'Group' => [
    'admin',
    'users'
  ],
  'GroupRo' => [
    ''
  ],
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'c',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=AdminCustomerUser;Nav=Agent',
      'LinkOption' => '',
      'Name' => 'Customer User Administration',
      'NavBar' => 'Customers',
      'Prio' => '9000',
      'Type' => ''
    }
  ],
  'NavBarModule' => {
    'Block' => 'Customer',
    'Description' => 'Create and manage customers.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Customers',
    'Prio' => '300'
  },
  'NavBarName' => 'Customers',
  'Title' => 'Customers'
};
```

2.33.7. Frontend::Module###AdminCustomerCompany

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminCustomerCompany'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin',
    'users'
  ],
  'GroupRo' => [
    ''
  ],
  'NavBar' => [
    {
      'AccessKey' => 'c',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=AdminCustomerCompany;Nav=Agent',
      'LinkOption' => '',
      'Name' => 'Customer Company Administration',
      'NavBar' => 'Customers',
      'Prio' => '9100',
      'Type' => ''
    }
  ],
  'NavBarModule' => {
    'Block' => 'Customer',
    'Description' => 'Create and manage companies.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Customer Companies',
    'Prio' => '310'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Customer Companies'
};
```

2.33.8. Frontend::Module###AdminCustomerUserGroup

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration

Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminCustomerUserGroup'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Customer',
    'Description' => 'Link customers to groups.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Customers <-> Groups',
    'Prio' => '400'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Customers <-> Groups'
};
```

2.33.9. Frontend::Module###AdminCustomerUserService

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminCustomerUserService'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Customer',
    'Description' => 'Link customers to services.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Customers <-> Services',
    'Prio' => '500'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Customers <-> Services'
};
```

2.33.10. Frontend::Module###AdminRole

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminRole'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Agent',
    'Description' => 'Create and manage roles.',
```


```
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Roles',
    'Prio' => '600'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Roles'
};
```

2.33.11. Frontend::Module###AdminRoleUser

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminRoleUser'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Agent',
    'Description' => 'Link agents to roles.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Agents <-> Roles',
    'Prio' => '700'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Agents <-> Roles'
};
```

2.33.12. Frontend::Module###AdminRoleGroup

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminRoleGroup'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Agent',
    'Description' => 'Link roles to groups.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Roles <-> Groups',
    'Prio' => '800'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Roles <-> Groups'
};
```

2.33.13. Frontend::Module###AdminSMIME

Description:	Frontend module registration for the agent interface.

Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminSMIME'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Email',
    'Description' => 'Manage S/MIME certificates for email
 encryption.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'S/MIME Certificates',
    'Prio' => '1100'
  },
  'NavBarName' => 'Admin',
  'Title' => 'S/MIME Management'
};
```

2.33.14. Frontend::Module###AdminPGP

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminPGP'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Email',
    'Description' => 'Manage PGP keys for email encryption.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'PGP Keys',
    'Prio' => '1200'
  },
  'NavBarName' => 'Admin',
  'Title' => 'PGP Key Management'
};
```

2.33.15. Frontend::Module###AdminMailAccount

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminMailAccount'} =  {
  'Description' => 'Admin',
  'Group' => [
```


```
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Email',
    'Description' => 'Manage POP3 or IMAP accounts to fetch email
 from.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'PostMaster Mail Accounts',
    'Prio' => '100'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Mail Accounts'
};
```

2.33.16. Frontend::Module###AdminPostMasterFilter

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminPostMasterFilter'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Email',
    'Description' => 'Filter incoming emails.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'PostMaster Filters',
    'Prio' => '200'
  },
  'NavBarName' => 'Admin',
  'Title' => 'PostMaster Filters'
};
```

2.33.17. Frontend::Module###AdminEmail

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminEmail'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'Send notifications to users.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Admin Notification',
    'Prio' => '400'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Admin Notification'
```

```
};
```

2.33.18. Frontend::Module###AdminSession

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminSession'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'Manage existing sessions.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Session Management',
    'Prio' => '500'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Session Management'
};
```

2.33.19. Frontend::Module###AdminPerformanceLog

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminPerformanceLog'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.PerformanceLog.css'
    ]
  },
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'View performance benchmark results.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Performance Log',
    'Prio' => '550'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Performance Log'
};
```

2.33.20. Frontend::Module###AdminLog

Description:	Frontend module registration for the agent interface.

Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminLog'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'View system log messages.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'System Log',
    'Prio' => '600'
  },
  'NavBarName' => 'Admin',
  'Title' => 'System Log'
};
```

2.33.21. Frontend::Module###AdminSelectBox

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminSelectBox'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'Execute SQL statements.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'SQL Box',
    'Prio' => '700'
  },
  'NavBarName' => 'Admin',
  'Title' => 'SQL Box'
};
```

2.33.22. Frontend::Module###AdminPackageManager

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminPackageManager'} =  {
  'Description' => 'Software Package Manager',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'Update and extend your system with software
 packages.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Package Manager',
    'Prio' => '1000'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Package Manager'
};
```

2.34. Frontend::Agent

2.34.1. AgentLogo

Description:	The logo shown in the header of the agent interface. The URL to the image can be a relative URL to the skin image directory, or a full URL to a remote web server.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'AgentLogo'} =  {
  'StyleHeight' => '67px',
  'StyleRight' => '38px',
  'StyleTop' => '-4px',
  'StyleWidth' => '244px',
  'URL' => 'skins/Agent/default/img/logo_bg.png'
};
```

2.34.2. AgentLoginLogo

Description:	The logo shown on top of the login box of the agent interface. The URL to the image must be relative URL to the skin image directory.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'AgentLoginLogo'} =  {
  'StyleHeight' => '100px',
  'URL' => 'skins/Agent/default/img/loginlogo_default.png'
};
```

2.34.3. LoginURL

Description:	Defines an alternate URL, where the login link refers to.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	0
Required:	0
Config-Setting:	`$Self->{'LoginURL'} =  'http://host.example.com/login.html';`

2.34.4. LogoutURL

Description:	Defines an alternate URL, where the logout link refers to.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	0
Required:	0
Config-Setting:	`$Self->{'LogoutURL'} =  'http://host.example.com/thanks-for-using-otrs.html';`

2.34.5. PreApplicationModule###AgentInfo

Description:	Defines a useful module to load specific user options or to display news.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	0
Required:	0
Config-Setting:	`$Self->{'PreApplicationModule'}->{'AgentInfo'} = 'Kernel::Modules::AgentInfo';`

2.34.6. InfoKey

Description:	Defines the key to be checked with Kernel::Modules::AgentInfo module. If this user preferences key is true, the message is accepted by the system.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'InfoKey'} =  'wpt22';`

2.34.7. InfoFile

Description:	File that is displayed in the Kernel::Modules::AgentInfo module, if located under Kernel/Output/HTML/Standard/AgentInfo.dtl.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'InfoFile'} =  'AgentInfo';`

2.34.8. LostPassword

Description:	Activates lost password feature for agents, in the agent interface.
Group:	Framework

SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LostPassword'} =  '1';`

2.34.9. ShowMotd

Description:	Shows the message of the day on login screen of the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'ShowMotd'} =  '0';`

2.34.10. NotificationSubjectLostPasswordToken

Description:	Defines the subject for notification mails sent to agents, with token about new requested password.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'NotificationSubjectLostPasswordToken'} =  'New OTRS password request';`

2.34.11. NotificationBodyLostPasswordToken

Description:	Defines the body text for notification mails sent to agents, with token about new requested password (after using this link the new password will be sent).
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'NotificationBodyLostPasswordToken'} =  'Hi
 <OTRS_USERFIRSTNAME>,

You or someone impersonating you has requested to change your OTRS
password.

If you want to do this, click on the link below. You will receive
 another email containing the password.

<OTRS_CONFIG_HttpType>://<OTRS_CONFIG_FQDN>/
<OTRS_CONFIG_ScriptAlias>index.pl?
Action=LostPassword;Token=<OTRS_TOKEN>

If you did not request a new password, please ignore this email.
            ';
```

2.34.12. NotificationSubjectLostPassword

Description:	Defines the subject for notification mails sent to agents, about new password.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'NotificationSubjectLostPassword'} =  'New OTRS password';`

2.34.13. NotificationBodyLostPassword

Description:	Defines the body text for notification mails sent to agents, about new password (after using this link the new password will be sent).
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'NotificationBodyLostPassword'} =  'Hi <OTRS_USERFIRSTNAME>,


Here\'s your new OTRS password.

New password: <OTRS_NEWPW>

You can log in via the following URL:

<OTRS_CONFIG_HttpType>://<OTRS_CONFIG_FQDN>/
<OTRS_CONFIG_ScriptAlias>index.pl
            ';
```

2.34.14. OpenMainMenuOnHover

Description:	If enabled, the first level of the main menu opens on mouse hover (instead of click only).
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'OpenMainMenuOnHover'} =  '0';`

2.34.15. Loader::Agent::Skin###000-default

Description:	Default skin for interface.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Loader::Agent::Skin'}->{'000-default'} =  {
  'Description' => 'This is the default orange - black skin.',
  'HomePage' => 'www.otrs.org',
  'InternalName' => 'default',
  'VisibleName' => 'Default'
};
```

2.34.16. Loader::Agent::Skin###001-ivory

Description:	Balanced white skin by Felix Niklas.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Loader::Agent::Skin'}->{'001-ivory'} =  {
  'Description' => 'Balanced white skin by Felix Niklas',
  'HomePage' => 'www.felixniklas.de',
  'InternalName' => 'ivory',
  'VisibleName' => 'Ivory'
};
```

2.34.17. Loader::Agent::Skin###001-slim

Description:	"Slim" Skin which tries to save screen space for power users.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Loader::Agent::Skin'}->{'001-slim'} =  {
  'Description' => '"Slim" skin which tries to save screen space for
power users.',
  'HomePage' => 'www.otrs.org',
  'InternalName' => 'slim',
  'VisibleName' => 'Slim'
};
```

2.34.18. Loader::Agent::DefaultSelectedSkin::HostBased

Description:	It is possible to configure different skins, for example to distinguish between diferent agents, to be used on a per-domain basis within the application. Using a regular expression (regex), you can configure a Key/Content pair to match a domain. The value in "Key" should match the domain, and the value in "Content" should be a valid skin on your system. Please see the example entries for the proper form of the regex.
Group:	Framework
SubGroup:	Frontend::Agent
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Loader::Agent::DefaultSelectedSkin::HostBased'} =  {
  'host1\\.example\\.com' => 'SomeSkin1',
  'host2\\.example\\.com' => 'SomeSkin2'
};
```

2.35. Frontend::Agent::Dashboard

2.35.1. AgentCustomerInformationCenter::Backend###0600-CIC-CustomerCompanyInformation

Description:	Parameters for the dashboard backend of the customer company information of the agent interface . "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'AgentCustomerInformationCenter::Backend'}->{'0600-CIC-
CustomerCompanyInformation'} =  {
  'Attributes' => '',
  'Block' => 'ContentSmall',
  'Default' => '1',
  'Description' => 'Customer Company Information',
  'Group' => '',
  'Module' =>
 'Kernel::Output::HTML::DashboardCustomerCompanyInformation',
  'Title' => 'Customer Company Information'
};
```

2.35.2. DashboardBackend###0000-ProductNotify

Description:	Defines the parameters for the dashboard backend. "Group" is used to restrict access to the plugin (e. g. Group: admin;group1;group2;). "Default" indicates if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" defines the cache expiration period in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'DashboardBackend'}->{'0000-ProductNotify'} =  {
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '1440',
  'Default' => '1',
  'Description' => 'News about OTRS releases!',
  'Group' => 'admin',
  'Module' => 'Kernel::Output::HTML::DashboardProductNotify',
  'Title' => 'Product News',
  'URL' => 'http://otrs.org/product.xml'
};
```

2.35.3. DashboardBackend###0390-UserOutOfOffice

Description:	Defines the parameters for the dashboard backend. "Limit" defines the number of entries displayed by default. "Group" is used to restrict

	access to the plugin (e. g. Group: admin;group1;group2;). "Default" indicates if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" defines the cache expiration period in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DashboardBackend'}->{'0390-UserOutOfOffice'} =  {
  'Block' => 'ContentSmall',
  'CacheTTLLocal' => '5',
  'Default' => '1',
  'Description' => '',
  'Group' => '',
  'IdleMinutes' => '60',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardUserOutOfOffice',
  'SortBy' => 'UserLastname',
  'Title' => 'Out Of Office'
};
```

2.35.4. DashboardBackend###0400-UserOnline

Description:	Defines the parameters for the dashboard backend. "Limit" defines the number of entries displayed by default. "Group" is used to restrict access to the plugin (e. g. Group: admin;group1;group2;). "Default" indicates if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" defines the cache expiration period in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DashboardBackend'}->{'0400-UserOnline'} =  {
  'Block' => 'ContentSmall',
  'CacheTTLLocal' => '5',
  'Default' => '0',
  'Description' => '',
  'Filter' => 'Agent',
  'Group' => '',
  'IdleMinutes' => '60',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardUserOnline',
  'ShowEmail' => '0',
  'SortBy' => 'UserLastname',
  'Title' => 'Online'
};
```

2.35.5. DashboardBackend###0410-RSS

Description:	Defines the parameters for the dashboard backend. "Limit" defines the number of entries displayed by default. "Group" is used to restrict access to the plugin (e. g. Group: admin;group1;group2;). "Default" indicates if the plugin is enabled by default or if the user needs to

	enable it manually. "CacheTTL" indicates the cache expiration period in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DashboardBackend'}->{'0410-RSS'} =  {
  'Block' => 'ContentSmall',
  'CacheTTL' => '360',
  'Default' => '1',
  'Description' => '',
  'Group' => '',
  'Limit' => '6',
  'Module' => 'Kernel::Output::HTML::DashboardRSS',
  'Title' => 'OTRS News',
  'URL' => 'http://www.otrs.com/en/rss.xml',
  'URL_de' => 'http://www.otrs.com/de/rss.xml',
  'URL_es' => 'http://www.otrs.com/es/rss.xml',
  'URL_nl' => 'http://www.otrs.com/nl/rss.xml',
  'URL_ru' => 'http://www.otrs.com/ru/rss.xml',
  'URL_zh' => 'http://www.otrs.com/cn/rss.xml'
};
```

2.35.6. DashboardBackend###0200-Image

Description:	Defines the parameters for the dashboard backend. "Group" is used to restrict access to the plugin (e. g. Group: admin;group1;group2;). "Default" indicates if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTL" indicates the cache expiration period in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'DashboardBackend'}->{'0200-Image'} =  {
  'Block' => 'ContentLarge',
  'Default' => '1',
  'Description' => 'Some picture description!',
  'Group' => '',
  'Height' => '140',
  'Link' => 'http://otrs.org/',
  'LinkTitle' => 'http://otrs.org/',
  'Module' => 'Kernel::Output::HTML::DashboardImage',
  'Title' => 'A picture',
  'URL' => 'http://www.otrs.com/uploads/pics/jointhecommunity_02.jpg',
  'Width' => '198'
};
```

2.35.7. DashboardBackend###0210-MOTD

Description:	Shows the message of the day (MOTD) in the agent dashboard. "Group" is used to restrict access to the plugin (e. g. Group: admin;group1;group2;). "Default" indicates if the plugin is enabled by default or if the user needs to enable it manually.
Group:	Framework

SubGroup:	Frontend::Agent::Dashboard
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'DashboardBackend'}->{'0210-MOTD'} =  {
  'Block' => 'ContentLarge',
  'Default' => '1',
  'Group' => '',
  'Module' => 'Kernel::Output::HTML::DashboardMOTD',
  'Title' => 'Message of the Day'
};
```

2.35.8. DashboardBackend###0300-IFrame

Description:	Defines the parameters for the dashboard backend. "Group" is used to restrict access to the plugin (e. g. Group: admin;group1;group2;). "Default" indicates if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTL" indicates the cache expiration period in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'DashboardBackend'}->{'0300-IFrame'} =  {
  'Align' => 'left',
  'Block' => 'ContentLarge',
  'Default' => '1',
  'Description' => 'Some description!',
  'Frameborder' => '1',
  'Group' => '',
  'Height' => '800',
  'Link' => 'http://otrs.org/',
  'LinkTitle' => 'OTRS.org/',
  'Marginheight' => '5',
  'Marginwidth' => '5',
  'Module' => 'Kernel::Output::HTML::DashboardIFrame',
  'Scrolling' => 'auto',
  'Title' => 'A Website',
  'URL' => 'http://www.otrs.org/',
  'Width' => '1024'
};
```

2.35.9. AgentCustomerInformationCenter::Backend###0050-CIC-CustomerUserList

Description:	Parameters for the dashboard backend of the customer user list overview of the agent interface . "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Framework
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'AgentCustomerInformationCenter::Backend'}->{'0050-CIC-
CustomerUserList'} =  {
  'Attributes' => '',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All customer users of a CustomerID',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardCustomerUserList',
  'Permission' => 'ro',
  'Title' => 'Customer Users'
};
```

2.36. Frontend::Agent::LinkObject

2.36.1. Frontend::AgentLinkObject::WildcardSearch

Description:	Starts a wildcard search of the active object after the link object mask is started.
Group:	Framework
SubGroup:	Frontend::Agent::LinkObject
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::AgentLinkObject::WildcardSearch'} =  '0';`

2.37. Frontend::Agent::ModuleMetaHead

2.37.1. Frontend::HeaderMetaModule###100-Refresh

Description:	Defines the module to generate html refresh headers of html sites.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleMetaHead
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Frontend::HeaderMetaModule'}->{'100-Refresh'} =  {
  'Module' => 'Kernel::Output::HTML::HeaderMetaRefresh'
};
```

2.38. Frontend::Agent::ModuleNotify

2.38.1. Frontend::NotifyModule###200-UID-Check

Description:	Defines the module to display a notification in the agent interface, if the system is used by the admin user (normally you shouldn't work as admin).
Group:	Framework
SubGroup:	Frontend::Agent::ModuleNotify
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Frontend::NotifyModule'}->{'200-UID-Check'} =  {
  'Module' => 'Kernel::Output::HTML::NotificationUIDCheck'
```

```
};
```

2.38.2. Frontend::NotifyModule###300-ShowAgentOnline

Description:	Defines the module that shows all the currently logged in agents in the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleNotify
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::NotifyModule'}->{'300-ShowAgentOnline'} =  {
  'IdleMinutes' => '60',
  'Module' => 'Kernel::Output::HTML::NotificationAgentOnline',
  'ShowEmail' => '1'
};
```

2.38.3. Frontend::NotifyModule###400-ShowCustomerOnline

Description:	Defines the module that shows all the currently loged in customers in the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleNotify
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::NotifyModule'}->{'400-ShowCustomerOnline'} =  {
  'IdleMinutes' => '60',
  'Module' => 'Kernel::Output::HTML::NotificationCustomerOnline',
  'ShowEmail' => '1'
};
```

2.38.4. Frontend::NotifyModule###500-OutofOffice-Check

Description:	Defines the module to display a notification in the agent interface, if the agent is logged in while having out-of-office active.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleNotify
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Frontend::NotifyModule'}->{'500-OutofOffice-Check'} =  {
  'Module' => 'Kernel::Output::HTML::NotificationOutofOfficeCheck'
};
```

2.38.5. Frontend::NotifyModule###900-Generic

Description:	Defines the module that shows a generic notification in the agent interface. Either "Text" - if configured - or the contents of "File" will be displayed.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleNotify

Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::NotifyModule'}->{'900-Generic'} =  {
  'File' => '<OTRS_CONFIG_Home>/var/notify.txt',
  'Link' => 'http://www.otrs.com',
  'Module' => 'Kernel::Output::HTML::NotificationGeneric',
  'Priority' => 'Warning',
  'Text' => 'The OTRS Website'
};
```

2.39. Frontend::Agent::ModuleRegistration

2.39.1. Frontend::Module###Logout

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'Logout'} =  {
  'Description' => 'Logout',
  'NavBarName' => '',
  'Title' => ''
};
```

2.39.2. Frontend::Module###AgentDashboard

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentDashboard'} =  {
  'Description' => 'Agent Dashboard',
  'Loader' => {
    'JavaScript' => [
      'thirdparty/flot-0.7/excanvas.js',
      'thirdparty/flot-0.7/jquery.flot.js',
      'Core.UI.Chart.js',
      'Core.UI.DnD.js',
      'Core.Agent.Dashboard.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'd',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=AgentDashboard',
      'LinkOption' => '',
      'Name' => 'Dashboard',
      'NavBar' => 'Dashboard',
      'Prio' => '50',
      'Type' => 'Menu'
    }
  ],
  'NavBarName' => 'Dashboard',
  'Title' => ''
};
```

2.39.3. Frontend::Module###AgentCustomerInformationCenter

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentCustomerInformationCenter'} =  {
  'Description' => 'Customer Information Center',
  'Loader' => {
    'JavaScript' => [
      'thirdparty/flot-0.7/excanvas.js',
      'thirdparty/flot-0.7/jquery.flot.js',
      'Core.UI.Chart.js',
      'Core.UI.DnD.js',
      'Core.Agent.Dashboard.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'c',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=AgentCustomerInformationCenter',
      'LinkOption' => 'onclick="window.setTimeout(function()
{Core.Agent.CustomerInformationCenterSearch.OpenSearchDialog();}, 0);
 return false;"',
      'Name' => 'Customer Information Center',
      'NavBar' => 'Customers',
      'Prio' => '50',
      'Type' => ''
    },
    {
      'AccessKey' => 'c',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=AgentCustomerInformationCenter',
      'LinkOption' => '',
      'Name' => 'Customers',
      'NavBar' => 'Customers',
      'Prio' => '60',
      'Type' => 'Menu'
    }
  ],
  'NavBarName' => 'Customer Information Center',
  'Title' => ''
};
```

2.39.4. Frontend::Module###AgentCustomerInformationCenterSearch

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentCustomerInformationCenterSearch'}
=  {
  'Description' => 'Customer Information Center Search',
  'Title' => ''
};
```

2.39.5. Frontend::Module###AgentPreferences

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentPreferences'} =  {
  'Description' => 'Agent Preferences',
  'NavBarName' => 'Preferences',
  'Title' => ''
};
```

2.39.6. Frontend::Module###PictureUpload

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Frontend::Module'}->{'PictureUpload'} =  {
  'Description' => 'Picture upload module',
  'NavBarName' => 'Ticket',
  'Title' => 'Picture-Upload'
};
```

2.39.7. Frontend::Module###AgentSpelling

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentSpelling'} =  {
  'Description' => 'Spell checker',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => '',
  'Title' => 'Spell Checker'
};
```

2.39.8. Frontend::Module###SpellingInline

Description:	Frontend module registration for the agent interface.

Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'SpellingInline'} =  {
  'Description' => 'Spell checker',
  'NavBarName' => '',
  'Title' => 'Spell Checker'
};
```

2.39.9. Frontend::Module###AgentBook

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentBook'} =  {
  'Description' => 'Address book of CustomerUser sources',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.CustomerSearch.js',
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => '',
  'Title' => 'Address Book'
};
```

2.39.10. Frontend::Module###AgentLinkObject

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentLinkObject'} =  {
  'Description' => 'Link Object',
  'NavBarName' => '',
  'Title' => 'Link Object'
};
```

2.39.11. Frontend::Module###AgentInfo

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentInfo'} =  {
  'Description' => 'Generic Info module',
  'NavBarName' => '',
  'Title' => 'Info'
};
```

2.39.12. Frontend::Module###AgentSearch

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentSearch'} =  {
  'Description' => 'Global Search Module',
  'NavBarName' => '',
  'Title' => 'Search'
};
```

2.39.13. CustomerFrontend::Module###SpellingInline

Description:	Frontend module registration for the customer interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerFrontend::Module'}->{'SpellingInline'} =  {
  'Description' => 'Spell checker',
  'NavBarName' => '',
  'Title' => 'Spell Checker'
};
```

2.39.14. Frontend::Module###AgentHTMLReference

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentHTMLReference'} =  {
  'Description' => 'HTML Reference',
  'Group' => [
    'users'
  ],
  'GroupRo' => [
    'users'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.HTMLReference.css'
    ]
  },
  'NavBarName' => '',
  'Title' => 'HTML Reference'
```

```
};
```

2.39.15. Frontend::Module###AgentStats

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AgentStats'} =  {
  'Description' => 'Stats',
  'Group' => [
    'stats'
  ],
  'GroupRo' => [
    'stats'
  ],
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.Stats.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => '',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=AgentStats;Subaction=Overview',
      'LinkOption' => '',
      'Name' => 'Statistics',
      'NavBar' => 'Stats',
      'Prio' => '8500',
      'Type' => 'Menu'
    },
    {
      'AccessKey' => '',
      'Block' => '',
      'Description' => 'Overview',
      'GroupRo' => [
        'stats'
      ],
      'Link' => 'Action=AgentStats;Subaction=Overview',
      'LinkOption' => '',
      'Name' => 'Overview',
      'NavBar' => 'Stats',
      'Prio' => '100',
      'Type' => ''
    },
    {
      'AccessKey' => '',
      'Block' => '',
      'Description' => 'New',
      'Group' => [
        'stats'
      ],
      'Link' => 'Action=AgentStats;Subaction=Add',
      'LinkOption' => '',
      'Name' => 'New',
      'NavBar' => 'Stats',
      'Prio' => '200',
      'Type' => ''
    },
    {
      'AccessKey' => '',
      'Block' => '',
      'Description' => 'Import',
      'Group' => [
        'stats'
      ],
      'Link' => 'Action=AgentStats;Subaction=Import',
      'LinkOption' => '',
      'Name' => 'Import',
      'NavBar' => 'Stats',
      'Prio' => '300',
      'Type' => ''
    }
  ],
  'NavBarName' => 'Stats',
  'Title' => 'Stats'
};
```

2.40. Frontend::Agent::NavBarModule

2.40.1. Frontend::NavBarModule###6-CustomerCompany

Description:	Frontend module registration (disable company link if no company feature is used).
Group:	Framework
SubGroup:	Frontend::Agent::NavBarModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::NavBarModule'}->{'6-CustomerCompany'} =  {
  'Module' => 'Kernel::Output::HTML::NavBarCustomerCompany'
};
```

2.41. Frontend::Agent::Preferences

2.41.1. PreferencesTableValue

Description:	Defines the name of the column to store the data in the preferences table.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PreferencesTableValue'} =  'preferences_value';`

2.41.2. PreferencesTableUserID

Description:	Defines the name of the column to store the user identifier in the preferences table.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PreferencesTableUserID'} =  'user_id';`

2.41.3. PreferencesView

Description:	Sets the display order of the different items in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'PreferencesView'} =  [
  'User Profile',
  'Email Settings',
  'Other Settings'
];
```

2.41.4. PreferencesGroups###Password

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'PreferencesGroups'}->{'Password'} =  {
  'Active' => '1',
  'Area' => 'Agent',
  'Column' => 'User Profile',
  'Label' => 'Change password',
  'Module' => 'Kernel::Output::HTML::PreferencesPassword',
  'PasswordMaxLoginFailed' => '0',
  'PasswordMin2Characters' => '0',
  'PasswordMin2Lower2UpperCharacters' => '0',
  'PasswordMinSize' => '0',
  'PasswordNeedDigit' => '0',
  'PasswordRegExp' => '',
  'Prio' => '0500'
};
```

2.41.5. PreferencesGroups###SpellDict

Description:	Defines the config parameters of this item, to be shown in the preferences view. Take care to maintain the dictionaries installed in the system in the data section.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'PreferencesGroups'}->{'SpellDict'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Data' => {
    'deutsch' => 'Deutsch',
    'english' => 'English'
  },
  'DataSelected' => 'english',
  'Key' => 'Default spelling dictionary',
  'Label' => 'Spelling Dictionary',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserSpellDict',
  'Prio' => '2000'
};
```

2.41.6. PreferencesGroups###Comment

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'PreferencesGroups'}->{'Comment'} =  {
  'Active' => '0',
  'Block' => 'Input',
  'Column' => 'Other Settings',
  'Data' => '$Env{"UserComment"}',
  'Key' => 'Comment',
  'Label' => 'Comment',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserComment',
  'Prio' => '6000'
};
```

2.41.7. PreferencesGroups###Language

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'PreferencesGroups'}->{'Language'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Key' => 'Frontend language',
  'Label' => 'Language',
  'Module' => 'Kernel::Output::HTML::PreferencesLanguage',
  'PrefKey' => 'UserLanguage',
  'Prio' => '1000'
};
```

2.41.8. PreferencesGroups###Skin

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'PreferencesGroups'}->{'Skin'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Key' => 'Wear this frontend skin',
  'Label' => 'Skin',
  'Module' => 'Kernel::Output::HTML::PreferencesSkin',
  'PrefKey' => 'UserSkin',
  'Prio' => '2000'
};
```

2.41.9. PreferencesGroups###Theme

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'PreferencesGroups'}->{'Theme'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Key' => 'Frontend theme',
  'Label' => 'Theme',
  'Module' => 'Kernel::Output::HTML::PreferencesTheme',
  'PrefKey' => 'UserTheme',
  'Prio' => '3000'
};
```

2.41.10. PreferencesGroups###OutOfOffice

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'PreferencesGroups'}->{'OutOfOffice'} =  {
  'Active' => '1',
  'Block' => 'OutOfOffice',
  'Column' => 'User Profile',
  'Key' => '',
  'Label' => 'Out Of Office Time',
  'Module' => 'Kernel::Output::HTML::PreferencesOutOfOffice',
  'PrefKey' => 'UserOutOfOffice',
  'Prio' => '4000'
};
```

2.41.11. PreferencesGroups###TimeZone

Description:	Defines the config parameters of this item, to be shown in the preferences view.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'PreferencesGroups'}->{'TimeZone'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Key' => 'Time Zone',
  'Label' => 'Time Zone',
  'Module' => 'Kernel::Output::HTML::PreferencesTimeZone',
  'PrefKey' => 'UserTimeZone',
  'Prio' => '5000'
};
```

2.41.12. PreferencesGroups###CSVSeparator

Description:	Gives end users the possibility to override the separator character for CSV files, defined in the translation files.
Group:	Framework
SubGroup:	Frontend::Agent::Preferences
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'PreferencesGroups'}->{'CSVSeparator'} =  {
  'Active' => '1',
  'Column' => 'Other Settings',
  'Data' => {
    '' => '',
    ',' => ',',
    ';' => ';',
    '\\t' => 'tab',
    '|' => '|'
  },
  'DataSelected' => '0',
  'Desc' => 'Select the separator character used in CSV files (stats
and searches). If you don\'t select a separator here, the default
separator for your language will be used.',
  'Key' => 'CSV Separator',
  'Label' => 'CSV Separator',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserCSVSeparator',
  'Prio' => '4000'
};
```

2.42. Frontend::Agent::SearchRouter

2.42.1. Frontend::SearchDefault

Description:	Search backend default router.
Group:	Framework
SubGroup:	Frontend::Agent::SearchRouter
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::SearchDefault'} =
'Action=AgentTicketSearch;Subaction=AJAX';
```

2.43. Frontend::Agent::Stats

2.43.1. Stats::SearchPageShown

Description:	Defines the default maximum number of search results shown on the overview page.

Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::SearchPageShown'} =  '20';`

2.43.2. Stats::DefaultSelectedDynamicObject

Description:	Defines the default selection at the drop down menu for dynamic objects (Form: Common Specification).
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::DefaultSelectedDynamicObject'} =  'Ticket';`

2.43.3. Stats::DefaultSelectedPermissions

Description:	Defines the default selection at the drop down menu for permissions (Form: Common Specification).
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Stats::DefaultSelectedPermissions'} =  [
  'stats'
];
```

2.43.4. Stats::DefaultSelectedFormat

Description:	Defines the default selection at the drop down menu for stats format (Form: Common Specification). Please insert the format key (see Stats::Format).
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Stats::DefaultSelectedFormat'} =  [
  'Print',
  'CSV'
];
```

2.43.5. Stats::SearchLimit

Description:	Defines the search limit for the stats.
Group:	Framework
SubGroup:	Frontend::Agent::Stats

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::SearchLimit'} =  '500';`

2.43.6. Stats::Format

Description:	Defines all the possible stats output formats.
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Stats::Format'} =  {
  'CSV' => 'CSV',
  'GD::Graph::area' => 'graph-area',
  'GD::Graph::bars' => 'graph-bars',
  'GD::Graph::hbars' => 'graph-hbars',
  'GD::Graph::lines' => 'graph-lines',
  'GD::Graph::linespoints' => 'graph-lines-points',
  'GD::Graph::pie' => 'graph-pie',
  'GD::Graph::points' => 'graph-points',
  'Print' => 'Print'
};
```

2.43.7. Stats::GraphSize

Description:	Sets the size of the statistic graph.
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Stats::GraphSize'} =  {
  '1200x800' => '1200x800',
  '1600x1200' => '1600x1200',
  '800x600' => '800x600'
};
```

2.43.8. Stats::TimeType

Description:	Sets the time type which should be shown.
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Stats::TimeType'} =  'Extended';`

2.43.9. Stats::ExchangeAxis

Description:	Allows agents to exchange the axis of a stat if they generate one.
Group:	Framework
SubGroup:	Frontend::Agent::Stats

Valid:	0
Required:	0
Config-Setting:	`$Self->{'Stats::ExchangeAxis'} =  '0';`

2.43.10. Stats::UseAgentElementInStats

Description:	Allows agents to generate individual-related stats.
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Stats::UseAgentElementInStats'} =  '0';`

2.43.11. Stats::CustomerIDAsMultiSelect

Description:	Shows all the customer identifiers in a multi-select field (not useful if you have a lot of customer identifiers).
Group:	Framework
SubGroup:	Frontend::Agent::Stats
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Stats::CustomerIDAsMultiSelect'} =  '1';`

2.44. Frontend::Customer

2.44.1. CustomerHeadline

Description:	The headline shown in the customer interface.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerHeadline'} =  'Example Company Support';`

2.44.2. CustomerLogo

Description:	The logo shown in the header of the customer interface. The URL to the image can be a relative URL to the skin image directory, or a full URL to a remote web server.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'CustomerLogo'} =  {
  'StyleHeight' => '50px',
  'StyleRight' => '25px',
  'StyleTop' => '2px',
  'StyleWidth' => '135px',
  'URL' => 'skins/Customer/default/img/logo.png'
};
```

2.44.3. CustomerPanelUserID

Description:	Defines the user identifier for the customer panel.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelUserID'} =  '1';`

2.44.4. CustomerGroupSupport

Description:	Activates support for customer groups.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerGroupSupport'} =  '0';`

2.44.5. CustomerGroupAlwaysGroups

Description:	Defines the groups every customer user will be in (if CustomerGroupSupport is enabled and you don't want to manage every user for these groups).
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'CustomerGroupAlwaysGroups'} =  [
  'users'
];
```

2.44.6. CustomerPanelLoginURL

Description:	Defines an alternate login URL for the customer panel..
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'CustomerPanelLoginURL'} =  'http://host.example.com/cgi-bin/
login.pl';
```

2.44.7. CustomerPanelLogoutURL

Description:	Defines an alternate logout URL for the customer panel.

Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'CustomerPanelLogoutURL'} =  'http://host.example.com/cgi-bin/login.pl';
```

2.44.8. Frontend::CustomerUser::Item###1-GoogleMaps

Description:	Defines a customer item, which generates a google maps icon at the end of a customer info block.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::CustomerUser::Item'}->{'1-GoogleMaps'} =  {
  'Attributes' => 'UserStreet;UserCity;UserCountry;',
  'CSS' => 'Core.Agent.CustomerUser.GoogleMaps.css',
  'CSSClass' => 'GoogleMaps',
  'Module' => 'Kernel::Output::HTML::CustomerUserGeneric',
  'Required' => 'UserStreet;UserCity;',
  'Target' => '_blank',
  'Text' => 'Location',
  'URL' => 'http://maps.google.com/maps?z=7&q='
};
```

2.44.9. Frontend::CustomerUser::Item###2-Google

Description:	Defines a customer item, which generates a google icon at the end of a customer info block.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::CustomerUser::Item'}->{'2-Google'} =  {
  'Attributes' => 'UserFirstname;UserLastname;',
  'CSS' => 'Core.Agent.CustomerUser.Google.css',
  'CSSClass' => 'Google',
  'Module' => 'Kernel::Output::HTML::CustomerUserGeneric',
  'Required' => 'UserFirstname;UserLastname;',
  'Target' => '_blank',
  'Text' => 'Google',
  'URL' => 'http://google.com/search?q='
};
```

2.44.10. Frontend::CustomerUser::Item###2-LinkedIn

Description:	Defines a customer item, which generates a LinkedIn icon at the end of a customer info block.
Group:	Framework
SubGroup:	Frontend::Customer

Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Frontend::CustomerUser::Item'}->{'2-LinkedIn'} =  {
  'Attributes' => 'UserFirstname;UserLastname;',
  'CSS' => 'Core.Agent.CustomerUser.LinkedIn.css',
  'CSSClass' => 'LinkedIn',
  'Module' => 'Kernel::Output::HTML::CustomerUserGeneric',
  'Required' => 'UserFirstname;UserLastname;',
  'Target' => '_blank',
  'Text' => 'LinkedIn',
  'URL' => 'http://www.linkedin.com/commonSearch?
type=people&keywords='
};
```

2.44.11. Frontend::CustomerUser::Item###3-XING

Description:	Defines a customer item, which generates a XING icon at the end of a customer info block.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Frontend::CustomerUser::Item'}->{'3-XING'} =  {
  'Attributes' => 'UserFirstname;UserLastname;',
  'CSS' => 'Core.Agent.CustomerUser.Xing.css',
  'CSSClass' => 'Xing',
  'Module' => 'Kernel::Output::HTML::CustomerUserGeneric',
  'Required' => 'UserFirstname;UserLastname;',
  'Target' => '_blank',
  'Text' => 'XING',
  'URL' => 'https://www.xing.com/app/search?op=search;keywords='
};
```

2.44.12. CustomerPanelPreApplicationModule###CustomerAccept

Description:	This module and its PreRun() function will be executed, if defined, for every request. This module is useful to check some user options or to display news about new applications.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'CustomerPanelPreApplicationModule'}->{'CustomerAccept'} =
'Kernel::Modules::CustomerAccept';
```

2.44.13. CustomerPanel::InfoKey

Description:	Defines the key to check with CustomerAccept. If this user preferences key is true, then the message is accepted by the system.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0

Required:	0
Config-Setting:	`$Self->{'CustomerPanel::InfoKey'} =  'CustomerAccept1';`

2.44.14. CustomerPanel::InfoFile

Description:	Defines the path of the shown info file, that is located under Kernel/Output/HTML/Standard/CustomerAccept.dtl.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	0
Required:	0
Config-Setting:	`$Self->{'CustomerPanel::InfoFile'} =  'CustomerAccept';`

2.44.15. CustomerPanelLostPassword

Description:	Activates lost password feature for customers.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelLostPassword'} =  '1';`

2.44.16. CustomerPanelCreateAccount

Description:	Enables customers to create their own accounts.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelCreateAccount'} =  '1';`

2.44.17. CustomerPanelSubjectLostPasswordToken

Description:	Defines the subject for notification mails sent to customers, with token about new requested password.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelSubjectLostPasswordToken'} =  'New OTRS password request';`

2.44.18. CustomerPanelBodyLostPasswordToken

Description:	Defines the body text for notification mails sent to customers, with token about new requested password (after using this link the new password will be sent).

Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'CustomerPanelBodyLostPasswordToken'} =  'Hi
 <OTRS_USERFIRSTNAME>,

You or someone impersonating you has requested to change your OTRS
password.

If you want to do this, click on this link. You will receive another
 email containing the password.

<OTRS_CONFIG_HttpType>://<OTRS_CONFIG_FQDN>/
<OTRS_CONFIG_ScriptAlias>customer.pl?
Action=CustomerLostPassword;Token=<OTRS_TOKEN>

If you did not request a new password, please ignore this email.
            ';
```

2.44.19. CustomerPanelSubjectLostPassword

Description:	Defines the subject for notification mails sent to customers, about new password.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelSubjectLostPassword'} =  'New OTRS password';`

2.44.20. CustomerPanelBodyLostPassword

Description:	Defines the body text for notification mails sent to customers, about new password (after using this link the new password will be sent).
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'CustomerPanelBodyLostPassword'} =  'Hi <OTRS_USERFIRSTNAME>,


New password: <OTRS_NEWPW>

<OTRS_CONFIG_HttpType>://<OTRS_CONFIG_FQDN>/
<OTRS_CONFIG_ScriptAlias>customer.pl
            ';
```

2.44.21. CustomerPanelSubjectNewAccount

Description:	Defines the subject for notification mails sent to customers, about new account.
Group:	Framework

SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelSubjectNewAccount'} =  'New OTRS Account!';`

2.44.22. CustomerPanelBodyNewAccount

Description:	Defines the body text for notification mails sent to customers, about new account.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'CustomerPanelBodyNewAccount'} =  'Hi <OTRS_USERFIRSTNAME>,

You or someone impersonating you has created a new OTRS account for
you.

Full name: <OTRS_USERFIRSTNAME> <OTRS_USERLASTNAME>
User name: <OTRS_USERLOGIN>
Password : <OTRS_USERPASSWORD>

You can log in via the following URL. We encourage you to change your
 password
via the Preferences button after logging in.

<OTRS_CONFIG_HttpType>://<OTRS_CONFIG_FQDN>/
<OTRS_CONFIG_ScriptAlias>customer.pl
            ';
```

2.44.23. Loader::Customer::Skin###000-default

Description:	Default skin for OTRS 3.0 interface.
Group:	Framework
SubGroup:	Frontend::Customer
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Loader::Customer::Skin'}->{'000-default'} =  {
  'Description' => 'This is the default orange - black skin for OTRS
 3.0.',
  'HomePage' => 'www.otrs.org',
  'InternalName' => 'default',
  'VisibleName' => 'Default'
};
```

2.44.24. Loader::Customer::SelectedSkin

Description:	The customer skin's InternalName which should be used in the customer interface. Please check the available skins in Frontend::Customer::Skins.
Group:	Framework
SubGroup:	Frontend::Customer

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Loader::Customer::SelectedSkin'} =  'default';`

2.45. Frontend::Customer::Auth

2.45.1. Customer::AuthModule

Description:	Defines the module to authenticate customers.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Customer::AuthModule'} = 'Kernel::System::CustomerAuth::DB';`

2.45.2. Customer::AuthModule::DB::CryptType

Description:	If "DB" was selected for Customer::AuthModule, the crypt type of passwords must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Customer::AuthModule::DB::CryptType'} =  'md5';`

2.45.3. Customer::AuthModule::DB::Table

Description:	If "DB" was selected for Customer::AuthModule, the name of the table where your customer data should be stored must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Customer::AuthModule::DB::Table'} =  'customer_user';`

2.45.4. Customer::AuthModule::DB::CustomerKey

Description:	If "DB" was selected for Customer::AuthModule, the name of the column for the CustomerKey in the customer table must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Customer::AuthModule::DB::CustomerKey'} =  'login';`

2.45.5. Customer::AuthModule::DB::CustomerPassword

Description:	If "DB" was selected for Customer::AuthModule, the column name for the CustomerPassword in the customer table must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Customer::AuthModule::DB::CustomerPassword'} =  'pw';`

2.45.6. Customer::AuthModule::DB::DSN

Description:	If "DB" was selected for Customer::AuthModule, the DSN for the connection to the customer table must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::DB::DSN'} =` `'DBI:mysql:database=customerdb;host=customerdbhost';`

2.45.7. Customer::AuthModule::DB::User

Description:	If "DB" was selected for Customer::AuthModule, a username to connect to the customer table can be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::DB::User'} =  'some_user';`

2.45.8. Customer::AuthModule::DB::Password

Description:	If "DB" was selected for Customer::AuthModule, a password to connect to the customer table can be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::DB::Password'} =  'some_password';`

2.45.9. Customer::AuthModule::DB::Type

Description:	If "DB" was selected for Customer::AuthModule, a database driver (normally autodetection is used) can be specified.
Group:	Framework

SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::DB::Type'} =  'mysql';`

2.45.10. Customer::AuthModule::HTTPBasicAuth::Replace

Description:	If "HTTPBasicAuth" was selected for Customer::AuthModule, you can specify to strip leading parts of user names (e. g. for domains like example_domain\user to user).
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::HTTPBasicAuth::Replace'} = 'example_domain\\\\';`

2.45.11. Customer::AuthModule::HTTPBasicAuth::ReplaceRegExp

Description:	If "HTTPBasicAuth" was selected for Customer::AuthModule, you can specify (by using a RegExp) to strip parts of REMOTE_USER (e. g. for to remove trailing domains). RegExp-Note, $1 will be the new Login.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::HTTPBasicAuth::ReplaceRegExp'} =  '^(.+?)@.+?$';`

2.45.12. Customer::AuthModule::LDAP::Host

Description:	If "LDAP" was selected for Customer::AuthModule, the LDAP host can be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::Host'} =  'ldap.example.com';`

2.45.13. Customer::AuthModule::LDAP::BaseDN

Description:	If "LDAP" was selected for Customer::AuthModule, the BaseDN must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0

Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::BaseDN'} =  'dc=example,dc=com';`

2.45.14. Customer::AuthModule::LDAP::UID

Description:	If "LDAP" was selected for Customer::AuthModule, the user identifier must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::UID'} =  'uid';`

2.45.15. Customer::AuthModule::LDAP::GroupDN

Description:	If "LDAP" was selected for Customer::Authmodule, you can check if the user is allowed to authenticate because he is in a posixGroup, e.g. user needs to be in a group xyz to use OTRS. Specify the group, who may access the system.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::GroupDN'} =` `'cn=otrsallow,ou=posixGroups,dc=example,dc=com';`

2.45.16. Customer::AuthModule::LDAP::AccessAttr

Description:	If "LDAP" was selected for Customer::AuthModule, you can specify access attributes here.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::AccessAttr'} =  'memberUid';`

2.45.17. Customer::AuthModule::LDAP::UserAttr

Description:	If "LDAP" was selected for Customer::AuthModule, user attributes can be specified. For LDAP posixGroups use UID, for non LDAP posixGroups use full user DN.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0

Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::UserAttr'} =  'UID';`

2.45.18. Customer::AuthModule::LDAP::SearchUserDN

Description:	If "LDAP" was selected for Customer::AuthModule and your users have only anonymous access to the LDAP tree, but you want to search through the data, you can do this with a user who has access to the LDAP directory. Specify the username for this special user here.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::SearchUserDN'} = 'cn=binduser,ou=users,dc=example,dc=com';`

2.45.19. Customer::AuthModule::LDAP::SearchUserPw

Description:	If "LDAP" was selected for Customer::AuthModule and your users have only anonymous access to the LDAP tree, but you want to search through the data, you can do this with a user who has access to the LDAP directory. Specify the password for this special user here.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::SearchUserPw'} = 'some_password';`

2.45.20. Customer::AuthModule::LDAP::AlwaysFilter

Description:	If "LDAP" was selected, you can add a filter to each LDAP query, e.g. (mail=*), (objectclass=user) or (!objectclass=computer).
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::AlwaysFilter'} =  '(!objectclass=computer)';`

2.45.21. Customer::AuthModule::LDAP::UserSuffix

Description:	If "LDAP" was selected for Customer::AuthModule and if you want to add a suffix to every customer login name, specifiy it here, e. g. you just want to write the username user but in your LDAP directory exists user@domain.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0

Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::UserSuffix'} =  '@domain.com';`

2.45.22. Customer::AuthModule::LDAP::Params

Description:	If "LDAP" was selected for Customer::AuthModule and special paramaters are needed for the Net::LDAP perl module, you can specify them here. See "perldoc Net::LDAP" for more information about the parameters.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Customer::AuthModule::LDAP::Params'} =  {
  'async' => '0',
  'port' => '389',
  'timeout' => '120',
  'version' => '3'
};
```

2.45.23. Customer::AuthModule::LDAP::Die

Description:	If "LDAP" was selected for Customer::AuthModule, you can specify if the applications will stop if e. g. a connection to a server can't be established due to network problems.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::LDAP::Die'} =  '1';`

2.45.24. Customer::AuthModule::Radius::Host

Description:	If "Radius" was selected for Customer::AuthModule, the radius host must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::Radius::Host'} =  'radiushost';`

2.45.25. Customer::AuthModule::Radius::Password

Description:	If "Radius" was selected for Customer::AuthModule, the password to authenticate to the radius host must be specified.
Group:	Framework
SubGroup:	Frontend::Customer::Auth

Valid:	0
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::Radius::Password'} =  'radiussecret';`

2.45.26. Customer::AuthModule::Radius::Die

Description:	If "Radius" was selected for Customer::AuthModule, you can specify if the applications will stop if e. g. a connection to a server can't be established due to network problems.
Group:	Framework
SubGroup:	Frontend::Customer::Auth
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Customer::AuthModule::Radius::Die'} =  '1';`

2.46. Frontend::Customer::ModuleMetaHead

2.46.1. CustomerFrontend::HeaderMetaModule###1-Refresh

Description:	Defines the module to generate html refresh headers of html sites, in the customer interface.
Group:	Framework
SubGroup:	Frontend::Customer::ModuleMetaHead
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'CustomerFrontend::HeaderMetaModule'}->{'1-Refresh'} =  {
  'Module' => 'Kernel::Output::HTML::HeaderMetaRefresh'
};
```

2.47. Frontend::Customer::ModuleNotify

2.47.1. CustomerFrontend::NotifyModule###1-ShowAgentOnline

Description:	Defines the module that shows the currently loged in agents in the customer interface.
Group:	Framework
SubGroup:	Frontend::Customer::ModuleNotify
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'CustomerFrontend::NotifyModule'}->{'1-ShowAgentOnline'} =  {
  'IdleMinutes' => '60',
  'Module' => 'Kernel::Output::HTML::NotificationAgentOnline',
  'ShowEmail' => '1'
};
```

2.47.2. CustomerFrontend::NotifyModule###1-ShowCustomerOnline

Description:	Defines the module that shows the currently loged in customers in the customer interface.

Group:	Framework
SubGroup:	Frontend::Customer::ModuleNotify
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'CustomerFrontend::NotifyModule'}->{'1-ShowCustomerOnline'} =
  {
  'Module' => 'Kernel::Output::HTML::NotificationCustomerOnline',
  'ShowEmail' => '1'
};
```

2.48. Frontend::Customer::ModuleRegistration

2.48.1. CustomerFrontend::Module###Logout

Description:	Frontend module registration for the customer interface.
Group:	Framework
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'CustomerFrontend::Module'}->{'Logout'} =  {
  'Description' => 'Logout of customer panel',
  'NavBarName' => '',
  'Title' => ''
};
```

2.48.2. CustomerFrontend::Module###CustomerPreferences

Description:	Frontend module registration for the customer interface.
Group:	Framework
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'CustomerFrontend::Module'}->{'CustomerPreferences'} =  {
  'Description' => 'Customer preferences',
  'NavBarName' => '',
  'Title' => 'Preferences'
};
```

2.48.3. CustomerFrontend::Module###CustomerAccept

Description:	Frontend module registration for the customer interface.
Group:	Framework
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'CustomerFrontend::Module'}->{'CustomerAccept'} =  {
  'Description' => 'To accept login information, such as an EULA or
license.',
  'NavBarName' => '',
  'Title' => 'Info'
};
```

2.48.4. CustomerFrontend::Module###PictureUpload

Description:	Frontend module registration for the customer interface.
Group:	Framework
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'CustomerFrontend::Module'}->{'PictureUpload'} =  {
  'Description' => 'Picture upload module',
  'NavBarName' => 'Ticket',
  'Title' => 'Picture-Upload'
};
```

2.49. Frontend::Customer::Preferences

2.49.1. PreferencesTable

Description:	Defines the name of the table, where the customer preferences are stored.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PreferencesTable'} =  'user_preferences';`

2.49.2. PreferencesTableKey

Description:	Defines the column to store the keys for the preferences table.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PreferencesTableKey'} =  'preferences_key';`

2.49.3. CustomerPreferences

Description:	Defines the parameters for the customer preferences table.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'CustomerPreferences'} =  {
  'Module' => 'Kernel::System::CustomerUser::Preferences::DB',
  'Params' => {
    'Table' => 'customer_preferences',
    'TableKey' => 'preferences_key',
    'TableUserID' => 'user_id',
    'TableValue' => 'preferences_value'
  }
};
```

2.49.4. CustomerPreferencesView

Description:	Sets the order of the different items in the customer preferences view.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'CustomerPreferencesView'} =  [
  'User Profile',
  'Other Settings'
];
```

2.49.5. CustomerPreferencesGroups###Password

Description:	Defines all the parameters for this item in the customer preferences.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerPreferencesGroups'}->{'Password'} =  {
  'Active' => '1',
  'Area' => 'Customer',
  'Column' => 'Other Settings',
  'Label' => 'Change password',
  'Module' => 'Kernel::Output::HTML::PreferencesPassword',
  'PasswordMin2Characters' => '0',
  'PasswordMin2Lower2UpperCharacters' => '0',
  'PasswordMinSize' => '0',
  'PasswordNeedDigit' => '0',
  'PasswordRegExp' => '',
  'Prio' => '1000'
};
```

2.49.6. CustomerPreferencesGroups###Language

Description:	Defines all the parameters for this item in the customer preferences.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerPreferencesGroups'}->{'Language'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Key' => 'Language',
  'Label' => 'Interface language',
  'Module' => 'Kernel::Output::HTML::PreferencesLanguage',
  'PrefKey' => 'UserLanguage',
  'Prio' => '2000'
};
```

2.49.7. CustomerPreferencesGroups###Theme

Description:	Defines all the parameters for this item in the customer preferences.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerPreferencesGroups'}->{'Theme'} =  {
  'Active' => '0',
  'Column' => 'User Profile',
  'Key' => 'Select your frontend Theme.',
  'Label' => 'Theme',
  'Module' => 'Kernel::Output::HTML::PreferencesTheme',
  'PrefKey' => 'UserTheme',
  'Prio' => '1000'
};
```

2.49.8. CustomerPreferencesGroups###TimeZone

Description:	Defines all the parameters for this item in the customer preferences.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerPreferencesGroups'}->{'TimeZone'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Key' => 'Time Zone',
  'Label' => 'Time Zone',
  'Module' => 'Kernel::Output::HTML::PreferencesTimeZone',
  'PrefKey' => 'UserTimeZone',
  'Prio' => '5000'
};
```

2.49.9. CustomerPreferencesGroups###PGP

Description:	Defines all the parameters for this item in the customer preferences.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerPreferencesGroups'}->{'PGP'} =  {
  'Active' => '1',
  'Column' => 'Other Settings',
  'Key' => 'PGP Key Upload',
  'Label' => 'PGP Key',
  'Module' => 'Kernel::Output::HTML::PreferencesPGP',
  'PrefKey' => 'UserPGPKey',
  'Prio' => '10000'
};
```

2.49.10. CustomerPreferencesGroups###SMIME

Description:	Defines all the parameters for this item in the customer preferences.
Group:	Framework
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'CustomerPreferencesGroups'}->{'SMIME'} =  {
  'Active' => '1',
  'Column' => 'Other Settings',
  'Key' => 'S/MIME Certificate Upload',
  'Label' => 'S/MIME Certificate',
  'Module' => 'Kernel::Output::HTML::PreferencesSMIME',
  'PrefKey' => 'UserSMIMEKey',
  'Prio' => '11000'
};
```

2.50. Frontend::Public

2.50.1. PublicFrontend::CommonParam###Action

Description:	Defines the default value for the action parameter for the public frontend. The action parameter is used in the scripts of the system.
Group:	Framework
SubGroup:	Frontend::Public
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'PublicFrontend::CommonParam'}->{'Action'} =
 'PublicDefault';
```

2.51. Frontend::Public::ModuleRegistration

2.51.1. PublicFrontend::Module###PublicDefault

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Public::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'PublicFrontend::Module'}->{'PublicDefault'} =  {
  'Description' => 'PublicDefault',
  'NavBarName' => '',
  'Title' => 'PublicDefault'
};
```

2.51.2. PublicFrontend::Module###PublicRepository

Description:	Frontend module registration for the agent interface.
Group:	Framework
SubGroup:	Frontend::Public::ModuleRegistration
Valid:	1

Required:	0
Config-Setting:	

```
$Self->{'PublicFrontend::Module'}->{'PublicRepository'} =  {
  'Description' => 'PublicRepository',
  'NavBarName' => '',
  'Title' => 'PublicRepository'
};
```

3. GenericInterface

3.1. Core::Ticket

3.1.1. Ticket::EventModulePost###1000-GenericInterface

Description:	Performs the configured action for each event (as an Invoker) for each configured Webservice.
Group:	GenericInterface
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::EventModulePost'}->{'1000-GenericInterface'} =  {
  'Event' => '',
  'Module' => 'Kernel::GenericInterface::Event::Handler',
  'Transaction' => '1'
};
```

3.2. Frontend::Admin::ModuleRegistration

3.2.1. Frontend::Module###AdminGenericInterfaceDebugger

Description:	Frontend module registration for the agent interface.
Group:	GenericInterface
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminGenericInterfaceDebugger'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.GenericInterface.css'
    ],
    'CSS_IE7' => [
      'Core.Agent.Admin.GenericInterface.IE7.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.GenericInterfaceDebugger.js'
    ]
  },
  'Title' => 'GenericInterface Debugger GUI'
};
```

3.2.2. Frontend::Module###AdminGenericInterfaceWebservice

Description:	Frontend module registration for the agent interface.

Group:	GenericInterface
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminGenericInterfaceWebservice'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.GenericInterface.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.GenericInterfaceWebservice.js'
    ]
  },
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'Create and manage web services.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Web Services',
    'Prio' => '1000'
  },
  'NavBarName' => 'Admin',
  'Title' => 'GenericInterface Web Service GUI'
};
```

3.2.3. Frontend::Module###AdminGenericInterfaceTransportHTTPSOAP

Description:	Frontend module registration for the agent interface.
Group:	GenericInterface
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}-
>{'AdminGenericInterfaceTransportHTTPSOAP'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.GenericInterface.css'
    ]
  },
  'Title' => 'GenericInterface TransportHTTPSOAP GUI'
};
```

3.2.4. Frontend::Module###AdminGenericInterfaceWebserviceHistory

Description:	Frontend module registration for the agent interface.
Group:	GenericInterface
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}-
>{'AdminGenericInterfaceWebserviceHistory'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.GenericInterface.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.GenericInterfaceWebserviceHistory.js'
    ]
  },
  'Title' => 'GenericInterface Webservice History GUI'
};
```

3.2.5. Frontend::Module###AdminGenericInterfaceOperationDefault

Description:	Frontend module registration for the agent interface.
Group:	GenericInterface
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}-
>{'AdminGenericInterfaceOperationDefault'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.GenericInterface.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.GenericInterfaceOperation.js'
    ]
  },
  'Title' => 'GenericInterface Operation GUI'
};
```

3.2.6. Frontend::Module###AdminGenericInterfaceInvokerDefault

Description:	Frontend module registration for the agent interface.
Group:	GenericInterface
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminGenericInterfaceInvokerDefault'}
 =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.GenericInterface.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.GenericInterfaceInvoker.js'
    ]
  },
  'Title' => 'GenericInterface Invoker GUI'
};
```

3.2.7. Frontend::Module###AdminGenericInterfaceMappingSimple

Description:	Frontend module registration for the agent interface.
Group:	GenericInterface
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminGenericInterfaceMappingSimple'} =
  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.GenericInterface.css'
    ],
    'CSS_IE7' => [
      'Core.Agent.Admin.GenericInterface.IE7.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.GenericInterfaceMappingSimple.js'
    ]
  },
  'Title' => 'GenericInterface Webservice Mapping GUI'
};
```

3.3. GenericInterface::Invoker

3.3.1. GenericInterface::Invoker::Event###Ticket

Description:	Event list to be displayed on GUI to trigger generic interface invokers.
Group:	GenericInterface
SubGroup:	GenericInterface::Invoker
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Invoker::Event'}->{'Ticket'} =  {
  'EscalationResponseTimeNotifyBefore' => '1',
  'EscalationResponseTimeStart' => '1',
  'EscalationResponseTimeStop' => '1',
  'EscalationSolutionTimeNotifyBefore' => '1',
  'EscalationSolutionTimeStart' => '1',
  'EscalationSolutionTimeStop' => '1',
  'EscalationUpdateTimeNotifyBefore' => '1',
  'EscalationUpdateTimeStart' => '1',
  'EscalationUpdateTimeStop' => '1',
  'HistoryAdd' => '1',
  'HistoryDelete' => '1',
  'TicketAccountTime' => '1',
  'TicketArchiveFlagUpdate' => '1',
  'TicketCreate' => '1',
  'TicketCustomerUpdate' => '1',
  'TicketDelete' => '1',
  'TicketFlagDelete' => '1',
  'TicketFlagSet' => '1',
  'TicketLockUpdate' => '1',
  'TicketMasterLinkDelete' => '1',
  'TicketMerge' => '1',
  'TicketOwnerUpdate' => '1',
  'TicketPendingTimeUpdate' => '1',
  'TicketPriorityUpdate' => '1',
  'TicketQueueUpdate' => '1',
  'TicketResponsibleUpdate' => '1',
  'TicketSLAUpdate' => '1',
  'TicketServiceUpdate' => '1',
  'TicketSlaveLinkAdd' => '1',
  'TicketSlaveLinkDelete' => '1',
  'TicketStateUpdate' => '1',
  'TicketSubscribe' => '1',
  'TicketTitleUpdate' => '1',
  'TicketTypeUpdate' => '1',
  'TicketUnlockTimeoutUpdate' => '1',
  'TicketUnsubscribe' => '1'
};
```

3.3.2. GenericInterface::Invoker::Event###Article

Description:	Event list to be displayed on GUI to trigger generic interface invokers.
Group:	GenericInterface
SubGroup:	GenericInterface::Invoker
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Invoker::Event'}->{'Article'} =  {
  'ArticleAgentNotification' => '1',
  'ArticleAutoResponse' => '1',
  'ArticleBounce' => '1',
  'ArticleCreate' => '1',
  'ArticleCustomerNotification' => '1',
  'ArticleFlagDelete' => '1',
  'ArticleFlagSet' => '1',
  'ArticleSend' => '1',
  'ArticleUpdate' => '1'
};
```

3.4. GenericInterface::Invoker::ModuleRegistration

3.4.1. GenericInterface::Invoker::Module###Test::Test

Description:	GenericInterface module registration for the invoker layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Invoker::ModuleRegistration
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Invoker::Module'}->{'Test::Test'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceInvokerDefault',
  'Controller' => 'Test',
  'Name' => 'Test'
};
```

3.4.2. GenericInterface::Invoker::Module###Test::TestSimple

Description:	GenericInterface module registration for the invoker layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Invoker::ModuleRegistration
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'GenericInterface::Invoker::Module'}->{'Test::TestSimple'} =
 {
  'ConfigDialog' => 'AdminGenericInterfaceInvokerDefault',
  'Controller' => 'Test',
  'Name' => 'TestSimple'
};
```

3.5. GenericInterface::Mapping::ModuleRegistration

3.5.1. GenericInterface::Mapping::Module###Test

Description:	GenericInterface module registration for the mapping layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Mapping::ModuleRegistration
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'GenericInterface::Mapping::Module'}->{'Test'} =  {
  'ConfigDialog' => ''
};
```

3.5.2. GenericInterface::Mapping::Module###Simple

Description:	GenericInterface module registration for the mapping layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Mapping::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'GenericInterface::Mapping::Module'}->{'Simple'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceMappingSimple'
};
```

3.6. GenericInterface::Operation::ModuleRegistration

3.6.1. GenericInterface::Operation::Module###Test::Test

Description:	GenericInterface module registration for the operation layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::ModuleRegistration
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'GenericInterface::Operation::Module'}->{'Test::Test'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceOperationDefault',
  'Controller' => 'Test',
  'Name' => 'Test'
};
```

3.6.2. GenericInterface::Operation::Module###Session::SessionCreate

Description:	GenericInterface module registration for the operation layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Operation::Module'}-
>{'Session::SessionCreate'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceOperationDefault',
  'Controller' => 'Session',
  'Name' => 'SessionCreate'
};
```

3.6.3. GenericInterface::Operation::Module###Ticket::TicketCreate

Description:	GenericInterface module registration for the operation layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Operation::Module'}-
>{'Ticket::TicketCreate'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceOperationDefault',
  'Controller' => 'Ticket',
  'Name' => 'TicketCreate'
};
```

3.6.4. GenericInterface::Operation::Module###Ticket::TicketUpdate

Description:	GenericInterface module registration for the operation layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Operation::Module'}-
>{'Ticket::TicketUpdate'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceOperationDefault',
  'Controller' => 'Ticket',
  'Name' => 'TicketUpdate'
};
```

3.6.5. GenericInterface::Operation::Module###Ticket::TicketGet

Description:	GenericInterface module registration for the operation layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::ModuleRegistration

Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Operation::Module'}->{'Ticket::TicketGet'}
 =  {
  'ConfigDialog' => 'AdminGenericInterfaceOperationDefault',
  'Controller' => 'Ticket',
  'Name' => 'TicketGet'
};
```

3.6.6. GenericInterface::Operation::Module###Ticket::TicketSearch

Description:	GenericInterface module registration for the operation layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Operation::Module'}-
>{'Ticket::TicketSearch'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceOperationDefault',
  'Controller' => 'Ticket',
  'Name' => 'TicketGet'
};
```

3.7. GenericInterface::Operation::TicketCreate

3.7.1. GenericInterface::Operation::TicketCreate###ArticleType

Description:	Defines the default type of the article for this operation.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketCreate
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'GenericInterface::Operation::TicketCreate'}->{'ArticleType'}
 =  'webrequest';
```

3.7.2. GenericInterface::Operation::TicketCreate###HistoryType

Description:	Defines the history type for this operation, which gets used for ticket history in the agent interface.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketCreate
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'GenericInterface::Operation::TicketCreate'}->{'HistoryType'}
 =  'NewTicket';
```

3.7.3. GenericInterface::Operation::TicketCreate###HistoryComment

Description:	Defines the history comment for this operation, which gets used for ticket history in the agent interface.

Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketCreate
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::TicketCreate'}->{'HistoryComment'} =  '%%GenericInterface Create';`

3.7.4. GenericInterface::Operation::TicketCreate###AutoResponseType

Description:	Defines the default auto response type of the article for this operation.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketCreate
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::TicketCreate'}->{'AutoResponseType'} =  'auto reply';`

3.8. GenericInterface::Operation::TicketUpdate

3.8.1. GenericInterface::Operation::TicketUpdate###ArticleType

Description:	Defines the default type of the article for this operation.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketUpdate
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::TicketUpdate'}->{'ArticleType'} =  'webrequest';`

3.8.2. GenericInterface::Operation::TicketUpdate###HistoryType

Description:	Defines the history type for this operation, which gets used for ticket history in the agent interface.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketUpdate
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::TicketUpdate'}->{'HistoryType'} =  'AddNote';`

3.8.3. GenericInterface::Operation::TicketUpdate###HistoryComment

Description:	Defines the history comment for this operation, which gets used for ticket history in the agent interface.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketUpdate

Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::TicketUpdate'}->{'HistoryComment'} =  '%%GenericInterface Note';`

3.8.4. GenericInterface::Operation::TicketUpdate###AutoResponseType

Description:	Defines the default auto response type of the article for this operation.
Group:	GenericInterface
SubGroup:	GenericInterface::Operation::TicketUpdate
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::TicketUpdate'}->{'AutoResponseType'} =  'auto follow up';`

3.9. GenericInterface::Transport::ModuleRegistration

3.9.1. GenericInterface::Transport::Module###HTTP::SOAP

Description:	GenericInterface module registration for the transport layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Transport::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Transport::Module'}->{'HTTP::SOAP'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceTransportHTTPSOAP',
  'Name' => 'SOAP',
  'Protocol' => 'HTTP'
};
```

3.9.2. GenericInterface::Transport::Module###HTTP::Test

Description:	GenericInterface module registration for the transport layer.
Group:	GenericInterface
SubGroup:	GenericInterface::Transport::ModuleRegistration
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'GenericInterface::Transport::Module'}->{'HTTP::Test'} =  {
  'ConfigDialog' => 'AdminGenericInterfaceTransportHTTPTest',
  'Name' => 'Test',
  'Protocol' => 'HTTP'
};
```

3.10. GenericInterface::Webservice

3.10.1. GenericInterface::WebserviceConfig::CacheTTL

Description:	Cache time in seconds for the web service config backend.
Group:	GenericInterface

SubGroup:	GenericInterface::Webservice
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::WebserviceConfig::CacheTTL'} =  '86400';`

3.10.2. GenericInterface::Operation::Common::CachedAuth::AgentCacheTTL

Description:	Cache time in seconds for agent authentication in the GenericInterface.
Group:	GenericInterface
SubGroup:	GenericInterface::Webservice
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::Common::CachedAuth::AgentCacheTTL'} =  '300';`

3.10.3. GenericInterface::Operation::Common::CachedAuth::CustomerCacheTTL

Description:	Cache time in seconds for customer authentication in the GenericInterface.
Group:	GenericInterface
SubGroup:	GenericInterface::Webservice
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Operation::Common::CachedAuth::CustomerCacheTTL'} =  '300';`

3.10.4. GenericInterface::Webservice::Path::Separator

Description:	Webservice path separator.
Group:	GenericInterface
SubGroup:	GenericInterface::Webservice
Valid:	1
Required:	1
Config-Setting:	`$Self->{'GenericInterface::Webservice::Path::Separator'} =  '»';`

4. ProcessManagement

4.1. Core

4.1.1. Process::DynamicFieldProcessManagementProcessID

Description:	This option defines the dynamic field in which a Process Management process entity id is stored.
Group:	ProcessManagement
SubGroup:	Core

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::DynamicFieldProcessManagementProcessID'} = 'ProcessManagementProcessID';`

4.1.2. Process::DynamicFieldProcessManagementActivityID

Description:	This option defines the dynamic field in which a Process Management activity entity id is stored.
Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::DynamicFieldProcessManagementActivityID'} = 'ProcessManagementActivityID';`

4.1.3. Process::DefaultQueue

Description:	This option defines the process tickets default queue.
Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::DefaultQueue'} =  'Raw';`

4.1.4. Process::DefaultState

Description:	This option defines the process tickets default state.
Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::DefaultState'} =  'new';`

4.1.5. Process::DefaultLock

Description:	This option defines the process tickets default lock.
Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::DefaultLock'} =  'unlock';`

4.1.6. Process::DefaultPriority

Description:	This option defines the process tickets default priority.

Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::DefaultPriority'} =  '3 normal';`

4.1.7. Process::Entity::Prefix

Description:	Default ProcessManagement entity prefixes for entity IDs that are automatically generated.
Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Process::Entity::Prefix'} =  {
  'Activity' => 'A',
  'ActivityDialog' => 'AD',
  'Process' => 'P',
  'Transition' => 'T',
  'TransitionAction' => 'TA'
};
```

4.1.8. Process::CacheTTL

Description:	Cache time in seconds for the DB process backend.
Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::CacheTTL'} =  '3600';`

4.1.9. Process::NavBarOutput::CacheTTL

Description:	Cache time in seconds for the ticket process navigation bar output module.
Group:	ProcessManagement
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Process::NavBarOutput::CacheTTL'} =  '900';`

4.2. Core::Ticket

4.2.1. Ticket::EventModulePost###TicketProcessTransitions

Description:	Event module registration. For more performance you can define a trigger event (e. g. Event => TicketCreate).

Group:	ProcessManagement
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::EventModulePost'}->{'TicketProcessTransitions'} =  {
  'Event' => '',
  'Module' =>
 'Kernel::System::Ticket::Event::TicketProcessTransitions',
  'Transaction' => '1'
};
```

4.3. Frontend::Admin::ModuleRegistration

4.3.1. Frontend::Module###AdminProcessManagement

Description:	Frontend module registration for the agent interface.
Group:	ProcessManagement
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminProcessManagement'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.ProcessManagement.css',
      'Core.AllocationList.css'
    ],
    'JavaScript' => [
      'thirdparty/jsplumb-1.4.2/jsplumb.js',
      'Core.Agent.Admin.ProcessManagement.js',
      'Core.Agent.Admin.ProcessManagement.Canvas.js',
      'Core.UI.AllocationList.js'
    ]
  },
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'Configure Processes.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Process Management',
    'Prio' => '750'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Process Management'
};
```

4.3.2. Frontend::Module###AdminProcessManagementActivity

Description:	Frontend module registration for the agent interface.
Group:	ProcessManagement
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminProcessManagementActivity'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.ProcessManagement.css',
      'Core.AllocationList.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.ProcessManagement.js',
      'Core.UI.AllocationList.js'
    ]
  },
  'Title' => 'Process Management Activity GUI'
};
```

4.3.3. Frontend::Module###AdminProcessManagementActivityDialog

Description:	Frontend module registration for the agent interface.
Group:	ProcessManagement
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminProcessManagementActivityDialog'}
 =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.ProcessManagement.css',
      'Core.AllocationList.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.ProcessManagement.js',
      'Core.UI.AllocationList.js'
    ]
  },
  'Title' => 'Process Management Activity Dialog GUI'
};
```

4.3.4. Frontend::Module###AdminProcessManagementTransition

Description:	Frontend module registration for the agent interface.
Group:	ProcessManagement
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminProcessManagementTransition'} =
{
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.ProcessManagement.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.ProcessManagement.js'
    ]
  },
  'Title' => 'Process Management Transition GUI'
};
```

4.3.5. Frontend::Module###AdminProcessManagementTransitionAction

Description:	Frontend module registration for the agent interface.
Group:	ProcessManagement
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}-
>{'AdminProcessManagementTransitionAction'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.ProcessManagement.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.ProcessManagement.js'
    ]
  },
  'Title' => 'Process Management Transition Action GUI'
};
```

4.3.6. Frontend::Module###AdminProcessManagementPath

Description:	Frontend module registration for the agent interface.
Group:	ProcessManagement
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminProcessManagementPath'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Loader' => {
    'CSS' => [
      'Core.Agent.Admin.ProcessManagement.css',
      'Core.AllocationList.css'
    ],
    'JavaScript' => [
      'Core.Agent.Admin.ProcessManagement.js',
      'Core.UI.AllocationList.js'
    ]
  },
  'Title' => 'Process Management Path GUI'
};
```

4.4. Frontend::Agent::ModuleRegistration

4.4.1. Frontend::Module###AgentTicketProcess

Description:	Frontend module registration for the agent interface.
Group:	ProcessManagement
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AgentTicketProcess'} =  {
  'Description' => 'Create new process ticket',
  'Loader' => {
    'CSS' => [
      'Core.Agent.TicketProcess.css'
    ],
    'JavaScript' => [
      'Core.Agent.CustomerSearch.js',
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'p',
      'Block' => '',
      'Description' => 'Create New process ticket',
      'Link' => 'Action=AgentTicketProcess',
      'LinkOption' => '',
      'Name' => 'New process ticket',
      'NavBar' => 'Ticket',
      'Prio' => '220',
      'Type' => ''
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'New process ticket'
};
```

4.5. Frontend::Agent::NavBarModule

4.5.1. Frontend::NavBarOutputModule###1-TicketProcesses

Description:	Frontend module registration (disable ticket processes screen if no process available).
Group:	ProcessManagement
SubGroup:	Frontend::Agent::NavBarModule
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::NavBarOutputModule'}->{'1-TicketProcesses'} =  {
  'Module' =>
 'Kernel::Output::HTML::NavBarOutputModuleAgentTicketProcess'
};
```

4.6. Frontend::Agent::Ticket::ViewProcess

4.6.1. Ticket::Frontend::AgentTicketProcess###StateType

Description:	Determines the next possible ticket states, for process tickets in the agent interface.

Group:	ProcessManagement
SubGroup:	Frontend::Agent::Ticket::ViewProcess
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketProcess'}->{'StateType'} =  [
  'new',
  'open',
  'pending auto',
  'pending reminder',
  'closed'
];
```

4.7. Frontend::Agent::Ticket::ViewZoom

4.7.1. Ticket::Frontend::AgentTicketZoom###ProcessDisplay

Description:	Display settings to override defaults for Process Tickets.
Group:	ProcessManagement
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketZoom'}->{'ProcessDisplay'} =  {
  'NavBarName' => 'Processes',
  'WidgetTitle' => 'Process Information'
};
```

4.7.2. Ticket::Frontend::AgentTicketZoom###ProcessWidgetDynamicFieldGroups

Description:	Dynamic fields groups for process widget. The key is the name of the group, the value contains the fields to be shown. Example: 'Key => My Group', 'Content: Name_X, NameY'.
Group:	ProcessManagement
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketZoom'}-
>{'ProcessWidgetDynamicFieldGroups'} =  {};
```

4.7.3. Ticket::Frontend::AgentTicketZoom###ProcessWidgetDynamicField

Description:	Dynamic fields shown in the process widget in ticket zoom screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	ProcessManagement
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketZoom'}-
>{'ProcessWidgetDynamicField'} =  {};
```

4.8. Frontend::Customer::ModuleRegistration

4.8.1. CustomerFrontend::Module###CustomerTicketProcess

Description:	Frontend module registration for the customer interface.
Group:	ProcessManagement
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'CustomerFrontend::Module'}->{'CustomerTicketProcess'} =  {
  'Description' => 'Process Ticket',
  'Loader' => {
    'CSS' => [
      'Core.Customer.TicketProcess.css'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Process'
};
```

5. Scheduler

5.1. Core

5.1.1. Scheduler::SleepTime

Description:	Defines scheduler sleep time in seconds after processing all available tasks (floating point number).
Group:	Scheduler
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Scheduler::SleepTime'} =  '1.0';`

5.1.2. Scheduler::PIDUpdateTime

Description:	Defines scheduler PID update time in seconds (floating point number).
Group:	Scheduler
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Scheduler::PIDUpdateTime'} =  '60.0';`

5.1.3. Scheduler::RestartAfterSeconds

Description:	Defines the time in seconds after which the Scheduler performs an automatic self-restart.
Group:	Scheduler
SubGroup:	Core

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Scheduler::RestartAfterSeconds'} =  '86400';`

5.1.4. Scheduler::TaskDataLength

Description:	Defines the maximum length (in characters) for a scheduler task data. WARNING: Do not modify this setting unless you are sure of the current Database length for 'task_data' filed from 'scheduler_data_list' table.
Group:	Scheduler
SubGroup:	Core
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Scheduler::TaskDataLength'} =  '8000';`

5.2. Core::Log

5.2.1. Scheduler::LogPath

Description:	Defines the path for scheduler to store its console output (SchedulerOUT.log and SchedulerERR.log).
Group:	Scheduler
SubGroup:	Core::Log
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Scheduler::LogPath'} =  '<OTRS_CONFIG_Home>/var/log';`

5.2.2. Scheduler::Log::DaysToKeep

Description:	Defines the time in days to keep log backup files.
Group:	Scheduler
SubGroup:	Core::Log
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Scheduler::Log::DaysToKeep'} =  '10';`

5.3. Core::Web

5.3.1. Loader::Agent::CommonJS###000-GenericInterface

Description:	List of JS files to always be loaded for the agent interface.
Group:	Scheduler
SubGroup:	Core::Web
Valid:	1
Required:	1

Config-Setting:	

```
 $Self->{'Loader::Agent::CommonJS'}->{'000-GenericInterface'} =  [
  'Core.Agent.Admin.Scheduler.js'
];
```

5.4. Frontend::Admin::ModuleRegistration

5.4.1. Frontend::Module###AdminScheduler

Description:	Frontend module registration for the agent interface.
Group:	Scheduler
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminScheduler'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'Title' => 'GenericInterface Webservice Mapping GUI'
};
```

5.5. Frontend::Agent::ModuleNotify

5.5.1. Frontend::NotifyModule###800-Scheduler-Check

Description:	Defines the module to display a notification in the agent interface, (only for agents on the admin group) if the scheduler is not running.
Group:	Scheduler
SubGroup:	Frontend::Agent::ModuleNotify
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::NotifyModule'}->{'800-Scheduler-Check'} =  {
  'Module' => 'Kernel::Output::HTML::NotificationSchedulerCheck',
  'NotifyGroups' => {
    'admin' => '1',
    'users' => '0'
  }
};
```

6. Ticket

6.1. Core

6.1.1. OTRSEscalationEvents::DecayTime

Description:	The duration in minutes after emitting an event, in which the new escalation notify and start events are suppressed.
Group:	Ticket
SubGroup:	Core
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'OTRSEscalationEvents::DecayTime'} =  '1440';
```

6.2. Core::FulltextSearch

6.2.1. Ticket::SearchIndexModule

Description:	Helps to extend your articles full-text search (From, To, Cc, Subject and Body search). Runtime will do full-text searches on live data (it works fine for up to 50.000 tickets). StaticDB will strip all articles and will build an index after article creation, increasing fulltext searches about 50%. To create an initial index use "bin/otrs.RebuildFulltextIndex.pl".
Group:	Ticket
SubGroup:	Core::FulltextSearch
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::SearchIndexModule'} =
 'Kernel::System::Ticket::ArticleSearchIndex::RuntimeDB';
```

6.2.2. Ticket::SearchIndex::Attribute

Description:	Basic fulltext index settings. Execute "bin/otrs.RebuildFulltextIndex.pl" in order to generate a new index.
Group:	Ticket
SubGroup:	Core::FulltextSearch
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::SearchIndex::Attribute'} =  {
  'WordCountMax' => '1000',
  'WordLengthMax' => '30',
  'WordLengthMin' => '3'
};
```

6.2.3. Ticket::SearchIndex::Filters

Description:	Fulltext index regex filters to remove parts of the text.
Group:	Ticket
SubGroup:	Core::FulltextSearch
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::SearchIndex::Filters'} =  [
  '[,\\&\\<\\>\\?"\\!\\*\\|;\\[\\]\\(\\)\\+\\$\\^=]',
  '^[\':.]|[\':.]$'
];
```

6.2.4. Ticket::SearchIndex::StopWords

Description:	Stop words for fulltext index. These words will be removed.
Group:	Ticket
SubGroup:	Core::FulltextSearch
Valid:	1

Required:	1

Config-Setting:

```
 $Self->{'Ticket::SearchIndex::StopWords'} =  {
  'a' => '1',
  'als' => '1',
  'and' => '1',
  'auf' => '1',
  'das' => '1',
  'den' => '1',
  'der' => '1',
  'die' => '1',
  'for' => '1',
  'im' => '1',
  'in' => '1',
  'is' => '1',
  'of' => '1',
  'the' => '1',
  'to' => '1',
  'und' => '1',
  'vom' => '1',
  'zu' => '1'
};
```

6.2.5. Ticket::EventModulePost###98-ArticleSearchIndex

Description:	Builds an article index right after the article's creation.
Group:	Ticket
SubGroup:	Core::FulltextSearch
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'98-ArticleSearchIndex'} =  {
  'Event' => '(ArticleCreate|ArticleUpdate)',
  'Module' => 'Kernel::System::Ticket::Event::ArticleSearchIndex'
};
```

6.3. Core::LinkObject

6.3.1. LinkObject::PossibleLink###0200

Description:	Links 2 tickets with a "Normal" type link.
Group:	Ticket
SubGroup:	Core::LinkObject
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'LinkObject::PossibleLink'}->{'0200'} =  {
  'Object1' => 'Ticket',
  'Object2' => 'Ticket',
  'Type' => 'Normal'
};
```

6.3.2. LinkObject::PossibleLink###0201

Description:	Links 2 tickets with a "ParentChild" type link.
Group:	Ticket
SubGroup:	Core::LinkObject
Valid:	1

Required:	0
Config-Setting:	

```
$Self->{'LinkObject::PossibleLink'}->{'0201'} =  {
  'Object1' => 'Ticket',
  'Object2' => 'Ticket',
  'Type' => 'ParentChild'
};
```

6.4. Core::PostMaster

6.4.1. PostmasterMaxEmails

Description:	Maximal auto email responses to own email-address a day (Loop-Protection).
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterMaxEmails'} =  '40';`

6.4.2. PostMasterMaxEmailSize

Description:	Maximal size in KBytes for mails that can be fetched via POP3/POP3S/IMAP/IMAPS (KBytes).
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostMasterMaxEmailSize'} =  '16384';`

6.4.3. PostMasterReconnectMessage

Description:	The "bin/PostMasterMailAccount.pl" will reconnect to POP3/POP3S/IMAP/IMAPS host after the specified count of messages.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostMasterReconnectMessage'} =  '20';`

6.4.4. LoopProtectionModule

Description:	Default loop protection module.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'LoopProtectionModule'} =
'Kernel::System::PostMaster::LoopProtection::DB';
```

6.4.5. LoopProtectionLog

Description:	Path for the log file (it only applies if "FS" was selected for LoopProtectionModule and it is mandatory).
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'LoopProtectionLog'} =  '<OTRS_CONFIG_Home>/var/log/LoopProtection';`

6.4.6. PostmasterAutoHTML2Text

Description:	Converts HTML mails into text messages.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterAutoHTML2Text'} =  '1';`

6.4.7. PostmasterFollowUpSearchInReferences

Description:	Executes follow up checks on In-Reply-To or References headers for mails that don't have a ticket number in the subject.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterFollowUpSearchInReferences'} =  '0';`

6.4.8. PostmasterFollowUpSearchInBody

Description:	Executes follow up mail body checks in mails that don't have a ticket number in the subject.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterFollowUpSearchInBody'} =  '0';`

6.4.9. PostmasterFollowUpSearchInAttachment

Description:	Executes follow up mail attachments checks in mails that don't have a ticket number in the subject.
Group:	Ticket
SubGroup:	Core::PostMaster

Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterFollowUpSearchInAttachment'} =  '0';`

6.4.10. PostmasterFollowUpSearchInRaw

Description:	Executes follow up plain/raw mail checks in mails that don't have a ticket number in the subject.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterFollowUpSearchInRaw'} =  '0';`

6.4.11. PostmasterUserID

Description:	Specifies user id of the postmaster data base.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterUserID'} =  '1';`

6.4.12. PostmasterDefaultQueue

Description:	Defines the postmaster default queue.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterDefaultQueue'} =  'Raw';`

6.4.13. PostmasterDefaultPriority

Description:	Defines the default priority of new tickets.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterDefaultPriority'} =  '3 normal';`

6.4.14. PostmasterDefaultState

Description:	Defines the default state of new tickets.
Group:	Ticket

SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterDefaultState'} =  'new';`

6.4.15. PostmasterFollowUpState

Description:	Defines the state of a ticket if it gets a follow-up.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterFollowUpState'} =  'open';`

6.4.16. PostmasterFollowUpStateClosed

Description:	Defines the state of a ticket if it gets a follow-up and the ticket was already closed.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	0
Required:	0
Config-Setting:	`$Self->{'PostmasterFollowUpStateClosed'} =  'open';`

6.4.17. PostmasterFollowUpOnUnlockAgentNotifyOnlyToOwner

Description:	Sends agent follow-up notification only to the owner, if a ticket is unlocked (the default is to send the notification to all agents).
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	`$Self->{'PostmasterFollowUpOnUnlockAgentNotifyOnlyToOwner'} =  '0';`

6.4.18. PostmasterX-Header

Description:	Defines all the X-headers that should be scanned.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'PostmasterX-Header'} =  [
  'From',
  'To',
  'Cc',
    'Reply-To',
    'ReplyTo',
    'Subject',
    'Message-ID',
    'Message-Id',
    'Resent-To',
    'Resent-From',
    'Precedence',
    'Mailing-List',
    'List-Id',
    'List-Archive',
    'Errors-To',
    'References',
    'In-Reply-To',
    'X-Loop',
    'X-Spam-Flag',
    'X-Spam-Level',
    'X-Spam-Score',
    'X-Spam-Status',
    'X-No-Loop',
    'X-Priority',
    'Importance',
    'X-Mailer',
    'User-Agent',
    'Organization',
    'X-Original-To',
    'Delivered-To',
    'Envelope-To',
    'Return-Path',
    'X-OTRS-Loop',
    'X-OTRS-Priority',
    'X-OTRS-Queue',
    'X-OTRS-Lock',
    'X-OTRS-Ignore',
    'X-OTRS-State',
    'X-OTRS-State-PendingTime',
    'X-OTRS-Type',
    'X-OTRS-Service',
    'X-OTRS-SLA',
    'X-OTRS-CustomerNo',
    'X-OTRS-CustomerUser',
    'X-OTRS-SenderType',
    'X-OTRS-ArticleType',
    'X-OTRS-FollowUp-Priority',
    'X-OTRS-FollowUp-Queue',
    'X-OTRS-FollowUp-Lock',
    'X-OTRS-FollowUp-State',
    'X-OTRS-FollowUp-State-PendingTime',
    'X-OTRS-FollowUp-Type',
    'X-OTRS-FollowUp-Service',
    'X-OTRS-FollowUp-SLA',
    'X-OTRS-FollowUp-SenderType',
    'X-OTRS-FollowUp-ArticleType'
];
```

6.4.19. PostMaster::PreFilterModule###1-Match

Description:	Module to filter and manipulate incoming messages. Block/ignore all spam email with From: noreply@ address.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'PostMaster::PreFilterModule'}->{'1-Match'} =  {
  'Match' => {
    'From' => 'noreply@'
  },
  'Module' => 'Kernel::System::PostMaster::Filter::Match',
  'Set' => {
    'X-OTRS-Ignore' => 'yes'
  },
  'StopAfterMatch' => '0'
};
```

6.4.20. PostMaster::PreFilterModule###2-Match

Description:	Module to filter and manipulate incoming messages. Get a 4 digit number to ticket free text, use regex in Match e. g. From => '(.+?)@.+?', and use () as [***] in Set =>.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'PostMaster::PreFilterModule'}->{'2-Match'} =  {
  'Match' => {
    'Subject' => 'SomeNumber:(\\d\\d\\d\\d)'
  },
  'Module' => 'Kernel::System::PostMaster::Filter::Match',
  'Set' => {
    'X-OTRS-DynamicField-TicketFreeKey1' => 'SomeNumber',
    'X-OTRS-DynamicField-TicketFreeText1' => '[***]'
  },
  'StopAfterMatch' => '0'
};
```

6.4.21. PostMaster::PreFilterModule###3-NewTicketReject

Description:	Blocks all the incoming emails that do not have a valid ticket number in subject with From: @example.com address.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'PostMaster::PreFilterModule'}->{'3-NewTicketReject'} =  {
  'Match' => {
    'From' => '@example.com'
  },
  'Module' => 'Kernel::System::PostMaster::Filter::NewTicketReject',
  'Set' => {
    'X-OTRS-Ignore' => 'yes'
  },
  'StopAfterMatch' => '0'
};
```

6.4.22. PostMaster::PreFilterModule::NewTicketReject::Sender

Description:	Defines the sender for rejected emails.
Group:	Ticket

SubGroup:	Core::PostMaster
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'PostMaster::PreFilterModule::NewTicketReject::Sender'} =
'noreply@example.com';
```

6.4.23. PostMaster::PreFilterModule::NewTicketReject::Subject

Description:	Defines the subject for rejected emails.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'PostMaster::PreFilterModule::NewTicketReject::Subject'} =
'Email Rejected';
```

6.4.24. PostMaster::PreFilterModule::NewTicketReject::Body

Description:	Defines the body text for rejected emails.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'PostMaster::PreFilterModule::NewTicketReject::Body'} =  '
Dear Customer,

Unfortunately we could not detect a valid ticket number
in your subject, so this email can\'t be processed.

Please create a new ticket via the customer panel.

Thanks for your help!

 Your Helpdesk Team
';
```

6.4.25. PostMaster::PreFilterModule###4-CMD

Description:	CMD example setup. Ignores emails where external CMD returns some output on STDOUT (email will be piped into STDIN of some.bin).
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'PostMaster::PreFilterModule'}->{'4-CMD'} =  {
  'CMD' => '/usr/bin/some.bin',
  'Module' => 'Kernel::System::PostMaster::Filter::CMD',
  'Set' => {
    'X-OTRS-Ignore' => 'yes'
  }
};
```

6.4.26. PostMaster::PreFilterModule###5-SpamAssassin

Description:	Spam Assassin example setup. Ignores emails that are marked with SpamAssassin.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'PostMaster::PreFilterModule'}->{'5-SpamAssassin'} =  {
  'CMD' => '/usr/bin/spamassassin | grep -i "X-Spam-Status: yes"',
  'Module' => 'Kernel::System::PostMaster::Filter::CMD',
  'Set' => {
    'X-OTRS-Ignore' => 'yes'
  }
};
```

6.4.27. PostMaster::PreFilterModule###6-SpamAssassin

Description:	Spam Assassin example setup. Moves marked mails to spam queue.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'PostMaster::PreFilterModule'}->{'6-SpamAssassin'} =  {
  'CMD' => '/usr/bin/spamassassin | grep -i "X-Spam-Status: yes"',
  'Module' => 'Kernel::System::PostMaster::Filter::CMD',
  'Set' => {
    'X-OTRS-Queue' => 'spam'
  }
};
```

6.4.28. PostMaster::PreFilterModule###000-MatchDBSource

Description:	Module to use database filter storage.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'PostMaster::PreFilterModule'}->{'000-MatchDBSource'} =  {
  'Module' => 'Kernel::System::PostMaster::Filter::MatchDBSource'
};
```

6.4.29. PostMaster::PostFilterModule###000-FollowUpArticleTypeCheck

Description:	Module to check if arrived emails should be marked as email-internal (because of original forwarded internal email it college). ArticleType and SenderType define the values for the arrived email/article.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1

Required:	0

Config-Setting:

```
 $Self->{'PostMaster::PostFilterModule'}->{'000-
FollowUpArticleTypeCheck'} =  {
  'ArticleType' => 'email-internal',
  'Module' =>
 'Kernel::System::PostMaster::Filter::FollowUpArticleTypeCheck',
  'SenderType' => 'customer'
};
```

6.4.30. SendNoAutoResponseRegExp

Description:	If this regex matches, no message will be send by the autoresponder.
Group:	Ticket
SubGroup:	Core::PostMaster
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'SendNoAutoResponseRegExp'} =  '(MAILER-DAEMON|postmaster|
abuse)@.+?\\..+?';
```

6.5. Core::Stats

6.5.1. Stats::DynamicObjectRegistration###Ticket

Description:	Module to generate ticket statistics.
Group:	Ticket
SubGroup:	Core::Stats
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Stats::DynamicObjectRegistration'}->{'Ticket'} =  {
  'Module' => 'Kernel::System::Stats::Dynamic::Ticket'
};
```

6.5.2. Stats::DynamicObjectRegistration###TicketList

Description:	Determines if the statistics module may generate ticket lists.
Group:	Ticket
SubGroup:	Core::Stats
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Stats::DynamicObjectRegistration'}->{'TicketList'} =  {
  'Module' => 'Kernel::System::Stats::Dynamic::TicketList'
};
```

6.5.3. Stats::DynamicObjectRegistration###TicketAccountedTime

Description:	Module to generate accounted time ticket statistics.
Group:	Ticket
SubGroup:	Core::Stats
Valid:	1

Required:	0
Config-Setting:	

```
 $Self->{'Stats::DynamicObjectRegistration'}->{'TicketAccountedTime'}
 =  {
  'Module' => 'Kernel::System::Stats::Dynamic::TicketAccountedTime'
};
```

6.5.4. Stats::DynamicObjectRegistration###TicketSolutionResponseTime

Description:	Module to generate ticket solution and response time statistics.
Group:	Ticket
SubGroup:	Core::Stats
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Stats::DynamicObjectRegistration'}-
>{'TicketSolutionResponseTime'} =  {
  'Module' =>
 'Kernel::System::Stats::Dynamic::TicketSolutionResponseTime'
};
```

6.6. Core::Ticket

6.6.1. Ticket::Hook

Description:	The identifier for a ticket, e.g. Ticket#, Call#, MyTicket#. The default is Ticket#.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Hook'} =  'Ticket#';`

6.6.2. Ticket::HookDivider

Description:	The divider between TicketHook and ticket number. E.g ': '.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::HookDivider'} =  '';`

6.6.3. Ticket::SubjectSize

Description:	Max size of the subjects in an email reply.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1

Config-Setting:	`$Self->{'Ticket::SubjectSize'} =  '100';`

6.6.4. Ticket::SubjectRe

Description:	The text at the beginning of the subject in an email reply, e.g. RE, AW, or AS.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::SubjectRe'} =  'Re';`

6.6.5. Ticket::SubjectFwd

Description:	The text at the beginning of the subject when an email is forwarded, e.g. FW, Fwd, or WG.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::SubjectFwd'} =  'Fwd';`

6.6.6. Ticket::SubjectFormat

Description:	The format of the subject. 'Left' means '[TicketHook#:12345] Some Subject', 'Right' means 'Some Subject [TicketHook#:12345]', 'None' means 'Some Subject' and no ticket number. In the last case you should enable PostmasterFollowupSearchInRaw or PostmasterFollowUpSearchInReferences to recognize followups based on email headers and/or body.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::SubjectFormat'} =  'Left';`

6.6.7. Ticket::CustomQueue

Description:	Name of custom queue. The custom queue is a queue selection of your preferred queues and can be selected in the preferences settings.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::CustomQueue'} =  'My Queues';`

6.6.8. Ticket::NewArticleIgnoreSystemSender

Description:	Ignore article with system sender type for new article feature (e. g. auto responses or email notifications).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::NewArticleIgnoreSystemSender'} =  '0';`

6.6.9. Ticket::ChangeOwnerToEveryone

Description:	Changes the owner of tickets to everyone (useful for ASP). Normally only agent with rw permissions in the queue of the ticket will be shown.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::ChangeOwnerToEveryone'} =  '0';`

6.6.10. Ticket::Responsible

Description:	Enables ticket responsible feature, to keep track of a specific ticket.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Responsible'} =  '0';`

6.6.11. Ticket::ResponsibleAutoSet

Description:	Automatically sets the owner of a ticket as the responsible for it (if ticket responsible feature is enabled).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::ResponsibleAutoSet'} =  '1';`

6.6.12. Ticket::Type

Description:	Allows defining new types for ticket (if ticket type feature is enabled).
Group:	Ticket
SubGroup:	Core::Ticket

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Type'} =  '0';`

6.6.13. Ticket::Service

Description:	Allows defining services and SLAs for tickets (e. g. email, desktop, network, ...), and escalation attributes for SLAs (if ticket service/SLA feature is enabled).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Service'} =  '0';`

6.6.14. Ticket::Service::Default::UnknownCustomer

Description:	Allows default services to be selected also for non existing customers.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Service::Default::UnknownCustomer'} =  '0';`

6.6.15. Ticket::ArchiveSystem

Description:	Activates the ticket archive system to have a faster system by moving some tickets out of the daily scope. To search for these tickets, the archive flag has to be enabled in the ticket search.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::ArchiveSystem'} =  '0';`

6.6.16. Ticket::ArchiveSystem::RemoveSeenFlags

Description:	Controls if the ticket and article seen flags are removed when a ticket is archived.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::ArchiveSystem::RemoveSeenFlags'} =  '1';`

6.6.17. Ticket::ArchiveSystem::RemoveTicketWatchers

Description:	Removes the ticket watcher information when a ticket is archived.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::ArchiveSystem::RemoveTicketWatchers'} =  '1';`

6.6.18. Ticket::CustomerArchiveSystem

Description:	Activates the ticket archive system search in the customer interface.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::CustomerArchiveSystem'} =  '0';`

6.6.19. Ticket::NumberGenerator

Description:	Selects the ticket number generator module. "AutoIncrement" increments the ticket number, the SystemID and the counter are used with SystemID.counter format (e.g. 1010138, 1010139). With "Date" the ticket numbers will be generated by the current date, the SystemID and the counter. The format looks like Year.Month.Day.SystemID.counter (e.g. 200206231010138, 200206231010139). With "DateChecksum" the counter will be appended as checksum to the string of date and SystemID. The checksum will be rotated on a daily basis. The format looks like Year.Month.Day.SystemID.Counter.CheckSum (e.g. 2002070110101520, 2002070110101535). "Random" generates randomized ticket numbers in the format "SystemID.Random" (e.g. 100057866352, 103745394596).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::NumberGenerator'} = 'Kernel::System::Ticket::Number::DateChecksum';`

6.6.20. Ticket::NumberGenerator::MinCounterSize

Description:	Sets the minimal ticket counter size (if "AutoIncrement" was selected as TicketNumberGenerator). Default is 5, this means the counter starts from 10000.
Group:	Ticket
SubGroup:	Core::Ticket

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::NumberGenerator::MinCounterSize'} =  '5';`

6.6.21. Ticket::NumberGenerator::CheckSystemID

Description:	Checks the SystemID in ticket number detection for follow-ups (use "No" if SystemID has been changed after using the system).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::NumberGenerator::CheckSystemID'} =  '1';`

6.6.22. Ticket::CounterLog

Description:	Log file for the ticket counter.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::CounterLog'} =  '<OTRS_CONFIG_Home>/var/log/TicketCounter.log';`

6.6.23. Ticket::IndexModule

Description:	IndexAccelerator: to choose your backend TicketViewAccelerator module. "RuntimeDB" generates each queue view on the fly from ticket table (no performance problems up to approx. 60.000 tickets in total and 6.000 open tickets in the system). "StaticDB" is the most powerful module, it uses an extra ticket-index table that works like a view (recommended if more than 80.000 and 6.000 open tickets are stored in the system). Use the script "bin/otrs.RebuildTicketIndex.pl" for initial index update.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::IndexModule'} = 'Kernel::System::Ticket::IndexAccelerator::RuntimeDB';`

6.6.24. Ticket::StorageModule

Description:	Saves the attachments of articles. "DB" stores all data in the database (not recommended for storing big attachments). "FS" stores the data on the filesystem; this is faster but the webserver should run under the OTRS user. You can switch between the modules

	even on a system that is already in production without any loss of data.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::StorageModule'} = 'Kernel::System::Ticket::ArticleStorageDB';`

6.6.25. ArticleDir

Description:	Specifies the directory to store the data in, if "FS" was selected for TicketStorageModule.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'ArticleDir'} =  '<OTRS_CONFIG_Home>/var/article';`

6.6.26. Ticket::EventModulePost###100-ArchiveRestore

Description:	Restores a ticket from the archive (only if the event is a state change, from closed to any open available state).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::EventModulePost'}->{'100-ArchiveRestore'} =  {
  'Event' => 'TicketStateUpdate',
  'Module' => 'Kernel::System::Ticket::Event::ArchiveRestore'
};
```

6.6.27. Ticket::EventModulePost###110-AcceleratorUpdate

Description:	Updates the ticket index accelerator.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::EventModulePost'}->{'110-AcceleratorUpdate'} =  {
  'Event' => 'TicketStateUpdate|TicketQueueUpdate|TicketLockUpdate',
  'Module' => 'Kernel::System::Ticket::Event::TicketAcceleratorUpdate'
};
```

6.6.28. Ticket::EventModulePost###120-ForceOwnerResetOnMove

Description:	Resets and unlocks the owner of a ticket if it was moved to another queue.

Group:	Ticket
SubGroup:	Core::Ticket
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'120-ForceOwnerResetOnMove'} =
 {
  'Event' => 'TicketQueueUpdate',
  'Module' => 'Kernel::System::Ticket::Event::ForceOwnerReset'
};
```

6.6.29. Ticket::EventModulePost###130-ForceStateChangeOnLock

Description:	Forces to choose a different ticket state (from current) after lock action. Define the current state as key, and the next state after lock action as content.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'130-ForceStateChangeOnLock'} =
 {
  'Event' => 'TicketLockUpdate',
  'Module' => 'Kernel::System::Ticket::Event::ForceState',
  'new' => 'open'
};
```

6.6.30. Ticket::EventModulePost###140-ResponsibleAutoSet

Description:	Automatically sets the responsible of a ticket (if it is not set yet) after the first owner update.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'140-ResponsibleAutoSet'} =  {
  'Event' => 'TicketOwnerUpdate',
  'Module' => 'Kernel::System::Ticket::Event::ResponsibleAutoSet'
};
```

6.6.31. Ticket::EventModulePost###150-TicketPendingTimeReset

Description:	Sets the PendingTime of a ticket to 0 if the state is changed to a non-pending state.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'150-TicketPendingTimeReset'} =
 {
  'Event' => 'TicketStateUpdate',
  'Module' => 'Kernel::System::Ticket::Event::TicketPendingTimeReset'
};
```

6.6.32. Ticket::EventModulePost###500-NotificationEvent

Description:	Sends the notifications which are configured in the admin interface under "Notfication (Event)".
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'500-NotificationEvent'} =  {
  'Event' => '',
  'Module' => 'Kernel::System::Ticket::Event::NotificationEvent',
  'Transaction' => '1'
};
```

6.6.33. Ticket::EventModulePost###900-EscalationIndex

Description:	Updates the ticket escalation index after a ticket attribute got updated.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'900-EscalationIndex'} =  {
  'Event' => 'TicketSLAUpdate|TicketQueueUpdate|TicketStateUpdate|
TicketCreate|ArticleCreate',
  'Module' => 'Kernel::System::Ticket::Event::TicketEscalationIndex'
};
```

6.6.34. Ticket::EventModulePost###900-EscalationStopEvents

Description:	Ticket event module that triggers the escalation stop events.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::EventModulePost'}->{'900-EscalationStopEvents'} =  {
  'Event' => 'TicketSLAUpdate|TicketQueueUpdate|TicketStateUpdate|
ArticleCreate',
  'Module' =>
 'Kernel::System::Ticket::Event::TriggerEscalationStopEvents'
};
```

6.6.35. Ticket::EventModulePost###910-ForceUnlockOnMove

Description:	Forces to unlock tickets after being moved to another queue.
Group:	Ticket
SubGroup:	Core::Ticket

Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::EventModulePost'}->{'910-ForceUnlockOnMove'} =  {
  'Event' => 'TicketQueueUpdate',
  'Module' => 'Kernel::System::Ticket::Event::ForceUnlock'
};
```

6.6.36. Ticket::EventModulePost###920-TicketArticleNewMessageUpdate

Description:	Update Ticket "Seen" flag if every article got seen or a new Article got created.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::EventModulePost'}->{'920-
TicketArticleNewMessageUpdate'} =  {
  'Event' => 'ArticleCreate|ArticleFlagSet',
  'Module' => 'Kernel::System::Ticket::Event::TicketNewMessageUpdate'
};
```

6.6.37. Ticket::CustomModule###001-CustomModule

Description:	Overloads (redefines) existing functions in Kernel::System::Ticket. Used to easily add customizations.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::CustomModule'}->{'001-CustomModule'} =
 'Kernel::System::Ticket::Custom';
```

6.6.38. Ticket::ViewableSenderTypes

Description:	Defines the default viewable sender types of a ticket (default: customer).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::ViewableSenderTypes'} =  [
  '\'customer\''
];
```

6.6.39. Ticket::ViewableLocks

Description:	Defines the viewable locks of a ticket. Default: unlock, tmp_lock.
Group:	Ticket
SubGroup:	Core::Ticket

Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::ViewableLocks'} =  [
  '\'unlock\'',
  '\'tmp_lock\''
];
```

6.6.40. Ticket::ViewableStateType

Description:	Defines the valid state types for a ticket.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::ViewableStateType'} =  [
  'new',
  'open',
  'pending reminder',
  'pending auto'
];
```

6.6.41. Ticket::UnlockStateType

Description:	Defines the valid states for unlocked tickets. To unlock tickets the script "bin/otrs.UnlockTickets.pl" can be used.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::UnlockStateType'} =  [
  'new',
  'open'
];
```

6.6.42. Ticket::PendingNotificationOnlyToOwner

Description:	Sends reminder notifications of unlocked ticket after reaching the reminder date (only sent to ticket owner).
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::PendingNotificationOnlyToOwner'} =  '0';`

6.6.43. Ticket::PendingNotificationNotToResponsible

Description:	Disables sending reminder notifications to the responsible agent of a ticket (Ticket::Responsible needs to be activated).
Group:	Ticket

SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::PendingNotificationNotToResponsible'} =  '0';`

6.6.44. Ticket::PendingReminderStateType

Description:	Defines the state type of the reminder for pending tickets.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::PendingReminderStateType'} =  [
  'pending reminder'
];
```

6.6.45. Ticket::PendingAutoStateType

Description:	Determines the possible states for pending tickets that changed state after reaching time limit.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::PendingAutoStateType'} =  [
  'pending auto'
];
```

6.6.46. Ticket::StateAfterPending

Description:	Defines which states should be set automatically (Content), after the pending time of state (Key) has been reached.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::StateAfterPending'} =  {
  'pending auto close+' => 'closed successful',
  'pending auto close-' => 'closed unsuccessful'
};
```

6.6.47. System::Permission

Description:	Standard available permissions for agents within the application. If more permissions are needed, they can be entered here. Permissions must be defined to be effective. Some other good permissions have also been provided built-in: note, close, pending, customer, freetext, move, compose, responsible, forward, and bounce. Make sure that "rw" is always the last registered permission.

Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'System::Permission'} =  [
  'ro',
  'move_into',
  'create',
  'note',
  'owner',
  'priority',
  'rw'
];
```

6.6.48. Ticket::Permission###1-OwnerCheck

Description:	Module to check the owner of a ticket.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Permission'}->{'1-OwnerCheck'} =  {
  'Granted' => '1',
  'Module' => 'Kernel::System::Ticket::Permission::OwnerCheck',
  'Required' => '0'
};
```

6.6.49. Ticket::Permission###2-ResponsibleCheck

Description:	Module to check the agent responsible of a ticket.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Permission'}->{'2-ResponsibleCheck'} =  {
  'Granted' => '1',
  'Module' => 'Kernel::System::Ticket::Permission::ResponsibleCheck',
  'Required' => '0'
};
```

6.6.50. Ticket::Permission###3-GroupCheck

Description:	Module to check if a user is in a special group. Access is granted, if the user is in the specified group and has ro and rw permissions.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Permission'}->{'3-GroupCheck'} =  {
  'Granted' => '1',
  'Module' => 'Kernel::System::Ticket::Permission::GroupCheck',
  'Required' => '0'
};
```

6.6.51. Ticket::Permission###4-WatcherCheck

Description:	Module to check the watcher agents of a ticket.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::Permission'}->{'4-WatcherCheck'} =  {
  'Granted' => '1',
  'Module' => 'Kernel::System::Ticket::Permission::WatcherCheck',
  'Required' => '0'
};
```

6.6.52. CustomerTicket::Permission###1-GroupCheck

Description:	Module to check the group permissions for the access to customer tickets.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'CustomerTicket::Permission'}->{'1-GroupCheck'} =  {
  'Granted' => '0',
  'Module' =>
 'Kernel::System::Ticket::CustomerPermission::GroupCheck',
  'Required' => '1'
};
```

6.6.53. CustomerTicket::Permission###2-CustomerUserIDCheck

Description:	Grants access, if the customer ID of the ticket matches the customer user's ID and the customer user has group permissions on the queue the ticket is in.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'CustomerTicket::Permission'}->{'2-CustomerUserIDCheck'} =  {
  'Granted' => '1',
  'Module' =>
 'Kernel::System::Ticket::CustomerPermission::CustomerUserIDCheck',
  'Required' => '0'
};
```

6.6.54. CustomerTicket::Permission###3-CustomerIDCheck

Description:	Module to check customer permissions.
Group:	Ticket

SubGroup:	Core::Ticket
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'CustomerTicket::Permission'}->{'3-CustomerIDCheck'} =  {
  'Granted' => '1',
  'Module' =>
 'Kernel::System::Ticket::CustomerPermission::CustomerIDCheck',
  'Required' => '0'
};
```

6.6.55. Ticket::DefineEmailFrom

Description:	Defines how the From field from the emails (sent from answers and email tickets) should look like.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::DefineEmailFrom'} =  'SystemAddressName';`

6.6.56. Ticket::DefineEmailFromSeparator

Description:	Defines the separator between the agents real name and the given queue email address.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::DefineEmailFromSeparator'} =  'via';`

6.6.57. CustomerNotifyJustToRealCustomer

Description:	Sends customer notifications just to the mapped customer. Normally, if no customer is mapped, the latest customer sender gets the notification.
Group:	Ticket
SubGroup:	Core::Ticket
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerNotifyJustToRealCustomer'} =  '0';`

6.6.58. AgentSelfNotifyOnAction

Description:	Specifies if an agent should receive email notification of his own actions.
Group:	Ticket
SubGroup:	Core::Ticket

Valid:	1
Required:	1
Config-Setting:	`$Self->{'AgentSelfNotifyOnAction'} =  '0';`

6.7. Core::TicketACL

6.7.1. Ticket::Acl::Module###1-Ticket::Acl::Module

Description:	ACL module that allows closing parent tickets only if all its children are already closed ("State" shows which states are not available for the parent ticket until all child tickets are closed).
Group:	Ticket
SubGroup:	Core::TicketACL
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Acl::Module'}->{'1-Ticket::Acl::Module'} =  {
  'Module' =>
'Kernel::System::Ticket::Acl::CloseParentAfterClosedChilds',
  'State' => [
    'closed successful',
    'closed unsuccessful'
  ]
};
```

6.7.2. TicketACL::Default::Action

Description:	Default ACL values for ticket actions.
Group:	Ticket
SubGroup:	Core::TicketACL
Valid:	1
Required:	1
Config-Setting:	`$Self->{'TicketACL::Default::Action'} =  {};`

6.8. Core::TicketBulkAction

6.8.1. Ticket::Frontend::BulkFeature

Description:	Enables ticket bulk action feature for the agent frontend to work on more than one ticket at a time.
Group:	Ticket
SubGroup:	Core::TicketBulkAction
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::BulkFeature'} =  '1';`

6.8.2. Ticket::Frontend::BulkFeatureGroup

Description:	Enables ticket bulk action feature only for the listed groups.
Group:	Ticket

SubGroup:	Core::TicketBulkAction
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::BulkFeatureGroup'} =  [
  'admin',
  'users'
];
```

6.9. Core::TicketDynamicFieldDefault

6.9.1. Ticket::EventModulePost###TicketDynamicFieldDefault

Description:	Event module registration. For more performance you can define a trigger event (e. g. Event => TicketCreate). This is only possible if all Ticket dynamic fields need the same event.
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::EventModulePost'}->{'TicketDynamicFieldDefault'} =
{
  'Module' =>
'Kernel::System::Ticket::Event::TicketDynamicFieldDefault',
  'Transaction' => '1'
};
```

6.9.2. Ticket::TicketDynamicFieldDefault###Element1

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element1'} =  {
  'Event' => 'TicketCreate',
  'Name' => 'Field1',
  'Value' => 'Default'
};
```

6.9.3. Ticket::TicketDynamicFieldDefault###Element2

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket

SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element2'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.4. Ticket::TicketDynamicFieldDefault###Element3

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element3'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.5. Ticket::TicketDynamicFieldDefault###Element4

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element4'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.6. Ticket::TicketDynamicFieldDefault###Element5

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket

SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element5'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.7. Ticket::TicketDynamicFieldDefault###Element6

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element6'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.8. Ticket::TicketDynamicFieldDefault###Element7

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element7'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.9. Ticket::TicketDynamicFieldDefault###Element8

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket

SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element8'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.10. Ticket::TicketDynamicFieldDefault###Element9

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element9'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.11. Ticket::TicketDynamicFieldDefault###Element10

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element10'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.12. Ticket::TicketDynamicFieldDefault###Element11

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket

SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element11'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.13. Ticket::TicketDynamicFieldDefault###Element12

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element12'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.14. Ticket::TicketDynamicFieldDefault###Element13

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element13'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.15. Ticket::TicketDynamicFieldDefault###Element14

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket

SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element14'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.16. Ticket::TicketDynamicFieldDefault###Element15

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element15'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.9.17. Ticket::TicketDynamicFieldDefault###Element16

Description:	Configures a default TicketDynmicField setting. "Name" defines the dynamic field which should be used, "Value" is the data that will be set, and "Event" defines the trigger event. Please check the developer manual (http://doc.otrs.org/), chapter "Ticket Event Module".
Group:	Ticket
SubGroup:	Core::TicketDynamicFieldDefault
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::TicketDynamicFieldDefault'}->{'Element16'} =  {
  'Event' => '',
  'Name' => '',
  'Value' => ''
};
```

6.10. Core::TicketWatcher

6.10.1. Ticket::Watcher

Description:	Enables or disables the ticket watcher feature, to keep track of tickets without being the owner nor the responsible.
Group:	Ticket
SubGroup:	Core::TicketWatcher

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Watcher'} =  '0';`

6.10.2. Ticket::WatcherGroup

Description:	Enables ticket watcher feature only for the listed groups.
Group:	Ticket
SubGroup:	Core::TicketWatcher
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::WatcherGroup'} =  [
  'admin',
  'users'
];
```

6.11. Frontend::Admin::ModuleRegistration

6.11.1. Frontend::Module###AdminQueue

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminQueue'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Create and manage queues.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Queues',
    'Prio' => '100'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Queues'
};
```

6.11.2. Frontend::Module###AdminResponse

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminResponse'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Create and manage response templates.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Responses',
    'Prio' => '200'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Responses'
};
```

6.11.3. Frontend::Module###AdminQueueResponses

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminQueueResponses'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Link responses to queues.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Responses <-> Queues',
    'Prio' => '300'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Responses <-> Queues'
};
```

6.11.4. Frontend::Module###AdminAutoResponse

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminAutoResponse'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Create and manage responses that are
 automatically sent.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Auto Responses',
    'Prio' => '400'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Auto Responses'
};
```

6.11.5. Frontend::Module###AdminQueueAutoResponse

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminQueueAutoResponse'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Link queues to auto responses.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Auto Responses <-> Queues',
    'Prio' => '500'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Auto Responses <-> Queues'
};
```

6.11.6. Frontend::Module###AdminAttachment

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminAttachment'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Create and manage attachments.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Attachments',
    'Prio' => '600'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Attachments'
};
```

6.11.7. Frontend::Module###AdminResponseAttachment

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminResponseAttachment'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Link attachments to responses templates.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Attachments <-> Responses',
    'Prio' => '700'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Attachments <-> Responses'
};
```

6.11.8. Frontend::Module###AdminSalutation

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminSalutation'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Create and manage salutations.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Salutations',
    'Prio' => '800'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Salutations'
};
```

6.11.9. Frontend::Module###AdminSignature

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminSignature'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Queue',
    'Description' => 'Create and manage signatures.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Signatures',
    'Prio' => '900'
  },
  'NavBarName' => 'Admin',
```

	`'Title' => 'Signatures'` `};`

6.11.10. Frontend::Module###AdminSystemAddress

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminSystemAddress'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Email',
    'Description' => 'Set sender email addresses for this system.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Email Addresses',
    'Prio' => '300'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Email Addresses'
};
```

6.11.11. Frontend::Module###AdminNotification

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AdminNotification'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Manage notifications that are sent to agents.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Agent Notifications',
    'Prio' => '400'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Agent Notifications'
};
```

6.11.12. Frontend::Module###AdminNotificationEvent

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1

Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminNotificationEvent'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Create and manage event based notifications.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Notifications (Event)',
    'Prio' => '400'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Notifications (Event)'
};
```

6.11.13. Frontend::Module###AdminService

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminService'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Create and manage services.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Services',
    'Prio' => '900'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Services'
};
```

6.11.14. Frontend::Module###AdminSLA

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AdminSLA'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Create and manage Service Level Agreements
(SLAs).',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Service Level Agreements',
    'Prio' => '1000'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Service Level Agreements'
};
```

6.11.15. Frontend::Module###AdminType

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminType'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Create and manage ticket types.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Types',
    'Prio' => '700'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Types'
};
```

6.11.16. Frontend::Module###AdminState

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AdminState'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Create and manage ticket states.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'States',
    'Prio' => '800'
  },
  'NavBarName' => 'Admin',
  'Title' => 'States'
};
```

6.11.17. Frontend::Module###AdminPriority

Description:	Frontend module registration for the agent interface.

Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminPriority'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'Ticket',
    'Description' => 'Create and manage ticket priorities.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'Priorities',
    'Prio' => '850'
  },
  'NavBarName' => 'Admin',
  'Title' => 'Priorities'
};
```

6.11.18. Frontend::Module###AdminGenericAgent

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Admin::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AdminGenericAgent'} =  {
  'Description' => 'Admin',
  'Group' => [
    'admin'
  ],
  'NavBarModule' => {
    'Block' => 'System',
    'Description' => 'Manage periodic tasks.',
    'Module' => 'Kernel::Output::HTML::NavBarModuleAdmin',
    'Name' => 'GenericAgent',
    'Prio' => '300'
  },
  'NavBarName' => 'Admin',
  'Title' => 'GenericAgent'
};
```

6.12. Frontend::Agent

6.12.1. Ticket::Frontend::PendingDiffTime

Description:	Time in seconds that gets added to the actual time if setting a pending-state (default: 86400 = 1 day).
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::PendingDiffTime'} =  '86400';`

6.12.2. Ticket::Frontend::MaxQueueLevel

Description:	Define the max depth of queues.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::MaxQueueLevel'} =  '5';`

6.12.3. Ticket::Frontend::ListType

Description:	Shows existing parent/child queue lists in the system in the form of a tree or a list.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::ListType'} =  'tree';`

6.12.4. Ticket::Frontend::TextAreaEmail

Description:	Permitted width for compose email windows.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::TextAreaEmail'} =  '82';`

6.12.5. Ticket::Frontend::TextAreaNote

Description:	Permitted width for compose note windows.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::TextAreaNote'} =  '78';`

6.12.6. Ticket::Frontend::InformAgentMaxSize

Description:	Max size (in rows) of the informed agents box in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::InformAgentMaxSize'} =  '3';`

6.12.7. Ticket::Frontend::InvolvedAgentMaxSize

Description:	Max size (in rows) of the involved agents box in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::InvolvedAgentMaxSize'} =  '3';`

6.12.8. Ticket::Frontend::CustomerInfoCompose

Description:	Shows the customer user information (phone and email) in the compose screen.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerInfoCompose'} =  '1';`

6.12.9. Ticket::Frontend::CustomerInfoComposeMaxSize

Description:	Max size (in characters) of the customer information table (phone and email) in the compose screen.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerInfoComposeMaxSize'} =  '22';`

6.12.10. Ticket::Frontend::CustomerInfoZoom

Description:	Shows the customer user's info in the ticket zoom view.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerInfoZoom'} =  '1';`

6.12.11. Ticket::Frontend::CustomerInfoZoomMaxSize

Description:	Maximum size (in characters) of the customer information table in the ticket zoom view.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1

Config-Setting:	`$Self->{'Ticket::Frontend::CustomerInfoZoomMaxSize'} =  '22';`

6.12.12. Ticket::Frontend::AccountTime

Description:	Activates time accounting.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AccountTime'} =  '1';`

6.12.13. Ticket::Frontend::TimeUnits

Description:	Sets the prefered time units (e.g. work units, hours, minutes).
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::TimeUnits'} =  ' (work units)';`

6.12.14. Ticket::Frontend::NeedAccountedTime

Description:	Defines if time accounting is mandatory in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::NeedAccountedTime'} =  '0';`

6.12.15. Ticket::Frontend::BulkAccountedTime

Description:	Defines if time accounting must be set to all tickets in bulk action.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::BulkAccountedTime'} =  '1';`

6.12.16. Ticket::Frontend::NeedSpellCheck

Description:	Defines if composed messages have to be spell checked in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::NeedSpellCheck'} =  '0';`

6.12.17. Ticket::Frontend::NewOwnerSelection

Description:	Shows an owner selection in phone and email tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::NewOwnerSelection'} =  '1';`

6.12.18. Ticket::Frontend::NewResponsibleSelection

Description:	Show a responsible selection in phone and email tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::NewResponsibleSelection'} =  '1';`

6.12.19. Ticket::Frontend::NewQueueSelectionType

Description:	Defines the receipent target of the phone ticket and the sender of the email ticket ("Queue" shows all queues, "SystemAddress" displays all system addresses) in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::NewQueueSelectionType'} =  'Queue';`

6.12.20. Ticket::Frontend::NewQueueSelectionString

Description:	Determines the strings that will be shown as receipent (To:) of the phone ticket and as sender (From:) of the email ticket in the agent interface. For Queue as NewQueueSelectionType "<Queue>" shows the names of the queues and for SystemAddress "<Realname> <<Email>>" shows the name and email of the receipent.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::NewQueueSelectionString'} =  '<Queue>';`

6.12.21. Ticket::Frontend::NewQueueOwnSelection

Description:	Determines which options will be valid of the recepient (phone ticket) and the sender (email ticket) in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::NewQueueOwnSelection'} =  {
  '1' => 'First Queue!',
  '2' => 'Second Queue!'
};
```

6.12.22. Ticket::Frontend::ShowCustomerTickets

Description:	Shows customer history tickets in AgentTicketPhone, AgentTicketEmail and AgentTicketCustomer.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::ShowCustomerTickets'} =  '1';`

6.12.23. NewTicketInNewWindow::Enabled

Description:	If enabled, TicketPhone and TicketEmail will be open in new windows.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'NewTicketInNewWindow::Enabled'} =  '0';`

6.12.24. CustomerDBLink

Description:	Defines an external link to the database of the customer (e.g. 'http://yourhost/customer.php?CID=$Data{"CustomerID"}' or '').
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'CustomerDBLink'} =  '$Env{"CGIHandle"}?
Action=AgentCustomerInformationCenter;CustomerID=
$QData{"CustomerID"}';
```

6.12.25. CustomerDBLinkTarget

Description:	Defines the target attribute in the link to external customer database. E.g. 'target="cdb"'.

Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerDBLinkTarget'} =  '';`

6.12.26. CustomerDBLinkClass

Description:	Defines the target attribute in the link to external customer database. E.g. 'AsPopup PopupType_TicketAction'.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerDBLinkClass'} =  '';`

6.12.27. Frontend::CommonObject###QueueObject

Description:	Path of the file that stores all the settings for the QueueObject object for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::CommonObject'}->{'QueueObject'} = 'Kernel::System::Queue';`

6.12.28. Frontend::CommonObject###TicketObject

Description:	Path of the file that stores all the settings for the TicketObject for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::CommonObject'}->{'TicketObject'} = 'Kernel::System::Ticket';`

6.12.29. Frontend::CommonParam###Action

Description:	Defines the default used Frontend-Module if no Action parameter given in the url on the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1

Config-Setting:	`$Self->{'Frontend::CommonParam'}->{'Action'} =  'AgentDashboard';`

6.12.30. Frontend::CommonParam###QueueID

Description:	Default queue ID used by the system in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::CommonParam'}->{'QueueID'} =  '0';`

6.12.31. Frontend::CommonParam###TicketID

Description:	Default ticket ID used by the system in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Frontend::CommonParam'}->{'TicketID'} =  '';`

6.13. Frontend::Agent::CustomerSearch

6.13.1. Ticket::Frontend::CustomerSearchAutoComplete###Active

Description:	Enables or disables the autocomplete feature for the customer search in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::CustomerSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerSearchAutoComplete'}->{'Active'} = '1';`

6.13.2. Ticket::Frontend::CustomerSearchAutoComplete###MinQueryLength

Description:	Sets the minimum number of characters before autocomplete query is sent.
Group:	Ticket
SubGroup:	Frontend::Agent::CustomerSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerSearchAutoComplete'}->{'MinQueryLength'} =  '2';`

6.13.3. Ticket::Frontend::CustomerSearchAutoComplete###QueryDelay

Description:	Delay time between autocomplete queries in milliseconds.

Group:	Ticket
SubGroup:	Frontend::Agent::CustomerSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerSearchAutoComplete'}->{'QueryDelay'} =  '100';`

6.13.4. Ticket::Frontend::CustomerSearchAutoComplete###MaxResultsDisplayed

Description:	Sets the number of search results to be displayed for the autocomplete feature.
Group:	Ticket
SubGroup:	Frontend::Agent::CustomerSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerSearchAutoComplete'}->{'MaxResultsDisplayed'} =  '20';`

6.13.5. Ticket::Frontend::CustomerSearchAutoComplete::DynamicWidth

Description:	Determines if the search results container for the autocomplete feature should adjust its width dynamically.
Group:	Ticket
SubGroup:	Frontend::Agent::CustomerSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerSearchAutoComplete::DynamicWidth'} =  '1';`

6.14. Frontend::Agent::Dashboard

6.14.1. DashboardBackend###0100-TicketPendingReminder

Description:	Parameters for the dashboard backend of the ticket pending reminder overview of the agent interface . "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'DashboardBackend'}->{'0100-TicketPendingReminder'} =  {
  'Attributes' => 'TicketPendingTimeOlderMinutes=1;StateType=pending
reminder;SortBy=PendingTime;OrderBy=Down;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All tickets with a reminder set where the reminder
 date has been reached',
  'Filter' => 'Locked',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'rw',
  'Time' => 'UntilTime',
  'Title' => 'Reminder Tickets'
};
```

6.14.2. DashboardBackend###0110-TicketEscalation

Description:	Parameters for the dashboard backend of the ticket escalation overview of the agent interface . "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DashboardBackend'}->{'0110-TicketEscalation'} =  {
  'Attributes' =>
 'TicketEscalationTimeOlderMinutes=1;SortBy=EscalationTime;OrderBy=Down;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All escalated tickets',
  'Filter' => 'All',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'rw',
  'Time' => 'EscalationTime',
  'Title' => 'Escalated Tickets'
};
```

6.14.3. DashboardBackend###0120-TicketNew

Description:	Parameters for the dashboard backend of the new tickets overview of the agent interface. "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'DashboardBackend'}->{'0120-TicketNew'} =  {
  'Attributes' => 'StateType=new;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All new tickets, these tickets have not been
 worked on yet',
  'Filter' => 'All',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'rw',
  'Time' => 'Age',
  'Title' => 'New Tickets'
};
```

6.14.4. DashboardBackend###0130-TicketOpen

Description:	Parameters for the dashboard backend of the ticket pending reminder overview of the agent interface. "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'DashboardBackend'}->{'0130-TicketOpen'} =  {
  'Attributes' => 'StateType=open;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All open tickets, these tickets have already been
 worked on, but need a response',
  'Filter' => 'All',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'rw',
  'Time' => 'Age',
  'Title' => 'Open Tickets / Need to be answered'
};
```

6.14.5. DashboardBackend###0250-TicketStats

Description:	Parameters for the dashboard backend of the ticket stats of the agent interface. "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'DashboardBackend'}->{'0250-TicketStats'} =  {
  'Block' => 'ContentSmall',
  'CacheTTL' => '30',
  'Closed' => '1',
  'Created' => '1',
  'Default' => '1',
  'Group' => '',
  'Module' => 'Kernel::Output::HTML::DashboardTicketStatsGeneric',
  'Permission' => 'rw',
  'Title' => '7 Day Stats'
};
```

6.14.6. DashboardBackend###0260-TicketCalendar

Description:	Parameters for the dashboard backend of the ticket calendar of the agent interface. "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'DashboardBackend'}->{'0260-TicketCalendar'} =  {
  'Block' => 'ContentSmall',
  'CacheTTL' => '2',
  'Default' => '1',
  'Group' => '',
  'Limit' => '6',
  'Module' => 'Kernel::Output::HTML::DashboardCalendar',
  'OwnerOnly' => '',
  'Permission' => 'rw',
  'Title' => 'Upcoming Events'
};
```

6.14.7. AgentCustomerInformationCenter::Backend###0100-CIC-TicketPendingReminder

Description:	Parameters for the dashboard backend of the ticket pending reminder overview of the agent interface . "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'AgentCustomerInformationCenter::Backend'}->{'0100-CIC-
TicketPendingReminder'} =  {
  'Attributes' => 'TicketPendingTimeOlderMinutes=1;StateType=pending
 reminder;SortBy=PendingTime;OrderBy=Down;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All tickets with a reminder set where the reminder
 date has been reached',
  'Filter' => 'Locked',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'ro',
  'Time' => 'UntilTime',
  'Title' => 'Reminder Tickets'
};
```

6.14.8. AgentCustomerInformationCenter::Backend###0110-CIC-TicketEscalation

Description:	Parameters for the dashboard backend of the ticket escalation overview of the agent interface . "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'AgentCustomerInformationCenter::Backend'}->{'0110-CIC-
TicketEscalation'} =  {
  'Attributes' =>
 'TicketEscalationTimeOlderMinutes=1;SortBy=EscalationTime;OrderBy=Down;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All escalated tickets',
  'Filter' => 'All',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'ro',
  'Time' => 'EscalationTime',
  'Title' => 'Escalated Tickets'
};
```

6.14.9. AgentCustomerInformationCenter::Backend###0120-CIC-TicketNew

Description:	Parameters for the dashboard backend of the new tickets overview of the agent interface. "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'AgentCustomerInformationCenter::Backend'}->{'0120-CIC-
TicketNew'} =  {
  'Attributes' => 'StateType=new;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All new tickets, these tickets have not been
 worked on yet',
  'Filter' => 'All',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'ro',
  'Time' => 'Age',
  'Title' => 'New Tickets'
};
```

6.14.10. AgentCustomerInformationCenter::Backend###0130-CIC-TicketOpen

Description:	Parameters for the dashboard backend of the ticket pending reminder overview of the agent interface. "Limit" is the number of entries shown by default. "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'AgentCustomerInformationCenter::Backend'}->{'0130-CIC-
TicketOpen'} =  {
  'Attributes' => 'StateType=open;',
  'Block' => 'ContentLarge',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'All open tickets, these tickets have already been
 worked on, but need a response',
  'Filter' => 'All',
  'Group' => '',
  'Limit' => '10',
  'Module' => 'Kernel::Output::HTML::DashboardTicketGeneric',
  'Permission' => 'ro',
  'Time' => 'Age',
  'Title' => 'Open Tickets / Need to be answered'
};
```

6.14.11. AgentCustomerInformationCenter::Backend###0500-CIC-CustomerIDStatus

Description:	Parameters for the dashboard backend of the customer id status widget of the agent interface . "Group" is used to restrict the access to the plugin (e. g. Group: admin;group1;group2;). "Default" determines if the plugin is enabled by default or if the user needs to enable it manually. "CacheTTLLocal" is the cache time in minutes for the plugin.
Group:	Ticket
SubGroup:	Frontend::Agent::Dashboard
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'AgentCustomerInformationCenter::Backend'}->{'0500-CIC-
CustomerIDStatus'} =  {
  'Attributes' => '',
  'Block' => 'ContentSmall',
  'CacheTTLLocal' => '0.5',
  'Default' => '1',
  'Description' => 'Company Status',
  'Group' => '',
  'Module' => 'Kernel::Output::HTML::DashboardCustomerIDStatus',
  'Permission' => 'ro',
  'Title' => 'Company Status'
};
```

6.15. Frontend::Agent::ModuleMetaHead

6.15.1. Frontend::HeaderMetaModule###2-TicketSearch

Description:	Module to generate html OpenSearch profile for short ticket search in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleMetaHead
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::HeaderMetaModule'}->{'2-TicketSearch'} =  {
  'Action' => 'AgentTicketSearch',
  'Module' => 'Kernel::Output::HTML::HeaderMetaTicketSearch'
};
```

6.16. Frontend::Agent::ModuleNotify

6.16.1. Frontend::NotifyModule###5-Ticket::TicketEscalation

Description:	Module to show notifications and escalations (ShownMax: max. shown escalations, EscalationInMinutes: Show ticket which will escalation in, CacheTime: Cache of calculated escalations in seconds).
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleNotify
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::NotifyModule'}->{'5-Ticket::TicketEscalation'} =
{
  'CacheTime' => '40',
  'EscalationInMinutes' => '120',
  'Module' =>
'Kernel::Output::HTML::NotificationAgentTicketEscalation',
  'ShownMax' => '25'
};
```

6.17. Frontend::Agent::ModuleRegistration

6.17.1. Frontend::Module###AgentTicketQueue

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1

Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AgentTicketQueue'} =  {
  'Description' => 'Overview of all open Tickets',
  'Loader' => {
    'CSS' => [
      'Core.AgentTicketQueue.css'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'o',
      'Block' => '',
      'Description' => 'Overview of all open Tickets',
      'Link' => 'Action=AgentTicketQueue',
      'LinkOption' => '',
      'Name' => 'Queue view',
      'NavBar' => 'Ticket',
      'Prio' => '100',
      'Type' => ''
    },
    {
      'AccessKey' => 't',
      'Block' => 'ItemArea',
      'Description' => '',
      'Link' => 'Action=AgentTicketQueue',
      'LinkOption' => '',
      'Name' => 'Tickets',
      'NavBar' => 'Ticket',
      'Prio' => '200',
      'Type' => 'Menu'
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'QueueView'
};
```

6.17.2. Frontend::Module###AgentTicketPhone

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AgentTicketPhone'} =  {
  'Description' => 'Create new phone ticket',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.CustomerSearch.js',
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'n',
      'Block' => '',
      'Description' => 'Create new phone ticket (inbound)',
      'Link' => 'Action=AgentTicketPhone',
      'LinkOption' => '',
      'Name' => 'New phone ticket',
      'NavBar' => 'Ticket',
      'Prio' => '200',
      'Type' => ''
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'New phone ticket'
};
```

6.17.3. Frontend::Module###AgentTicketPhoneOutbound

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketPhoneOutbound'} =  {
  'Description' => 'Phone Call',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Phone-Ticket'
};
```

6.17.4. Frontend::Module###AgentTicketPhoneInbound

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketPhoneInbound'} =  {
  'Description' => 'Incoming Phone Call',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Phone-Ticket'
};
```

6.17.5. Frontend::Module###AgentTicketEmail

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketEmail'} =  {
  'Description' => 'Create new email ticket',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.CustomerSearch.js',
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBar' => [
    {
      'AccessKey' => 'm',
      'Block' => '',
      'Description' => 'Create new email ticket and send this out
 (outbound)',
      'Link' => 'Action=AgentTicketEmail',
      'LinkOption' => '',
      'Name' => 'New email ticket',
      'NavBar' => 'Ticket',
      'Prio' => '210',
      'Type' => ''
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'New email ticket'
};
```

6.17.6. Frontend::Module###AgentTicketSearch

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketSearch'} =  {
  'Description' => 'Search Ticket',
  'NavBar' => [
    {
      'AccessKey' => 's',
      'Block' => '',
      'Description' => 'Search Tickets',
      'Link' => 'Action=AgentTicketSearch',
      'LinkOption' => 'onclick="window.setTimeout(function()
{Core.Agent.Search.OpenSearchDialog(\'AgentTicketSearch\');}, 0);
 return false;"',
      'Name' => 'Search',
      'NavBar' => 'Ticket',
      'Prio' => '300',
      'Type' => ''
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'Search'
};
```

6.17.7. Frontend::Module###AgentTicketLockedView

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketLockedView'} =  {
  'Description' => 'Locked Tickets',
  'NavBarName' => 'Ticket',
  'Title' => 'Locked Tickets'
};
```

6.17.8. Frontend::Module###AgentTicketResponsibleView

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketResponsibleView'} =  {
  'Description' => 'Responsible Tickets',
  'NavBarName' => 'Ticket',
  'Title' => 'Responsible Tickets'
};
```

6.17.9. Frontend::Module###AgentTicketWatchView

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketWatchView'} =  {
  'Description' => 'Watched Tickets',
  'NavBarName' => 'Ticket',
  'Title' => 'Watched Tickets'
};
```

6.17.10. Frontend::Module###AgentCustomerSearch

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentCustomerSearch'} =  {
  'Description' => 'AgentCustomerSearch',
  'NavBarName' => 'Ticket',
  'Title' => 'AgentCustomerSearch'
};
```

6.17.11. Frontend::Module###AgentTicketStatusView

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketStatusView'} =  {
  'Description' => 'Overview of all open tickets',
  'NavBar' => [
    {
      'AccessKey' => 'v',
      'Block' => '',
      'Description' => 'Overview of all open Tickets.',
      'Link' => 'Action=AgentTicketStatusView',
      'LinkOption' => '',
      'Name' => 'Status view',
      'NavBar' => 'Ticket',
      'Prio' => '110',
      'Type' => ''
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'Status view'
};
```

6.17.12. Frontend::Module###AgentTicketEscalationView

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketEscalationView'} =  {
  'Description' => 'Overview of all escalated tickets',
  'NavBar' => [
    {
      'AccessKey' => 'e',
      'Block' => '',
      'Description' => 'Overview Escalated Tickets',
      'Link' => 'Action=AgentTicketEscalationView',
      'LinkOption' => '',
      'Name' => 'Escalation view',
      'NavBar' => 'Ticket',
      'Prio' => '120',
      'Type' => ''
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'Escalation view'
};
```

6.17.13. Frontend::Module###AgentZoom

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentZoom'} =  {
  'Description' => 'compat module for AgentZoom to AgentTicketZoom',
  'NavBarName' => 'Ticket',
  'Title' => ''
};
```

6.17.14. Frontend::Module###AgentTicketZoom

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AgentTicketZoom'} =  {
  'Description' => 'Ticket Zoom',
  'Loader' => {
    'CSS' => [
      'Core.Agent.TicketProcess.css'
    ],
    'JavaScript' => [
      'thirdparty/jquery-tablesorter-2.0.5/jquery.tablesorter.js',
      'Core.UI.Table.Sort.js',
      'Core.Agent.TicketZoom.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Zoom'
};
```

6.17.15. Frontend::Module###AgentTicketAttachment

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AgentTicketAttachment'} =  {
  'Description' => 'To download attachments',
  'NavBarName' => 'Ticket',
  'Title' => ''
};
```

6.17.16. Frontend::Module###AgentTicketPlain

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Frontend::Module'}->{'AgentTicketPlain'} =  {
  'Description' => 'Ticket plain view of an email',
  'NavBarName' => 'Ticket',
  'Title' => 'Plain'
};
```

6.17.17. Frontend::Module###AgentTicketNote

Description:	Frontend module registration for the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentTicketNote'} =  {
  'Description' => 'Ticket Note',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Note'
};
```

6.17.18. Frontend::Module###AgentTicketMerge

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentTicketMerge'} =  {
  'Description' => 'Ticket Merge',
  'NavBarName' => 'Ticket',
  'Title' => 'Merge'
};
```

6.17.19. Frontend::Module###AgentTicketPending

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentTicketPending'} =  {
  'Description' => 'Ticket Pending',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Pending'
};
```

6.17.20. Frontend::Module###AgentTicketWatcher

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentTicketWatcher'} =  {
  'Description' => 'A TicketWatcher Module',
  'NavBarName' => 'Ticket-Watcher',
  'Title' => 'Ticket-Watcher'
};
```

6.17.21. Frontend::Module###AgentTicketPriority

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentTicketPriority'} =  {
  'Description' => 'Ticket Priority',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Priority'
};
```

6.17.22. Frontend::Module###AgentTicketLock

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentTicketLock'} =  {
  'Description' => 'Ticket Lock',
  'NavBarName' => 'Ticket',
  'Title' => 'Lock'
};
```

6.17.23. Frontend::Module###AgentTicketMove

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::Module'}->{'AgentTicketMove'} =  {
  'Description' => 'Ticket Move',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Move'
};
```

6.17.24. Frontend::Module###AgentTicketHistory

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketHistory'} =  {
  'Description' => 'Ticket History',
  'NavBarName' => 'Ticket',
  'Title' => 'History'
};
```

6.17.25. Frontend::Module###AgentTicketOwner

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketOwner'} =  {
  'Description' => 'Ticket Owner',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Owner'
};
```

6.17.26. Frontend::Module###AgentTicketResponsible

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Module'}->{'AgentTicketResponsible'} =  {
  'Description' => 'Ticket Responsible',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Responsible'
};
```

6.17.27. Frontend::Module###AgentTicketCompose

Description:	Frontend module registration for the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketCompose'} =  {
  'Description' => 'Ticket Compose email Answer',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.CustomerSearch.js',
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Compose'
};
```

6.17.28. Frontend::Module###AgentTicketBounce

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketBounce'} =  {
  'Description' => 'Ticket Compose Bounce Email',
  'NavBarName' => 'Ticket',
  'Title' => 'Bounce'
};
```

6.17.29. Frontend::Module###AgentTicketForward

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketForward'} =  {
  'Description' => 'Ticket Forward Email',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.CustomerSearch.js',
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Forward'
};
```

6.17.30. Frontend::Module###AgentTicketCustomer

Description:	Frontend module registration for the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketCustomer'} =  {
  'Description' => 'Ticket Customer',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.CustomerSearch.js',
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Customer'
};
```

6.17.31. Frontend::Module###AgentTicketClose

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketClose'} =  {
  'Description' => 'Ticket Close',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Close'
};
```

6.17.32. Frontend::Module###AgentTicketFreeText

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketFreeText'} =  {
  'Description' => 'Ticket FreeText',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Free Fields'
};
```

6.17.33. Frontend::Module###AgentTicketPrint

Description:	Frontend module registration for the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketPrint'} =  {
  'Description' => 'Ticket Print',
  'NavBarName' => 'Ticket',
  'Title' => 'Print'
};
```

6.17.34. Frontend::Module###AgentTicketBulk

Description:	Frontend module registration for the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::ModuleRegistration
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::Module'}->{'AgentTicketBulk'} =  {
  'Description' => 'Ticket bulk module',
  'Loader' => {
    'JavaScript' => [
      'Core.Agent.TicketAction.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Bulk-Action'
};
```

6.18. Frontend::Agent::Preferences

6.18.1. PreferencesGroups###NewTicketNotify

Description:	Parameters for the NewTicketNotify object in the preferences view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'PreferencesGroups'}->{'NewTicketNotify'} =  {
  'Active' => '1',
  'Column' => 'Email Settings',
  'Data' => {
    '0' => 'No',
    '1' => 'Yes'
  },
  'DataSelected' => '0',
  'Desc' => 'Send me a notification if there is a new ticket in "My
 Queues".',
  'Key' => 'Send new ticket notifications',
  'Label' => 'New ticket notification',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserSendNewTicketNotification',
  'Prio' => '1000'
};
```

6.18.2. PreferencesGroups###FollowUpNotify

Description:	Parameters for the FollowUpNotify object in the preference view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'PreferencesGroups'}->{'FollowUpNotify'} =  {
  'Active' => '1',
  'Column' => 'Email Settings',
  'Data' => {
    '0' => 'No',
    '1' => 'Yes'
  },
  'DataSelected' => '0',
  'Desc' => 'Send me a notification if a customer sends a follow up
and I\'m the owner of the ticket or the ticket is unlocked and is in
one of my subscribed queues.',
  'Key' => 'Send ticket follow up notifications',
  'Label' => 'Ticket follow up notification',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserSendFollowUpNotification',
  'Prio' => '2000'
};
```

6.18.3. PreferencesGroups###LockTimeoutNotify

Description:	Parameters for the LockTimeoutNotify object in the preference view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'PreferencesGroups'}->{'LockTimeoutNotify'} =  {
  'Active' => '1',
  'Column' => 'Email Settings',
  'Data' => {
    '0' => 'No',
    '1' => 'Yes'
  },
  'DataSelected' => '0',
  'Desc' => 'Send me a notification if a ticket is unlocked by the
system.',
  'Key' => 'Send ticket lock timeout notifications',
  'Label' => 'Ticket lock timeout notification',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserSendLockTimeoutNotification',
  'Prio' => '3000'
};
```

6.18.4. PreferencesGroups###MoveNotify

Description:	Parameters for the MoveNotify object in the preference view of the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'PreferencesGroups'}->{'MoveNotify'} =  {
  'Active' => '1',
  'Column' => 'Email Settings',
  'Data' => {
    '0' => 'No',
    '1' => 'Yes'
  },
  'DataSelected' => '0',
  'Desc' => 'Send me a notification if a ticket is moved into one of
"My Queues".',
  'Key' => 'Send ticket move notifications',
  'Label' => 'Ticket move notification',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserSendMoveNotification',
  'Prio' => '4000'
};
```

6.18.5. PreferencesGroups###WatcherNotify

Description:	Parameters for the WatcherNotify object in the preference view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'PreferencesGroups'}->{'WatcherNotify'} =  {
  'Active' => '1',
  'Column' => 'Email Settings',
  'Data' => {
    '0' => 'No',
    '1' => 'Yes'
  },
  'DataSelected' => '0',
  'Desc' => 'Send me the same notifications for my watched tickets
that the ticket owners will get.',
  'Key' => 'Send ticket watch notifications',
  'Label' => 'Ticket watch notification',
  'Module' => 'Kernel::Output::HTML::PreferencesTicketWatcher',
  'PrefKey' => 'UserSendWatcherNotification',
  'Prio' => '5000'
};
```

6.18.6. PreferencesGroups###CustomQueue

Description:	Parameters for the CustomQueue object in the preference view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'PreferencesGroups'}->{'CustomQueue'} =  {
  'Active' => '1',
  'Column' => 'Other Settings',
  'Desc' => 'Your queue selection of your favorite queues. You also
get notified about those queues via email if enabled.',
  'Key' => 'My Queues',
  'Label' => 'My Queues',
  'Module' => 'Kernel::Output::HTML::PreferencesCustomQueue',
  'Permission' => 'ro',
  'Prio' => '1000'
};
```

6.18.7. PreferencesGroups###RefreshTime

Description:	Parameters for the RefreshTime object in the preference view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'PreferencesGroups'}->{'RefreshTime'} =  {
  'Active' => '1',
  'Column' => 'Other Settings',
  'Data' => {
    '0' => 'off',
    '10' => '10 minutes',
    '15' => '15 minutes',
    '2' => ' 2 minutes',
    '5' => ' 5 minutes',
    '7' => ' 7 minutes'
  },
  'DataSelected' => '0',
  'Desc' => 'If enabled, the different overviews (Dashboard,
LockedView, QueueView) will automatically refresh after the specified
time.',
  'Key' => 'Refresh Overviews after',
  'Label' => 'Overview Refresh Time',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserRefreshTime',
  'Prio' => '2000'
};
```

6.18.8. PreferencesGroups###TicketOverviewSmallPageShown

Description:	Parameters for the pages (in which the tickets are shown) of the small ticket overview.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'PreferencesGroups'}->{'TicketOverviewSmallPageShown'} =  {
  'Active' => '0',
  'Column' => 'Other Settings',
  'Data' => {
    '10' => '10',
    '15' => '15',
    '20' => '20',
    '25' => '25',
    '30' => '30',
    '35' => '35'
  },
  'DataSelected' => '25',
  'Key' => 'Ticket limit per page for Ticket Overview "Small"',
  'Label' => 'Ticket Overview "Small" Limit',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserTicketOverviewSmallPageShown',
  'Prio' => '8000'
};
```

6.18.9. PreferencesGroups###TicketOverviewMediumPageShown

Description:	Parameters for the pages (in which the tickets are shown) of the medium ticket overview.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'PreferencesGroups'}->{'TicketOverviewMediumPageShown'} =  {
  'Active' => '0',
  'Column' => 'Other Settings',
  'Data' => {
    '10' => '10',
    '15' => '15',
    '20' => '20',
    '25' => '25',
    '30' => '30',
    '35' => '35'
  },
  'DataSelected' => '20',
  'Key' => 'Ticket limit per page for Ticket Overview "Medium"',
  'Label' => 'Ticket Overview "Medium" Limit',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserTicketOverviewMediumPageShown',
  'Prio' => '8100'
};
```

6.18.10. PreferencesGroups###TicketOverviewPreviewPageShown

Description:	Parameters for the pages (in which the tickets are shown) of the ticket preview overview.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'PreferencesGroups'}->{'TicketOverviewPreviewPageShown'} =  {
  'Active' => '0',
  'Column' => 'Other Settings',
  'Data' => {
    '10' => '10',
    '15' => '15',
    '20' => '20',
    '25' => '25',
    '30' => '30',
    '35' => '35'
  },
  'DataSelected' => '15',
  'Key' => 'Ticket limit per page for Ticket Overview "Preview"',
  'Label' => 'Ticket Overview "Preview" Limit',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserTicketOverviewPreviewPageShown',
  'Prio' => '8200'
};
```

6.18.11. PreferencesGroups###CreateNextMask

Description:	Parameters for the CreateNextMask object in the preference view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'PreferencesGroups'}->{'CreateNextMask'} =  {
  'Active' => '1',
  'Column' => 'Other Settings',
  'Data' => {
    '' => 'CreateTicket',
    'AgentTicketZoom' => 'TicketZoom'
  },
  'DataSelected' => '',
  'Key' => 'Show this screen after I created a new ticket',
  'Label' => 'Screen after new ticket',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserCreateNextMask',
  'Prio' => '3000'
};
```

6.19. Frontend::Agent::SearchRouter

6.19.1. Frontend::Search###AgentCustomerInformationCenter

Description:	Search backend router.
Group:	Ticket
SubGroup:	Frontend::Agent::SearchRouter
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Search'}->{'AgentCustomerInformationCenter'} =  {
  '^AgentCustomerInformationCenter' =>
 'Action=AgentCustomerInformationCenter'
};
```

6.19.2. Frontend::Search###Ticket

Description:	Search backend router.
Group:	Ticket
SubGroup:	Frontend::Agent::SearchRouter
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Frontend::Search'}->{'Ticket'} =  {
  '^AgentTicket' => 'Action=AgentTicketSearch;Subaction=AJAX'
};
```

6.20. Frontend::Agent::Ticket::ArticleAttachmentModule

6.20.1. Ticket::Frontend::ArticleAttachmentModule###1-Download

Description:	Shows a link to download article attachments in the zoom view of the article in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleAttachmentModule
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::Frontend::ArticleAttachmentModule'}->{'1-Download'}
 =  {
  'Module' => 'Kernel::Output::HTML::ArticleAttachmentDownload'
};
```

6.20.2. Ticket::Frontend::ArticleAttachmentModule###2-HTML-Viewer

Description:	Shows a link to access article attachments via a html online viewer in the zoom view of the article in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleAttachmentModule
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::Frontend::ArticleAttachmentModule'}->{'2-HTML-
Viewer'} =  {
  'Module' => 'Kernel::Output::HTML::ArticleAttachmentHTMLViewer'
};
```

6.21. Frontend::Agent::Ticket::ArticleComposeModule

6.21.1. Ticket::Frontend::ArticleComposeModule###1-SignEmail

Description:	Module to compose signed messages (PGP or S/MIME).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleComposeModule
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::Frontend::ArticleComposeModule'}->{'1-SignEmail'} =
 {
  'Module' => 'Kernel::Output::HTML::ArticleComposeSign'
};
```

6.21.2. Ticket::Frontend::ArticleComposeModule###2-CryptEmail

Description:	Module to crypt composed messages (PGP or S/MIME).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleComposeModule
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::ArticleComposeModule'}->{'2-CryptEmail'} =
  {
  'Module' => 'Kernel::Output::HTML::ArticleComposeCrypt'
};
```

6.22. Frontend::Agent::Ticket::ArticleViewModule

6.22.1. Ticket::Frontend::ArticleViewModule###1-PGP

Description:	Agent interface article notification module to check PGP.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleViewModule
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::ArticleViewModule'}->{'1-PGP'} =  {
  'Module' => 'Kernel::Output::HTML::ArticleCheckPGP'
};
```

6.22.2. Ticket::Frontend::ArticleViewModule###1-SMIME

Description:	Agent interface module to check incoming emails in the Ticket-Zoom-View if the S/MIME-key is available and true.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleViewModule
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::ArticleViewModule'}->{'1-SMIME'} =  {
  'Module' => 'Kernel::Output::HTML::ArticleCheckSMIME'
};
```

6.23. Frontend::Agent::Ticket::ArticleViewModulePre

6.23.1. Ticket::Frontend::ArticlePreViewModule###1-PGP

Description:	Agent interface article notification module to check PGP.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleViewModulePre
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::ArticlePreViewModule'}->{'1-PGP'} =  {
  'Module' => 'Kernel::Output::HTML::ArticleCheckPGP'
};
```

6.23.2. Ticket::Frontend::ArticlePreViewModule###1-SMIME

Description:	Agent interface article notification module to check S/MIME.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ArticleViewModulePre
Valid:	1
Required:	1

Config-Setting:	

```
$Self->{'Ticket::Frontend::ArticlePreViewModule'}->{'1-SMIME'} =  {
  'Module' => 'Kernel::Output::HTML::ArticleCheckSMIME'
};
```

6.24. Frontend::Agent::Ticket::MenuModule

6.24.1. Ticket::Frontend::MenuModule###000-Back

Description:	Shows a link in the menu to go back in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::MenuModule'}->{'000-Back'} =  {
  'Action' => '',
  'Description' => 'Back',
  'Link' => '$Env{"LastScreenOverview"};TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Back',
  'PopupType' => '',
  'Target' => ''
};
```

6.24.2. Ticket::Frontend::MenuModule###100-Lock

Description:	Shows a link in the menu to lock/unlock tickets in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::MenuModule'}->{'100-Lock'} =  {
  'Action' => 'AgentTicketLock',
  'Module' => 'Kernel::Output::HTML::TicketMenuLock',
  'Name' => 'Lock',
  'Target' => ''
};
```

6.24.3. Ticket::Frontend::MenuModule###200-History

Description:	Shows a link in the menu to access the history of a ticket in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::MenuModule'}->{'200-History'} =  {
  'Action' => 'AgentTicketHistory',
  'Description' => 'Show the ticket history',
  'Link' => 'Action=AgentTicketHistory;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'History',
  'PopupType' => 'TicketHistory',
  'Target' => ''
};
```

6.24.4. Ticket::Frontend::MenuModule###210-Print

Description:	Shows a link in the menu to print a ticket or an article in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::MenuModule'}->{'210-Print'} =  {
  'Action' => 'AgentTicketPrint',
  'Description' => 'Print this ticket',
  'Link' => 'Action=AgentTicketPrint;TicketID=$QData{"TicketID"}',
  'LinkParam' => 'target="print"',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Print',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.5. Ticket::Frontend::MenuModule###300-Priority

Description:	Shows a link in the menu to see the priority of a ticket in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::MenuModule'}->{'300-Priority'} =  {
  'Action' => 'AgentTicketPriority',
  'Description' => 'Change the ticket priority',
  'Link' => 'Action=AgentTicketPriority;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Priority',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.6. Ticket::Frontend::MenuModule###310-FreeText

Description:	Shows a link in the menu to add a free text field in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::MenuModule'}->{'310-FreeText'} =  {
  'Action' => 'AgentTicketFreeText',
  'Description' => 'Change the free fields for this ticket',
  'Link' => 'Action=AgentTicketFreeText;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Free Fields',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.7. Ticket::Frontend::MenuModule###320-Link

Description:	Shows a link in the menu that allows linking a ticket with another object in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'320-Link'} =  {
  'Action' => 'AgentLinkObject',
  'Description' => 'Link this ticket to other objects',
  'Link' => 'Action=AgentLinkObject;SourceObject=Ticket;SourceKey=
$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Link',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.8. Ticket::Frontend::MenuModule###400-Owner

Description:	Shows a link in the menu to see the owner of a ticket in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'400-Owner'} =  {
  'Action' => 'AgentTicketOwner',
  'Description' => 'Change the owner for this ticket',
  'Link' => 'Action=AgentTicketOwner;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Owner',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.9. Ticket::Frontend::MenuModule###410-Responsible

Description:	Shows a link in the menu to see the responsible agent of a ticket in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'410-Responsible'} =  {
  'Action' => 'AgentTicketResponsible',
  'Description' => 'Change the responsible person for this ticket',
  'Link' => 'Action=AgentTicketResponsible;TicketID=
$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuResponsible',
  'Name' => 'Responsible',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.10. Ticket::Frontend::MenuModule###420-Customer

Description:	Shows a link in the menu to see the customer who requested the ticket in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'420-Customer'} =  {
  'Action' => 'AgentTicketCustomer',
  'Description' => 'Change the customer for this ticket',
  'Link' => 'Action=AgentTicketCustomer;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Customer',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.11. Ticket::Frontend::MenuModule###420-Note

Description:	Shows a link in the menu to add a note in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'420-Note'} =  {
  'Action' => 'AgentTicketNote',
  'Description' => 'Add a note to this ticket',
  'Link' => 'Action=AgentTicketNote;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Note',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.12. Ticket::Frontend::MenuModule###425-Phone Call Outbound

Description:	Shows a link in the menu to add a note in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'425-Phone Call Outbound'}
 =  {
  'Action' => 'AgentTicketPhoneOutbound',
  'Description' => 'Phone Call Outbound',
  'Link' => 'Action=AgentTicketPhoneOutbound;TicketID=
$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Phone Call Outbound',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.13. Ticket::Frontend::MenuModule###426-Phone Call Inbound

Description:	Shows a link in the menu to add a note in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'426-Phone Call Inbound'} =
  {
  'Action' => 'AgentTicketPhoneInbound',
  'Description' => 'Phone Call Inbound',
  'Link' => 'Action=AgentTicketPhoneInbound;TicketID=
$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Phone Call Inbound',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.14. Ticket::Frontend::MenuModule###430-Merge

Description:	Shows a link in the menu that allows merging tickets in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'430-Merge'} =  {
  'Action' => 'AgentTicketMerge',
  'Description' => 'Merge into a different ticket',
  'Link' => 'Action=AgentTicketMerge;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Merge',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.15. Ticket::Frontend::MenuModule###440-Pending

Description:	Shows a link in the menu to set a ticket as pending in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'440-Pending'} =  {
  'Action' => 'AgentTicketPending',
  'Description' => 'Set this ticket to pending',
  'Link' => 'Action=AgentTicketPending;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Pending',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.16. Ticket::Frontend::MenuModule###448-Watch

Description:	Shows a link in the menu for subscribing / unsubscribing from a ticket in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'448-Watch'} =  {
  'Action' => 'AgentTicketWatcher',
  'Module' => 'Kernel::Output::HTML::TicketMenuTicketWatcher',
  'Name' => 'Watch',
  'Target' => ''
};
```

6.24.17. Ticket::Frontend::MenuModule###450-Close

Description:	Shows a link in the menu to close a ticket in the ticket zoom view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'450-Close'} =  {
  'Action' => 'AgentTicketClose',
  'Description' => 'Close this ticket',
  'Link' => 'Action=AgentTicketClose;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Close',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.24.18. Ticket::Frontend::MenuModule###460-Delete

Description:	Shows a link in the menu to delete a ticket in the ticket zoom view of the agent interface. Additional access control to show or not show this link can be done by using Key "Group" and Content like "rw:group1;move_into:group2".
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	0
Required:	0

Config-Setting:	

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'460-Delete'} =  {
  'Action' => 'AgentTicketMove',
  'Description' => 'Delete this ticket',
  'Link' => 'Action=AgentTicketMove;TicketID=
$Data{"TicketID"};DestQueue=Delete',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Delete',
  'PopupType' => '',
  'Target' => ''
};
```

6.24.19. Ticket::Frontend::MenuModule###470-Spam

Description:	Shows a link to set a ticket as spam in the ticket zoom view of the agent interface. Additional access control to show or not show this link can be done by using Key "Group" and Content like "rw:group1;move_into:group2".
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModule
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::Frontend::MenuModule'}->{'470-Spam'} =  {
  'Action' => 'AgentTicketMove',
  'Description' => 'Mark as Spam!',
  'Link' => 'Action=AgentTicketMove;TicketID=
$Data{"TicketID"};DestQueue=Delete',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Spam',
  'PopupType' => '',
  'Target' => ''
};
```

6.25. Frontend::Agent::Ticket::MenuModulePre

6.25.1. Ticket::Frontend::PreMenuModule###100-Lock

Description:	Shows a link in the menu to lock / unlock a ticket in the ticket overviews of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::Frontend::PreMenuModule'}->{'100-Lock'} =  {
  'Action' => 'AgentTicketLock',
  'Module' => 'Kernel::Output::HTML::TicketMenuLock',
  'Name' => 'Lock',
  'PopupType' => '',
  'Target' => ''
};
```

6.25.2. Ticket::Frontend::PreMenuModule###200-Zoom

Description:	Shows a link in the menu to zoom a ticket in the ticket overviews of the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::PreMenuModule'}->{'200-Zoom'} =  {
  'Action' => 'AgentTicketZoom',
  'Description' => 'Look into a ticket!',
  'Link' => 'Action=AgentTicketZoom;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Zoom',
  'PopupType' => '',
  'Target' => ''
};
```

6.25.3. Ticket::Frontend::PreMenuModule###210-History

Description:	Shows a link in the menu to see the history of a ticket in every ticket overview of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::PreMenuModule'}->{'210-History'} =  {
  'Action' => 'AgentTicketHistory',
  'Description' => 'Show the ticket history',
  'Link' => 'Action=AgentTicketHistory;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'History',
  'PopupType' => 'TicketHistory',
  'Target' => ''
};
```

6.25.4. Ticket::Frontend::PreMenuModule###300-Priority

Description:	Shows a link in the menu to set the priority of a ticket in every ticket overview of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::PreMenuModule'}->{'300-Priority'} =  {
  'Action' => 'AgentTicketPriority',
  'Description' => 'Change the priority for this ticket',
  'Link' => 'Action=AgentTicketPriority;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Priority',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.25.5. Ticket::Frontend::PreMenuModule###420-Note

Description:	Shows a link in the menu to add a note to a ticket in every ticket overview of the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::PreMenuModule'}->{'420-Note'} =  {
  'Action' => 'AgentTicketNote',
  'Description' => 'Add a note to this ticket',
  'Link' => 'Action=AgentTicketNote;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Note',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.25.6. Ticket::Frontend::PreMenuModule###440-Close

Description:	Shows a link in the menu to close a ticket in every ticket overview of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::PreMenuModule'}->{'440-Close'} =  {
  'Action' => 'AgentTicketClose',
  'Description' => 'Close this ticket',
  'Link' => 'Action=AgentTicketClose;TicketID=$QData{"TicketID"}',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Close',
  'PopupType' => 'TicketAction',
  'Target' => ''
};
```

6.25.7. Ticket::Frontend::PreMenuModule###445-Move

Description:	Shows a link in the menu to move a ticket in every ticket overview of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::PreMenuModule'}->{'445-Move'} =  {
  'Action' => 'AgentTicketMove',
  'Description' => 'Change queue!',
  'Module' => 'Kernel::Output::HTML::TicketMenuMove',
  'Name' => 'Move'
};
```

6.25.8. Ticket::Frontend::PreMenuModule###450-Delete

Description:	Shows a link in the menu to delete a ticket in every ticket overview of the agent interface. Additional access control to show or not show this link can be done by using Key "Group" and Content like "rw:group1;move_into:group2".
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::PreMenuModule'}->{'450-Delete'} =  {
  'Action' => 'AgentTicketMove',
  'Description' => 'Delete this ticket',
  'Link' => 'Action=AgentTicketMove;TicketID=
$Data{"TicketID"};DestQueue=Delete',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Delete',
  'PopupType' => '',
  'Target' => ''
};
```

6.25.9. Ticket::Frontend::PreMenuModule###460-Spam

Description:	Shows a link in the menu to set a ticket as spam in every ticket overview of the agent interface. Additional access control to show or not show this link can be done by using Key "Group" and Content like "rw:group1;move_into:group2".
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::MenuModulePre
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::PreMenuModule'}->{'460-Spam'} =  {
  'Action' => 'AgentTicketMove',
  'Description' => 'Mark as Spam!',
  'Link' => 'Action=AgentTicketMove;TicketID=
$Data{"TicketID"};DestQueue=Delete',
  'Module' => 'Kernel::Output::HTML::TicketMenuGeneric',
  'Name' => 'Spam',
  'PopupType' => '',
  'Target' => ''
};
```

6.26. Frontend::Agent::Ticket::ViewBounce

6.26.1. Ticket::Frontend::AgentTicketBounce###Permission

Description:	Required permissions to use the ticket bounce screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBounce
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketBounce'}->{'Permission'} =
 'bounce';
```

6.26.2. Ticket::Frontend::AgentTicketBounce###RequiredLock

Description:	Defines if a ticket lock is required in the ticket bounce screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBounce
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBounce'}->{'RequiredLock'} = '1';`

6.26.3. Ticket::Frontend::AgentTicketBounce###StateDefault

Description:	Defines the default next state of a ticket after being bounced, in the ticket bounce screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBounce
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBounce'}->{'StateDefault'} = 'closed successful';`

6.26.4. Ticket::Frontend::AgentTicketBounce###StateType

Description:	Defines the next state of a ticket after being bounced, in the ticket bounce screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBounce
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketBounce'}->{'StateType'} =  [
  'open',
  'closed'
];
```

6.26.5. Ticket::Frontend::BounceText

Description:	Defines the default ticket bounced notification for customer/sender in the ticket bounce screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBounce
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::BounceText'} =  'Your email with ticket number "<OTRS_TICKET>" is bounced to "<OTRS_BOUNCE_TO>". Contact this address for further information.';`

6.27. Frontend::Agent::Ticket::ViewBulk

6.27.1. Ticket::Frontend::AgentTicketBulk###RequiredLock

Description:	Automatically lock and set owner to current Agent after selecting for an Bulk Action.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'RequiredLock'} = '1';`

6.27.2. Ticket::Frontend::AgentTicketBulk###TicketType

Description:	Sets the ticket type in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'TicketType'} =  '1';`

6.27.3. Ticket::Frontend::AgentTicketBulk###Owner

Description:	Sets the ticket owner in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'Owner'} =  '1';`

6.27.4. Ticket::Frontend::AgentTicketBulk###Responsible

Description:	Sets the responsible agent of the ticket in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'Responsible'} =  '1';`

6.27.5. Ticket::Frontend::AgentTicketBulk###State

Description:	If a note is added by an agent, sets the state of a ticket in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'State'} =  '1';`

6.27.6. Ticket::Frontend::AgentTicketBulk###StateType

Description:	Defines the next state of a ticket after adding a note, in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'StateType'} =  [
  'open',
  'closed',
  'pending reminder',
  'pending auto'
];
```

6.27.7. Ticket::Frontend::AgentTicketBulk###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'StateDefault'} = 'open';`

6.27.8. Ticket::Frontend::AgentTicketBulk###Priority

Description:	Shows the ticket priority options in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'Priority'} =  '1';`

6.27.9. Ticket::Frontend::AgentTicketBulk###PriorityDefault

Description:	Defines the default ticket priority in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'PriorityDefault'} = '3 normal';`

6.27.10. Ticket::Frontend::AgentTicketBulk###ArticleTypeDefault

Description:	Defines the default type of the note in the ticket bulk screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'ArticleTypeDefault'} =  'note-internal';`

6.27.11. Ticket::Frontend::AgentTicketBulk###ArticleTypes

Description:	Specifies the different note types that will be used in the system.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewBulk
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketBulk'}->{'ArticleTypes'} =  {
  'note-external' => '1',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.28. Frontend::Agent::Ticket::ViewClose

6.28.1. Ticket::Frontend::AgentTicketClose###Permission

Description:	Required permissions to use the close ticket screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Permission'} = 'close';`

6.28.2. Ticket::Frontend::AgentTicketClose###RequiredLock

Description:	Defines if a ticket lock is required in the close ticket screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'RequiredLock'} = '1';`

6.28.3. Ticket::Frontend::AgentTicketClose###TicketType

Description:	Sets the ticket type in the close ticket screen of the agent interface (Ticket::Type needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'TicketType'} =  '0';`

6.28.4. Ticket::Frontend::AgentTicketClose###Service

Description:	Sets the service in the close ticket screen of the agent interface (Ticket::Service needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Service'} =  '0';`

6.28.5. Ticket::Frontend::AgentTicketClose###Queue

Description:	Sets the queue in the ticket close screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Queue'} =  '0';`

6.28.6. Ticket::Frontend::AgentTicketClose###Owner

Description:	Sets the ticket owner in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Owner'} =  '0';`

6.28.7. Ticket::Frontend::AgentTicketClose###OwnerMandatory

Description:	Sets if ticket owner must be selected by the agent.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'OwnerMandatory'} = '0';`

6.28.8. Ticket::Frontend::AgentTicketClose###Responsible

Description:	Sets the responsible agent of the ticket in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Responsible'} = '0';`

6.28.9. Ticket::Frontend::AgentTicketClose###State

Description:	If a note is added by an agent, sets the state of a ticket in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'State'} =  '1';`

6.28.10. Ticket::Frontend::AgentTicketClose###StateType

Description:	Defines the next state of a ticket after adding a note, in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketClose'}->{'StateType'} =  [
  'closed'
];
```

6.28.11. Ticket::Frontend::AgentTicketClose###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'StateDefault'} = 'closed successful';`

6.28.12. Ticket::Frontend::AgentTicketClose###Note

Description:	Allows adding notes in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Note'} =  '1';`

6.28.13. Ticket::Frontend::AgentTicketClose###Subject

Description:	Sets the default subject for notes added in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Subject'} = '$Text{"Close"}';`

6.28.14. Ticket::Frontend::AgentTicketClose###Body

Description:	Sets the default body text for notes added in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Body'} =  '';`

6.28.15. Ticket::Frontend::AgentTicketClose###InvolvedAgent

Description:	Shows a list of all the involved agents on this ticket, in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'InvolvedAgent'} = '0';`

6.28.16. Ticket::Frontend::AgentTicketClose###InformAgent

Description:	Shows a list of all the possible agents (all agents with note permissions on the queue/ticket) to determine who should be informed about this note, in the close ticket screen of the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'InformAgent'} = '0';`

6.28.17. Ticket::Frontend::AgentTicketClose###ArticleTypeDefault

Description:	Defines the default type of the note in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'ArticleTypeDefault'} =  'note-internal';`

6.28.18. Ticket::Frontend::AgentTicketClose###ArticleTypes

Description:	Specifies the available note types for this ticket mask. If the option is deselected, ArticleTypeDefault is used and the option is removed from the mask.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Ticket::Frontend::AgentTicketClose'}->{'ArticleTypes'} =  {
  'note-external' => '0',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.28.19. Ticket::Frontend::AgentTicketClose###Priority

Description:	Shows the ticket priority options in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Priority'} =  '0';`

6.28.20. Ticket::Frontend::AgentTicketClose###PriorityDefault

Description:	Defines the default ticket priority in the close ticket screen of the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'PriorityDefault'} = '3 normal';`

6.28.21. Ticket::Frontend::AgentTicketClose###Title

Description:	Shows the title fields in the close ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'Title'} =  '0';`

6.28.22. Ticket::Frontend::AgentTicketClose###HistoryType

Description:	Defines the history type for the close ticket screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'HistoryType'} = 'AddNote';`

6.28.23. Ticket::Frontend::AgentTicketClose###HistoryComment

Description:	Defines the history comment for the close ticket screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'HistoryComment'} = '%%Close';`

6.28.24. Ticket::Frontend::AgentTicketClose###DynamicField

Description:	Dynamic fields shown in the ticket close screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'DynamicField'} = {};`

6.28.25. Ticket::Frontend::AgentTicketClose###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'RichTextWidth'} = '620';`

6.28.26. Ticket::Frontend::AgentTicketClose###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewClose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketClose'}->{'RichTextHeight'} = '100';`

6.29. Frontend::Agent::Ticket::ViewCompose

6.29.1. Ticket::Frontend::AgentTicketCompose###Permission

Description:	Required permissions to use the ticket compose screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketCompose'}->{'Permission'} = 'compose';`

6.29.2. Ticket::Frontend::AgentTicketCompose###RequiredLock

Description:	Defines if a ticket lock is required in the ticket compose screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1

Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketCompose'}->{'RequiredLock'} =
'1';
```

6.29.3. Ticket::Frontend::AgentTicketCompose###StateDefault

Description:	Defines the default next state of a ticket if it is composed / answered in the ticket compose screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketCompose'}->{'StateDefault'} =
'open';
```

6.29.4. Ticket::Frontend::AgentTicketCompose###StateType

Description:	Defines the next possible states after composing / answering a ticket in the ticket compose screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketCompose'}->{'StateType'} =  [
  'open',
  'closed',
  'pending auto',
  'pending reminder'
];
```

6.29.5. Ticket::Frontend::AgentTicketCompose###ArticleTypes

Description:	Specifies the different article types that will be used in the system.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketCompose'}->{'ArticleTypes'} =
[
  'email-external',
  'email-internal'
];
```

6.29.6. Ticket::Frontend::ResponseFormat

Description:	Defines the format of responses in the ticket compose screen of the agent interface ($QData{"OrigFrom"} is From 1:1, $QData{"OrigFromName"} is only realname of From).
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::ResponseFormat'} =  '$QData{"Salutation"}
$QData{"StdResponse"}
$QData{"Signature"}

$TimeShort{"$QData{"Created"}"} - $QData{"OrigFromName"}
 $Text{"wrote"}:
$QData{"Body"}
';
```

6.29.7. Ticket::Frontend::Quote

Description:	Defines the used character for email quotes in the ticket compose screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::Quote'} =  '>';`

6.29.8. Ticket::Frontend::ComposeAddCustomerAddress

Description:	Adds customers email addresses to recipients in the ticket compose screen of the agent interface. The customers email address won't be added if the article type is email-internal.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::ComposeAddCustomerAddress'} =  '1';`

6.29.9. Ticket::Frontend::ComposeReplaceSenderAddress

Description:	Replaces the original sender with current customer's email address on compose answer in the ticket compose screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::ComposeReplaceSenderAddress'} =  '0';`

6.29.10. Ticket::Frontend::ComposeExcludeCcRecipients

Description:	Uses Cc recipients in reply Cc list on compose an email answer in the ticket compose screen of the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::ComposeExcludeCcRecipients'} =  '0';`

6.29.11. Ticket::Frontend::AgentTicketCompose###DynamicField

Description:	Dynamic fields shown in the ticket compose screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCompose
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketCompose'}->{'DynamicField'} = {};`

6.30. Frontend::Agent::Ticket::ViewCustomer

6.30.1. Ticket::Frontend::AgentTicketCustomer###Permission

Description:	Required permissions to change the customer of a ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCustomer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketCustomer'}->{'Permission'} = 'customer';`

6.30.2. Ticket::Frontend::AgentTicketCustomer###RequiredLock

Description:	Defines if a ticket lock is required to change the customer of a ticket in the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewCustomer
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketCustomer'}->{'RequiredLock'} = '0';`

6.31. Frontend::Agent::Ticket::ViewEmailNew

6.31.1. Ticket::Frontend::AgentTicketEmail###Priority

Description:	Sets the default priority for new email tickets in the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'Priority'} =  '3 normal';`

6.31.2. Ticket::Frontend::AgentTicketEmail###ArticleType

Description:	Sets the default article type for new email tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'ArticleType'} = 'email-external';`

6.31.3. Ticket::Frontend::AgentTicketEmail###SenderType

Description:	Sets the default sender type for new email tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'SenderType'} = 'agent';`

6.31.4. Ticket::Frontend::AgentTicketEmail###Subject

Description:	Sets the default subject for new email tickets (e.g. 'email Outbound') in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'Subject'} =  '';`

6.31.5. Ticket::Frontend::AgentTicketEmail###Body

Description:	Sets the default text for new email tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'Body'} =  '';`

6.31.6. Ticket::Frontend::AgentTicketEmail###StateDefault

Description:	Sets the default next ticket state, after the creation of an email ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'StateDefault'} = 'open';`

6.31.7. Ticket::Frontend::AgentTicketEmail###StateType

Description:	Determines the next possible ticket states, after the creation of a new email ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'StateType'} =  [
  'open',
  'pending auto',
  'pending reminder',
  'closed'
];
```

6.31.8. Ticket::Frontend::AgentTicketEmail###HistoryType

Description:	Defines the history type for the email ticket screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'HistoryType'} = 'EmailAgent';`

6.31.9. Ticket::Frontend::AgentTicketEmail###HistoryComment

Description:	Defines the history comment for the email ticket screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'HistoryComment'} = '';`

6.31.10. Ticket::Frontend::AgentTicketEmail###DynamicField

Description:	Dynamic fields shown in the ticket email screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'DynamicField'} = {};`

6.31.11. Ticket::Frontend::AgentTicketEmail###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'RichTextWidth'} = '620';`

6.31.12. Ticket::Frontend::AgentTicketEmail###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEmailNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEmail'}->{'RichTextHeight'} = '320';`

6.32. Frontend::Agent::Ticket::ViewEscalation

6.32.1. Ticket::Frontend::AgentTicketEscalationView###TicketPermission

Description:	Defines the required permission to show a ticket in the escalation view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEscalation
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEscalationView'}->{'TicketPermission'} =  'rw';`

6.32.2. Ticket::Frontend::AgentTicketEscalationView###ViewableTicketsPage

Description:	Shows all open tickets (even if they are locked) in the escalation view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEscalation
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEscalationView'}->{'ViewableTicketsPage'} =  '50';`

6.32.3. Ticket::Frontend::AgentTicketEscalationView###SortBy::Default

Description:	Defines the default ticket attribute for ticket sorting in the escalation view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEscalation
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEscalationView'}->{'SortBy::Default'} =  'EscalationTime';`

6.32.4. Ticket::Frontend::AgentTicketEscalationView###Order::Default

Description:	Defines the default ticket order (after priority sort) in the escalation view of the agent interface. Up: oldest on top. Down: latest on top.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewEscalation
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketEscalationView'}->{'Order::Default'} =  'Up';`

6.33. Frontend::Agent::Ticket::ViewForward

6.33.1. Ticket::Frontend::AgentTicketForward###Permission

Description:	Required permissions to use the ticket forward screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	1

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketForward'}->{'Permission'} = 'forward';`

6.33.2. Ticket::Frontend::AgentTicketForward###RequiredLock

Description:	Defines if a ticket lock is required in the ticket forward screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketForward'}->{'RequiredLock'} = '1';`

6.33.3. Ticket::Frontend::AgentTicketForward###StateDefault

Description:	Defines the default next state of a ticket after being forwarded, in the ticket forward screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketForward'}->{'StateDefault'} = 'closed successful';`

6.33.4. Ticket::Frontend::AgentTicketForward###StateType

Description:	Defines the next possible states after forwarding a ticket in the ticket forward screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketForward'}->{'StateType'} =  [
  'open',
  'closed',
  'pending reminder',
  'pending auto'
];
```

6.33.5. Ticket::Frontend::AgentTicketForward###ArticleTypeDefault

Description:	Defines the default type of forwarded message in the ticket forward screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1

Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketForward'}-
>{'ArticleTypeDefault'} =  'email-external';
```

6.33.6. Ticket::Frontend::AgentTicketForward###ArticleTypes

Description:	Specifies the different article types that will be used in the system.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketForward'}->{'ArticleTypes'} =
[
  'email-external',
  'email-internal'
];
```

6.33.7. Ticket::Frontend::AgentTicketForward###DynamicField

Description:	Dynamic fields shown in the ticket forward screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketForward'}->{'DynamicField'} =
{};
```

6.33.8. Ticket::Frontend::AgentTicketForward###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketForward'}->{'RichTextWidth'} =
'620';
```

6.33.9. Ticket::Frontend::AgentTicketForward###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewForward
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketForward'}->{'RichTextHeight'} = '100';`

6.34. Frontend::Agent::Ticket::ViewFreeText

6.34.1. Ticket::Frontend::AgentTicketFreeText###Permission

Description:	Required permissions to use the ticket free text screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Permission'} = 'rw';`

6.34.2. Ticket::Frontend::AgentTicketFreeText###RequiredLock

Description:	Defines if a ticket lock is required in the ticket free text screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'RequiredLock'} = '0';`

6.34.3. Ticket::Frontend::AgentTicketFreeText###TicketType

Description:	Sets the ticket type in the ticket free text screen of the agent interface (Ticket::Type needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'TicketType'} = '1';`

6.34.4. Ticket::Frontend::AgentTicketFreeText###Service

Description:	Sets the service in the ticket free text screen of the agent interface (Ticket::Service needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Service'} =  '1';`

6.34.5. Ticket::Frontend::AgentTicketFreeText###Queue

Description:	Sets the queue in the ticket free text screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Queue'} =  '0';`

6.34.6. Ticket::Frontend::AgentTicketFreeText###Owner

Description:	Sets the ticket owner in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Owner'} =  '0';`

6.34.7. Ticket::Frontend::AgentTicketFreeText###OwnerMandatory

Description:	Sets if ticket owner must be selected by the agent.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'OwnerMandatory'} =  '0';`

6.34.8. Ticket::Frontend::AgentTicketFreeText###Responsible

Description:	Sets the responsible agent of the ticket in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Responsible'} = '0';`

6.34.9. Ticket::Frontend::AgentTicketFreeText###State

Description:	If a note is added by an agent, sets the state of a ticket in the ticket free text screen of the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'State'} =  '0';`

6.34.10. Ticket::Frontend::AgentTicketFreeText###StateType

Description:	Defines the next state of a ticket after adding a note, in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'StateType'} =  [
  'open',
  'closed',
  'pending reminder',
  'pending auto'
];
```

6.34.11. Ticket::Frontend::AgentTicketFreeText###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'StateDefault'} =
'open';
```

6.34.12. Ticket::Frontend::AgentTicketFreeText###Note

Description:	Allows adding notes in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Note'} =  '0';`

6.34.13. Ticket::Frontend::AgentTicketFreeText###Subject

Description:	Defines the default subject of a note in the ticket free text screen of the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Subject'} = '$Text{"Note"}';`

6.34.14. Ticket::Frontend::AgentTicketFreeText###Body

Description:	Defines the default body of a note in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Body'} =  '';`

6.34.15. Ticket::Frontend::AgentTicketFreeText###InvolvedAgent

Description:	Shows a list of all the involved agents on this ticket, in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'InvolvedAgent'} = '0';`

6.34.16. Ticket::Frontend::AgentTicketFreeText###InformAgent

Description:	Shows a list of all the possible agents (all agents with note permissions on the queue/ticket) to determine who should be informed about this note, in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'InformAgent'} = '0';`

6.34.17. Ticket::Frontend::AgentTicketFreeText###ArticleTypeDefault

Description:	Defines the default type of the note in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText

Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'ArticleTypeDefault'} =  'note-internal';`

6.34.18. Ticket::Frontend::AgentTicketFreeText###ArticleTypes

Description:	Specifies the available note types for this ticket mask. If the option is deselected, ArticleTypeDefault is used and the option is removed from the mask.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'ArticleTypes'} =
{
  'note-external' => '1',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.34.19. Ticket::Frontend::AgentTicketFreeText###Priority

Description:	Shows the ticket priority options in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Priority'} = '0';`

6.34.20. Ticket::Frontend::AgentTicketFreeText###PriorityDefault

Description:	Defines the default ticket priority in the ticket free text screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'PriorityDefault'} =  '3 normal';`

6.34.21. Ticket::Frontend::AgentTicketFreeText###Title

Description:	Shows the title fields in the ticket free text screen of the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'Title'} =  '1';`

6.34.22. Ticket::Frontend::AgentTicketFreeText###HistoryType

Description:	Defines the history type for the ticket free text screen action, which gets used for ticket history.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'HistoryType'} = 'AddNote';`

6.34.23. Ticket::Frontend::AgentTicketFreeText###HistoryComment

Description:	Defines the history comment for the ticket free text screen action, which gets used for ticket history.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'HistoryComment'} =  '%%FreeText';`

6.34.24. Ticket::Frontend::AgentTicketFreeText###DynamicField

Description:	Dynamic fields shown in the ticket free text screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'DynamicField'} = {};`

6.34.25. Ticket::Frontend::AgentTicketFreeText###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'RichTextWidth'} = '620';`

6.34.26. Ticket::Frontend::AgentTicketFreeText###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewFreeText
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketFreeText'}->{'RichTextHeight'} =  '100';`

6.35. Frontend::Agent::Ticket::ViewHistory

6.35.1. Ticket::Frontend::HistoryOrder

Description:	Shows the ticket history (reverse ordered) in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewHistory
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::HistoryOrder'} =  'normal';`

6.36. Frontend::Agent::Ticket::ViewMailbox

6.36.1. Ticket::Frontend::AgentTicketLockedView###SortBy::Default

Description:	Defines the default ticket attribute for ticket sorting in the locked ticket view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMailbox
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketLockedView'}->{'SortBy::Default'} =  'Age';`

6.36.2. Ticket::Frontend::AgentTicketLockedView###Order::Default

Description:	Defines the default ticket order in the ticket locked view of the agent interface. Up: oldest on top. Down: latest on top.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMailbox
Valid:	1
Required:	1

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketLockedView'}->{'Order::Default'} =  'Up';`

6.36.3. Ticket::Frontend::AgentTicketResponsibleView###SortBy::Default

Description:	Defines the default ticket attribute for ticket sorting in the responsible view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMailbox
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsibleView'}->{'SortBy::Default'} =  'Age';`

6.36.4. Ticket::Frontend::AgentTicketResponsibleView###Order::Default

Description:	Defines the default ticket order in the responsible view of the agent interface. Up: oldest on top. Down: latest on top.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMailbox
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsibleView'}->{'Order::Default'} =  'Up';`

6.36.5. Ticket::Frontend::AgentTicketWatchView###SortBy::Default

Description:	Defines the default ticket attribute for ticket sorting in the watch view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMailbox
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketWatchView'}->{'SortBy::Default'} =  'Age';`

6.36.6. Ticket::Frontend::AgentTicketWatchView###Order::Default

Description:	Defines the default ticket order in the watch view of the agent interface. Up: oldest on top. Down: latest on top.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMailbox
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketWatchView'}->{'Order::Default'} =  'Up';`

6.37. Frontend::Agent::Ticket::ViewMerge

6.37.1. Ticket::Frontend::AgentTicketMerge###Permission

Description:	Required permissions to use the ticket merge screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMerge
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMerge'}->{'Permission'} = 'rw';`

6.37.2. Ticket::Frontend::AgentTicketMerge###RequiredLock

Description:	Defines if a ticket lock is required in the ticket merge screen of a zoomed ticket in the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMerge
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMerge'}->{'RequiredLock'} = '1';`

6.37.3. Ticket::Frontend::MergeText

Description:	When tickets are merged, the customer can be informed per email by setting the check box "Inform Sender". In this text area, you can define a pre-formatted text which can later be modified by the agents.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMerge
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::MergeText'} =  'Your email with ticket number "<OTRS_TICKET>" is merged to "<OTRS_MERGE_TO_TICKET>".';`

6.37.4. Ticket::Frontend::AutomaticMergeText

Description:	When tickets are merged, a note will be added automatically to the ticket which is no longer active. In this text area you can define this text (This text cannot be changed by the agent).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMerge
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AutomaticMergeText'} =  'Merged Ticket <OTRS_TICKET> to <OTRS_MERGE_TO_TICKET>.';`

6.37.5. Ticket::Frontend::AgentTicketMerge###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMerge
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMerge'}->{'RichTextWidth'} = '620';`

6.37.6. Ticket::Frontend::AgentTicketMerge###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMerge
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMerge'}->{'RichTextHeight'} = '100';`

6.38. Frontend::Agent::Ticket::ViewMove

6.38.1. Ticket::Frontend::MoveType

Description:	Determines if the list of possible queues to move to ticket into should be displayed in a dropdown list or in a new window in the agent interface. If "New Window" is set you can add a move note to the ticket.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::MoveType'} =  'form';`

6.38.2. Ticket::Frontend::AgentTicketMove###State

Description:	Allows to set a new ticket state in the move ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	1

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMove'}->{'State'} =  '1';`

6.38.3. Ticket::DefaultNextMoveStateType

Description:	Defines the next state of a ticket after being moved to another queue, in the move ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::DefaultNextMoveStateType'} =  [
  'open',
  'closed'
];
```

6.38.4. Ticket::Frontend::AgentTicketMove###Priority

Description:	Shows the ticket priority options in the move ticket screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMove'}->{'Priority'} =  '0';`

6.38.5. Ticket::Frontend::AgentTicketMove###NextScreen

Description:	Determines the next screen after the ticket is moved. LastScreenOverview will return to search results, queueview, dashboard or the like, LastScreenView will return to TicketZoom.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::Frontend::AgentTicketMove'}->{'NextScreen'} =
 'LastScreenView';
```

6.38.6. Ticket::Frontend::AgentTicketMove###Subject

Description:	Sets the default subject for notes added in the ticket move screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMove'}->{'Subject'} = '$Text{"Change Queue"}';`

6.38.7. Ticket::Frontend::AgentTicketMove###Body

Description:	Sets the default body text for notes added in the ticket move screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMove'}->{'Body'} =  '';`

6.38.8. Ticket::Frontend::AgentTicketMove###DynamicField

Description:	Dynamic fields shown in the ticket move screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewMove
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketMove'}->{'DynamicField'} =  {};`

6.39. Frontend::Agent::Ticket::ViewNote

6.39.1. Ticket::Frontend::AgentTicketNote###Permission

Description:	Required permissions to use the ticket note screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Permission'} = 'note';`

6.39.2. Ticket::Frontend::AgentTicketNote###RequiredLock

Description:	Defines if a ticket lock is required in the ticket note screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'RequiredLock'} = '0';`

6.39.3. Ticket::Frontend::AgentTicketNote###TicketType

Description:	Sets the ticket type in the ticket note screen of the agent interface (Ticket::Type needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'TicketType'} =  '0';`

6.39.4. Ticket::Frontend::AgentTicketNote###Service

Description:	Sets the service in the ticket note screen of the agent interface (Ticket::Service needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Service'} =  '0';`

6.39.5. Ticket::Frontend::AgentTicketNote###Queue

Description:	Sets the queue in the ticket note screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Queue'} =  '0';`

6.39.6. Ticket::Frontend::AgentTicketNote###Owner

Description:	Sets the ticket owner in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Owner'} =  '0';`

6.39.7. Ticket::Frontend::AgentTicketNote###OwnerMandatory

Description:	Sets if ticket owner must be selected by the agent.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'OwnerMandatory'} = '0';`

6.39.8. Ticket::Frontend::AgentTicketNote###Responsible

Description:	Sets the responsible agent of the ticket in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Responsible'} =  '0';`

6.39.9. Ticket::Frontend::AgentTicketNote###State

Description:	If a note is added by an agent, sets the state of a ticket in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'State'} =  '0';`

6.39.10. Ticket::Frontend::AgentTicketNote###StateType

Description:	Defines the next state of a ticket after adding a note, in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketNote'}->{'StateType'} =  [
  'open',
  'closed',
  'pending reminder',
  'pending auto'
];
```

6.39.11. Ticket::Frontend::AgentTicketNote###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the ticket note screen of the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'StateDefault'} = 'open';`

6.39.12. Ticket::Frontend::AgentTicketNote###Note

Description:	Allows adding notes in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Note'} =  '1';`

6.39.13. Ticket::Frontend::AgentTicketNote###Subject

Description:	Sets the default subject for notes added in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Subject'} = '$Text{"Note"}';`

6.39.14. Ticket::Frontend::AgentTicketNote###Body

Description:	Sets the default body text for notes added in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Body'} =  '';`

6.39.15. Ticket::Frontend::AgentTicketNote###InvolvedAgent

Description:	Shows a list of all the involved agents on this ticket, in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'InvolvedAgent'} = '0';`

6.39.16. Ticket::Frontend::AgentTicketNote###InformAgent

Description:	Shows a list of all the possible agents (all agents with note permissions on the queue/ticket) to determine who should be informed about this note, in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'InformAgent'} =  '0';`

6.39.17. Ticket::Frontend::AgentTicketNote###ArticleTypeDefault

Description:	Defines the default type of the note in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'ArticleTypeDefault'} =  'note-internal';`

6.39.18. Ticket::Frontend::AgentTicketNote###ArticleTypes

Description:	Specifies the available note types for this ticket mask. If the option is deselected, ArticleTypeDefault is used and the option is removed from the mask.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketNote'}->{'ArticleTypes'} =  {
  'note-external' => '1',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.39.19. Ticket::Frontend::AgentTicketNote###Priority

Description:	Shows the ticket priority options in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Priority'} =  '0';`

6.39.20. Ticket::Frontend::AgentTicketNote###PriorityDefault

Description:	Defines the default ticket priority in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'PriorityDefault'} = '3 normal';`

6.39.21. Ticket::Frontend::AgentTicketNote###Title

Description:	Shows the title fields in the ticket note screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'Title'} =  '0';`

6.39.22. Ticket::Frontend::AgentTicketNote###HistoryType

Description:	Defines the history type for the ticket note screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'HistoryType'} = 'AddNote';`

6.39.23. Ticket::Frontend::AgentTicketNote###HistoryComment

Description:	Defines the history comment for the ticket note screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'HistoryComment'} = '%%Note';`

6.39.24. Ticket::Frontend::AgentTicketNote###DynamicField

Description:	Dynamic fields shown in the ticket note screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'DynamicField'} =  {};`

6.39.25. Ticket::Frontend::AgentTicketNote###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'RichTextWidth'} = '620';`

6.39.26. Ticket::Frontend::AgentTicketNote###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewNote
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketNote'}->{'RichTextHeight'} = '100';`

6.40. Frontend::Agent::Ticket::ViewOwner

6.40.1. Ticket::Frontend::AgentTicketOwner###Permission

Description:	Required permissions to use the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Permission'} = 'owner';`

6.40.2. Ticket::Frontend::AgentTicketOwner###RequiredLock

Description:	Defines if a ticket lock is required in the ticket owner screen of a zoomed ticket in the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'RequiredLock'} = '0';`

6.40.3. Ticket::Frontend::AgentTicketOwner###TicketType

Description:	Sets the ticket type in the ticket owner screen of a zoomed ticket in the agent interface (Ticket::Type needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'TicketType'} =  '0';`

6.40.4. Ticket::Frontend::AgentTicketOwner###Service

Description:	Sets the service in the ticket owner screen of a zoomed ticket in the agent interface (Ticket::Service needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Service'} =  '0';`

6.40.5. Ticket::Frontend::AgentTicketOwner###Queue

Description:	Sets the queue in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Queue'} =  '0';`

6.40.6. Ticket::Frontend::AgentTicketOwner###Owner

Description:	Sets the ticket owner in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Owner'} =  '1';`

6.40.7. Ticket::Frontend::AgentTicketOwner###OwnerMandatory

Description:	Sets if ticket owner must be selected by the agent.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'OwnerMandatory'} = '1';`

6.40.8. Ticket::Frontend::AgentTicketOwner###Responsible

Description:	Sets the responsible agent of the ticket in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Responsible'} = '0';`

6.40.9. Ticket::Frontend::AgentTicketOwner###State

Description:	If a note is added by an agent, sets the state of the ticket in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'State'} =  '0';`

6.40.10. Ticket::Frontend::AgentTicketOwner###StateType

Description:	Defines the next state of a ticket after adding a note, in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'StateType'} =  [
  'open',
  'pending reminder',
  'pending auto'
];
```

6.40.11. Ticket::Frontend::AgentTicketOwner###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'StateDefault'} = 'open';`

6.40.12. Ticket::Frontend::AgentTicketOwner###Note

Description:	Allows adding notes in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Note'} =  '1';`

6.40.13. Ticket::Frontend::AgentTicketOwner###Subject

Description:	Sets the default subject for notes added in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Subject'} = '$Text{"Owner Update"}!';`

6.40.14. Ticket::Frontend::AgentTicketOwner###Body

Description:	Sets the default body text for notes added in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Body'} =  '';`

6.40.15. Ticket::Frontend::AgentTicketOwner###InvolvedAgent

Description:	Shows a list of all the involved agents on this ticket, in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'InvolvedAgent'} = '0';`

6.40.16. Ticket::Frontend::AgentTicketOwner###InformAgent

Description:	Shows a list of all the possible agents (all agents with note permissions on the queue/ticket) to determine who should be informed about this note, in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'InformAgent'} = '0';`

6.40.17. Ticket::Frontend::AgentTicketOwner###ArticleTypeDefault

Description:	Defines the default type of the note in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'ArticleTypeDefault'} =  'note-internal';`

6.40.18. Ticket::Frontend::AgentTicketOwner###ArticleTypes

Description:	Specifies the available note types for this ticket mask. If the option is deselected, ArticleTypeDefault is used and the option is removed from the mask.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'ArticleTypes'} =  {
  'note-external' => '0',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.40.19. Ticket::Frontend::AgentTicketOwner###Priority

Description:	Shows the ticket priority options in the ticket owner screen of a zoomed ticket in the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Priority'} =  '0';`

6.40.20. Ticket::Frontend::AgentTicketOwner###PriorityDefault

Description:	Defines the default ticket priority in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'PriorityDefault'} = '3 normal';`

6.40.21. Ticket::Frontend::AgentTicketOwner###Title

Description:	Shows the title fields in the ticket owner screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'Title'} =  '0';`

6.40.22. Ticket::Frontend::AgentTicketOwner###HistoryType

Description:	Defines the history type for the ticket owner screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'HistoryType'} = 'AddNote';`

6.40.23. Ticket::Frontend::AgentTicketOwner###HistoryComment

Description:	Defines the history comment for the ticket owner screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'HistoryComment'} = '%%Owner';`

6.40.24. Ticket::Frontend::AgentTicketOwner###DynamicField

Description:	Dynamic fields shown in the ticket owner screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'DynamicField'} = {};`

6.40.25. Ticket::Frontend::AgentTicketOwner###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'RichTextWidth'} = '620';`

6.40.26. Ticket::Frontend::AgentTicketOwner###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewOwner
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketOwner'}->{'RichTextHeight'} = '100';`

6.41. Frontend::Agent::Ticket::ViewPending

6.41.1. Ticket::Frontend::AgentTicketPending###Permission

Description:	Required permissions to use the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Permission'} = 'pending';`

6.41.2. Ticket::Frontend::AgentTicketPending###RequiredLock

Description:	Defines if a ticket lock is required in the ticket pending screen of a zoomed ticket in the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'RequiredLock'} = '1';`

6.41.3. Ticket::Frontend::AgentTicketPending###TicketType

Description:	Sets the ticket type in the ticket pending screen of a zoomed ticket in the agent interface (Ticket::Type needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'TicketType'} = '0';`

6.41.4. Ticket::Frontend::AgentTicketPending###Service

Description:	Sets the service in the ticket pending screen of a zoomed ticket in the agent interface (Ticket::Service needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Service'} =  '0';`

6.41.5. Ticket::Frontend::AgentTicketPending###Queue

Description:	Sets the queue in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Queue'} =  '0';`

6.41.6. Ticket::Frontend::AgentTicketPending###Owner

Description:	Sets the ticket owner in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Owner'} =  '0';`

6.41.7. Ticket::Frontend::AgentTicketPending###OwnerMandatory

Description:	Sets if ticket owner must be selected by the agent.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'OwnerMandatory'} =  '0';`

6.41.8. Ticket::Frontend::AgentTicketPending###Responsible

Description:	Sets the responsible agent of the ticket in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Responsible'} = '0';`

6.41.9. Ticket::Frontend::AgentTicketPending###State

Description:	If a note is added by an agent, sets the state of the ticket in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'State'} =  '1';`

6.41.10. Ticket::Frontend::AgentTicketPending###StateType

Description:	Defines the next state of a ticket after adding a note, in the ticket pending screen of a zoomed ticket in the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketPending'}->{'StateType'} =  [
  'pending reminder',
  'pending auto'
];
```

6.41.11. Ticket::Frontend::AgentTicketPending###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'StateDefault'} = 'pending reminder';`

6.41.12. Ticket::Frontend::AgentTicketPending###Note

Description:	Allows adding notes in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Note'} =  '1';`

6.41.13. Ticket::Frontend::AgentTicketPending###Subject

Description:	Sets the default subject for notes added in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Subject'} = '$Text{"Pending"}!';`

6.41.14. Ticket::Frontend::AgentTicketPending###Body

Description:	Sets the default body text for notes added in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Body'} =  '';`

6.41.15. Ticket::Frontend::AgentTicketPending###InvolvedAgent

Description:	Shows a list of all the involved agents on this ticket, in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'InvolvedAgent'} = '0';`

6.41.16. Ticket::Frontend::AgentTicketPending###InformAgent

Description:	Shows a list of all the possible agents (all agents with note permissions on the queue/ticket) to determine who should be informed about this note, in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'InformAgent'} = '0';`

6.41.17. Ticket::Frontend::AgentTicketPending###ArticleTypeDefault

Description:	Defines the default type of the note in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'ArticleTypeDefault'} =  'note-internal';`

6.41.18. Ticket::Frontend::AgentTicketPending###ArticleTypes

Description:	Specifies the available note types for this ticket mask. If the option is deselected, ArticleTypeDefault is used and the option is removed from the mask.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketPending'}->{'ArticleTypes'} =
 {
  'note-external' => '0',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.41.19. Ticket::Frontend::AgentTicketPending###Priority

Description:	Shows the ticket priority options in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Priority'} =  '0';`

6.41.20. Ticket::Frontend::AgentTicketPending###PriorityDefault

Description:	Defines the default ticket priority in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'PriorityDefault'} =  '3 normal';`

6.41.21. Ticket::Frontend::AgentTicketPending###Title

Description:	Shows the title fields in the ticket pending screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'Title'} =  '0';`

6.41.22. Ticket::Frontend::AgentTicketPending###HistoryType

Description:	Defines the history type for the ticket pending screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending

Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'HistoryType'} = 'AddNote';`

6.41.23. Ticket::Frontend::AgentTicketPending###HistoryComment

Description:	Defines the history comment for the ticket pending screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'HistoryComment'} = '%%Pending';`

6.41.24. Ticket::Frontend::AgentTicketPending###DynamicField

Description:	Dynamic fields shown in the ticket pending screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'DynamicField'} = {};`

6.41.25. Ticket::Frontend::AgentTicketPending###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'RichTextWidth'} = '620';`

6.41.26. Ticket::Frontend::AgentTicketPending###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPending
Valid:	1

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPending'}->{'RichTextHeight'} = '100';`

6.42. Frontend::Agent::Ticket::ViewPhoneInbound

6.42.1. Ticket::Frontend::AgentTicketPhoneInbound###Permission

Description:	Required permissions to use the ticket phone inbound screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'Permission'} = 'phone';`

6.42.2. Ticket::Frontend::AgentTicketPhoneInbound###RequiredLock

Description:	Defines if a ticket lock is required in the ticket phone inbound screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'RequiredLock'} = '0';`

6.42.3. Ticket::Frontend::AgentTicketPhoneInbound###ArticleType

Description:	Defines the default type of the note in the ticket phone inbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'ArticleType'} = 'phone';`

6.42.4. Ticket::Frontend::AgentTicketPhoneInbound###SenderType

Description:	Defines the default sender type for phone tickets in the ticket phone inbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'SenderType'} =  'customer';`

6.42.5. Ticket::Frontend::AgentTicketPhoneInbound###Subject

Description:	Defines the default subject for phone tickets in the ticket phone inbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'Subject'} = '$Text{"Phone call"}!';`

6.42.6. Ticket::Frontend::AgentTicketPhoneInbound###Body

Description:	Defines the default note body text for phone tickets in the ticket phone inbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'Body'} =  '';`

6.42.7. Ticket::Frontend::AgentTicketPhoneInbound###State

Description:	Defines the default ticket next state after adding a phone note in the ticket phone inbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'State'} = 'open';`

6.42.8. Ticket::Frontend::AgentTicketPhoneInbound###StateType

Description:	Next possible ticket states after adding a phone note in the ticket phone inbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'StateType'} =
  [
  'open',
  'pending auto',
  'pending reminder',
  'closed'
];
```

6.42.9. Ticket::Frontend::AgentTicketPhoneInbound###HistoryType

Description:	Defines the history type for the ticket phone inbound screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'HistoryType'} =  'PhoneCallCustomer';`

6.42.10. Ticket::Frontend::AgentTicketPhoneInbound###HistoryComment

Description:	Defines the history comment for the ticket phone inbound screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'HistoryComment'} =  '';`

6.42.11. Ticket::Frontend::AgentTicketPhoneInbound###DynamicField

Description:	Dynamic fields shown in the ticket phone inbound screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'DynamicField'} =  {};`

6.42.12. Ticket::Frontend::AgentTicketPhoneInbound###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'RichTextWidth'} =  '620';`

6.42.13. Ticket::Frontend::AgentTicketPhoneInbound###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneInbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneInbound'}->{'RichTextHeight'} =  '320';`

6.43. Frontend::Agent::Ticket::ViewPhoneNew

6.43.1. Ticket::Frontend::AgentTicketPhone###Priority

Description:	Sets the default priority for new phone tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'Priority'} =  '3 normal';`

6.43.2. Ticket::Frontend::AgentTicketPhone###ArticleType

Description:	Sets the default article type for new phone tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'ArticleType'} = 'phone';`

6.43.3. Ticket::Frontend::AgentTicketPhone###SenderType

Description:	Sets the default sender type for new phone ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'SenderType'} = 'customer';`

6.43.4. Ticket::Frontend::AgentTicketPhone::AllowMultipleFrom

Description:	Controls if more than one from entry can be set in the new phone ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone::AllowMultipleFrom'} = '1';`

6.43.5. Ticket::Frontend::AgentTicketPhone###Subject

Description:	Sets the default subject for new phone tickets (e.g. 'Phone call') in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'Subject'} =  '';`

6.43.6. Ticket::Frontend::AgentTicketPhone###Body

Description:	Sets the default note text for new telephone tickets. E.g 'New ticket via call' in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'Body'} =  '';`

6.43.7. Ticket::Frontend::AgentTicketPhone###StateDefault

Description:	Sets the default next state for new phone tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'StateDefault'} = 'open';`

6.43.8. Ticket::Frontend::AgentTicketPhone###StateType

Description:	Determines the next possible ticket states, after the creation of a new phone ticket in the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'StateType'} =  [
  'open',
  'pending auto',
  'pending reminder',
  'closed'
];
```

6.43.9. Ticket::Frontend::AgentTicketPhone###HistoryType

Description:	Defines the history type for the phone ticket screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'HistoryType'} =
'PhoneCallCustomer';
```

6.43.10. Ticket::Frontend::AgentTicketPhone###HistoryComment

Description:	Defines the history comment for the phone ticket screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'HistoryComment'} =
'';
```

6.43.11. Ticket::Frontend::AgentTicketPhone###SplitLinkType

Description:	Sets the default link type of splitted tickets in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'SplitLinkType'} =  {
  'Direction' => 'Target',
  'LinkType' => 'ParentChild'
};
```

6.43.12. Ticket::Frontend::AgentTicketPhone###DynamicField

Description:	Dynamic fields shown in the ticket phone screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'DynamicField'} = {};`

6.43.13. Ticket::Frontend::AgentTicketPhone###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'RichTextWidth'} = '620';`

6.43.14. Ticket::Frontend::AgentTicketPhone###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhone'}->{'RichTextHeight'} = '320';`

6.44. Frontend::Agent::Ticket::ViewPhoneOutbound

6.44.1. Ticket::Frontend::AgentTicketPhoneOutbound###Permission

Description:	Required permissions to use the ticket phone outbound screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'Permission'} =  'phone';`

6.44.2. Ticket::Frontend::AgentTicketPhoneOutbound###RequiredLock

Description:	Defines if a ticket lock is required in the ticket phone outbound screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'RequiredLock'} =  '1';`

6.44.3. Ticket::Frontend::AgentTicketPhoneOutbound###ArticleType

Description:	Defines the default type of the note in the ticket phone outbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'ArticleType'} =  'phone';`

6.44.4. Ticket::Frontend::AgentTicketPhoneOutbound###SenderType

Description:	Defines the default sender type for phone tickets in the ticket phone outbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'SenderType'} =  'agent';`

6.44.5. Ticket::Frontend::AgentTicketPhoneOutbound###Subject

Description:	Defines the default subject for phone tickets in the ticket phone outbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'Subject'} = '$Text{"Phone call"}!';`

6.44.6. Ticket::Frontend::AgentTicketPhoneOutbound###Body

Description:	Defines the default note body text for phone tickets in the ticket phone outbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'Body'} = '';`

6.44.7. Ticket::Frontend::AgentTicketPhoneOutbound###State

Description:	Defines the default ticket next state after adding a phone note in the ticket phone outbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'State'} = 'closed successful';`

6.44.8. Ticket::Frontend::AgentTicketPhoneOutbound###StateType

Description:	Next possible ticket states after adding a phone note in the ticket phone outbound screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	1
Config-Setting:	

```
 $Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'StateType'}
 =  [
  'open',
  'pending auto',
  'pending reminder',
  'closed'
];
```

6.44.9. Ticket::Frontend::AgentTicketPhoneOutbound###HistoryType

Description:	Defines the history type for the ticket phone outbound screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'HistoryType'} =  'PhoneCallAgent';`

6.44.10. Ticket::Frontend::AgentTicketPhoneOutbound###HistoryComment

Description:	Defines the history comment for the ticket phone outbound screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'HistoryComment'} =  '';`

6.44.11. Ticket::Frontend::AgentTicketPhoneOutbound###DynamicField

Description:	Dynamic fields shown in the ticket phone outbound screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'DynamicField'} =  {};`

6.44.12. Ticket::Frontend::AgentTicketPhoneOutbound###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'RichTextWidth'} =  '620';`

6.44.13. Ticket::Frontend::AgentTicketPhoneOutbound###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPhoneOutbound
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPhoneOutbound'}->{'RichTextHeight'} =  '320';`

6.45. Frontend::Agent::Ticket::ViewPrint

6.45.1. Ticket::Frontend::AgentTicketPrint###DynamicField

Description:	Dynamic fields shown in the ticket print screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPrint
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPrint'}->{'DynamicField'} = {};`

6.46. Frontend::Agent::Ticket::ViewPriority

6.46.1. Ticket::Frontend::AgentTicketPriority###Permission

Description:	Required permissions to use the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Permission'} = 'priority';`

6.46.2. Ticket::Frontend::AgentTicketPriority###RequiredLock

Description:	Defines if a ticket lock is required in the ticket priority screen of a zoomed ticket in the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'RequiredLock'} = '1';`

6.46.3. Ticket::Frontend::AgentTicketPriority###TicketType

Description:	Sets the ticket type in the ticket priority screen of a zoomed ticket in the agent interface (Ticket::Type needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'TicketType'} = '0';`

6.46.4. Ticket::Frontend::AgentTicketPriority###Service

Description:	Sets the service in the ticket priority screen of a zoomed ticket in the agent interface (Ticket::Service needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Service'} =  '0';`

6.46.5. Ticket::Frontend::AgentTicketPriority###Queue

Description:	Sets the queue in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Queue'} =  '0';`

6.46.6. Ticket::Frontend::AgentTicketPriority###Owner

Description:	Sets the ticket owner in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Owner'} =  '0';`

6.46.7. Ticket::Frontend::AgentTicketPriority###OwnerMandatory

Description:	Sets if ticket owner must be selected by the agent.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'OwnerMandatory'} =  '0';`

6.46.8. Ticket::Frontend::AgentTicketPriority###Responsible

Description:	Sets the responsible agent of the ticket in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Responsible'} = '0';`

6.46.9. Ticket::Frontend::AgentTicketPriority###State

Description:	If a note is added by an agent, sets the state of the ticket in the ticket priority screen of a zoomed ticket in the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'State'} =  '0';`

6.46.10. Ticket::Frontend::AgentTicketPriority###StateType

Description:	Defines the next state of a ticket after adding a note, in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'StateType'} =  [
  'open',
  'pending reminder',
  'pending auto'
];
```

6.46.11. Ticket::Frontend::AgentTicketPriority###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'StateDefault'} =
'open';
```

6.46.12. Ticket::Frontend::AgentTicketPriority###Note

Description:	Allows adding notes in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Note'} =  '1';`

6.46.13. Ticket::Frontend::AgentTicketPriority###Subject

Description:	Sets the default subject for notes added in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority

Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Subject'} = '$Text{"Priority Update"}!';`

6.46.14. Ticket::Frontend::AgentTicketPriority###Body

Description:	Sets the default body text for notes added in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Body'} =  '';`

6.46.15. Ticket::Frontend::AgentTicketPriority###InvolvedAgent

Description:	Shows a list of all the involved agents on this ticket, in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'InvolvedAgent'} = '0';`

6.46.16. Ticket::Frontend::AgentTicketPriority###InformAgent

Description:	Shows a list of all the possible agents (all agents with note permissions on the queue/ticket) to determine who should be informed about this note, in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'InformAgent'} = '0';`

6.46.17. Ticket::Frontend::AgentTicketPriority###ArticleTypeDefault

Description:	Defines the default type of the note in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketPriority'}-
>{'ArticleTypeDefault'} =  'note-internal';
```

6.46.18. Ticket::Frontend::AgentTicketPriority###ArticleTypes

Description:	Specifies the available note types for this ticket mask. If the option is deselected, ArticleTypeDefault is used and the option is removed from the mask.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketPriority'}->{'ArticleTypes'} =
 {
  'note-external' => '0',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.46.19. Ticket::Frontend::AgentTicketPriority###Priority

Description:	Shows the ticket priority options in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Priority'} =
 '1';
```

6.46.20. Ticket::Frontend::AgentTicketPriority###PriorityDefault

Description:	Defines the default ticket priority in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::AgentTicketPriority'}->{'PriorityDefault'}
 =  '3 normal';
```

6.46.21. Ticket::Frontend::AgentTicketPriority###Title

Description:	Shows the title fields in the ticket priority screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'Title'} =  '0';`

6.46.22. Ticket::Frontend::AgentTicketPriority###HistoryType

Description:	Defines the history type for the ticket priority screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'HistoryType'} = 'AddNote';`

6.46.23. Ticket::Frontend::AgentTicketPriority###HistoryComment

Description:	Defines the history comment for the ticket priority screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'HistoryComment'} =  '%%Priority';`

6.46.24. Ticket::Frontend::AgentTicketPriority###DynamicField

Description:	Dynamic fields shown in the ticket priority screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'DynamicField'} = {};`

6.46.25. Ticket::Frontend::AgentTicketPriority###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'RichTextWidth'} =  '620';`

6.46.26. Ticket::Frontend::AgentTicketPriority###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewPriority
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketPriority'}->{'RichTextHeight'} =  '100';`

6.47. Frontend::Agent::Ticket::ViewQueue

6.47.1. Ticket::Frontend::AgentTicketQueue###StripEmptyLines

Description:	Strips empty lines on the ticket preview in the queue view.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'StripEmptyLines'} = '0';`

6.47.2. Ticket::Frontend::AgentTicketQueue###ViewAllPossibleTickets

Description:	Shows all both ro and rw queues in the queue view.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'ViewAllPossibleTickets'} =  '0';`

6.47.3. Ticket::Frontend::AgentTicketQueue###HighlightAge1

Description:	Sets the age in minutes (first level) for highlighting queues that contain untouched tickets.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'HighlightAge1'} = '1440';`

6.47.4. Ticket::Frontend::AgentTicketQueue###HighlightAge2

Description:	Sets the age in minutes (second level) for highlighting queues that contain untouched tickets.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'HighlightAge2'} = '2880';`

6.47.5. Ticket::Frontend::AgentTicketQueue###Blink

Description:	Activates a blinking mechanism of the queue that contains the oldest ticket.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'Blink'} =  '1';`

6.47.6. Ticket::Frontend::AgentTicketQueue###QueueSort

Description:	Sorts the tickets (ascendingly or descendingly) when a single queue is selected in the queue view and after the tickets are sorted by priority. Values: 0 = ascending (oldest on top, default), 1 = descending (youngest on top). Use the QueueID for the key and 0 or 1 for value.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'QueueSort'} =  {
  '3' => '0',
  '7' => '1'
};
```

6.47.7. Ticket::Frontend::AgentTicketQueue###SortBy::Default

Description:	Defines the default sort criteria for all queues displayed in the queue view.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'SortBy::Default'} = 'Age';`

6.47.8. Ticket::Frontend::AgentTicketQueue###PreSort::ByPriority

Description:	Defines if a pre-sorting by priority should be done in the queue view.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'PreSort::ByPriority'} = '1';`

6.47.9. Ticket::Frontend::AgentTicketQueue###Order::Default

Description:	Defines the default sort order for all queues in the queue view, after priority sort.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewQueue
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketQueue'}->{'Order::Default'} = 'Up';`

6.48. Frontend::Agent::Ticket::ViewResponsible

6.48.1. Ticket::Frontend::AgentTicketResponsible###Permission

Description:	Required permissions to use the ticket responsible screen in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Permission'} = 'responsible';`

6.48.2. Ticket::Frontend::AgentTicketResponsible###RequiredLock

Description:	Defines if a ticket lock is required in the ticket responsible screen of the agent interface (if the ticket isn't locked yet, the ticket gets locked and the current agent will be set automatically as its owner).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'RequiredLock'} = '0';`

6.48.3. Ticket::Frontend::AgentTicketResponsible###TicketType

Description:	Sets the ticket type in the ticket responsible screen of the agent interface (Ticket::Type needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible

Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'TicketType'} = '0';`

6.48.4. Ticket::Frontend::AgentTicketResponsible###Service

Description:	Sets the service in the ticket responsible screen of the agent interface (Ticket::Service needs to be activated).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Service'} = '0';`

6.48.5. Ticket::Frontend::AgentTicketResponsible###Queue

Description:	Sets the queue in the ticket responsible screen of a zoomed ticket in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Queue'} = '0';`

6.48.6. Ticket::Frontend::AgentTicketResponsible###Owner

Description:	Sets the ticket owner in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Owner'} = '0';`

6.48.7. Ticket::Frontend::AgentTicketResponsible###OwnerMandatory

Description:	Sets if ticket owner must be selected by the agent.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'OwnerMandatory'} =  '0';`

6.48.8. Ticket::Frontend::AgentTicketResponsible###Responsible

Description:	Sets the responsible agent of the ticket in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Responsible'} =  '1';`

6.48.9. Ticket::Frontend::AgentTicketResponsible###State

Description:	If a note is added by an agent, sets the state of a ticket in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'State'} = '0';`

6.48.10. Ticket::Frontend::AgentTicketResponsible###StateType

Description:	Defines the next state of a ticket after adding a note, in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'StateType'} =
[
  'open',
  'pending reminder',
  'pending auto'
];
```

6.48.11. Ticket::Frontend::AgentTicketResponsible###StateDefault

Description:	Defines the default next state of a ticket after adding a note, in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'StateDefault'} =  'open';`

6.48.12. Ticket::Frontend::AgentTicketResponsible###Note

Description:	Allows adding notes in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Note'} =  '1';`

6.48.13. Ticket::Frontend::AgentTicketResponsible###Subject

Description:	Sets the default subject for notes added in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Subject'} =` `'$Text{"Responsible Update"}!';`

6.48.14. Ticket::Frontend::AgentTicketResponsible###Body

Description:	Sets the default body text for notes added in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Body'} =  '';`

6.48.15. Ticket::Frontend::AgentTicketResponsible###InvolvedAgent

Description:	Shows a list of all the involved agents on this ticket, in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}-` `>{'InvolvedAgent'} =  '0';`

6.48.16. Ticket::Frontend::AgentTicketResponsible###InformAgent

Description:	Shows a list of all the possible agents (all agents with note permissions on the queue/ticket) to determine who should be informed about this note, in the ticket responsible screen of the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'InformAgent'} =  '0';`

6.48.17. Ticket::Frontend::AgentTicketResponsible###ArticleTypeDefault

Description:	Defines the default type of the note in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'ArticleTypeDefault'} =  'note-internal';`

6.48.18. Ticket::Frontend::AgentTicketResponsible###ArticleTypes

Description:	Specifies the available note types for this ticket mask. If the option is deselected, ArticleTypeDefault is used and the option is removed from the mask.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'ArticleTypes'} =  {
  'note-external' => '0',
  'note-internal' => '1',
  'note-report' => '0'
};
```

6.48.19. Ticket::Frontend::AgentTicketResponsible###Priority

Description:	Shows the ticket priority options in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Priority'} = '0';`

6.48.20. Ticket::Frontend::AgentTicketResponsible###PriorityDefault

Description:	Defines the default ticket priority in the ticket responsible screen of the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'PriorityDefault'} =  '3 normal';`

6.48.21. Ticket::Frontend::AgentTicketResponsible###Title

Description:	Shows the title fields in the ticket responsible screen of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'Title'} = '1';`

6.48.22. Ticket::Frontend::AgentTicketResponsible###HistoryType

Description:	Defines the history type for the ticket responsible screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'HistoryType'} =  'AddNote';`

6.48.23. Ticket::Frontend::AgentTicketResponsible###HistoryComment

Description:	Defines the history comment for the ticket responsible screen action, which gets used for ticket history in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'HistoryComment'} =  '%%Responsible';`

6.48.24. Ticket::Frontend::AgentTicketResponsible###DynamicField

Description:	Dynamic fields shown in the ticket responsible screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'DynamicField'} =  {};`

6.48.25. Ticket::Frontend::AgentTicketResponsible###RichTextWidth

Description:	Defines the width for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'RichTextWidth'} =  '620';`

6.48.26. Ticket::Frontend::AgentTicketResponsible###RichTextHeight

Description:	Defines the height for the rich text editor component for this screen. Enter number (pixels) or percent value (relative).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewResponsible
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketResponsible'}->{'RichTextHeight'} =  '100';`

6.49. Frontend::Agent::Ticket::ViewSearch

6.49.1. Ticket::Frontend::AgentTicketSearch###ExtendedSearchCondition

Description:	Allows extended search conditions in ticket search of the agent interface. With this feature you can search e. g. with this kind of conditions like "(key1&&key2)" or "(key1||key2)".
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'ExtendedSearchCondition'} =  '1';`

6.49.2. Ticket::Frontend::AgentTicketSearch###SearchLimit

Description:	Maximum number of tickets to be displayed in the result of a search in the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'SearchLimit'} = '2000';`

6.49.3. Ticket::Frontend::AgentTicketSearch###SearchPageShown

Description:	Number of tickets to be displayed in each page of a search result in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'SearchPageShown'} =  '40';`

6.49.4. Ticket::Frontend::AgentTicketSearch###SearchViewableTicketLines

Description:	Number of lines (per ticket) that are shown by the search utility in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'SearchViewableTicketLines'} =  '10';`

6.49.5. Ticket::Frontend::AgentTicketSearch###SortBy::Default

Description:	Defines the default ticket attribute for ticket sorting of the ticket search result of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'SortBy::Default'} =  'Age';`

6.49.6. Ticket::Frontend::AgentTicketSearch###Order::Default

Description:	Defines the default ticket order in the ticket search result of the agent interface. Up: oldest on top. Down: latest on top.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1

Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Order::Default'} = 'Down';`

6.49.7. Ticket::Frontend::AgentTicketSearch###SearchArticleCSVTree

Description:	Exports the whole article tree in search result (it can affect the system performance).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'SearchArticleCSVTree'} = '0';`

6.49.8. Ticket::Frontend::AgentTicketSearch###SearchCSVData

Description:	Data used to export the search result in CSV format.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'SearchCSVData'} =
[
  'TicketNumber',
  'Age',
  'Created',
  'Closed',
  'FirstLock',
  'FirstResponse',
  'State',
  'Priority',
  'Queue',
  'Lock',
  'Owner',
  'UserFirstname',
  'UserLastname',
  'CustomerID',
  'CustomerName',
  'From',
  'Subject',
  'AccountedTime',
  'ArticleTree',
  'SolutionInMin',
  'SolutionDiffInMin',
  'FirstResponseInMin',
  'FirstResponseDiffInMin'
];
```

6.49.9. Ticket::Frontend::AgentTicketSearch###ArticleCreateTime

Description:	Includes article create times in the ticket search of the agent interface.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'ArticleCreateTime'} =  '0';`

6.49.10. Ticket::Frontend::AgentTicketSearch###Defaults###Fulltext

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'Fulltext'} =  '';`

6.49.11. Ticket::Frontend::AgentTicketSearch###Defaults###TicketNumber

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketNumber'} =  '';`

6.49.12. Ticket::Frontend::AgentTicketSearch###Defaults###Title

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'Title'} =  '';`

6.49.13. Ticket::Frontend::AgentTicketSearch###Defaults###From

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0

Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'From'} =  '';`

6.49.14. Ticket::Frontend::AgentTicketSearch###Defaults###To

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'To'} =  '';`

6.49.15. Ticket::Frontend::AgentTicketSearch###Defaults###Cc

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'Cc'} =  '';`

6.49.16. Ticket::Frontend::AgentTicketSearch###Defaults###Subject

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'Subject'} =  '';`

6.49.17. Ticket::Frontend::AgentTicketSearch###Defaults###Body

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'Body'} =  '';`

6.49.18. Ticket::Frontend::AgentTicketSearch###Defaults###CustomerID

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'CustomerID'} =  '';`

6.49.19. Ticket::Frontend::AgentTicketSearch###Defaults###CustomerUserLogin

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'CustomerUserLogin'} =  '';`

6.49.20. Ticket::Frontend::AgentTicketSearch###Defaults###StateIDs

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'StateIDs'} =  [];`

6.49.21. Ticket::Frontend::AgentTicketSearch###Defaults###QueueIDs

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'QueueIDs'} =  [];`

6.49.22. Ticket::Frontend::AgentTicketSearch###Defaults###TicketCreateTimePoint

Description:	Default data to use on attribute for ticket search screen. Example: "TicketCreateTimePointFormat=year;TicketCreateTimePointStart=Last;TicketCreate

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketCreateTimePoint'} =  '';`

6.49.23. Ticket::Frontend::AgentTicketSearch###Defaults###TicketCreateTimeSlot

Description:	Default data to use on attribute for ticket search screen. Example: "TicketCreateTimeStartYear=2010;TicketCreateTimeStartMonth=10;TicketCreateTi
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketCreateTimeSlot'} =  '';`

6.49.24. Ticket::Frontend::AgentTicketSearch###Defaults###TicketChangeTimePoint

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketChangeTimePoint'} =  '';`

6.49.25. Ticket::Frontend::AgentTicketSearch###Defaults###TicketChangeTimeSlot

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketChangeTimeSlot'} =  '';`

6.49.26. Ticket::Frontend::AgentTicketSearch###Defaults###TicketCloseTimePoint

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch

Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketCloseTimePoint'} =  '';`

6.49.27. Ticket::Frontend::AgentTicketSearch###Defaults###TicketCloseTimeSlot

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketCloseTimeSlot'} =  '';`

6.49.28. Ticket::Frontend::AgentTicketSearch###Defaults###TicketEscalationTimePoint

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketEscalationTimePoint'} =  '';`

6.49.29. Ticket::Frontend::AgentTicketSearch###Defaults###TicketEscalationTimeSlot

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'TicketEscalationTimeSlot'} =  '';`

6.49.30. Ticket::Frontend::AgentTicketSearch###Defaults###ArticleCreateTimePoint

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0

Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'ArticleCreateTimePoint'} =  '';`

6.49.31. Ticket::Frontend::AgentTicketSearch###Defaults###ArticleCreateTimeSlot

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'ArticleCreateTimeSlot'} =  '';`

6.49.32. Ticket::Frontend::AgentTicketSearch###Defaults###SearchInArchive

Description:	Defines the default shown ticket search attribute for ticket search screen.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'SearchInArchive'} =  '';`

6.49.33. Ticket::Frontend::CustomerTicketSearch###SearchArticleCSVTree

Description:	Exports the whole article tree in search result (it can affect the system performance).
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketSearch'}->{'SearchArticleCSVTree'} =  '0';`

6.49.34. Ticket::Frontend::AgentTicketSearch###DynamicField

Description:	Dynamic fields shown in the ticket search screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and shown by default.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'DynamicField'} = {};`

6.49.35. Ticket::Frontend::AgentTicketSearch###Defaults###DynamicField

Description:	Defines the default shown ticket search attribute for ticket search screen. Example: a text, 1, Search_DynamicField_Field1StartYear=2002; Search_DynamicField_Field1StartMonth=12; Search_DynamicField_Field1StartDay=12; Search_DynamicField_Field1StartHour=00; Search_DynamicField_Field1StartMinute=00; Search_DynamicField_Field1StartSecond=00; Search_DynamicField_Field1StopYear=2009; Search_DynamicField_Field1StopMonth=02; Search_DynamicField_Field1StopDay=10; Search_DynamicField_Field1StopHour=23; Search_DynamicField_Field1StopMinute=59; Search_DynamicField_Field1StopSecond=59;.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'Defaults'}->{'DynamicField'} =  {};`

6.49.36. Ticket::Frontend::AgentTicketSearch###SearchCSVDynamicField

Description:	Dynamic Fields used to export the search result in CSV format.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewSearch
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketSearch'}->{'SearchCSVDynamicField'} =  {};`

6.50. Frontend::Agent::Ticket::ViewStatus

6.50.1. Ticket::Frontend::AgentTicketStatusView###ViewableTicketsPage

Description:	Shows all open tickets (even if they are locked) in the status view of the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewStatus
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketStatusView'}->{'ViewableTicketsPage'} =  '50';`

6.50.2. Ticket::Frontend::AgentTicketStatusView###SortBy::Default

Description:	Defines the default ticket attribute for ticket sorting in the status view of the agent interface.

Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewStatus
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketStatusView'}->{'SortBy::Default'} =  'Age';`

6.50.3. Ticket::Frontend::AgentTicketStatusView###Order::Default

Description:	Defines the default ticket order (after priority sort) in the status view of the agent interface. Up: oldest on top. Down: latest on top.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewStatus
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketStatusView'}->{'Order::Default'} =  'Down';`

6.51. Frontend::Agent::Ticket::ViewZoom

6.51.1. Ticket::Frontend::PlainView

Description:	Shows a link to see a zoomed email ticket in plain text.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::PlainView'} =  '0';`

6.51.2. Ticket::Frontend::ZoomExpand

Description:	Shows all the articles of the ticket (expanded) in the zoom view.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::ZoomExpand'} =  '0';`

6.51.3. Ticket::Frontend::ZoomExpandSort

Description:	Shows the articles sorted normally or in reverse, under ticket zoom in the agent interface.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1

Config-Setting:	`$Self->{'Ticket::Frontend::ZoomExpandSort'} =  'normal';`

6.51.4. Ticket::ZoomAttachmentDisplayCount

Description:	Shows a count of icons in the ticket zoom, if the article has attachments.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::ZoomAttachmentDisplayCount'} =  '20';`

6.51.5. Ticket::ZoomTimeDisplay

Description:	Displays the accounted time for an article in the ticket zoom view.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::ZoomTimeDisplay'} =  '0';`

6.51.6. Ticket::UseArticleColors

Description:	Shows colors for different article types in the article table.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::UseArticleColors'} =  '0';`

6.51.7. Ticket::Frontend::TicketArticleFilter

Description:	Activates the article filter in the zoom view to specify which articles should be shown.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::TicketArticleFilter'} =  '0';`

6.51.8. Ticket::Frontend::HTMLArticleHeightDefault

Description:	Set the default height (in pixels) of inline HTML articles in AgentTicketZoom.
Group:	Ticket

SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::HTMLArticleHeightDefault'} =  '100';`

6.51.9. Ticket::Frontend::HTMLArticleHeightMax

Description:	Set the maximum height (in pixels) of inline HTML articles in AgentTicketZoom.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::HTMLArticleHeightMax'} =  '2500';`

6.51.10. Ticket::Frontend::ZoomRichTextForce

Description:	Show article as rich text even if rich text writing is disabled.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::ZoomRichTextForce'} =  '1';`

6.51.11. Ticket::Frontend::AgentTicketZoom###DynamicField

Description:	Dynamic fields shown in the sidebar of the ticket zoom screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Agent::Ticket::ViewZoom
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::AgentTicketZoom'}->{'DynamicField'} =  {};`

6.52. Frontend::Agent::TicketOverview

6.52.1. Ticket::Frontend::Overview###Small

Description:	Allows having a small format ticket overview (CustomerInfo => 1 - shows also the customer information).
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::Overview'}->{'Small'} =  {`

```
  'CustomerInfo' => '1',
  'Module' => 'Kernel::Output::HTML::TicketOverviewSmall',
  'ModulePriority' => '100',
  'Name' => 'Small',
  'NameShort' => 'S'
};
```

6.52.2. Ticket::Frontend::OverviewSmall###ColumnHeader

Description:	Shows either the last customer article's subject or the ticket title in the small format overview.
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::OverviewSmall'}->{'ColumnHeader'} =
'LastCustomerSubject';
```

6.52.3. Ticket::Frontend::Overview###Medium

Description:	Allows having a medium format ticket overview (CustomerInfo => 1 - shows also the customer information).
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::Overview'}->{'Medium'} =  {
  'CustomerInfo' => '0',
  'Module' => 'Kernel::Output::HTML::TicketOverviewMedium',
  'ModulePriority' => '200',
  'Name' => 'Medium',
  'NameShort' => 'M',
  'TicketActionsPerTicket' => '0'
};
```

6.52.4. Ticket::Frontend::Overview###Preview

Description:	Shows a preview of the ticket overview (CustomerInfo => 1 - shows also Customer-Info, CustomerInfoMaxSize max. size in characters of Customer-Info).
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::Overview'}->{'Preview'} =  {
  'CustomerInfo' => '0',
  'CustomerInfoMaxSize' => '18',
  'DefaultPreViewLines' => '25',
  'DefaultViewNewLine' => '90',
  'Module' => 'Kernel::Output::HTML::TicketOverviewPreview',
  'ModulePriority' => '300',
  'Name' => 'Preview',
  'NameShort' => 'L',
  'StripEmptyLines' => '0',
  'TicketActionsPerTicket' => '0'
};
```

6.52.5. Ticket::Frontend::Overview::PreviewArticleSenderTypes

Description:	Defines which article sender types should be shown in the preview of a ticket.
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::Overview::PreviewArticleSenderTypes'} =  {
  'agent' => '1',
  'customer' => '1',
  'system' => '1'
};
```

6.52.6. Ticket::Frontend::Overview::PreviewArticleTypeExpanded

Description:	Defines wich article type should be expanded when entering the overview. If nothing defined, latest article will be expanded.
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::Overview::PreviewArticleTypeExpanded'} =
'';
```

6.52.7. Ticket::Frontend::OverviewSmall###DynamicField

Description:	Dynamic fields shown in the ticket small format overview screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::OverviewSmall'}->{'DynamicField'} =  {};`

6.52.8. Ticket::Frontend::OverviewMedium###DynamicField

Description:	Dynamic fields shown in the ticket medium format overview screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::OverviewMedium'}->{'DynamicField'} =  {};`

6.52.9. Ticket::Frontend::OverviewPreview###DynamicField

Description:	Dynamic fields shown in the ticket preview format overview screen of the agent interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Agent::TicketOverview
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::OverviewPreview'}->{'DynamicField'} =  {};`

6.53. Frontend::Agent::ToolBarModule

6.53.1. Frontend::ToolBarModule###1-Ticket::AgentTicketQueue

Description:	Toolbar Item for a shortcut.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::ToolBarModule'}->{'1-Ticket::AgentTicketQueue'} =
{
  'AccessKey' => 'q',
  'Action' => 'AgentTicketQueue',
  'CssClass' => 'QueueView',
  'Link' => 'Action=AgentTicketQueue',
  'Module' => 'Kernel::Output::HTML::ToolBarLink',
  'Name' => 'Queue view',
  'Priority' => '1010010'
};
```

6.53.2. Frontend::ToolBarModule###2-Ticket::AgentTicketStatus

Description:	Toolbar Item for a shortcut.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::ToolBarModule'}->{'2-Ticket::AgentTicketStatus'} =
  {
  'AccessKey' => 'o',
  'Action' => 'AgentTicketStatusView',
  'CssClass' => 'StatusView',
  'Link' => 'Action=AgentTicketStatusView',
  'Module' => 'Kernel::Output::HTML::ToolBarLink',
  'Name' => 'Status view',
  'Priority' => '1010020'
};
```

6.53.3. Frontend::ToolBarModule###3-Ticket::AgentTicketEscalation

Description:	Toolbar Item for a shortcut.
Group:	Ticket

SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::ToolBarModule'}->{'3-
Ticket::AgentTicketEscalation'} =  {
  'AccessKey' => 'w',
  'Action' => 'AgentTicketEscalationView',
  'CssClass' => 'EscalationView',
  'Link' => 'Action=AgentTicketEscalationView',
  'Module' => 'Kernel::Output::HTML::ToolBarLink',
  'Name' => 'Escalation view',
  'Priority' => '1010030'
};
```

6.53.4. Frontend::ToolBarModule###4-Ticket::AgentTicketPhone

Description:	Toolbar Item for a shortcut.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::ToolBarModule'}->{'4-Ticket::AgentTicketPhone'} =
 {
  'AccessKey' => 'l',
  'Action' => 'AgentTicketPhone',
  'CssClass' => 'PhoneTicket',
  'Link' => 'Action=AgentTicketPhone',
  'Module' => 'Kernel::Output::HTML::ToolBarLink',
  'Name' => 'New phone ticket',
  'Priority' => '1020010'
};
```

6.53.5. Frontend::ToolBarModule###5-Ticket::AgentTicketEmail

Description:	Toolbar Item for a shortcut.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::ToolBarModule'}->{'5-Ticket::AgentTicketEmail'} =
 {
  'AccessKey' => 'l',
  'Action' => 'AgentTicketEmail',
  'CssClass' => 'EmailTicket',
  'Link' => 'Action=AgentTicketEmail',
  'Module' => 'Kernel::Output::HTML::ToolBarLink',
  'Name' => 'New email ticket',
  'Priority' => '1020020'
};
```

6.53.6. Frontend::ToolBarModule###6-Ticket::TicketResponsible

Description:	Agent interface notification module to see the number of tickets an agent is responsible for.

Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::ToolBarModule'}->{'6-Ticket::TicketResponsible'} =
  {
  'CssClass' => 'Responsible',
  'CssClassNew' => 'Responsible New',
  'CssClassReached' => 'Responsible Reached',
  'Module' => 'Kernel::Output::HTML::ToolBarTicketResponsible',
  'Priority' => '1030010'
};
```

6.53.7. Frontend::ToolBarModule###7-Ticket::TicketWatcher

Description:	Agent interface notification module to see the number of watched tickets.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::ToolBarModule'}->{'7-Ticket::TicketWatcher'} =  {
  'CssClass' => 'Watcher',
  'CssClassNew' => 'Watcher New',
  'CssClassReached' => 'Watcher Reached',
  'Module' => 'Kernel::Output::HTML::ToolBarTicketWatcher',
  'Priority' => '1030020'
};
```

6.53.8. Frontend::ToolBarModule###8-Ticket::TicketLocked

Description:	Agent interface notification module to check the used charset.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	1
Required:	0
Config-Setting:	

```
 $Self->{'Frontend::ToolBarModule'}->{'8-Ticket::TicketLocked'} =  {
  'CssClass' => 'Locked',
  'CssClassNew' => 'Locked New',
  'CssClassReached' => 'Locked Reached',
  'Module' => 'Kernel::Output::HTML::ToolBarTicketLocked',
  'Priority' => '1030030'
};
```

6.53.9. Frontend::ToolBarModule###9-Ticket::TicketSearchProfile

Description:	Agent interface module to access search profiles via nav bar.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::ToolBarModule'}->{'9-Ticket::TicketSearchProfile'}
 =  {
  'Block' => 'ToolBarSearchProfile',
  'Description' => 'Search-Template',
  'MaxWidth' => '40',
  'Module' => 'Kernel::Output::HTML::ToolBarTicketSearchProfile',
  'Name' => 'Search-Template',
  'Priority' => '1990010'
};
```

6.53.10. Frontend::ToolBarModule###10-Ticket::TicketSearchFulltext

Description:	Agent interface module to access fulltext search via nav bar.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::ToolBarModule'}->{'10-
Ticket::TicketSearchFulltext'} =  {
  'Block' => 'ToolBarSearchFulltext',
  'CSS' => 'Core.Agent.Toolbar.FulltextSearch.css',
  'Description' => 'Fulltext-Search',
  'Module' => 'Kernel::Output::HTML::ToolBarGeneric',
  'Name' => 'Fulltext-Search',
  'Priority' => '1990020',
  'Size' => '10'
};
```

6.53.11. Frontend::ToolBarModule###11-CICSearchCustomerID

Description:	Agent interface module to access CIC search via nav bar.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::ToolBarModule'}->{'11-CICSearchCustomerID'} =  {
  'Block' => 'ToolBarCICSearchCustomerID',
  'CSS' => 'Core.Agent.Toolbar.CICSearch.css',
  'Data' => {
    'MaxResultsDisplayed' => '20',
    'MinQueryLength' => '2',
    'QueryDelay' => '100'
  },
  'Description' => 'CIC search for CustomerID',
  'Module' => 'Kernel::Output::HTML::ToolBarGeneric',
  'Name' => 'CostomerID search',
  'Priority' => '1990030',
  'Size' => '10'
};
```

6.53.12. Frontend::ToolBarModule###11-CICSearchCustomerUser

Description:	Agent interface module to access CIC search via nav bar.
Group:	Ticket
SubGroup:	Frontend::Agent::ToolBarModule

Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Frontend::ToolBarModule'}->{'11-CICSearchCustomerUser'} =  {
  'Block' => 'ToolBarCICSearchCustomerUser',
  'CSS' => 'Core.Agent.Toolbar.CICSearch.css',
  'Data' => {
    'MaxResultsDisplayed' => '20',
    'MinQueryLength' => '2',
    'QueryDelay' => '100'
  },
  'Description' => 'CIC search for CustomerUser',
  'Module' => 'Kernel::Output::HTML::ToolBarGeneric',
  'Name' => 'Costomer user search',
  'Priority' => '1990040',
  'Size' => '10'
};
```

6.54. Frontend::Customer

6.54.1. Ticket::Frontend::CustomerTicketOverviewCustomEmptyText

Description:	Custom text for the page shown to customers that have no tickets yet.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::CustomerTicketOverviewCustomEmptyText'} =
{
  'Button' => 'Create your first ticket',
  'Text' => 'Please click the button below to create your first
ticket.',
  'Title' => 'Welcome!'
};
```

6.54.2. Frontend::CustomerUser::Item###15-OpenTickets

Description:	Customer item (icon) which shows the open tickets of this customer as info block. Setting CustomerUserLogin to 1 searches for tickets based on login name rather than CustomerID.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Frontend::CustomerUser::Item'}->{'15-OpenTickets'} =  {
  'Action' => 'AgentTicketSearch',
  'Attributes' => 'StateType=Open;',
  'CSS' => 'Core.Agent.CustomerUser.OpenTicket.css',
  'CSSClassNoOpenTicket' => 'NoOpenTicket',
  'CSSClassOpenTicket' => 'OpenTicket',
  'CustomerUserLogin' => '0',
  'Module' => 'Kernel::Output::HTML::CustomerUserGenericTicket',
  'Subaction' => 'Search',
  'Target' => '_blank',
  'Text' => 'Open tickets'
};
```

6.54.3. Frontend::CustomerUser::Item###16-OpenTicketsForCustomerUserLogin

Description:	Customer item (icon) which shows the open tickets of this customer as info block. Setting CustomerUserLogin to 1 searches for tickets based on login name rather than CustomerID.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::CustomerUser::Item'}->{'16-OpenTicketsForCustomerUserLogin'} =  {
  'Action' => 'AgentTicketSearch',
  'Attributes' => 'StateType=Open;',
  'CSS' => 'Core.Agent.CustomerUser.OpenTicket.css',
  'CSSClassNoOpenTicket' => 'NoOpenTicket',
  'CSSClassOpenTicket' => 'OpenTicket',
  'CustomerUserLogin' => '1',
  'Module' => 'Kernel::Output::HTML::CustomerUserGenericTicket',
  'Subaction' => 'Search',
  'Target' => '_blank',
  'Text' => 'Open tickets of customer'
};
```

6.54.4. Frontend::CustomerUser::Item###17-ClosedTickets

Description:	Customer item (icon) which shows the closed tickets of this customer as info block. Setting CustomerUserLogin to 1 searches for tickets based on login name rather than CustomerID.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	0
Required:	0

Config-Setting:

```
 $Self->{'Frontend::CustomerUser::Item'}->{'17-ClosedTickets'} =  {
  'Action' => 'AgentTicketSearch',
  'Attributes' => 'StateType=Closed;',
  'CSS' => 'Core.Agent.CustomerUser.OpenTicket.css',
  'CSSClassNoOpenTicket' => 'NoOpenTicket',
  'CSSClassOpenTicket' => 'OpenTicket',
  'CustomerUserLogin' => '0',
  'Module' => 'Kernel::Output::HTML::CustomerUserGenericTicket',
  'Subaction' => 'Search',
  'Target' => '_blank',
  'Text' => 'Closed tickets'
};
```

6.54.5. Frontend::CustomerUser::Item###18-ClosedTicketsForCustomerUserLogin

Description:	Customer item (icon) which shows the closed tickets of this customer as info block. Setting CustomerUserLogin to 1 searches for tickets based on login name rather than CustomerID.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	0

Required:	0

Config-Setting:

```
 $Self->{'Frontend::CustomerUser::Item'}->{'18-
ClosedTicketsForCustomerUserLogin'} =  {
  'Action' => 'AgentTicketSearch',
  'Attributes' => 'StateType=Closed;',
  'CSS' => 'Core.Agent.CustomerUser.OpenTicket.css',
  'CSSClassNoOpenTicket' => 'NoOpenTicket',
  'CSSClassOpenTicket' => 'OpenTicket',
  'CustomerUserLogin' => '1',
  'Module' => 'Kernel::Output::HTML::CustomerUserGenericTicket',
  'Subaction' => 'Search',
  'Target' => '_blank',
  'Text' => 'Closed tickets of customer'
};
```

6.54.6. CustomerFrontend::CommonObject###QueueObject

Description:	Path of the file that stores all the settings for the QueueObject object for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'CustomerFrontend::CommonObject'}->{'QueueObject'} =
 'Kernel::System::Queue';
```

6.54.7. CustomerFrontend::CommonObject###TicketObject

Description:	Path of the file that stores all the settings for the TicketObject for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'CustomerFrontend::CommonObject'}->{'TicketObject'} =
 'Kernel::System::Ticket';
```

6.54.8. CustomerFrontend::CommonParam###Action

Description:	Defines the default used Frontend-Module if no Action parameter given in the url on the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'CustomerFrontend::CommonParam'}->{'Action'} =
 'CustomerTicketOverview';
```

6.54.9. CustomerFrontend::CommonParam###TicketID

Description:	Default ticket ID used by the system in the customer interface.

Group:	Ticket
SubGroup:	Frontend::Customer
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerFrontend::CommonParam'}->{'TicketID'} =  '';`

6.55. Frontend::Customer::ModuleMetaHead

6.55.1. CustomerFrontend::HeaderMetaModule###2-TicketSearch

Description:	Module to generate html OpenSearch profile for short ticket search in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::ModuleMetaHead
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'CustomerFrontend::HeaderMetaModule'}->{'2-TicketSearch'} =
{
  'Action' => 'CustomerTicketSearch',
  'Module' => 'Kernel::Output::HTML::CustomerHeaderMetaTicketSearch'
};
```

6.56. Frontend::Customer::ModuleRegistration

6.56.1. CustomerFrontend::Module###CustomerTicketOverview

Description:	Frontend module registration for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'CustomerFrontend::Module'}->{'CustomerTicketOverview'} =  {
  'Description' => 'Overview of customer tickets',
  'NavBar' => [
    {
      'AccessKey' => 'm',
      'Block' => '',
      'Description' => 'Tickets',
      'Link' => 'Action=CustomerTicketOverview;Subaction=MyTickets',
      'LinkOption' => '',
      'Name' => 'Tickets',
      'NavBar' => 'Ticket',
      'Prio' => '100',
      'Type' => 'Menu'
    },
    {
      'AccessKey' => 'm',
      'Block' => '',
      'Description' => 'My Tickets',
      'Link' => 'Action=CustomerTicketOverview;Subaction=MyTickets',
      'LinkOption' => '',
      'Name' => 'My Tickets',
      'NavBar' => 'Ticket',
      'Prio' => '110',
      'Type' => 'Submenu'
    },
    {
      'AccessKey' => 'c',
      'Block' => '',
      'Description' => 'Company Tickets',
      'Link' =>
 'Action=CustomerTicketOverview;Subaction=CompanyTickets',
      'LinkOption' => '',
      'Name' => 'Company Tickets',
      'NavBar' => 'Ticket',
      'Prio' => '120',
      'Type' => 'Submenu'
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'Overview'
};
```

6.56.2. CustomerFrontend::Module###CustomerTicketMessage

Description:	Frontend module registration for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerFrontend::Module'}->{'CustomerTicketMessage'} =  {
  'Description' => 'Create tickets',
  'NavBar' => [
    {
      'AccessKey' => 'n',
      'Block' => '',
      'Description' => 'Create new Ticket',
      'Link' => 'Action=CustomerTicketMessage',
      'LinkOption' => '',
      'Name' => 'New Ticket',
      'NavBar' => 'Ticket',
      'Prio' => '100',
      'Type' => 'Submenu'
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'New Ticket'
};
```

6.56.3. CustomerFrontend::Module###CustomerTicketZoom

Description:	Frontend module registration for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerFrontend::Module'}->{'CustomerTicketZoom'} =  {
  'Description' => 'Ticket zoom view',
  'Loader' => {
    'JavaScript' => [
      'Core.Customer.TicketZoom.js',
      'Core.UI.Popup.js'
    ]
  },
  'NavBarName' => 'Ticket',
  'Title' => 'Zoom'
};
```

6.56.4. CustomerFrontend::Module###CustomerTicketPrint

Description:	Frontend module registration for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerFrontend::Module'}->{'CustomerTicketPrint'} =  {
  'Description' => 'Customer Ticket Print Module',
  'NavBarName' => '',
  'Title' => 'Print'
};
```

6.56.5. CustomerFrontend::Module###CustomerTicketAttachment

Description:	Frontend module registration for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerFrontend::Module'}->{'CustomerTicketAttachment'} =
 {
  'Description' => 'To download attachments',
  'NavBarName' => '',
  'Title' => ''
};
```

6.56.6. CustomerFrontend::Module###CustomerTicketSearch

Description:	Frontend module registration for the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::ModuleRegistration
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerFrontend::Module'}->{'CustomerTicketSearch'} =  {
  'Description' => 'Customer ticket search',
  'NavBar' => [
    {
      'AccessKey' => 's',
      'Block' => '',
      'Description' => 'Search',
      'Link' => 'Action=CustomerTicketSearch',
      'LinkOption' => '',
      'Name' => 'Search',
      'NavBar' => 'Ticket',
      'Prio' => '300',
      'Type' => 'Submenu'
    }
  ],
  'NavBarName' => 'Ticket',
  'Title' => 'Search'
};
```

6.57. Frontend::Customer::Preferences

6.57.1. CustomerPreferencesGroups###ShownTickets

Description:	Defines all the parameters for the ShownTickets object in the customer preferences of the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerPreferencesGroups'}->{'ShownTickets'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Data' => {
    '15' => '15',
    '20' => '20',
    '25' => '25',
    '30' => '30'
  },
  'DataSelected' => '25',
  'Key' => 'Tickets per page',
  'Label' => 'Number of displayed tickets',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserShowTickets',
  'Prio' => '4000'
};
```

6.57.2. CustomerPreferencesGroups###RefreshTime

Description:	Defines all the parameters for the RefreshTime object in the customer preferences of the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Preferences
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'CustomerPreferencesGroups'}->{'RefreshTime'} =  {
  'Active' => '1',
  'Column' => 'User Profile',
  'Data' => {
    '' => 'off',
    '10' => '10 minutes',
    '15' => '15 minutes',
    '2' => ' 2 minutes',
    '5' => ' 5 minutes',
    '7' => ' 7 minutes'
  },
  'DataSelected' => '',
  'Key' => 'Refresh interval',
  'Label' => 'Ticket overview',
  'Module' => 'Kernel::Output::HTML::PreferencesGeneric',
  'PrefKey' => 'UserRefreshTime',
  'Prio' => '4000'
};
```

6.58. Frontend::Customer::Ticket::ViewNew

6.58.1. Ticket::Frontend::CustomerTicketMessage###NextScreenAfterNewTicket

Description:	Determines the next screen after new customer ticket in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'NextScreenAfterNewTicket'} =  'CustomerTicketOverview';`

6.58.2. Ticket::Frontend::CustomerTicketMessage###Priority

Description:	Allows customers to set the ticket priority in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'Priority'} =  '1';`

6.58.3. Ticket::Frontend::CustomerTicketMessage###PriorityDefault

Description:	Defines the default priority of new customer tickets in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'PriorityDefault'} =  '3 normal';`

6.58.4. Ticket::Frontend::CustomerTicketMessage###Queue

Description:	Allows customers to set the ticket queue in the customer interface. If this is set to 'No', QueueDefault should be configured.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'Queue'} =  '1';`

6.58.5. Ticket::Frontend::CustomerTicketMessage###QueueDefault

Description:	Defines the default queue for new customer tickets in the customer interface.

Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'QueueDefault'} =  'Postmaster';`

6.58.6. Ticket::Frontend::CustomerTicketMessage###TicketType

Description:	Allows customers to set the ticket type in the customer interface. If this is set to 'No', TicketTypeDefault should be configured.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'TicketType'} = '1';`

6.58.7. Ticket::Frontend::CustomerTicketMessage###TicketTypeDefault

Description:	Defines the default ticket type for new customer tickets in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	0
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'TicketTypeDefault'} =  'default';`

6.58.8. Ticket::Frontend::CustomerTicketMessage###Service

Description:	Allows customers to set the ticket service in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'Service'} = '1';`

6.58.9. Ticket::Frontend::CustomerTicketMessage###SLA

Description:	Allows customers to set the ticket SLA in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1

Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'SLA'} =  '1';`

6.58.10. Ticket::Frontend::CustomerTicketMessage###StateDefault

Description:	Defines the default state of new customer tickets in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'StateDefault'} =  'new';`

6.58.11. Ticket::Frontend::CustomerTicketMessage###ArticleType

Description:	Defines the default type for article in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'ArticleType'} =  'webrequest';`

6.58.12. Ticket::Frontend::CustomerTicketMessage###SenderType

Description:	Sender type for new tickets from the customer inteface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'SenderType'} = 'customer';`

6.58.13. Ticket::Frontend::CustomerTicketMessage###HistoryType

Description:	Defines the default history type in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'HistoryType'} =  'WebRequestCustomer';`

6.58.14. Ticket::Frontend::CustomerTicketMessage###HistoryComment

Description:	Comment for new history entries in the customer interface.
Group:	Ticket

SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'HistoryComment'} =  '';`

6.58.15. CustomerPanelSelectionType

Description:	Defines the receipent target of the tickets ("Queue" shows all queues, "SystemAddress" displays all system addresses) in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanelSelectionType'} =  'Queue';`

6.58.16. CustomerPanelSelectionString

Description:	Determines the strings that will be shown as receipent (To:) of the ticket in the customer interface. For Queue as CustomerPanelSelectionType, "<Queue>" shows the names of the queues, and for SystemAddress, "<Realname> <<Email>>" shows the name and email of the receipent.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'CustomerPanelSelectionString'} =  '<Queue>';`

6.58.17. CustomerPanelOwnSelection

Description:	Determines which queues will be valid for ticket's recepients in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	0
Required:	0
Config-Setting:	

```
$Self->{'CustomerPanelOwnSelection'} =  {
  'Junk' => 'First Queue',
  'Misc' => 'Second Queue'
};
```

6.58.18. CustomerPanel::NewTicketQueueSelectionModule

Description:	Module for To-selection in new ticket screen in the customer interface.
Group:	Ticket

SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	1
Config-Setting:	`$Self->{'CustomerPanel::NewTicketQueueSelectionModule'} = 'Kernel::Output::HTML::CustomerNewTicketQueueSelectionGeneric';`

6.58.19. Ticket::Frontend::CustomerTicketMessage###DynamicField

Description:	Dynamic fields options shown in the ticket message screen of the customer interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required. NOTE. If you want to display these fields also in the ticket zoom of the customer interface, you have to enable them in CustomerTicketZoom###DynamicField.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewNew
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketMessage'}->{'DynamicField'} =  {};`

6.59. Frontend::Customer::Ticket::ViewPrint

6.59.1. Ticket::Frontend::CustomerTicketPrint###DynamicField

Description:	Dynamic fields shown in the ticket print screen of the customer interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewPrint
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketPrint'}->{'DynamicField'} = {};`

6.60. Frontend::Customer::Ticket::ViewSearch

6.60.1. Ticket::CustomerTicketSearch::SearchLimit

Description:	Maximum number of tickets to be displayed in the result of a search in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::CustomerTicketSearch::SearchLimit'} =  '5000';`

6.60.2. Ticket::CustomerTicketSearch::SearchPageShown

Description:	Number of tickets to be displayed in each page of a search result in the customer interface.

Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::CustomerTicketSearch::SearchPageShown'} =  '40';`

6.60.3. Ticket::CustomerTicketSearch::SortBy::Default

Description:	Defines the default ticket attribute for ticket sorting in a ticket search of the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::CustomerTicketSearch::SortBy::Default'} =  'Age';`

6.60.4. Ticket::CustomerTicketSearch::Order::Default

Description:	Defines the default ticket order of a search result in the customer interface. Up: oldest on top. Down: latest on top.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::CustomerTicketSearch::Order::Default'} =  'Down';`

6.60.5. Ticket::Frontend::CustomerTicketSearch###ExtendedSearchCondition

Description:	Allows extended search conditions in ticket search of the customer interface. With this feature you can search w. g. with this kind of conditions like "(key1&&key2)" or "(key1||key2)".
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketSearch'}->{'ExtendedSearchCondition'} =  '1';`

6.60.6. Ticket::Frontend::CustomerTicketSearch###SearchCSVData

Description:	Data used to export the search result in CSV format.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	1

Config-Setting:

```
 $Self->{'Ticket::Frontend::CustomerTicketSearch'}->{'SearchCSVData'}
 =  [
  'TicketNumber',
  'Age',
  'Created',
  'Closed',
  'State',
  'Priority',
  'Lock',
  'CustomerID',
  'CustomerName',
  'From',
  'Subject'
];
```

6.60.7. Ticket::Frontend::CustomerTicketSearch###DynamicField

Description:	Dynamic fields shown in the ticket search screen of the customer interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::CustomerTicketSearch'}->{'DynamicField'} =
  {};
```

6.60.8. Ticket::Frontend::CustomerTicketSearch###SearchOverviewDynamicField

Description:	Dynamic fields shown in the ticket search overview results screen of the customer interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::CustomerTicketSearch'}-
>{'SearchOverviewDynamicField'} =  {};
```

6.60.9. Ticket::Frontend::CustomerTicketSearch###SearchCSVDynamicField

Description:	Dynamic Fields used to export the search result in CSV format.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewSearch
Valid:	1
Required:	0

Config-Setting:

```
 $Self->{'Ticket::Frontend::CustomerTicketSearch'}-
>{'SearchCSVDynamicField'} =  {};
```

6.61. Frontend::Customer::Ticket::ViewZoom

6.61.1. Ticket::Frontend::CustomerTicketZoom###NextScreenAfterFollowUp

Description:	Determines the next screen after the follow up screen of a zoomed ticket in the customer interface.

Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'NextScreenAfterFollowUp'} =  'CustomerTicketOverview';`

6.61.2. Ticket::Frontend::CustomerTicketZoom###ArticleType

Description:	Defines the default type of the note in the ticket zoom screen of the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'ArticleType'} = 'webrequest';`

6.61.3. Ticket::Frontend::CustomerTicketZoom###SenderType

Description:	Defines the default sender type for tickets in the ticket zoom screen of the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'SenderType'} = 'customer';`

6.61.4. Ticket::Frontend::CustomerTicketZoom###HistoryType

Description:	Defines the history type for the ticket zoom action, which gets used for ticket history in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'HistoryType'} = 'FollowUp';`

6.61.5. Ticket::Frontend::CustomerTicketZoom###HistoryComment

Description:	Defines the history comment for the ticket zoom action, which gets used for ticket history in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom

Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'HistoryComment'} = '';`

6.61.6. Ticket::Frontend::CustomerTicketZoom###Priority

Description:	Allows customers to change the ticket priority in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'Priority'} =  '1';`

6.61.7. Ticket::Frontend::CustomerTicketZoom###PriorityDefault

Description:	Defines the default priority of follow up customer tickets in the ticket zoom screen in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'PriorityDefault'} =  '3 normal';`

6.61.8. Ticket::Frontend::CustomerTicketZoom###State

Description:	Allows choosing the next compose state for customer tickets in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'State'} =  '1';`

6.61.9. Ticket::Frontend::CustomerTicketZoom###StateDefault

Description:	Defines the default next state for a ticket after customer follow up in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'StateDefault'} = 'open';`

6.61.10. Ticket::Frontend::CustomerTicketZoom###StateType

Description:	Defines the next possible states for customer tickets in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'StateType'} =  [
  'open',
  'closed'
];
```

6.61.11. Ticket::Frontend::CustomerTicketZoom###AttributesView

Description:	Shows the activated ticket attributes in the customer interface (0 = Disabled and 1 = Enabled).
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	1
Config-Setting:	

```
$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'AttributesView'} =
  {
  'Owner' => '0',
  'Priority' => '1',
  'Queue' => '1',
  'Responsible' => '0',
  'SLA' => '0',
  'Service' => '0',
  'State' => '1',
  'Type' => '0'
};
```

6.61.12. Ticket::Frontend::CustomerTicketZoom###DynamicField

Description:	Dynamic fields shown in the ticket zoom screen of the customer interface. Possible settings: 0 = Disabled, 1 = Enabled.
Group:	Ticket
SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	0
Config-Setting:	

```
$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'DynamicField'} =
{};
```

6.61.13. Ticket::Frontend::CustomerTicketZoom###FollowUpDynamicField

Description:	Dynamic fields options shown in the ticket reply section in the ticket zoom screen of the customer interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket

SubGroup:	Frontend::Customer::Ticket::ViewZoom
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketZoom'}->{'FollowUpDynamicField'} =  {};`

6.62. Frontend::Customer::TicketOverview

6.62.1. Ticket::Frontend::CustomerTicketOverviewSortable

Description:	Controls if customers have the ability to sort their tickets.
Group:	Ticket
SubGroup:	Frontend::Customer::TicketOverview
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketOverviewSortable'} =  '';`

6.62.2. Ticket::Frontend::CustomerTicketOverview###ColumnHeader

Description:	Shows either the last customer article's subject or the ticket title in the small format overview.
Group:	Ticket
SubGroup:	Frontend::Customer::TicketOverview
Valid:	1
Required:	0
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketOverview'}->{'ColumnHeader'} =  'TicketTitle';`

6.62.3. Ticket::Frontend::CustomerTicketOverview###Owner

Description:	Show the current owner in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::TicketOverview
Valid:	1
Required:	1
Config-Setting:	`$Self->{'Ticket::Frontend::CustomerTicketOverview'}->{'Owner'} = '0';`

6.62.4. Ticket::Frontend::CustomerTicketOverview###Queue

Description:	Show the current queue in the customer interface.
Group:	Ticket
SubGroup:	Frontend::Customer::TicketOverview
Valid:	1
Required:	1

Config-Setting:

```
$Self->{'Ticket::Frontend::CustomerTicketOverview'}->{'Queue'} =
'0';
```

6.62.5. Ticket::Frontend::CustomerTicketOverview###DynamicField

Description:	Dynamic fields shown in the ticket overview screen of the customer interface. Possible settings: 0 = Disabled, 1 = Enabled, 2 = Enabled and required.
Group:	Ticket
SubGroup:	Frontend::Customer::TicketOverview
Valid:	1
Required:	0

Config-Setting:

```
$Self->{'Ticket::Frontend::CustomerTicketOverview'}->{'DynamicField'}
=  {};
```

6.63. Frontend::Queue::Preferences

6.63.1. QueuePreferences###Comment2

Description:	Parameters of the example queue attribute Comment2.
Group:	Ticket
SubGroup:	Frontend::Queue::Preferences
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'QueuePreferences'}->{'Comment2'} =  {
  'Block' => 'TextArea',
  'Cols' => '50',
  'Desc' => 'Define the queue comment 2.',
  'Label' => 'Comment2',
  'Module' => 'Kernel::Output::HTML::QueuePreferencesGeneric',
  'PrefKey' => 'Comment2',
  'Rows' => '5'
};
```

6.64. Frontend::SLA::Preferences

6.64.1. SLAPreferences###Comment2

Description:	Parameters of the example SLA attribute Comment2.
Group:	Ticket
SubGroup:	Frontend::SLA::Preferences
Valid:	0
Required:	0

Config-Setting:

```
$Self->{'SLAPreferences'}->{'Comment2'} =  {
  'Block' => 'TextArea',
  'Cols' => '50',
  'Desc' => 'Define the sla comment 2.',
  'Label' => 'Comment2',
  'Module' => 'Kernel::Output::HTML::SLAPreferencesGeneric',
  'PrefKey' => 'Comment2',
  'Rows' => '5'
};
```

6.65. Frontend::Service::Preferences

6.65.1. ServicePreferences###Comment2

Description:	Parameters of the example service attribute Comment2.
Group:	Ticket
SubGroup:	Frontend::Service::Preferences
Valid:	0
Required:	0
Config-Setting:	

```
 $Self->{'ServicePreferences'}->{'Comment2'} =  {
  'Block' => 'TextArea',
  'Cols' => '50',
  'Desc' => 'Define the service comment 2.',
  'Label' => 'Comment2',
  'Module' => 'Kernel::Output::HTML::ServicePreferencesGeneric',
  'PrefKey' => 'Comment2',
  'Rows' => '5'
};
```

Appendix C. Credits

OTRS is an open source project, and we would like to thank many people for their help and support. The following list is surely incomplete, and we apologize for that! Just drop us a note if you are not on this list.

The following persons have especially pushed the project or are still active supporters:

- Robert Kehl, who created the Win32-installer for the 1.x releases. Thanks a lot, Robert!
- Torsten Werner, who maintains the Debian-installer for OTRS. Thanks a lot, Torsten, by making OTRS also available for the Debian community.
- Nils Jeppe (mirror Hamburg, Germany), Bryan Fullerton (mirror Toronto, Canada), Eberhard Mönkeberg (mirror Göttingen, Germany), Timo Dreger (mirror Düsseldorf, Germany) and Netmonic (mirror Vienna, Austria), who are mirroring our FTP server. Thanks a lot, with your help it is always possible to download OTRS quickly!
- Anja Schneider, who is helping with the translation of this manual and with the correction and revision of the German texts. Many thanks for your help and patience, Anja!
- We receive many ideas via the OTRS mailing lists for system improvements, patches or bugfixes. Great support for all users is always available there. Thanks a lot to all people on the mailing lists for your active assistance!

Also we would like to extend a big thank you to the following persons:

- Martin Scherbaum
- Carsten Gross
- Harald Müller
- Stefan Schmidt
- Milisav Radmanic
- Uli Hecht
- Norman Walsh
- Heiko Baumann
- Atif Ghaffar
- Pablo Ruiz Garcia
- Dan Rau
- Christoph Kaulich
- Mark Jackson
- Diane Shieh
- Bernard Choppy
- Carl Bailey

- Phil Davis
- Edwin D. Vinas
- Lars Müller
- Vladimir Gerdjikov
- Fred van Dijk
- Sebastien Guilbaud
- Wiktor Wodecki
- Arnold Ligtvoet
- Antti Kämäräinen
- Nicolas Goralski
- Gilberto Cezar de Almeida
- Jorge Becerra
- Eddie Urenda
- Stella Power
- Andreas Haase
- Reiner Keller
- Covert Jake
- Moshe Leibovitch
- Bjärn Jacke
- Remo Catelotti
- Alfons Obermeyer
- Michael Rech
- Danie Theron
- Richard Paradies
- Art Powell

Appendix D. GNU Free Documentation License

Version 1.1, March 2000

Copyright (C) 2000 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document, that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters), and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats, suitable for input to text formatters. A copy made in an otherwise Transparent file format, whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include: plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include: PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location, containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously, at no charge, using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location, until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version, as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License, for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections, in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

. How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation

> License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.