

T···Systems···

rvsEVO

Version 5.04

User Manual

The products listed in this manual are protected by copyright.

rvsEVO

Version 5.04

User Manual

© 2013 by T-Systems International GmbH

Holzhauser Straße 4 - 8

13509 Berlin

This manual is protected by copyright. All rights reserved. No part of this book may be used or reproduced in any form or by any means including photocopies, microfilm or any other means or stored in a database or retrieval system without obtaining prior permission from T-Systems. Rights are also reserved as far as lectures, radio and television is concerned.

We reserve the right to make changes to the content of this manual without giving prior notice. T-Systems is not liable for technical or printing errors or defects in this manual. Moreover, T-Systems shall not be liable for damage which is directly or indirectly caused by delivery, performance and use of this material.

1	Introduction	9
1.1	Short description of the system	9
1.2	rvsEVO Tiny Edition	11
1.3	rvsEVO Light Edition	12
1.4	rvsEVO Standard Edition	12
1.5	rvsEVO Enterprise Edition	13
1.6	Representation means	14
1.7	Target group	14
2	Installation	15
2.1	System requirements	15
2.2	Obtaining a license	15
2.3	Network Requirements	16
2.4	Fresh installation of rvsEVO	17
2.5	Graphical User Interface (brief description):	26
2.6	How to start rvsEVO?	27
2.7	How to stop rvsEVO?	28
2.8	rvsEVO as Windows service	29
2.9	rvsEVO update installation	30
2.10	Migration from rvs® portable to rvsEVO	30
2.11	Displaying license key information	33
2.12	Uninstall rvsEVO	34
3	Configuration	37
3.1	Customizing the global rvsEVO parameters	37
3.1.1	rvsEVO Environment	38
3.1.2	Notification (SNMP)	42
3.1.3	Observer	43
3.1.4	Resource Check	45
3.2	Customizing the station configuration	49
3.2.1	Customizing stations via GUI	49
3.2.2	Configuring a local station	50
3.2.3	Setting up of a neighbour station	58
3.2.4	Setting up a routed station	65
3.2.5	Setting up a virtual station	66
3.2.6	How to configure a TLS receiver?	66
3.2.7	Customizing stations via XML configuration file	71
3.3	Customizing the JobStarts	75
3.3.1	Customizing via GUI	75
3.3.2	Customizing via XML configuration file <code>rvsJobstart.xml</code>	79
4	Working with rvsEVO	89
4.1	Starting the rvsEVO server	89
4.2	Stopping the rvsEVO server	89
4.3	Displaying messages	90
4.3.1	Displaying Monitor messages	90
4.3.2	Messages from rvsEVO Server	94
4.3.3	Messages from command prompt and rvsEVO clients	94
4.4	Activating a station	95

4.5	Sending a file	95
4.5.1	States for Send Jobs and Receive Jobs	104
4.6	Synchronization of Send Jobs	106
4.7	Active Panel	111
4.8	Listing of all receive and send jobs	112
4.9	Deleting or releasing EERPs	119
4.10	Create an Info File for an External JobID	120
4.11	Archiving the entries of processed send or receive jobs in the revision log	126
4.12	Delivering a certificate	126
4.13	Requesting a Certificate	127
4.14	Replacing a certificate	128
4.15	Display a certificate list	129
4.16	Open the keystore file	129
4.17	Import a CRL	129
4.18	Import a TSL	130
4.19	Terminate a Session	131
4.20	Create and send a journal	131
4.21	Doing character set conversion	132
4.22	Command line tools for internal use	132
5	Backing Up and Recovering rvsEVO Data	133
5.1	Backup	133
5.1.1	What is backed up?	134
5.1.2	Redo Log	134
5.2	Recovering the rvsEVO data	135
6	Encrypted transmission with rvsEVO	137
6.1	Introduction: basics	137
6.2	System requirements	137
6.3	Principle and sequence of rvsEVO encryption	138
6.4	How do I create an own key pair?	140
6.5	How to import and export a certificate	141
6.6	How to import and export ComSecure public keys	142
6.7	How to create a CSR (Certificate Signing Request) and import the CA certificates 142	
6.8	How to receive files without decryption	144
6.9	How to delete a key entry	145
7	rvs® OFTP Proxy	147
7.1	Basics	147
7.2	rvs® OFTP Proxy Architecture	147
7.3	Configure Bastion Instance in rvsEVO	149
8	File Service Module	151
8.1	Basics	151
8.2	Architecture of File Service Module	151
8.3	Setting up of a Neighbourstation with File Service Module	152
9	Remote GUI	155

9.1	Starting the GUI from the remote computer	155
9.2	Features of the remote GUI	156
10	User Management	159
10.1	User Administration	159
10.1.1	Add user	159
10.1.2	Delete a user	160
10.1.3	Edit a user	160
11	PKI Bindung	161
11.1	Introduction	161
11.2	Configuration	161
11.2.1	Configuration of stations	161
11.2.2	PKI configuration file	162
12	rvsEVO Database	167
12.1	Derby	167
12.2	Oracle	167
12.3	How to Drop and Create the Database Tables?	169
12.4	How to view job data from a database?	170
13	rvsEVO Central Administration	171
13.1	Introduction	171
13.2	Command Tools of the Central Administration	172
13.3	How to work with the central administration features?	174
13.3.1	How to exchange a license key file?	175
13.3.2	How to change a station parameter?	176
13.3.3	How to make an update of rvsEVO?	178
14	rvsbat Batch Interface	181
14.1	Starting rvsbat	181
14.2	Create a Send Job with SEND Command.	182
14.3	Managing Jobstarts via rvsbat	185
14.3.1	The RESENTR command	185
14.3.2	The SENDJOB Command	186
14.3.3	The FAILURE Command	187
14.3.4	Jobstart Parameters	187
15	Appendix	191
15.1	ODETTE Protocol	191
Index	1

Change History

The following changes of User Manual were made:

Version 5.04

- New parameters for SEND command (`rvsbat`): INITTIME, TSTAMP, ALG, COMPRESSION, ENCRYPTION, FILEDESC, SFS, SIGN, SIGNRESP, XID
- additional functionality of `rvsbat`: managing jobstarts
- new environment variables for **parameterHandling=ENV**: RVS_ERROR_TEXT and RVS_ERROR_ID
- new start/stop scripts for installations on UNIX systems
- default settings for creating new transmissions
- revision files are configurable via MaxRevisionLogSize and MaxRevisionLogCount parameters
- installation of rvsEVO with default values
- stationlist as table configurable
- new chapter about setting up a virtual station
- saving Monitor Messages in Oracle database
- key management reachable via tab **Key**
- new chapter about the functionality connection pooling
- new depiction command tools: `rvsbat`, `restartJob`, `convertFile`, `getCertificateList`, `sendJournal`, `startKeyMgn`
- description of working with CA certificates
- description of batch files in `$RVS_HOME\bin\jobstart` directory
- minor updates

Version 5.03

- Update of chapter „Customizing the JobStarts“
JobStart configuration in case of failed jobs.
new JobStart parameters
- update of chapter „Displaying Monitor Messages“
configuration of display of Log Messages
after new start of the GUI: displaying of the last 25 Log Messages
- new chapter about license key details
- minor updates

Version 5.02

- Update of installation chapter:

- selection between Derby database and Oracle database with Enterprise Edition
- selection between Standard- and Client-Installation
- setting up of a neighbour station after installation
- new chapter about User Management
- new chapter about Remote GUI
- new chapter about receipt of files without decryption
- new components for `createSendJob` and `convertAndSend`
- new command tools: `getJobInfoList`, `deliverCertificate`, `requestCertificate`, `replaceCertificate`, `importCRL`, `importTSL`, `terminateSession`, `updateStationList`, `holdJob`, `releaseJob`, `deleteJob`
- Observer: using original filename in place of VDSN
- restart of send entries which were changed to SP_failed status
- minor updates

1 Introduction

In this chapter you will find a short description of rvs[®] and rvsEVO as well as an explanation of typographic conventions used in the present manual.

1.1 Short description of the system

What is rvs[®]

rvs[®] = Rechner-Verbund-System

The abbreviation rvs[®] stands for the German word Rechner-Verbund-System. The rvs[®] computer communication system is a well established base service for electronic data interchange, EDI.

rvs[®] serves to ensure transmission of electronic data between heterogeneous computer platforms using different network protocols.

To do so, rvs[®] implements a universal network model, which you can configure in each network node.

rvs[®] provides an efficient and reliable transport service for both standardized EDI message types and files of any format or contents. You can receive only such files that are explicitly destined for rvs[®]. This means that rvs[®] does not allow any unauthorized access to remote or to own data files.

The system was originally developed by Volkswagen AG and has been used in the German and European automobile industries for a number of years but also by banks, insurances and industry worldwide.

rvs[®] uses the OFTP protocol.

What rvs[®] is not

rvs[®] is not an online system. It neither supports direct terminal-like access to other sites, nor does it provide a communication pipe from application to application on a data record level. You cannot directly execute transfers in your own application. You rather can place send orders from within you application to rvs[®] which will be handled asynchronously.

rvs[®] is not a job scheduling system.

rvs[®] does not care about the contents of the files it is transporting. It only acts as a transparent transport medium and performs no semantic interpretation of the data it carries.

rvs[®] is not an EDI converter. You can, however, purchase additional components for converting between specific message formats (e.g. VDA, ODETTE, EDIFACT, XML) using rvs[®] as transport service from T-Systems GmbH.

rvs[®] is not a network control or monitoring tool.

What is rvsEVO

rvsEVO rvsEVO is a communication software with a graphical user interface based, like rvs[®], on the OFTP protocol.

- available features The following features are available in rvsEVO 5.04:
- graphical user interface
 - sending files to neighbour or to routed stations
 - receiving files from neighbour or from routed stations
 - receiving files on virtual stations; sending files from virtual stations
 - activating neighbour stations
 - support of the OFTP version 1.3, 1.4 and 2.0
 - viewing information about receive, send, failed and ended jobs
 - displaying monitor messages
 - log files for tracing the monitor activities and for troubleshooting
 - deleting or releasing EERPs (End-to-End-Response) if necessary
 - archiving information about the send or receive jobs in the revision log
 - code conversion with various code conversion tables
 - format conversion
 - defining job filters and actions when sending or receiving files
 - compression and encryption
 - backup and recovery
 - No limit for largness of transmitted files
 - Support of Central Journal functions (for more information see the Central Journal User Manual)
 - Support of SNMP Monitoring (for more information see the rvs[®] SNMP Agent User Manual)
 - Remote access to rvsEVO-GUI (Remote GUI)
 - Oracle database connection
 - PKI connection

rvsEVO is implemented in Java. There exist four options of rvsEVO: rvsEVO Enterprise Edition, rvsEVO Standard Edition, rvsEVO Light Edition, rvsEVO Tiny Edition. Please see chapter 1.2 to 1.5 for description.

rvsEVO uses a batch interface and the file system to communicate with the application. If capable to do so, the linked application can indicate successful processing and have successful dispatch indicated.

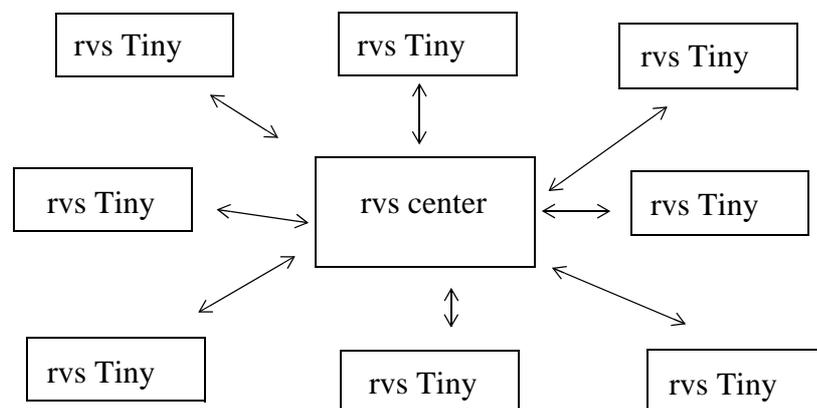
At present, rvsEVO supports the following networks: TCP/IP, TLS, ISDN (only for Windows), XOT, Proxy TCP/IP and Proxy TLS.

For more information on supported platforms please refer to the `$(RVS_HOME)\doku\readme.txt` release notes.

Note: Please read the chapter 1.6 for the explanation of `$(RVS_HOME)`.

1.2 rvsEVO Tiny Edition

The Tiny Edition supports the communication with only one neighbour station: rvs[®] center. The number of partner station is restricted to 1. Other stations can be reached via routing. Therefore rvsEVO Tiny Edition is particularly suitable for star topology.



Features of rvsEVO Tiny Edition (subject to modifications):

- 1 partner station
- 1 neighbour station (rvs[®] center)
- 1 Session
- max. 4 routing partner
- communication component TCP/IP
- encryption and compression
- Central Journal
- SNMP Agent
- code conversion (PC-Mainframe)

- central administration

1.3 rvsEVO Light Edition

Features of rvsEVO Light Edition (subject to modifications):

- max. 5 partner stations (direct or routing partner)
- no limit of parallel connections (sessions)
- routing functionality
- communication component TCP/IP
- encryption and compression
- Derby database embedded
- Central Journal
- SNMP Agent
- code conversion (PC-Mainframe)
- central administration

Additional the following components can be linked:

- communication components ISDN (for Windows) and XOT

1.4 rvsEVO Standard Edition

Features of rvsEVO Standard Edition (subject to modifications):

- no limit of partner stations (direct or routing partner)
- no limit of parallel connections (sessions)
- routing functionality
- communication component TCP/IP
- encryption and compression
- Derby database embedded
- Central Journal
- SNMP Agent
- code conversion (PC-Mainframe)
- central administration
- remote GUI (5 users)
- one virtual station

Additional the following components can be linked:

- communication components ISDN (for Windows) and XOT

- File Service Proxy
- further virtual stations

1.5 rvsEVO Enterprise Edition

rvsEVO Enterprise Edition provides all the basic requirements for secure data transfer with a simultaneous continuous performance even at high loads. rvsEVO Enterprise Edition includes an automatic adjustment of the processing capacity and uses a professional external database.

Features of rvsEVO Enterprise Edition (subject to modifications):

- no limit of partner stations (direct or routing partner)
- no limit of parallel connections (sessions)
- routing functionality
- all communication components
- encryption and compression
- Derby database embedded
- Oracle database connection
- Central Journal
- SNMP Agent
- code conversion (PC-Mainframe)
- central administration
- PKI connection
- File Service Proxy
- remote GUI (10 users)
- one virtual station

Additional the following components can be linked:

- rvs® OFTP Proxy for OFTP2
- further virtual stations

The available features are limited by the license key.

For more Information on rvsEVO Editions please contact your sales partner (Phone: +375 606 19 902; E-Mail: rvs-service@t-systems.com)

1.6 Representation means

This chapter describes the typographic conventions used in this manual and explains the meaning of specially highlighted expressions.

Typographic conventions

- Instructions begin with a bullet.
- Other lists begin with the en dash.

Character styles	<code>Courier</code>	Commands, menu commands, file names, path names, programs, examples, scripts, options, qualifiers, data sets, fields, modes, window names, dialog boxes and statuses
	BOLD and IN CAPITAL LETTERS	Parameters, environment variables, variables
	“Inverted commas”	Links to other manuals, sections and chapters, literature
	Bold	Important terms, names of operating systems, proper names, buttons, function keys.

Directories

\$RVS_HOME As user directories are found on different locations for the different operating systems we use the variable **\$RVS_HOME** in this manual. Default values are:

- `C:\Programs\rvsEVO` for **Windows XP** and **Windows 2000**

Substitute the variable with your correct path.

1.7 Target group

This manual is meant for regular users of rvsEVO as well as administrators. It provides an overview of the basic rvsEVO functions.

Skills The following skills are required to be able to use rvsEVO:

- good knowledge of the current operating system
- knowledge of the communications techniques in use
TCP/IP, TLS, ISDN, XOT, Proxy TCP/IP or Proxy TLS.

Before starting to work with rvsEVO it is advisable to have read this book.

2 Installation

The present chapter describes the system requirements as well as the rvsEVO installation procedure.

2.1 System requirements

To successfully operate rvsEVO 5.2 you need the following software:

- Software
- Operating system: Windows XP, Windows 2000, Windows 2003, Windows 7, Windows Server 2008, UNIX (AIX, Solaris/SunOS, HP-UX, Linux) or OpenVMS.
 - Java runtime environment (JRE 1.5._XX or Java Software Development Kit 1.5_XX).

Please make sure a Java runtime environment is present on your system prior to installing rvsEVO. The software is freely available for download from <http://java.sun.com>.

Hint: on Windows and Linux-Systems Java runtime environment can be installed with rvsEVO installer.

- Communication line based on ISDN (only for Windows), TCP/IP, TLS, XOT, Proxy TCP/IP and Proxy TLS.

Initially, you need at least 200 MB free space on your hard disk. Depending on the amount of usage, the retention period for old entries, and the time between database cleanups, the space requirement may be considerably larger.

You can download rvsEVO from the following web page: <https://servicenet.t-systems.de/tsi/de/267072/Startseite/Business-Integration/rvs>
If that is not possible for you, please contact your sales partner (telephone: +49 30 3497-1165; Fax: +49 (0) 1805 33 44 90 46 35; Email: rvs-sales@t-systems.com). We can send the software also on DVD to you.

2.2 Obtaining a license

You need a license key to work with rvsEVO.

- rvs[®] after-sales service
- Please contact the rvs[®] Service Support Center to receive a license key.
phone from Germany: 0800 664 77 45
phone from other countries: +375 606 19 902
email: rvs-service@t-systems.com

To receive a license key:

- Please send the hostname and the **Odette** ID to the rvs[®] Service Support Center. You get the hostname as typing `hostname` in the command prompt window (`Run -> cmd`). You can request the Odette

ID with the VAD (<http://www.vda.de/de/verband/fachabteilungen/logistik/infos/odette-id/index.html>)

- You will be sent your licensekey file by email.
- Save the license key file in the `$RVS_HOME\conf\` directory as `license.properties`.

Note: For more information about the ODETTE-ID see chapter 3.2.2 "Configuring a local station".

Please read the chapter 1.6 for the explanation of `$RVS_HOME`.

Testlicense For receiving a testlicense, please contact the rvs[®] Service Support Center too. In this case you must not inform the rvs[®] Service Support Center about your hostname and the Odette ID.

2.3 Network Requirements

ISDN connection **Important Note:** ISDN is only for Windows platform available. For Unix platforms XOT should be used.

If you want to use rvsEVO to exchange data via the **ISDN network** you need the following equipment:

- ISDN type telephone connection with s_0 bus having at least two data channels (B-channel) and one control channel (D-channel)
- ISDN adapter
- CAPI 2.0 driver software for operation of the ISDN card under Windows XP.

This is how you install the ISDN interface:

- Install the ISDN card into your computer and attach the ISDN card to the ISDN connection.
- Make sure that the ISDN card works properly.

Note: Many card manufacturers supply suitable software for a self-test, such as a call from one data channel to another. Restart your computer after installation and the test in order to make the CAPI 2.0 driver available for other applications.

External ISDN router If you use an external ISDN router with remote CAPI Interface (e.g. BinTec Brick), you do not need an internal ISDN card. In this configuration, several applications can share the same Brick router. The Brick router supports the "Remote CAPI" interface. This means that every computer in your LAN uses the Brick router as if it were a local ISDN card in the computer.

Note: T-Systems has successfully tested the following ISDN devices for use with rvsEVO:

Device	Manufacturer	Remarks
EICON DIVA Server BRI-2M	Eicon, http://www.eicon.de	
EICON DIVA Pro 2.0	Eicon, http://www.eicon.de	
EICON DIVA 2.0	Eicon, http://www.eicon.de	
Longshine LCS-8051A	Longshine, http://www.longshine.de	
BIANCA/BRICK-XS, -XL, -XM, X4100, X4300, R1200, R4100, R4300	Funkwerk EC, http://www.funkwerk-ec.com	

rvs[®] Service Support Center will provide you with the current list of tested devices:

phone from Germany: 0800 664 77 45
 phone from other countries: +375 606 19 902
 email: rvs-service@t-systems.com

XOT

XOT connection To use the XOT functionality in rvsEVO you need an IP connection to the XOT-capable router e.g. CISCO 801, CISCO 2600 or BINTEC X4300, R1200, R4100 and R4300. On demand we shall send you a separate document with examples how to configure CISCO or BinTec router for XOT.

2.4 Fresh installation of rvsEVO

Installation steps This chapter describes the installation of rvsEVO. Please read the chapter 2.1 „System requirements“ before installing.

First we describe how to perform installation on Windows systems. Then we briefly cover installation on UNIX systems because installation is identical on both operating systems.

Hint: rvsEVO can be installed without input of any data. You only have to press the return button. In this case the configuration will done with the default values. Exception: this procedure cannot be executed in german language on console.

Installation on Windows Systems

- Start Windows and log in as a Windows user with administrator rights.

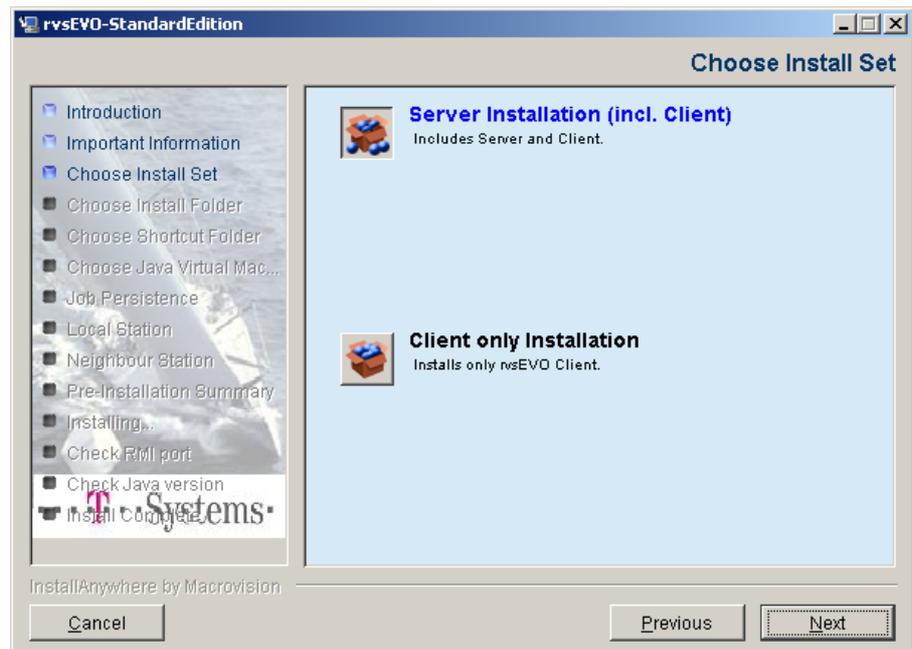
- Start the `rvsEVO_X.X_setup.exe` installer (where X.X indicates the rvsEVO version number) by double-clicking or using the Windows command: `Start -> Run`.
- Choose the installation language (German, English) in the first dialog. Click **<OK>** to go to the next installation step.



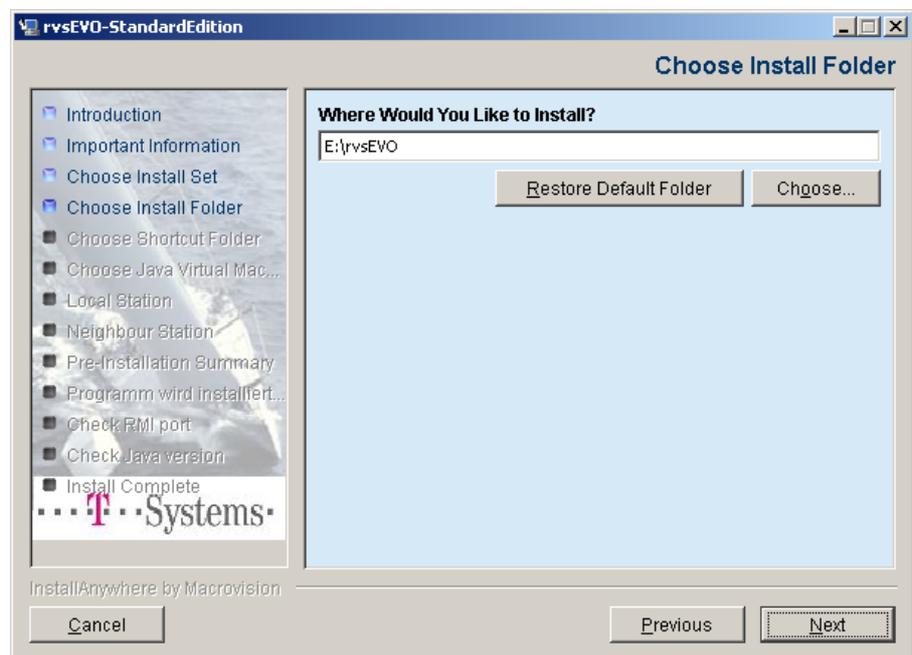
- With the next two dialogs you get information on installation and the version of rvsEVO.
- If there is already an rvsEVO installation on your machine, you have to decide whether you would like to install a new version or to update your current rvsEVO.



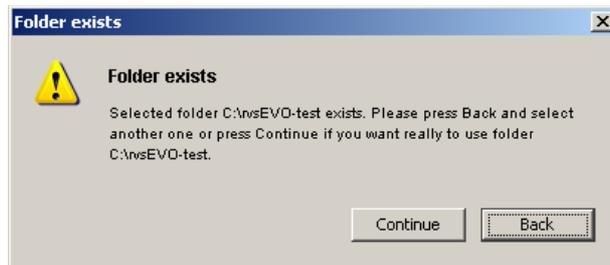
- If you decide for installing a new version you will be asked whether you wish to delete previous rvsEVO.
- Next you have to choose between a „Server“ installation and a „Client“ installation (see chapter 9 "Remote GUI").



- In the next dialog you can indicate the rvsEVO destination directory. This directory may not contain another rvsEVO installation.

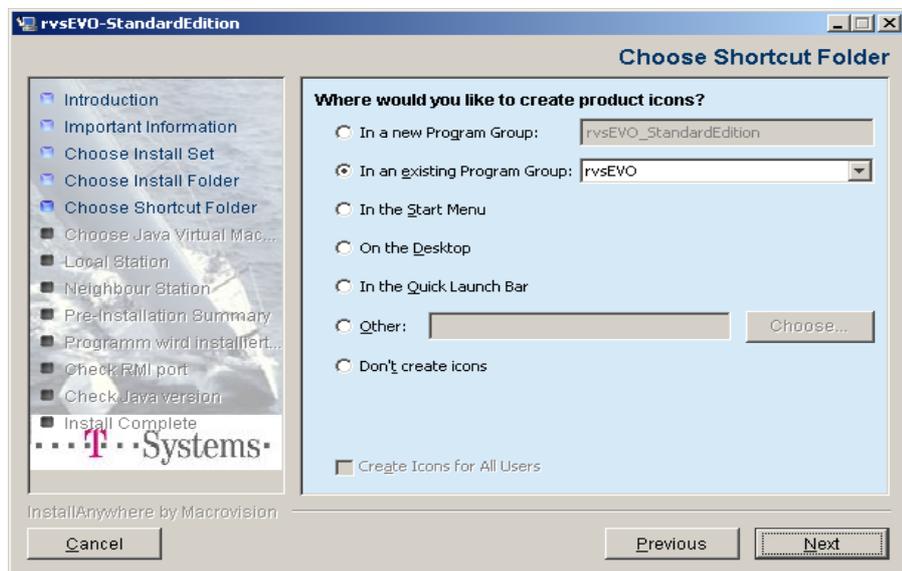


If you choose an existing directory the following warning appears:



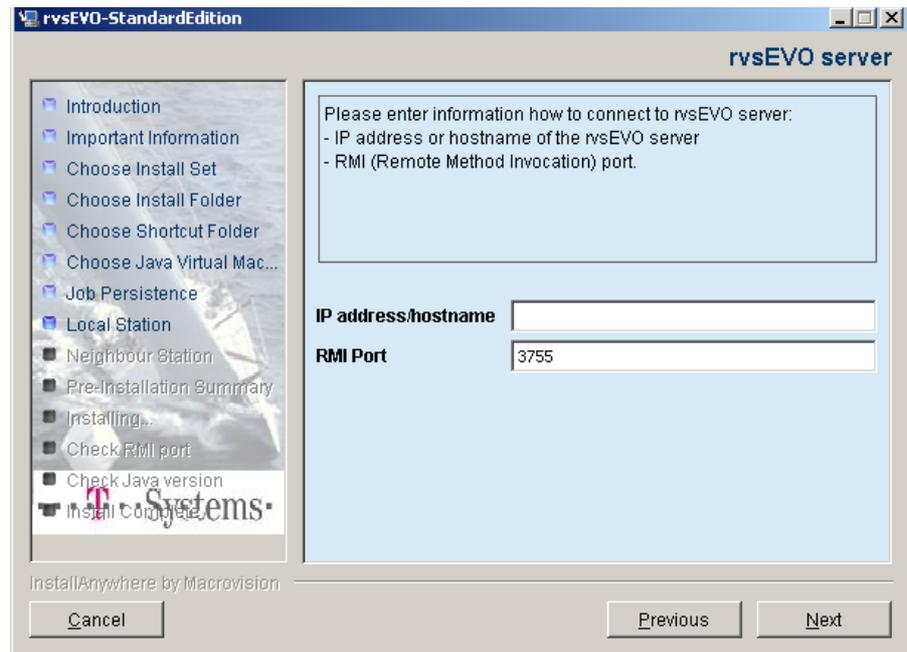
Press the **Continue** button for going on.

- Define the program group to install rvsEVO icons in the following dialog.



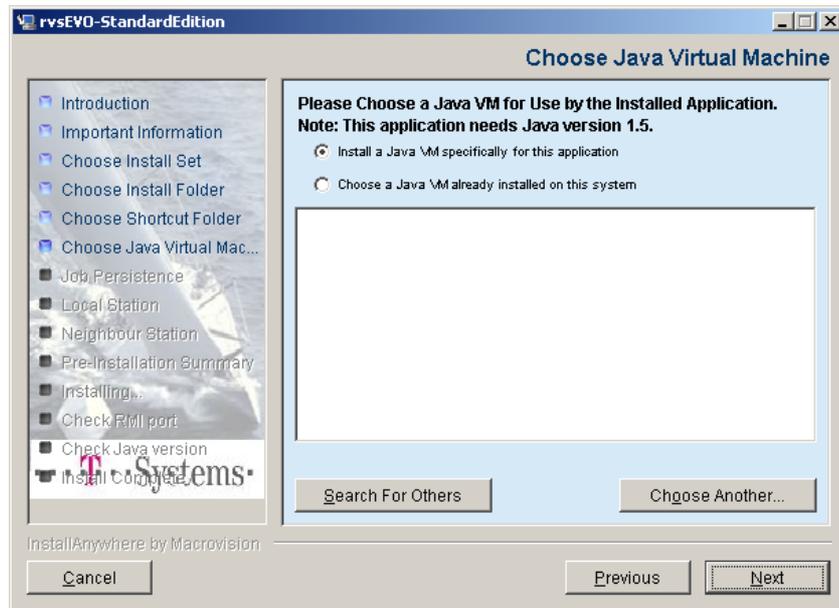
- Client Installation
- The next two dialogs relate to the „Client“ installation only. Please indicate the IP address (or hostname) and the RMI port (Standard:

3755) of the rvsEVO server.



- The last screen display informs you of the successful installation of rvsEVO Client.
- Server Installation
- The following dialogs describe the „Standard“ installation only. Define the Java runtime environment for rvsEVO operation in the next dialog. You can choose between the Java Virtual Machine, which will be installed by the installation routine especially for rvsEVO or Java Virtual Machine, which is already installed on your system. The installer searches for installed components and proposes the versions found in a dialog. rvsEVO has been released for use from version

1.5._X onwards.



Hint: If you choose to install the Java Virtual Machine especially for rvsEVO, you have to answer the next question **Install JCE Files** with **Yes** (We recommend this).

Due to import control restrictions of some countries, the JCE jurisdiction policy files shipped with the Java 2 SDK, v 1.5 allow "strong" but limited cryptography to be used. rvsEVO uses an extension of the Java runtime environment - JCE (Java Cryptography Extension) of Sun Microsystems, Inc - to implement the cryptographic features. It is necessary to install these extension of the Java runtime environment to use unlimited cryptographic strengths. This is available for most countries.

Hint: If you do not allow the installation routine to install this component (i.e. you answer the question **Install JCE Files** with **No**), or if you choose to use a JVM already existing on your system, you have to install the JCE files belated.

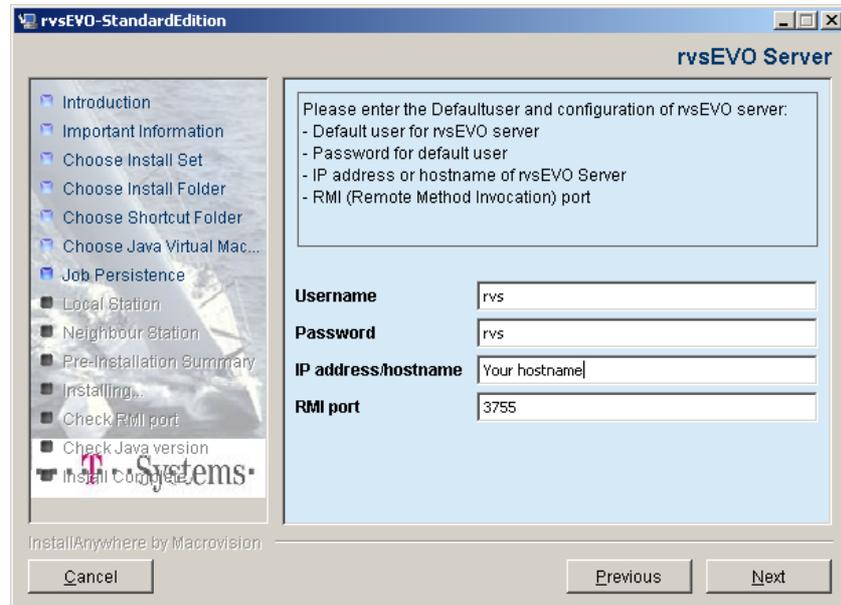
If you try to transfer encrypted files and you did not install JCE, you'll get the following message: „invalid key length“.

For more information about JCE files read the `$RVS_HOME/docu/readme.txt` file.

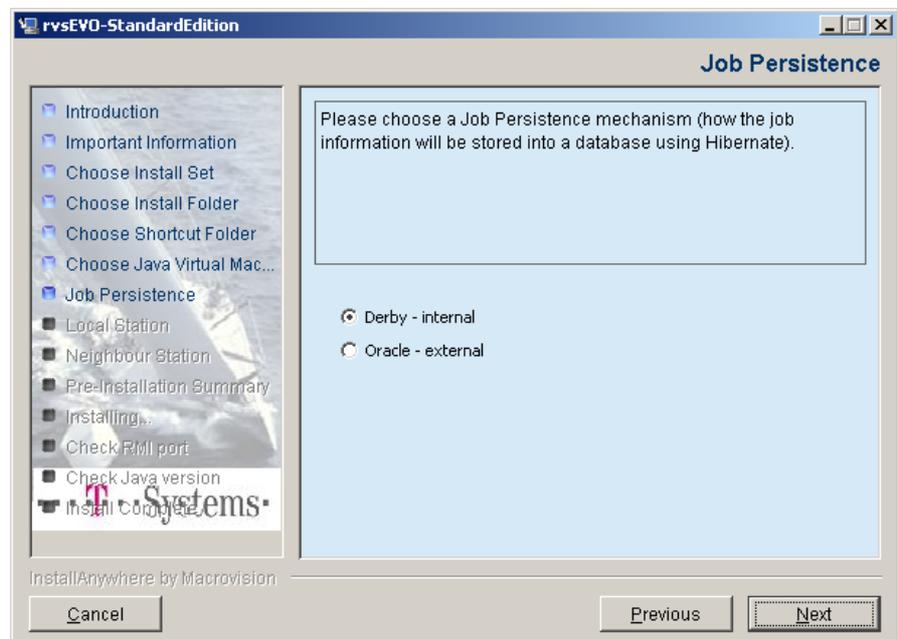
- Subsequent please define the username and the password of the default user with administrator rights as well as the IP address or hostname (default: hostname of your machine) and the RMI port (standard: 3755).

Hint: Due to the special role of the default user concerning the local Client Server communication you cannot change his data in the User

Management.



- With the installation of rvsEVO Enterprise Edition in the next dialog you can choose in which database your job data should be stored. The derby database is the part of rvsEVO installation and will be installed in the directory `$RVS_HOME/db`.



- The following three dialogs are related to Oracle database only. You can ignore them if you decided for a Derby database yourself. Please set your database connection parameters:

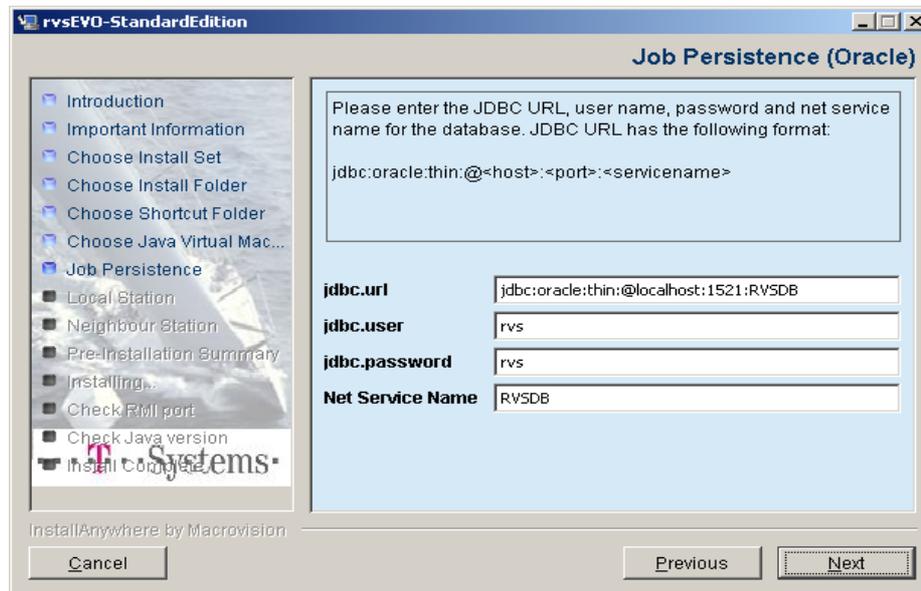
- `jdbc.url`: has the following syntax:
`jdbc:oracle:thin:@<server>:<port>/<service_name>`
`server` is the name or IP address of the machine where Oracle server is installed.

Default port for Oracle is 1521.

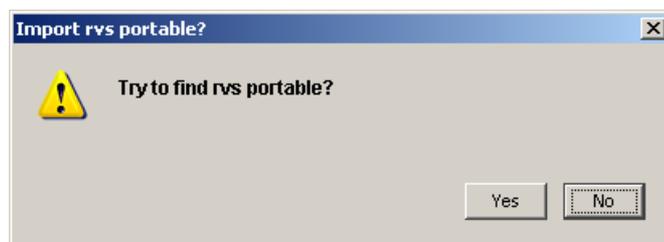
`service_name` is Oracle service name.

- `jdbc.user` is the user setting up on Oracle database
- `jdbc.password` is his password
- `Net Service Name`: name of the Oracle network service

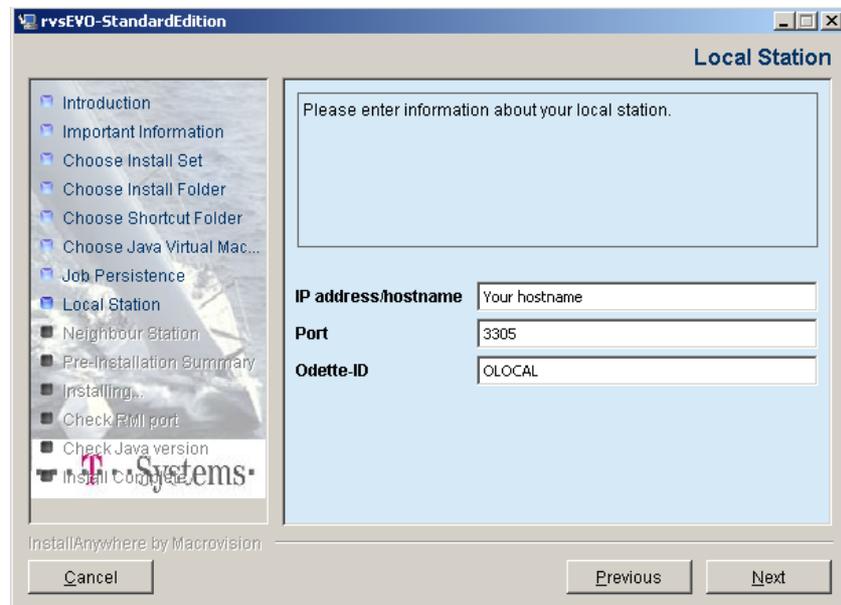
For more information see chapter 12.2 "Oracle".



- Subsequent you where asked whether you wish to search through your system for an existing rvs® portable installation. Then you can import the settings from rvs® portable. Please see chapter 2.10 "Migration from rvs® portable to rvsEVO" in case of migration.



- If you will not import the settings, the following dialog deals with the settings of the local station. Please read the chapter 3.2 for more information about setting station parameter (e.g. how to obtain the ODETTE ID). The neighbour station will be configured after installation via GUI. (Please read 3.2.3, „Setting up of a neighbour station“ page 58)



- In the next dialog you are given a brief overview of selections you have made (installation directory, link directory). The required and the currently available disk space is also indicated. Press the **Install** button to start installation and to copy the installation files into the directories you specified.
- The last dialogs informs you of the successful installation of rvsEVO.

UNIX Systems **Installation on UNIX Systems**

As mentioned earlier in this chapter, installation on UNIX systems runs analog to an installation on Windows systems. The installation file is named `rvsEVO_X.X.X_setup.bin` and can be started as a window-based installation under the X-Server.

The installation on UNIX systems can be done in different modes: `awt`, `swing`, `console` or `silent`.

For graphical modes (`awt` and `swing`) the UNIX environment variable **DISPLAY** for the X-Server should be set.

Example:

```
export DISPLAY=<IP address of X Server>:0.0
```

Note: Make sure to call the installation file as a shell script when you perform installation. Default mode is the console mode.

Example (command line):

```
sh ./rvsEVO_501_00_SE_setup.bin -i swing
```

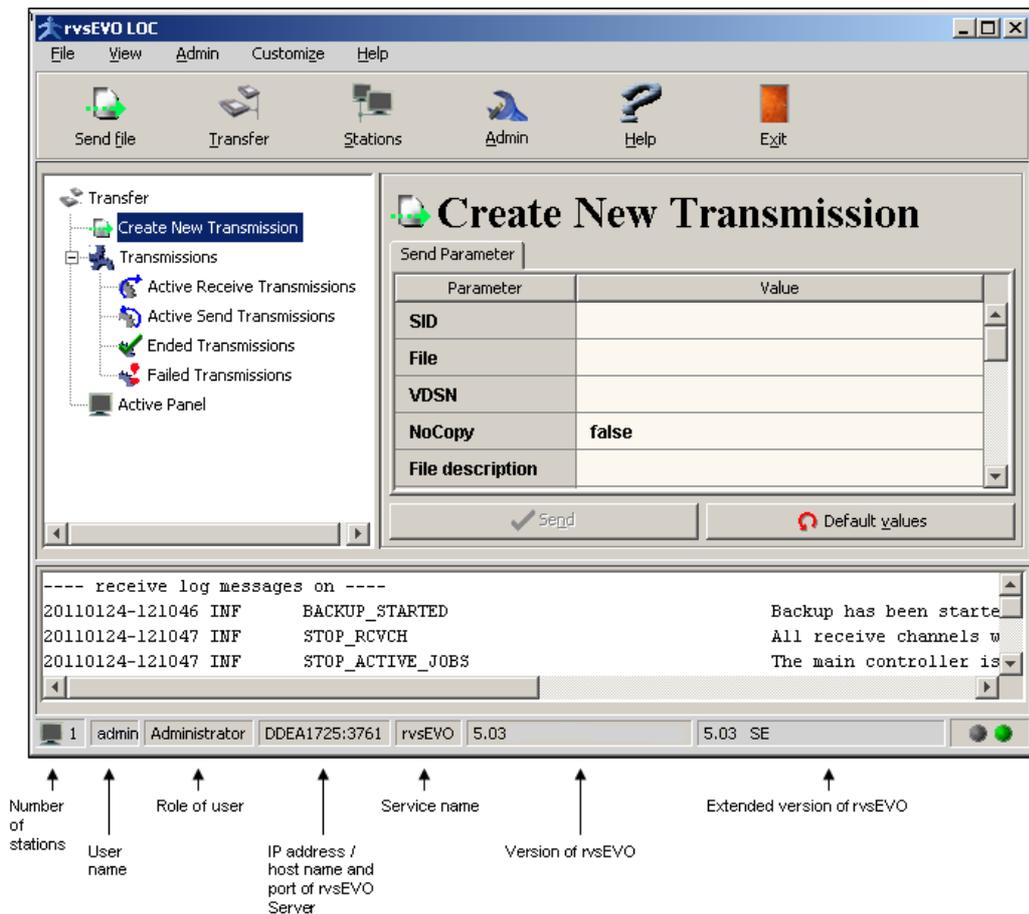
The installation prompts are identical in both modes (see section **Installation on Windows systems**).

There are minor differences for particular UNIX platforms mentioned in the release notes for the respective version (\$RVS_HOME\doku\readme.txt document).

Please notice the hint about JCE files in chapter „Installation on Windows Systems“.The JCE files have to be installed for encrypted data exchange.

GUI 2.5 Graphical User Interface (brief description):

By using the GUI you need not much practise for working with rvsEVO. The picture below indicates the GUI in the `Send file` window:



In the title bar the name of the program and the stationID of the local user is shown. Thereunder you can find the menu bar and the function bar.

With a click on one of the symbols (Send file, Transfer, Stations, Admin, Help and Exit) you can open the special menu item or terminate the GUI.

The middle part is composed in a navigation range and a work range.

At the bottom of the GUI the log messages were displayed. Thereunder the status line is arranged and gives information of rvsEVO installation: number of stations, user name, role of user, IP address / host name and port of rvsEVO server, service name, version of rvsEVO and extended version of rvsEVO.

The rvsEVO version can also be shown via the
`$RVS_HOME\tools\rvsver.bat` command tool.

2.6 How to start rvsEVO?

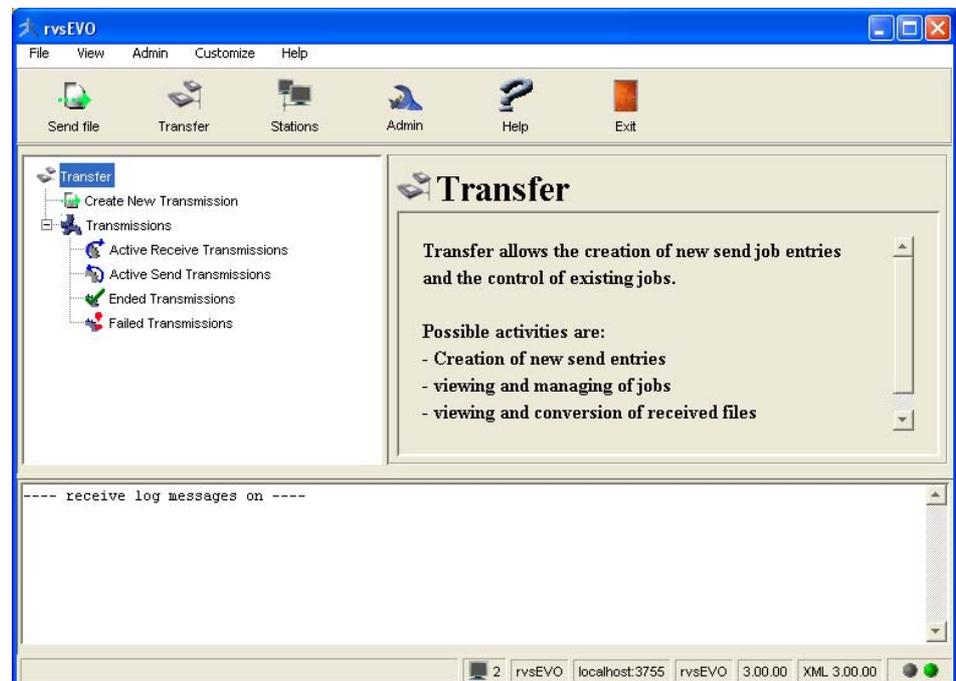
Windows: Start rvsEVO by choosing Start -> Programs -> rvsEVO -> startGUI (or the Start menu command you specified during installation) in the Start menu.

Unix: Start rvsEVO by starting the shell script
`$RVS_HOME\bin\startGUI.sh`.

This menu item (Program) first starts the user interface, which then starts the rvsEVO server.

Successful start:

Start A successful start is indicated as follows:



`startGUI` Use the `startGUI` program to start the user interface.

Syntax:

```
startGUI [-local -console -reset
-help -?]
```

Mögliche Parameter:

<code>-local</code>	local start of rvsEVO
<code>-console</code>	starts the GUI with console output
<code>-reset</code>	resets the GUI settings during startup
<code>-help</code>	Requests help information
<code>-?</code>	Requestshelp information

To start the rvsEVO server, use the `startService` or `startServer` script on Windows systems and `startServer` on UNIX systems. These scripts are located in the `$RVS_HOME/bin/` directory (on Windows systems as batch files, on UNIX systems as shell scripts).

Note:

Windows: By default, rvsEVO starts as a service on Windows systems. You can configure this in the `$RVS_HOME/conf/rvsConfig.xml` file using the `RvsStartScript` parameter (see also chapter 3.1).

As an alternative you can also start rvsEVO at the command prompt by defining the `$RVS_HOME/bin/startServer.bat` program as the `RvsStartScript` program.

Unix: By default, the `$RVS_HOME/bin/startServer.sh` shell script is used as the `RvsStartScript` on Unix systems.

The `$RVS_HOME\tools\scripts\Linux` and `$RVS_HOME\tools\scripts\Solaris` directories contain scripts that enable the automatic start and stop of rvsEVO together with the operation system. In the same directories you can find text files, in which the script installation steps are described.

2.7 How to stop rvsEVO?

Windows: Stop rvsEVO by choosing `Start -> Programs -> rvsEVO -> stop Server` (or the `Start` menu command you specified during installation) in the `Start` menu.

As an alternative you can also stop rvsEVO at the command prompt by the `$RVS_HOME/bin/stopServer.bat` program.

Unix: Stop rvsEVO by starting the shell script
`$RVS_HOME\bin\stopServer.sh.`

2.8 rvsEVO as Windows service

Per default is rvsEVO installed as a Windows service.

Note: The term Service means a program that can be started from the operating system and works in the background.

This is possible starting the batch script `rvsservice.bat` out of the directory `$RVS_HOME\bin`.

Usage:

```
rvsservice <parameters> [options]
```

The possible parameters are:

-c	starts rvsEVO on console
-i	installs rvsEVO as service
-r	removes rvsEVO as service
-s	starts rvsEVO as service
-h	usage

Example:

```
C:\Programme\rvsEVO\bin>rvsservice -i
-----
-- RUSTINY SERVICE LAUNCHER --
-----
rvs Server installed.
```

With the command `rvsservice -c rvsEVO` Server was started on the command line (console).

With the command `rvsservice -i rvsEVO` was installed as system (Windows) service.

Now you can find rvsEVO as a service in the list of system services (Start -> Control Panel -> Administrative Tools -> Services). If you want to start rvsEVO every time the system starts, you can set the startup type to `Automatic` by choosing `Automatic` from the combo box in the **Startup type** area.

Example (Windows XP german version):

2.9 rvsEVO update installation

An rvsEVO update installation is almost identical to a normal installation (see chapter 2.4 "Fresh installation of rvsEVO").

precondition The system variable **RVS_HOME** is to be set for the logged on user (the value of **RVS_HOME** is the installation directory of rvsEVO). Set **RVS_HOME** by choosing Start -> Control Panel -> System -> Advanced -> Environment Variables

Note: During the update installation a new \$RVS_HOME/conf/rvs-system.properties.new file is added. Please take over the configurations of your old rvs-system.properties file to the new one and rename rvs-system.properties.new in rvs-system.properties.

Please read chapter 13 to learn how you can use the Central Administration to update other rvsEVO installations.

2.10 Migration from rvs® portable to rvsEVO

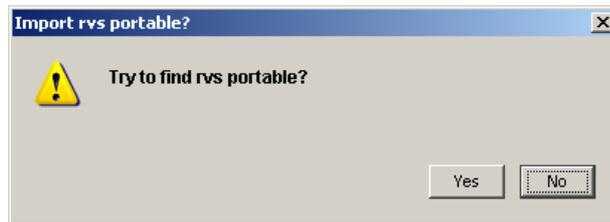
Requirements For migration of settings from rvs® portable you have to consider the following requirements:

- an executable rvs® portable must be installed on your system
- migration should only be started if the absence of communication has been assured (no file reception/transmission and no encryption/compression)
- the system variable **RVSENV** is to be set for logged on user. Set **RVSENV** by choosing Start -> Control Panel -> System -> Advanced -> Environment Variables

The following steps need to be done for migration from rvs® portable to rvsEVO:

- Install rvsEVO on the same machine where rvsXP / rvsX is installed and follow below instructions:

If you are asked whether you wish to search through your system for an existing rvs® portable installation, press the **Yes** button.



If you are asked whether you wish to import the settings from rvs® portable, press the **Yes** button.



- Only with encrypted transmission: import your own key pair with the program `importComSecureKeyPair` (see below).
- Only with encrypted transmission: import the public keys of your partners (see chapter 6.6 "How to import and export ComSecure public keys").
- Create the users which were defined in rvs® portable. They will not be created automatically.

`importComSecure
KeyPair`

Use the program

`$RVS_HOME\tools\csi\importComSecureKeyPair` to import a ComSecure key pair from rvsXP / rvsX.

Syntax:

```
importComSecureKeyPair -k <keystore1> [-k <keystore2>]
-pri <private key> -pub <public key> -x509 <x509
filename> -s <sid> [-help] [-?]>
```

Required parameters:

-k <keystore>	Name and path of rvsEVO keystore file, where the key pair is to be saved (it is possible to indicate several files).
-pri <private key>	Name and path of private ComSecure key.
-pub <public key>	Name and path of public ComSecure key (see also parameter -x509).
-s <sid>	StationID of the station, the key pair belongs to.

-x509 <x509 filename>	Name and path of public key (instead of -pub , if the certificate is in X.509 format).
------------------------------------	---

Optional parameters:

-help	Requests help information.
-?	Requests help information.

Important: If the public key is not existent in format X.509 you have to insert the additional information in the file

`$RVS_HOME\tools\csi\certificate-properties.xml`.

Sample file (certificate-properties.xml):

```
- <!-- common-name -->
  <entry key="subject.cn">rvsEVO-comsecure-j</entry>
- <!-- organisastion-unit -->
  <entry key="subject.ou">rvs</entry>
- <!-- organisastion -->
  <entry key="subject.o">T-Systems International GmbH</
entry>
- <!-- locality -->
  <entry key="subject.l">Berlin</entry>
- <!-- state -->
  <entry key="subject.st">Berlin</entry>
- <!-- country-code -->
  <entry key="subject.c">de</entry>
- <!-- email -->
  <entry key="subject.email">rvs-support@t-systems.com</
entry>
```

The following values are set in above example:

- common name = rvsEVO-comsecure-j
- organisation unit = rvs
- organisation = T-Systems International GmbH
- locality = Berlin
- state = Berlin
- contry code = de
- email = rvs-support@r-systems.com

The other elements of `certificate-properties.xml` must not be edited.

Note: Use the following command tools for importing the settings into an existing rvsEVO:

`$RVS_HOME\tools\portable2jobstart.bat` and
`$RVS_HOME\tools\portable2stationlist.bat`

2.11 Displaying license key information

To display information about the license key open the Administration window and select the item `License Key` in the Administration tree on the left hand side.



The following table describes the parameters:

Parameter	Beschreibung
Company	Name of the company or „Testinstallation“
Components	licensed components (see note below the table)
Key	expiration date of the key (format yyddd) and hashkey
KeyBackup	see Key
Neighbours	max. number of direct neighbour stations
Partners	max. number of partner stations (neighbour and routing stations)
ProxyStations	number of licensed rvs® OFTP Proxy stations
Routings	max. number of routing partner

Sessions	max. number of parallel connections
Users	number of users
Virtuals	number of licensed virtual stations

Components **Note:** The following components are available:

- **F:** File transfer
- **A:** server of central administration
- **I:** client of central administration
- **D:** external database
- **E:** EDI converter
- **S:** encryption
- **C:** compression
- **J:** Central Journal
- **M:** SNMP Traps
- **W:** administrative GUI over JMX (only rvsProxy)
- **Y:** File Service Proxy
- **K:** PKI connection
- **O:** certificate validation with OCSP
- **B:** certificate validation with CRL
- **R:** Remote GUI (not for Tiny Edition)
- **T:** Testkey

2.12 Uninstall rvsEVO

In order to uninstall rvsEVO, please follow the procedure given below:

Windows: start the program Uninstall rvsEVO_StandardEdition.exe in the directory `$RVS_HOME\UninstallerData` with double-click and follow the dialog.

Unix: use the shell-script `$RVS_HOME/UninstallerData/Uninstall rvsEVO_StandardEdition.sh`.

Uninstall procedure on UNIX-systems can be executed analogous to rvsEVO installation. Following modes are available: `awt`, `swing`, `silent` und `console`. For the graphical modes (`awt` and `swing`), the system environment variable **DISPLAY** must be set for the X server (see example in chapter 2.3). The uninstall routine is executed in the console mode by default. The uninstall queries are same in all the modes.

- Select **<uninstall>** in order to remove rvsEVO from your server.

- In the next dialog, you can decide, whether you would also like to delete the created or changed directories and files after the installation. We do not recommend this, since you would thus delete all the received files and archived file. The program has been uninstalled.
- In the following window, it is displayed, which components have not been deleted. Click on **<done>** in order to end the uninstall program.

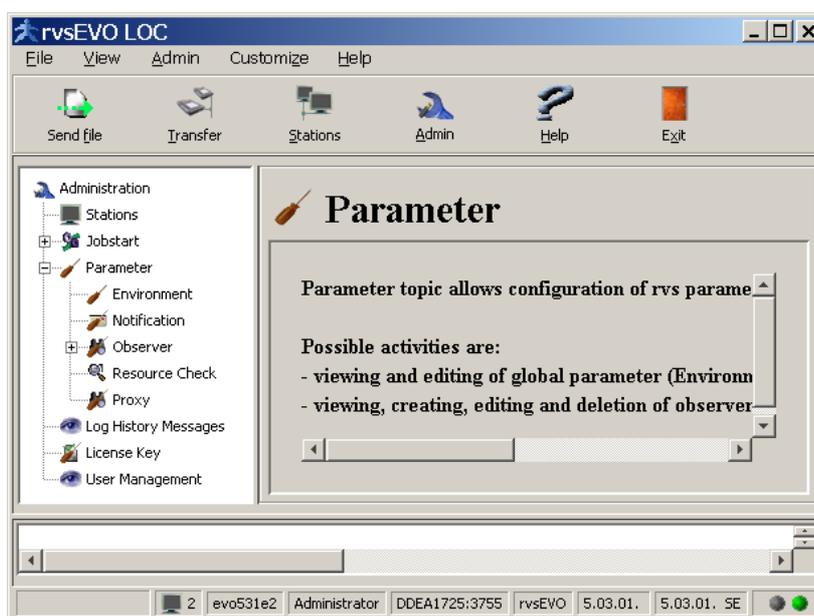
3 Configuration

The present chapter describes how to customize rvsEVO via the GUI or via the XML configuration files.

3.1 Customizing the global rvsEVO parameters

The rvsConfig file contains the rvsEVO parameters which are configurable via

- the graphical user interface GUI (Admin -> Parameter)



- or in the XML configuration file `$RVS_HOME/conf/rvsConfig.xml`

There are five following parameter groups:

- **rvsEVO Environment:** general parameters, which are important for the rvsEVO environment
- **Notification (SNMP):** this group refers to the feature rvs[®]-SNMP-Agent
- **Observer:** feature for automatically scanning of directories to find send entries. It is similar to the feature rvsjs in rvs[®] portable.
- **Resource Check:** error handling relating to resource check and expiring certificates
- **Proxy:** parameters for the Bastion Instance of rvs[®] OFTP Proxy. A table of the parameters you can find in chapter 7.3 "Configure Bastion Instance in rvsEVO".

These parameter groups are to be found:

- in GUI as subentries of parameter item in the Administration tree
- or as separate XML blocks in the file `rvsConfig.xml`.

3.1.1 rvsEVO Environment

In the following table you will find the parameters, which refers to the rvsEVO environment.

ELEMENT	DESCRIPTION
ARCDIR	Archive directory where archive files and backup files are stored. Default: <code>\$RVS_HOME\archive</code> . RevisionLog.xml files contain the entries of processed send or receive jobs (please see chapter 4.11). .jar files contain the backup data and RedoLog files contain any dynamic data from backup time on (see chapter 5).
BackupStartup	You can use this parameter to specify an automatic back-up to be performed each time you start rvsEVO. Possible values: Y (Yes); N (No): The default is Y. See chapter 5.1 for more information.
Browser	Name of browser, e.g. <code>explorer</code>
CentralJournalIn-stance	The rvs [®] destination station that is to receive the Journal files. This station must be present in the rvsEVO station list.
Cleanupdays	Specify days for archiving of completed or failed jobs. The <code>archiveJobs</code> program will save any jobs older than the time specified in this parameter to the <code>RevisionLog.xml</code> file (see chapter 4.11, see also the <code>PersistenceArchive</code> parameter). Default: 7 days Note: You can set only one of cleanup parameters: cleanupdays or cleanuptime .
Cleanupinterval	Time interval in minutes between two archiving cycles. This parameter has no relation to the other two cleanup parameters: cleanupdays or cleanuptime .
Cleanuptime	Specify time in the format HHmmss for archiving of completed and failed jobs. The <code>archiveJobs</code> program will save any jobs older than the time specified in this parameter to the <code>RevisionLog.xml</code> file (see chapter 4.11, see also the <code>PersistenceArchive</code> parameter). Note: You can set only one of cleanup parameters: cleanupdays or cleanuptime .

ELEMENT	DESCRIPTION
ConnSetupFail-WaitTime	Time in milliseconds rvsEVO waits after a connection failed to be established before rvsEVO tries to establish the connection again.
DB	<p>Obsolete. Job Data are now written in a database. Please see the chapter 12 "rvsEVO Database" for more information.</p> <p>In the versions 4.0 and under it the meaning of this parameter is the following:</p> <p>Directory for job administration with the ENDED, FAILED, RCV and SND subdirectories. The RCV and SND subdirectories are used to store temporary, not fully processed jobs. The FAILED directory holds the failed, the ENDED directory the completed jobs. These directories are also visible in the GUI (Transfer window, Transmissions).</p>
Description	Free text
EngdatConfigFile	Configuration file for the modul ENGDAT. Not in use at moment.
FirstLanguage	<p>Language of GUI. Values: de (German), en (English)</p> <p>In addition parameter <code>FirstLanguage</code> in <code>\$RVS_HOME/conf/rvsEvoClient.prefs</code> file must be aligned. The following example shows how to set English as GUI language: <code><entry key="client.FirstLanguage" value="en"/></code></p> <p>Please restart rvsEVO after changing.</p>
HelpFile	Path of help file
HostAllowFile	Configuration file containing DNS names or IP addresses of hosts that may send rvsEVO commands to the rvsEVO server.
HostDenyFile	Configuration file containing DNS names or IP addresses of hosts that may not send rvsEVO commands to the rvsEVO server.
INBOX	Directory where completely received files are stored.
JobstartConfigFile	Configuration file for job start.
JournalFilename-Prefix	The prefix for the Journal file name: default TINY.
LOGDIR	Directory for both log files: <code>startServer.log</code> and <code>rvsservice.log</code> .
LooptestNeighbourSID	ID of the station via which the loop test (transmission of a file to the own local station) is performed. The file is to be transmitted to the own station, and rvsEVO sends this file via a neighbour station back to the local station.

ELEMENT	DESCRIPTION
MailLocalAddress	E-mail address of the originator of warnings. (Please see also parameter Mail and Send e-mail if certificate expires/expired in Chapter 3.1.4 "Resource Check")
MailSMTPHost	Hostname or IP address of the mail server of the originator of warnings. (Please see also parameter Mail and Send e-mail if certificate expires/expired in Chapter 3.1.4 "Resource Check")
ManagementConfigFile	Name of the configuration file for the central administration (see chapter about the central administration in this manual).
MaxMonLogCount	without function
MaxMonLogSize	without function
MaxRevisionLogCount	Number of <code>RevisionLog.xml</code> revision files that can be generated. Refer to chapter 4.11 to find out how to generate a revision file. Default:: 100.
MaxRevisionLogSize	Maximum file size of the <code>RevisionLog.xml</code> revision file in lines (1 line has maximal 10 kbytes). Default: 1.000.000
MaxSessions	Maximum number of simultaneously running receiving processes for TCP/IP communication. Default: 2; maximum number is restricted by system resources.
MonlogStylesheet	without function
OFTPTimeout	Time-out in milliseconds at ODETTE level; default: 30 000, no maximum.
OUTBOX	Temporary directory for files to be sent.
PersistenceArchive	Name of the file with the transfer data (statistic file). Default: <code>\$RVS_HOME/archive/RevisionLog.xml</code> . See chapter 4.11 for more information.
RedoLog	This parameter defines, if a redolog file will be written or not. Possible values: – <code>N</code> (Default): redolog file will not be written – <code>Y</code> Redolog file will be written. See chapter 5.2 for more information.
RMIServiceHost	Host Name in the RMI registry. Default: localhost. RMI (Remote Method Invocation) is a protokoll for the internal process communication in Java
RMIServiceName	Name of the rvsEVO service in the RMI registry. Default: rvsEVO

ELEMENT	DESCRIPTION
RMIServicePort	Port number for the communication in the RMI registry. Default: 3755
RvsStartScript	Path of the script starting rvsEVO; default: \$RVS_HOME/bin/startServer.bat or as Windows service: \$RVS_HOME/bin/startService.bat
SendJournalInterval	Time interval in seconds between sending of two Journal files to the rvs [®] destination station (defined by the CentralJournalInstance parameter). No Journal file will be sent if no value or 0 is specified here.
SessionAliveTimeout	Time in milliseconds to consider a connection active; default 600 000, no maximum.
SessionWaitTime	SessionWaitTime is the time in milliseconds the OFTP session between the sender and the receiver station will be kept alive after an ended transmission for waiting for an other send job; default: 0
StationsConfigFile	Station configuration file: Comprises the configuration parameters for the local station, the neighbour station, and routed stations.
TEMP	Directory for temporary use.
Timestamp	Defines whether or not a file receives a time stamp in its name when it is received. Possible values: N (default) Time stamp is added only if the file name already exists; Y File name always receives the TransmissionID as time stamp.

ELEMENT	DESCRIPTION
Traceltem	<p>Parameter that enables tracing. The following values are possible: O: for Odette level, N : not active (default)</p> <p>Trace output is written to a file which name is composed as follows: <code>StationID</code> and „_“ and <code>SessionID</code> and <code>.trc</code> extension. The file is saved in the <code>\$RVS_HOME/log/trace/odette</code> directory. The output can be restricted to specific stations (in the following example <code>SID1,SID2</code> und <code>SID3</code>). This function is controlled by the properties file <code>\$RVS_HOME/conf/rvs-system.properties</code> with the following definition: <code>rvs_evo.tracing.odette.station_ids=SID1,SID2,SID3</code>. Enable the definition <code>rvs_evo.tracing.odette.enable_network_data=true</code> in order to write the network data into the trace file..</p>
TransmissionFail-WaitTime	Time in milliseconds for a transmission restart after a failure.

3.1.2 Notification (SNMP)

In this table are the parameters, which refer to the feature `rvs® SNMP Agent`.

`rvs® SNMP Agent` is an application that can respond to Network management systems (NMS) queries and send `rvsEVO` status information to NMS.

You can find the installer of `rvs® SNMP Agent` in the `$RVS_HOME/SNMP_Agent` directory. For more information about this feature, please read the `rvs® SNMP Agent User Manual`. After installation of `rvs® SNMP Agent` the manual is stored in `C:\Programme\rvsSNMPAgent\doc` directory by default.

You must edit the following parameters in `rvsEVO` to be able to work with `rvs® SNMP Agent`:

ELEMENT	DESCRIPTION
Active <AgentActive>	<p>This parameter defines whether or not <code>rvs® SNMP Agent</code> is enabled or disabled: Default: Y (Yes). Possible values: Y (Yes) or N (No).</p>

ELEMENT	DESCRIPTION
Interval <Agent-HeartbeatInterval>	This parameter defines the interval (in seconds) at which rvsEVO sends a Heartbeat message to the Agent UDP address (AgentHostname + AgentPort).
IP address <AgentHostname>	Agent computer name (or IP address). Default: localhost.
Port <AgentPort>	Agent IP port. Default: 3744.
Log-Level <AgentLogLevel>	This parameter defines whether or not rvsEVO sends log messages to the Agent. Possible values: 0, 1, 2, 3. 0: no log messages are sent 1: error messages are sent 2: warnings and error messages are sent 3: warnings, information and error messages are sent

3.1.3 Observer

The feature Observer (Admin -> Parameter -> Observer) generates send entries for files, which are in a configured directory. The observer checks in regular time intervals a directory for the specified file names (mask). If such files are found, the send entry will be created. The options for send entry can be configured, too.

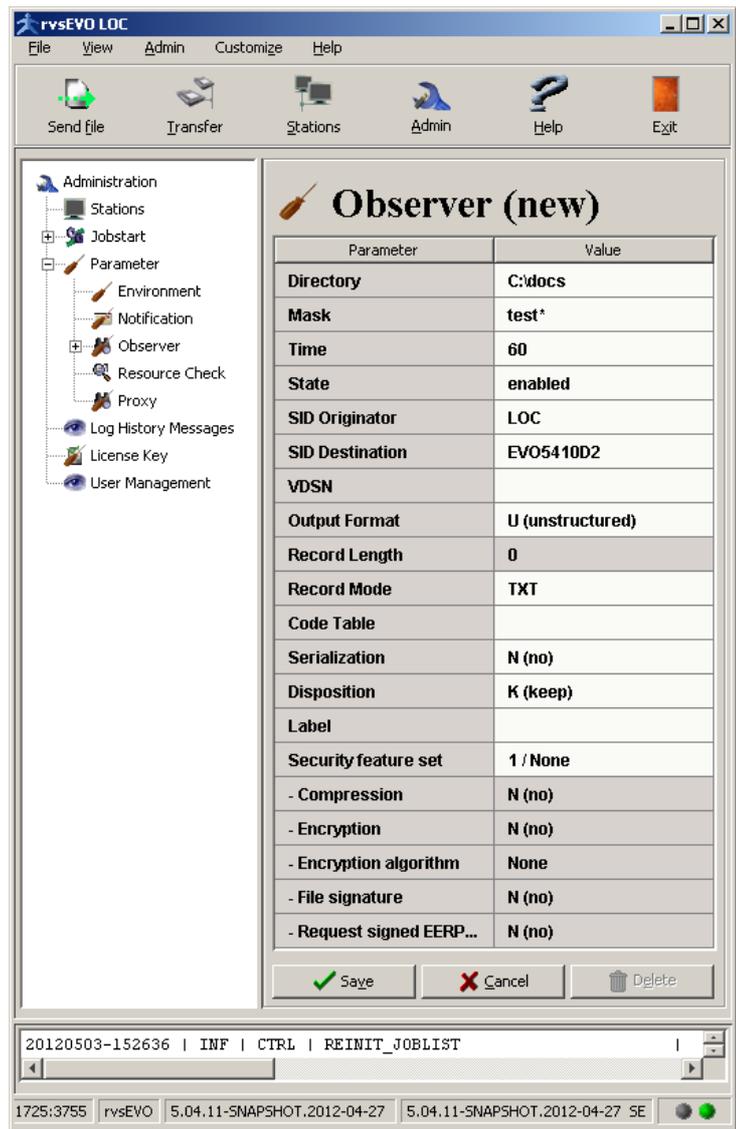
A right click on the Observer symbol opens a context menu with the possibility to add a new observer.

Note: You can configure few observers for scanning few different directories, if necessary.

Example:

In the following example the directory `c:\docs` will be scanned every 60 seconds for sending all files, which file name begins with `test` to the

station EVO5410D2



In the following table are the observer parameters described.

ELEMENT	DESCRIPTION
Directory	directory, which should be scanned.
Mask	The regular expression, which should be applied. Example: MKL* means, that the observer checks for files beginning with MKL. If it finds them, it will create a send entry.
State	This parameter defines, if the feature observer active is or not. Possible values: ENABLED and DISABLED.
Time	Time interval between two checks. Default: 30 seconds.

All other parameters from an Observer panel are for the sending a file. These are: SID of Originator, SID of Destination, VDSN, Output Format, Record Length, Record Mode, Code Table, Serialization, Disposition, Label, Security feature set, Compression, Encryption, Encryption algorithm, File signature and Request signed EERP/NERP.

To which values these parameters can be set, please read in the chapter 4.5. The program `createSendJob` or GUI are using the same parameters, when sending a file.

Note: You can also use the original filename as file description. This function is controlled by the properties file `$RVS_HOME/conf/rvs-system.properties` with the definition `observer.sfiddesc.usefilename=true`.

3.1.4 Resource Check

This parameter group refers to resource check of rvsEVO directories and checking the expiration dates of certificates. This functionality allows the operator to react early enough to lack of free disk space and expiring certificates.

Resource Check All active directories of rvsEVO will be checked (these are the directories, which are configured in the file `$RVS_HOME/conf/rvsConfig.xml` for the following variables: `<DB>`, `<TEMP>`, `<INPUT>`, `<OUTPUT>`, `<ARCDIR>` and `<LOGDIR>`.

In case of the low disk space you can see in the Monitor Log a message, that informs you in which rvsEVO directories is not enough disk space. At the same time an E-mail to the responsible administrator can be sent.

There are three levels of resource lack:

- first level: a warning will be displayed.
- second level: all receivers will be stopped, so that no file transfer will be possible
- third level: all processes and rvsEVO itself will be ended.

For all 3 levels a message in the log file will be issued and the responsible administrator can be informed.

certificate check Checking of expirationdates of you own keypairs and the certificates of your partners. If a certificate expires in preset time a message will be issued in the Monitor Log. At the same time an E-mail to the responsible administrator can be sent.

ELEMENT	DESCRIPTION
DiskSpace	Number of kilobytes, that has to be free in every checking rvsEVO directory, before a warning in Monitor Log will be issued. If the disk space is less than given in this parameter also an e-mail can be sent to the address defined in Mail parameter. Default: 150 000.
Error certificate expiration	If certificate expiration is reached in less days than defined in this parameter an error message will be issued in Monitor Log and an e-mail can be sent to the address defined in parameter Send e-mail if certificate expires/expired . Default: 7
Mail	E-mail address for sending warnings if the number of free kilobytes is less than given in DiskSpace parameter. An other e-mail is sent after the time given in parameter Time . Please see also parameter MailLocalAddress and MailSMTPHost in chapter 3.1.1 "rvsEVO Environment"
RcvStop-DiskSpace	Number of kilobytes, that has to be free in every checking rvsEVO directory, before a warning in Monitor Log will be issued. If the number of free kilobytes is less than a value of this parameter all rvsEVO receiver processes will be stopped. No file transfer will be possible. Default: 120 000
rvsEVO server stop disk space	Number of kilobytes, that has to be free in every checking rvsEVO directory. With this parameter the critical limit for the resource deficit should be configured. If the number of free kilobytes is less than a value of this parameter all rvsEVO processes and the rvsEVO server will be stopped. You have a possibility to start a script (see parameter System in this table). Default: 100 000.
Send e-mail if certificate expires/expired	E-mail address for sending warnings if the certificate expiration is reached in less days than defined in parameter Warning certificate expiration and error messages if the expiration is reached in less days than defined in parameter Error certificate expiration . Please see also parameter Mail Local Address and Mail SMTP Host in chapter 3.1.1 "rvsEVO Environment"
SuspendTime	Time between recourse checks after the value given in RcvStopDiskSpace parameter was underrun.
System	a script, which should be executed, if the critical limit (level) is reached. See CriticalDiskSpace in this table.

ELEMENT	DESCRIPTION
Time	Time in seconds between two resource checks. Default: 600.
Warning certificate expiration	If certificate expiration is reached in less days than defined in this parameter a warning will be issued in Monitor Log and an e-mail can be sent to the address defined in parameter Send e-mail if certificate expires/expired . Default: 30

XML configuration file `rvsConfig.xml`

In this section we give some notes about the structure of the XML configuration file `rvsConfig.xml`.

The parameters in this file must be specified in the respective XML element. The file itself must be a valid XML file.

Example:

```
### Excerpt from rvsConfig.xml #####
```

```
...
<Environment>
<TEMP>c:\Programs\rvsEVO\files\temp</TEMP>
<INBOX>c:\Programs\rvsEVO\files\inbox</INBOX>
<OUTBOX>c:\Programs\rvsEVO\files\outbox</OUTBOX>
<ARCDIR>c:\Programs\rvsEVO\archive</ARCDIR>
<JobstartConfigFile>rvsJobstart.xml</JobstartConfigFile>
<StationsConfigFile>rvsStationlist.xml</StationsConfigFile>
<MonlogStylesheet>MonlogStylesheet.xslt</MonlogStylesheet>
<PersistenceArchive>RevisionLog.xml</PersistenceArchive>
<HostAllowFile>host.allow</HostAllowFile>
<HostDenyFile>host.deny</HostDenyFile>
<RMIServiceName>rvsEVO</RMIServiceName>
<RMIServiceHost>localhost</RMIServiceHost>
<JobQueueTimeout>3000</JobQueueTimeout>
</Environment>
...
```

```
##### rvsConfig.xml #####
```

You can define and edit the paths for the TEMP, INBOX, OUTBOX, LOGDIR and ARCDIR elements.

Example:

```
<TEMP>C:\Programs\rvsEVO\temp</TEMP>
```

You are free to choose the names for the `rvsJobstart` and `rvsStationlist` files; the only requirement is that they are specified in the respective XML element, are valid XML files and are located in the conf directory. The same also applies to `HostAllowFile` and `HostDenyFile`. (Please see parameters of `rvsEVO Environment`).

Example:

```
<StationsConfigFile>stations.xml</StationsConfigFile>
```

The `stations.xml` file is a station list in XML format containing the required rvsEVO parameters (see chapter 3.2) and is located in the rvsEVO `conf` directory.

The entries for `RMIServiceName` and `RMIServiceHost` are for internal communication and must not be changed.

3.2 Customizing the station configuration

Station Configuration

During installation you set up a local station.

For further customization of stations (creation of new stations and their modification or deletion) you can use:

- the Graphical User Interface (GUI) or
- configuration file `$RVS_HOME\conf\rvsStationlist.xml` in XML format (see chapter 3.2.7 "Customizing stations via XML configuration file").
- command tool `updateStationList` (see page 74)

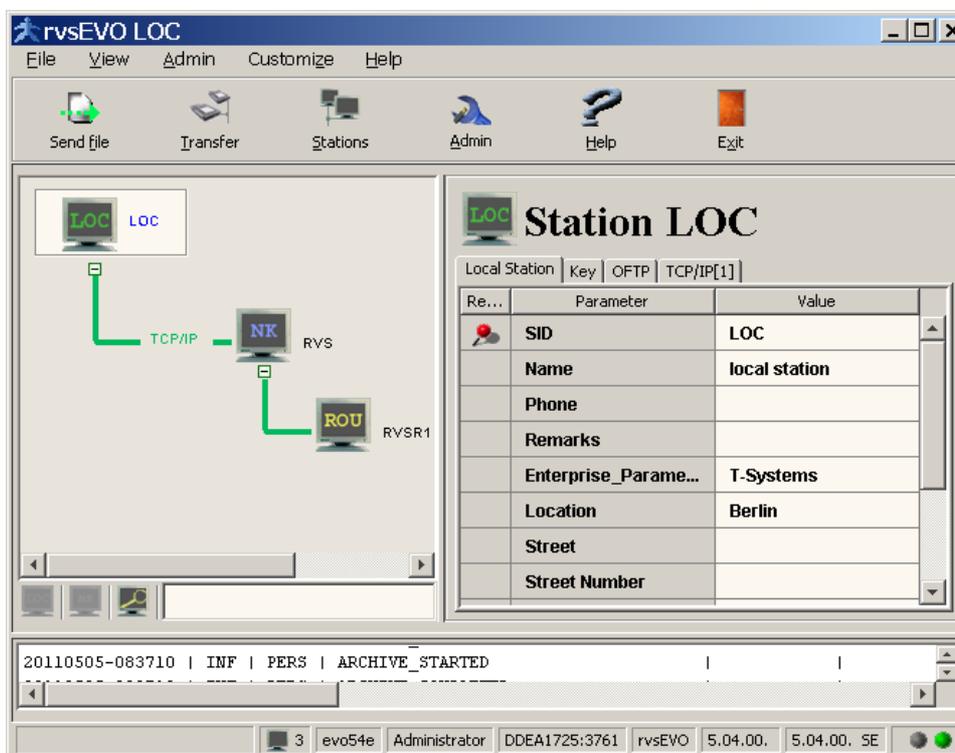
The obligatory parameters for the local station have already been polled during installation, and the

```
$RVS_HOME\conf\rvsStationlist.xml
```

station configuration file has been appropriately adapted. This configuration is also visible in the GUI. In Chapter 3.2.3 "Setting up of a neighbour station" (page 58) you can read how to set up a neighbour station, in chapter 3.2.4 "Setting up a routed station" you can read how to set up a routed station and in chapter 3.2.5 "Setting up a virtual station" you can read how to set up a virtual station.

3.2.1 Customizing stations via GUI

To open the station window, select the **Stations** icon in the function bar.



On the left side of the stations window you will see the station tree; to the right is a station parameter table.

Alternative to the station tree the stations can be shown as a list. This functionality is to configure in `$RVS_HOME/conf/rvsEvoClient.prefs` file with the following definition:
<entry key="client.stationlist.stationtree.enabled" value="false"/>.

The station tree/list depicts all of the stations which exist in rvsEVO (e.g. your local station, the virtual stations, the neighbour station and the routing station) as well as the connection type of the local station and the neighbour station (TCP/IP, TLS, ISDN, XOT, Proxy TCP/IP and Proxy TLS). With a click on the small button below the local station you can show or hide the neighbour stations and the virtual stations. The small button below a neighbour station blends or masks out the routing stations.

The station table on the right-hand side of the window displays all of the parameters for the station currently selected. With the aid of the various station tabs (OFTP, TCP/IP, ISDN, XOT, Proxy TCP/IP and Proxy TLS) you can configure various parameter groups.

Grayed fields indicate that these parameters cannot be edited.

The parameters which are obligatory for station configuration are marked in the column **Re..** (Required) with the symbol .



Example: `Odette Id` is obligatory in the OFTP tab.

On the right-hand side of the window beneath the station parameter table is a series of buttons: **S**ave, **C**ancel, **U**ndo, **U**ndo **A**ll. These allow you to save changes (Save), discard them (Cancel) or reverse them (Undo, Undo All).

3.2.2 Configuring a local station

Normally the local station parameters are already set during the installation. In this chapter you will find the explanation of all for local station possible parameters.

As mentioned above the configuration of the stations are possible via GUI or via XML configuration files. Thus you will find in the parameter table the name of the parameter from the XML file in <> .

The possible tabs for the local station configuration are: Local Station, Key, OFTP (Odette parameters), and network tabs for TCP/IP, TLS, ISDN, XOT, Proxy TCP/IP or Proxy TLS. A right click on the local station opens a context menu with the possibility to add a new receiver: TCP/IP, TLS, ISDN, XOT, Proxy TCP/IP or Proxy TLS. A TLS receiver is needed, if you want to encrypt all data over the network connection.

- Local Station: the mandatory parameters in this tab is SID and the network. The parameter SID is locally unique station ID which can consist of up to sixteen characters. It is a strictly local definition; remote stations do not have access to these names; they only know the ODETTE IDs. The parameter network was assigned during rvsEVO installation, while setting TCP/IP parameters. The rest parameters are optional and are contact data.
- OFTP (ODETTE Parameters): For the local station it is necessary only to set the ODETTE ID. ODETTE ID is a worldwide unique identification of all nodes using the ODETTE file transfer protocol (OFTP). This 25 character name consists of
 - the letter O,
 - an 18 character organization identifier provided by the ODETTE codification group, and
 - a 6 character computer sub address that is administrated by each organization.

Note: If you communicate within your own closed network only, the ODETTE ID may be freely chosen as long as it remains unique in your network.
- Key (key administration): Please look at chapter 6 "Encrypted transmission with rvsEVO" for further information.

TCP/IP

In the next table you will find the description of the TCP/IP parameters:

TCP/IP parameters

Parameter	Description
enabled <enabled>	This parameter defines whether or not the TCP/IP listener is enabled or disabled: Default: Y (Yes). Possible values: Y (Yes) or N (No).
IP Address <IPAddress>	IP address or DNS name of the own station. If you own station has only one IP address, you can left this field empty.
max. incoming sessions <Sessions>	Maximal number of simultaneously active receipt processes on the same channel. Maximum: 100
Port <Port>	Port on which a TCP/IP listener is to be started; 3305 by default

Receiver Number <ReceiverNumber>	Number for differentiating the various receiving channels through which the local station can be reached. Each number has a tab with a set of TCP/IP parameters. rvsEVO automatically assigns and manages this number. Default: 1
restart timeout <RestartTimeout>	Time interval in seconds for the restart of a new TCP/IP listener.
timeout <TimeOut>	Time Out in seconds after which the communication program the connection closes, if the partner station does not answers.

TLS

The TLS parameters are necessary for the encrypted communication with a partner. TLS (Transport Layer Security) is an encryption protocol for data transmission in an internet. It provides an encryption during the communication way, on the session level.

In the following table the most parameters are the same as for the TCP/IP network; added are some parameters important for the encryption. Please read the chapter 3.2.6 "How to configure a TLS receiver?" for configuration of the TLS receiver (listener).

TLS parameters

Parameter	Description
client authentication <ClientAuthentication>	Sometimes is for the TLS connection necessary, that a client (in this case a partner, from whom you receive data) should be authenticated. With this parameter you can choose, if this is: <ul style="list-style-type: none"> • NONE (no authentication) • WANTED or • NEEDED Note: The authentication should be done with the X.509 certificates.
enabled <enabled>	This parameter defines whether or not the TLS listener is enabled or disabled: Default: Y (Yes). – Possible values: Y (Yes) or N (No)
IP Address <IPAddress>	IP address or DNS name of the own station for the TLS connection.
Keystore file name <KeystoreFileName>	Name of the key store for the TLS connection. In this file are stored your keys (private and public key) for encrypted communication with TLS. We recommend to use the default key store: \$RVS_HOME/system/data/tlsKeyManagerKeyStore.p12 Please read the chapter 3.2.6 for more information, which steps are needed.
max. incoming sessions <Sessions>	Maximal number of simultaneous active receipt processes on the same channel. Maximum: 100
Port <Port>	Port for the TLS listener; default 6619
Receiver Number <ReceiverNumber>	Number for differentiating the various receiving channels (listeners) through which the local station can be reached. Each number has a tab with a set of TLS parameters. rvsEVO automatically assigns and manages this number. Default: 1
restart timeout <RestartTimeout>	Time interval in seconds for the restart of a new TCP/IP listener.
timeout <TimeOut>	Time Out in seconds after which the communication program the connection closes, if the partner station does not answer.

trusted certs keystore file name <TrustManagerKeystoreFileName>	Name of the key store for the trusted certificates needed for the TLS connection. In this file you should import X.509 certificates (public keys in form of X.509 certificates) of your partner stations. We recommend to use the default trusted certificates keystore file \$RVS_HOME/system/data/tlsTrustManagerKeyStore.p12. Please read the chapter 3.2.6 for more information, which steps are needed.
--	--

Proxy TCP/IP and Proxy TLS

The parameters of Proxy TCP/IP and Proxy TLS are almost identical with those of TCP/IP and TLS. There is only one additional parameter:

- Proxy Bastion: you can select the Bastion from a list of the Bastion instances created in rvsEVO in which the receiver / listener should be started.

ISDN

The ISDN parameters for a local station are described in the following table. Please see the chapter 2.3 "Network Requirements" for ISDN system requirements.

ISDN parameters

Parameter	Description
Card number <Device-ReceiverNumber>	Number of the ISDN card installed in the computer, beginning with "0"
enabled <enabled>	This parameter defines whether or not the ISDN listener is enabled or disabled: Default: Y (Yes). – Possible values: Y (Yes) or N (No)
ISDN Address<Address>	An ISDN number of your local station. For the 1TR6 ISDN standard the single-digit (EndSelectionNumber) (ESN) is assigned to the local station. Default: " "
ISDN Facilities	Special information or facilities of the ISDN transmission.
ISDN Protocol<Protocol>	Specifies the ISDN standard used. - 1TR6: German national standard - E-DSS1: EURO-ISDN Standard: E-DSS1

ISDN Terminal Identifier	Only required for X.31: Terminal End Identification . Default: 0 – no TEI assigned
ISDN Userdata	User data for the ISDN transmission
max.incoming sessions <sessions>	Maximal number of simultaneous active receipt processes on the same channel. Maximum: 100
OrdinalNumber <ReceiverNumber>	Number for differentiating the various receiving channels through which the local station can be reached. Each number has a tab with a set of ISDN parameters. rvsEVO automatically assigns and manages this number. Default: 1
RCV timeout <RestartTimeout>	Only valid for the local station: Cancellation time in seconds, after which the receiver interrupts the waiting for incoming calls and reconnects to the card driver.
timeout <TimeOut>	Time Out in seconds after which the communication program the connection closes, if the partner station does not answers.
Type <DeviceCard-Number>	not configurable. Determines the type of connection within your computer - CAPI2. Default: CAPI2A A = no Diehl/Eicon card
X.25 Address <X25Address>	15-digit X.25 DTE address of the local station. This number is optional. It is however advisable for an ISDN connection to enter the ISDN number here, as some partners expect an X.25 address.
X.25 Closed User Group	ISDN and X.25 allow for the formation of a closed user group. All members of such a group can communicate with each other via the public telecommunication network. Connection requests to group members received from participants not being a member of the closed user group will be rejected by the switching exchange. The same applies to connection requests from group members to participants not being a member of the closed user group. This service attribute is called Closed User Group (CUG).
X.25 DBit	D-bit: Delivery Confirmation;an X.25 data packet flag used to request end-to-end acknowledgment for the packet.

X.25 Facilities	Special information or facilities for an X.25 transmission; see the information of the X.25 service in use
X.25 PacketSize <PacketSize>	Size of data packets during data transmission
X.25 Userdata	User data for an X.25 transmission; see the information of the X.25 service in use.
X.25 Window Size <WindowSize>	- window size 7 (recommended for ISDN); - window size 2 (recommended for X.25 native) Note: Window size in X.25/ISDN communication is the number of packets that can be outstanding without acknowledgment. The window size will be negotiated during connection setup, but we recommend you to use the correct window size (depending on the partner network).

XOT

XOT (X.25 over TCP/IP) routers are able to route X.25 packets between a TCP/IP network on one side and an X.25 or ISDN network on the other side.

System requirements:

Please read the chapter 2.3 "Network Requirements" for XOT system requirements. On demand a separat document with an example description of a XOT router configuration can be sent.

The XOT parameters for a local station are described in the following table.

XOT parameters

Parameter	Description
enabled <enabled>	This parameter defines whether or not the XOT listener is enabled or disabled: Default: Y (Yes). – Possible values: Y (Yes) or N (No)
Local IP address	Local station only (optional): own IP address. IP addresses have the form "255.255.255.255". If you have not specified a value for the local station, you permit automatic definition of the IP address. If your own station has only one IP address, this field should be left empty.
Local port	Port of the local station Default: 1998 for XOT communication.

max.incoming sessions <sessions>	Maximal number of simultaneous active receipt processes on the same channel. Maximum: 100
OrdinalNumber <ReceiverNumber>	Number for differentiating the various receiving channels through which the local station can be reached. Each number has a tab with a set of XOT parameters. rvsEVO automatically assigns and manages this number. Default: 1
RCV timeout <RestartTimeout>	Time interval in seconds for the restart of a new XOT listener.
Router IP address	IP address of an XOT router. This parameter is optional for the local station and mandatory for the partner station
Router Port	Port of the XOT router Default: 1998 for XOT communication
timeout <TimeOut>	Time Out in seconds after which the communication program the connection closes, if the partner station does not answers.
X.25 Address <X25Address>	15-digit X.25 DTE address of the local station. This number is optional. It is however advisable for an ISDN connection to enter the ISDN number here, as some partners expect an X.25 address.
X.25 DBit	D-bit: Delivery Confirmation; an X.25 data packet flag used to request end-to-end acknowledgment for the packet.
X.25 Facilities	Special information or facilities for an X.25 transmission; see the information of the X.25 service in use
X.25 Modulo	In X.25 data transmission exist two modulo operating standards: Modulo 8 and Modulo 128. Modulo 128 means, that you must select bigger window size via parameter FACILITIES. The standard window size for modulo 8 is 2.
X.25 PacketSize <PacketSize>	Size of data packets during data transmission
X.25 Userdata	User data for an X.25 transmission; see the information of the X.25 service in use

<p>X.25 Window Size <WindowSize></p>	<p>window size 7 (recommended for ISDN) window size 2 (recommended for X.25 native) Note: Window size in X.25/ISDN communication is the number of packets that can be outstanding without acknowledgment. The window size will be negotiated during connection setup, but we recommend you to use the correct window size (depending on the partner network).</p>
--	--

3.2.3 Setting up of a neighbour station

A right click on the local station opens a context menu with the possibility to add a new neighbour station. You can choose one of the following networks: TCP/IP, TLS, ISDN, XOT, Proxy TCP/IP or Proxy TLS. A TLS receiver is needed, if you want to encrypt all data over the network connection.

The possible tabs for the neighbour station are: Neighbour Station, Key, OFTP (Odette parameters), Line Type and network tabs: TCP/IP, TLS, ISDN, XOT, Proxy TCP/IP or Proxy TLS.

- Neighbour Station: the mandatory parameters in this tab are SID and the network. The parameter SID is locally unique station ID which can consist of up to sixteen characters. 'A-Z', '0-9', '-', '_' and '.' are possible characters. '.' may not rank first. SID is a strictly local definition; remote stations do not have access to these names; they only know the ODETTE IDs. The network parameter was assigned during adding a neighbour station. The rest parameters are optional and are contact data.
- Key (key administration): Please look at chapter 6 "Encrypted transmission with rvsEVO" for further information.
- OFTP: In the following table you will find detailed explanation of ODETTE group parameters.

OFTP parameters

Parameter	Description
<p>Authentication <Authentication></p>	<p>authentication on ODETTE level at the beginning of the transmission. Possible values: - Yes (X.509 certificate has to be exchanged) - No</p>
<p>Exchange Buffer Size <BufferSize></p>	<p>Maximal size of ODETTE Exchange Buffer in Bytes. Possible values: 0 - 99999 (default: 10.000). Attention: With ISDN connection the value should not be over 4.000 bytes since otherwise the transmission of large files can cause problems</p>

Certificate Validation Type <CertificateValidation-Type>	please read chapter 11.2.1 "Configuration of stations"
Compression <Compression>	Send parameter on station level. Only available if parameter File Service Proxy = EXTERNAL . Please see table "Send parameters" on page 96 for detailed explanation.
Exchange Buffer Credit <Credit>	Maximal number of sent blocks (Exchange Buffer) without an acknowledgment. Possible values: 0 - 999 (default: 999).
End to End Response in <EERPin>	Procedure for receiving receipts (EERPs/NERPs). NORMAL: The transmission is successfully terminated after receiving of EERP. NEVER: no EERP is needed. The transmission can be terminated without receiving a receipt. Hint: some operator use EERP destination and EERP originator in a different way. This causes problems with the identification of EERPs. Set the parameter <code>rvs_evo.job.additional_erp_reverse_search=true</code> in <code>\$RVS_HOME\conf\rvssystem.properties</code> file to resolve this problem.
End to End Response Out <EERPOut>	Procedure for sending receipts (EERPs/NERPs). NORMAL: Generation of a receipt after successful file reception and immediate active transmission. NEVER: no Generation of a receipt. The job is changed to ENDED status. HOLD: Generation of a receipt after successful file reception. The receipt, however, is only sent after having been released with the handleEERP program. SYNC: The OFTP session between sender station and receiver station will be kept alive waiting for an EERP, that should come from the receiver station. ROUTING_SYNC: Hint: This parameter is important only for a routing station. The OFTP session between the router station and the sender station will be kept alive waiting for an EERP, that should come from the receiver station. When the EERP is arrived, it will be sent in the same session to the sender. Default: NORMAL (see also parameter End to End Response in)

Encryption <Encryption>	Send parameter on station level. Only available if parameter File Service Proxy = EXTERNAL . Please see table "Send parameters" on page 96 for detailed explanation.
Encryption Algorithm <EncryptionAlgorithm>	Send parameter on station level. Only available if parameter File Service Proxy = EXTERNAL . Please see table "Send parameters" on page 96 for detailed explanation.
OFTP Version <Level>	which OFTP version should be used. Possible values: 1.3, 1.4 and 2.0
Odette ID <OdetteID>	ODETTE ID is a worldwide unique identification of all nodes using the ODETTE file transfer protocol (OFTP). This 25 character name consists of <ul style="list-style-type: none"> – the letter O, – an 18 character organization identifier provided by the ODETTE codification group, and – a 6 character computer sub address that is administrated by each organization.
Receive Password <PasswordReceive>	The password rvsEVO expects from the neighbour station.
Send Password <PasswordSend>	The password rvsEVO sends to the neighbour. ODETTE password exchange between two neighbour stations and verification always occurs while a session is being established.
PKI <PkiEnabled>	please read chapter 11.2.1 "Configuration of stations"
File Service Proxy <Proxy>	Only for File Service Modul; possible values: <ul style="list-style-type: none"> – NON (default) – INTERNAL – EXTERNAL Please see chapter 8 "File Service Module" for detailed explanation.
Restart <restart>	This parameter defines whether a restart is allowed after an error message. Possible values: true (default) or false:
Security Feature Set <SecurityFeatureSet>	Send parameter on station level. Only available if parameter File Service Proxy = EXTERNAL . Please see table "Send parameters" on page 96 for detailed explanation.

Security <SecuritySet>	Defines whether or not encryption is to be used during file transmission. Values: SECURITY=NO Encryption is impossible. The job aborts with an error message if a send job requires encryption. SECURITY=OPT Encryption possible as an option and can be specified in the send job. SECURITY=FORCED Encryption is compulsory. A warning is issued and the send job is converted into an encrypted job if a send job is scheduled without encryption. Reception of the file is refused if the partner station sends an unencrypted file. A send job for station 'S' is processed according to the SECURITY entry for station 'S', regardless of whether 'S' is a neighbouring station or is reached via routing. Default: SECURITY=OPT
Sign <Sign>	Send parameter on station level. Only available if parameter File Service Proxy = EXTERNAL . Please see table "Send parameters" on page 96 for detailed explanation.
Sign ERP <SignERP>	Send parameter on station level. Only available if parameter File Service Proxy = EXTERNAL . Please see table "Send parameters" on page 96 for detailed explanation.
SFIDDESC as File-name <UseDescAsFile-name>	SFID (Start File ID) is used as filename (instead of VDSN) for the transmission.
VDSN charset <VdsnCharset>	For receive jobs only: allowed character set for VDSN. Possible values: - ODETTE: ODETTE character set is allowed only (default). ALL: all ASCII characters are allowed

- Line Type: the only parameter in this tab is **Active connection setup**. It is occasionally necessary for files to be made only available and not to be sent immediately. On the contrary, the partner station is to establish the connection and fetch the available files as and when required. The partner bears the costs for the connection. Active connection establishment must be switched off for this case.
- TCP/IP and TLS: For the partner station it is necessary to set the IP address and the port.

ISDN

The ISDN parameters for a partner station are described in the following table.

ISDN parameters

Parameter	Description
Card number	Number of the ISDN card installed in the computer, beginning with "0"
Dial Retry Count	Count of dial retries, if the partner does not answer
Dial Retry Wait Time	Wait time between two dial retries, if the partner does not answer
ISDN Address	An ISDN number of the partner station.
ISDN Facilities	Special information or facilities of the ISDN transmission.
ISDN Protocol	Specifies the ISDN standard used. – 1TR6: German national standard – E-DSS1: EURO-ISDN Standard: E-DSS1
ISDN Terminal Identifier	Only required for X.31: Terminal End Identification . Default: 0 – no TEI assigned
ISDN Userdata	User data for the ISDN transmission
ReceiverNumber <ReceiverNumber>	Number for differentiating the various receiving channels through which the local station can be reached. Each number has a tab with a set of ISDN parameters. rvsEVO automatically assigns and manages this number. Default: 1
timeout <TimeOut>	Time Out in seconds after which the communication program the connection closes, if the partner station does not answers.
Type	not configurable: Determines the type of connection within your computer - CAPI2. Default: CAPI2A A = no Diehl/Eicon card
X.25 Address <Address>	15-digit X.25 DTE address of the partner station.

X.25 Closed User Group	ISDN and X.25 allow for the formation of a closed user group. All members of such a group can communicate with each other via the public telecommunication network. Connection requests to group members received from participants not being a member of the closed user group will be rejected by the switching exchange. The same applies to connection requests from group members to participants not being a member of the closed user group. This service attribute is called Closed User Group (CUG).
X.25 DBit	D-bit: Delivery Confirmation; an X.25 data packet flag used to request end-to-end acknowledgment for the packet.
X.25 Facilities	Special information or facilities for an X.25 transmission; see the information of the X.25 service in use
X.25 PacketSize <PacketSize>	Size of data packets during data transmission
X.25 Userdata	User data for an X.25 transmission; see the information of the X.25 service in use.
X.25 Window Size <WindowSize>	window size 7 (recommended for ISDN) window size 2 (recommended for X.25 native) Note: Window size in X.25/ISDN communication is the number of packets that can be outstanding without acknowledgment. The window size will be negotiated during connection setup, but we recommend you to use the correct window size (depending on the partner network).

XOT

Please read the chapter 2.3 "Network Requirements" for XOT system requirements. If desired, it is possible to receive a separate document for the configuration of an ISDN router.

The XOT parameters for a partner station are described in the following table.

XOT parameters

Parameter	Description
ReceiverNumber <ReceiverNumber>	Remote station only: Number for differentiating the various receivers through which the local station can be reached. Note: Each receiver should be defined in one XOT tab of the local station. rvsEVO automatically assigns and manages this number. The first tab of the local receiver has the number 1. Default: 1
Router IP address	IP address of an XOT router. This parameter is optional for the local station and mandatory for the partner station
Router Port	Port of the XOT router Default: 1998
timeout <TimeOut>	Time Out in seconds after which the communication program the connection closes, if the partner station does not answers.
X.25 Address <>	15-digit X.25 DTE address of the local station. This number is optional. It is however advisable for an ISDN connection to enter the ISDN number here, as some partners expect an X.25 address.
X.25 DBit	D-bit: Delivery Confirmation; applies only to Data packets. It is used to ensure that acknowledgment of Data packet is end-to-end. When not set, a local node can acknowledge the receipt of a Data Packet before forwarding it on the next node.
X.25 Facilities	Special information or facilities for an X.25 transmission; see the information of the X.25 service in use
X.25 Modulo	In X.25 data transmission exist two modulo operation standards: Modulo 8 and Modulo 128. Modulo 128 means, that you must select bigger window size via parameter FACILITIES. The standard window size for modulo 8 is 2.
X.25 PacketSize <PacketSize>	Size of data packets during data transmission
X.25 Userdata	User data for an X.25 transmission; see the information of the X.25 service in use

X.25 Window Size <WindowSize>	window size 7 (recommended for ISDN) window size 2 (recommended for X.25 native) Note: Window size in X.25/ISDN communication is the number of packets that can be outstanding without acknowledgment. The window size will be negotiated during connection setup, but we recommend you to use the correct window size (depending on the partner network).
--	---

A right-click on the neighbour station opens the context menu, which offers then the options **Add routed station** and **Activate connection**. With the option **Add routed station** you can add a set of routed stations, that are reachable via the neighbour station. **Activate connection** activates the connection to the neighbour station.

3.2.4 Setting up a routed station

Pre-condition: You must already have set up a direct neighbouring node via which you can reach the routed station.

Add routed stations Set up a routed station by a right-click on the neighbour station and selecting from the context menu the item **Add routed station**. For you, the connection type (line type) by which this station is to be reached is of no importance (this is dealt with by the direct neighbouring node). For this reason, the connection type to the routed station is not shown.

The possible tabs for this type of station are: Routed Station, OFTP (Odette parameters). Please refer to the section **Setting up a neighbour station** for the explanation of the ODETTE parameters.

A routed station in rvsEVO can only be reached via the neighbour station. This is shown by the parameter Routing Station in the GUI that is equivalent to the parameter *Gateway* in the element (parameter group) *StationRouted* of the XML station configuration file `rvsStationlist.xml`.

Besides entering the freely selectable station name (SID) in the Routed Station tab, you only have to enter the *Odette ID* in the Odette tab to complete the station configuration of the routed station.

A right-click on the routed station opens the context menu, which offers then an option **Delete station**.

3.2.5 Setting up a virtual station

What are virtual stations?	<p>Virtual stations are used to represent stations outside the OFTP network for the OFTP network. They allow files to be sent to destination stations outside the OFTP network.</p> <p>ODETTE-IDs uniquely define the stations in an OFTP network. Each virtual station must also be assigned an own ODETTE ID. A virtual station can also send and receive.</p>
Add virtual stations	<p>Click the local station with the right mouse button and choose Add Virtual Station.</p> <p>The possible tabs for this type of station are:</p> <ul style="list-style-type: none"> – Virtual Station: the mandatory parameter in this tab is SID. The parameter SID is a unique station ID which can consist of up to sixteen characters. The other parameters are optional and are contact data. – OFTP (Odette parameters): the mandatory parameter in this tab is Odette ID. Please refer to the section 3.2.2 "Configuring a local station" for the explanation of the Odette ID. Please refer to the section 11 "PKI Bindung" for the explanation of parameters PKI and Certificate Validation Type.
Delete virtual stations	<p>A right-click on the routed station opens the context menu, which offers then an option Delete station.</p> <p>Hint: In the station configuration on the partner side the virtual stations should be configurates as routed stations.</p>

3.2.6 How to configure a TLS receiver?

A right click on the local station opens a context menu with the possibility to add a new TLS receiver (listener)

In this chapter we will describe how to configure an encrypted part of the TLS connection. This applies to the two parameters in the receiver configuration list: `keystore file name` and `trusted certs keystore file name`. The other parameters are described in chapter 3.2, table TLS for local station.

Hint: The encrypted part of the TLS connection cannot be configured via Remote GUI.

Which steps are needed for the TLS communication?

The following steps need to be done before you start with the TLS communication.

- create an own key pair for the TLS communication
- export an own public key in form of X.509 certificate
- send you own public key to the partner with whom you should communicate per TLS

- The partner has to import your X.509 certificate (public key) and send to you his X.509 certificate.
- You have to import the partners' X.509 certificate into your TrustManager key file.
- now try to activate the partner station and when OK send a test file

Hint: Do not forget to activate (enable) your TLS receiver.

Creating an own key pair

When adding a TLS receiver (listener) to you local station the first step is to create a key pair for the encrypted TLS communication.

To create a key pair click on the value of the parameter **key store file name**. In this line you will find two symbols: ... and -->.

The symbol ... opens a dialog to select an other key store file and not the default one. Default is: `$RVS_HOME/system/data/tlsKeyManagerKeyStore.p12`. We recommend to use the default one.

The symbol --> starts a program Portecle for the creation of a key pair.

The following steps are necessary:

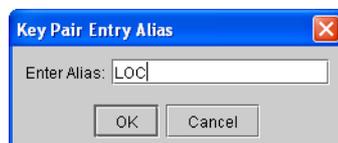
- Start Portecle with -->
- To open the functionality **Generate Key Pair**, select the **Tools** icon in the function bar.
- In the window **Generate Key Pair** use as the Key Algorithm **RSA** and as default key size **1024**.
- Press the button **OK** and the next window **Generate Certificate** opens.
- In the window **Generate Certificate** are the following fields to set:



The Common Name parameter applies in case of a connection to an existing PKI (Public Key Infrastructure) and this parameter is required for the LDAP structure. This parameter is mandatory.

All other data concerns your organization. rvsEVO does not stipulate how to complete the fields.

- After pressing **OK** button, opens the next window



The field **Entry Alias** is only for your internal use, to distinguish the certificates. It should only not be left empty. You can fill it with Common Name as proposed or with your Station ID.

- After pressing **OK** a key pair should be created successfully.

Export of the own public key as a certificate

- Click again on the value of **key store file name** and then on symbol -->. Porticle user interface appears.
- Click on your key pair and press the right mouse button. The context menu appears with the function **Export**.

The following windows appears:



- Use the default settings for the export: **Export Type:** Head Certificate; **Export Format:** DER Encoded.
- Confirm your settings with **OK**.
- In the next window you should name your certificate and store it. We recommend the file name with ending `.cer` or `.crt`. **Hint:** Make a copy of your certificate on another computer.

Import of a partner X.509 certificate

- Click on the value of **trusted certs keystore file name** and then on symbol `-->`. Portecle user interface appears.
- Select the **Tools** icon in the function bar and then the function **Import Trusted Certificate**.
- Select the certificate file in the next window
- The field **Entry Alias** is only for your internal use, to distinguish the partners. It should only not be left empty. You can fill it with Common Name as proposed or with the Station ID of the partner.

After you have done all steps mentioned at the beginning of this chapter try to activate the partner station and send a test file.

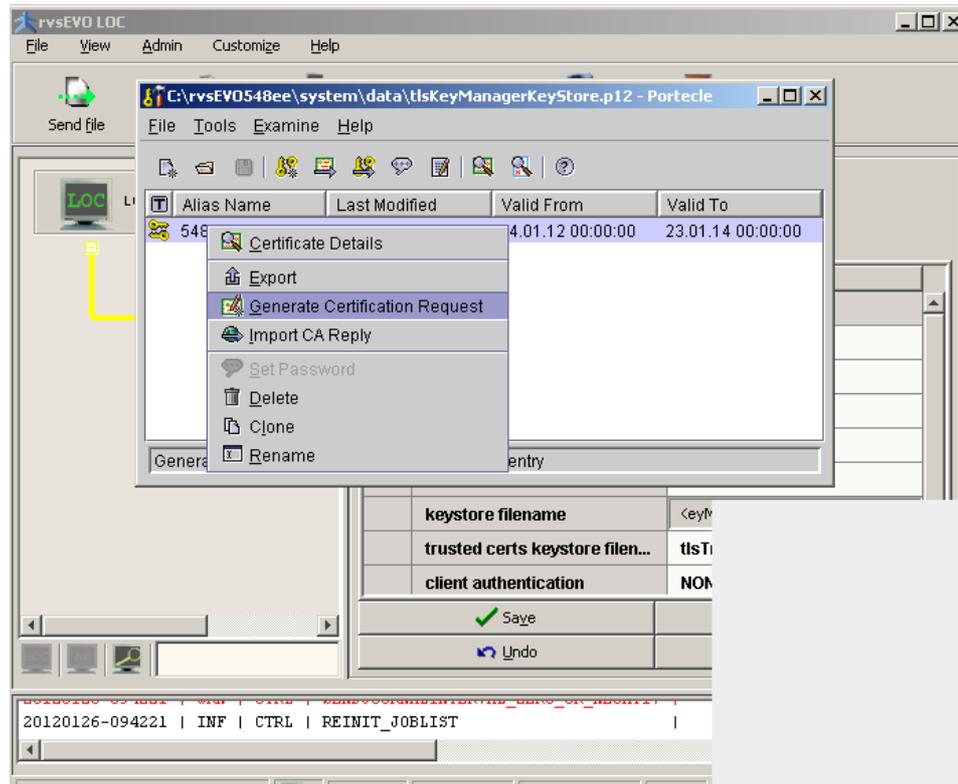
Procedure with CA certificates

This chapter describes the procedure for working with CA certificates:

Generate CSR

- Generate a key pair as described in this chapter.
- A right click on your key pair (click **keystore filename** and symbol `-->` for opening `tlsKeyManagerKeyStore.p12` file) opens a context menu with the possibility to generate a Certification Request. Define name and directory where the request should be saved. By default the

file is generated in PKCS#10 format.



Example of Certification Request:

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBWDCBwgIBADAbMRkwFwYDVQQDDDBlcm81NDhkLW9yaS1maWx1MIGfMAOGCSqGSI
A4GNADCBiQKBgQCezTte+HyTxy/1CCjtXWuADqO7H/LDDKfH7GZANKS2WqJ9+Ozw7:
Tcik5JS3Re8L3q6xGBzDxucD7N6Qp3ikincYTxaE92yuy+rbK43YJJRLdgiXKAu+a
YY/a2UjK0eGY+oVRmWJLlu+pbIs9VASTmWIDAQABMAOGCSqGSIb3DQEBBQUAA4GBA
p426CHVOREM6QzVWf2qNe40R/mxhuKT118kP1xx9oG4K7pkPbHiMuT1SdNmzb/GDp:
tisBm185o+fcDOKaOpIKidjC9aTact16YtSoRdc4v7RIw5t1zpRw+ANhwOwVwZMRg:
/pK79MbM
-----END NEW CERTIFICATE REQUEST-----
```

- Send the request to the TrustCenter (CA). With Odette CA you can make your application online: <https://www.odetteca.com>. Please see <https://forum.odette.org/repository/odette-ca-help.pdf> for more information.

You receive your server certificate (your public key signed by CA), a CA root- and a CA certificate if all entries of your certificate order are correct.

Import of own certificates

You should import the certificates in a specific sequential arrangement into \$RVS_HOME/system/data/tlsKeyManagerKeyStore.p12 file: at first the CA root-, thereafter the CA certificate and at last the server certificate.

- Start Portecle with **keystore filename** and -->.

- Import the CA Root certificate via function **Tools -> Import Trusted Certificate**
- Import the CA certificate via function **Tools -> Import Trusted Certificate**
- At last import the server certificate via function **Import CA Reply**. Highlight your key pair and after a right click on your key pair the context menu offers the option **Import CA Reply**.
- Save and close your keystore.

At least you must send your CA certificates to your partner and import whose certificates:

Import certificate of partner

- Open `$RVS_HOME/system/data/tlsTrustManagerKeyStore.p12` keystore file via Portecle (**trusted certs keystore file Name** and **-->**).
- Import the CA Root certificate of your partner via function **Tools -> Import Trusted Certificate**.
- Import the CA certificate of your partner via function **Tools -> Import Trusted Certificate**.
- Save and close your keystore.

Hint: You must not import the server certificate. Server certificates are exchanged during connection setup.

3.2.7 Customizing stations via XML configuration file

rvsStationlist.xml

On the other hand the station configuration can be done editing the XML station configuration file `$RVS_HOME/conf/rvsStationlist.xml` `rvsStationlist.xml`. The `StationLoc` element in the configuration file is equivalent to the Local Station in the GUI, `StationNeighbour` is Neighbour Station, `StationRouted` is the Routed Station and `StationVirtual` is the virtual station in the GUI.

```
<?xml version="1.0" encoding="ISO-8859-1"?> <rvsStationConfig>
  <StationLoc>
    ....
    <Line>
      <ReceiverNumber>1</ReceiverNumber>
      <Sessions>3</Sessions>
      <TcpBasic>
        <IPAddress>localhost</IPAddress>
        <Port>3305</Port>
      </TcpBasic>
      <TcpRec>
        <Enabled>Yes</Enabled>
        <RestartTimeOut>0</RestartTimeOut>
      </TcpRec>
      <Timeout>0</Timeout>
      <Type>TCP</Type>
    </Line>
  </StationLoc>
```

```

    <ReceiverNumber>1</ReceiverNumber>
    <Sessions>1</Sessions>
    <TcpBasic>
      <Port>3305</Port>
    </TcpBasic>
    <Timeout>0</Timeout>
    <TlsRec>
      <ClientAuthentication>NONE</ClientAuthentication>
      <Enabled>No</Enabled>
      <KeystoreFileName>tlsKeyManagerKeyStore.p12</KeystoreFileName>
      <RestartTimeOut>0</RestartTimeOut>
      <TrustManagerKeystoreFileName>tlsTrustManagerKeyStore.p12</
TrustManagerKeystoreFileName>
    </TlsRec>
    <Type>TLS</Type>
  </Line>
  <Pki>
    <CertificateValidationType>NONE</CertificateValidationType>
    <PkiEnabled>>false</PkiEnabled>
  </Pki>
  <Protocol>
    <Oftp>
      <BufferSize>0</BufferSize>
      <Credit>0</Credit>
      <OdetteID>LOCAL</OdetteID>
    </Oftp>
    <Type>OFTP</Type>
  </Protocol>
  <Sid>LOC</Sid>
</StationLoc>
<StationNeighbour>
  <Fileservice>
    <SecuritySet>NO</SecuritySet>
  </Fileservice>
<Line>
  <ReceiverNumber>1</ReceiverNumber>
  <Sessions>1</Sessions>
  <TcpBasic>
    <IPAddress>139.1.34.76</IPAddress>
    <Port>3678</Port>
  </TcpBasic>
  <Timeout>0</Timeout>
  <Type>TCP</Type>
</Line>
  ...
<Protocol>
  <Oftp>
    <ActiveConnSetup>Yes</ActiveConnSetup>
    <BufferSize>0</BufferSize>
    <Compression>ODETTE</Compression>
    <Credit>0</Credit>
    <EerpIn>NORMAL</EerpIn>
    <EerpOut>NORMAL</EerpOut>
    <Level>2.0</Level>
    <OdetteID>OFDST 001398XDSC</OdetteID>
    <PasswordReceive>111111</PasswordReceive>
    <PasswordSend>111111</PasswordSend>
    <VdsnCharset>ODETTE</VdsnCharset>
  </Oftp>
  <Type>OFTP</Type>
</Protocol>
  <Sid>RVS</Sid>
</StationNeighbour>
<StationRouted>
  <Gateway>RVS</Gateway>

```

```

<Pki>
  <CertificateValidationType>NONE</CertificateValidationType>
  <PkiEnabled>>false</PkiEnabled>
</Pki>
<Protocol>
  <Oftp>
    <BufferSize>0</BufferSize>
    <Credit>0</Credit>
    <EerpIn>NORMAL</EerpIn>
    <EerpOut>NORMAL</EerpOut>
    <OdetteID>OPOSTKLG654NKA</OdetteID>
    <VdsnCharset>ODETTE</VdsnCharset>
  </Oftp>
  <Type>OFTP</Type>
</Protocol>
<Sid>STATIONNAME</Sid>
</StationRouted>
<StationVirtual>
  <Contact/>
  <Pki>
    <CertificateValidationType>NONE</CertificateValidationType>
    <PkiEnabled>>false</PkiEnabled>
  </Pki>
  <Protocol>
    <Oftp>
      <ActiveConnSetup>Yes</ActiveConnSetup>
      <Authentication>>false</Authentication>
      <BufferSize>10000</BufferSize>
      <Compression>ODETTE</Compression>
      <Credit>999</Credit>
      <EerpIn>NORMAL</EerpIn>
      <EerpOut>NORMAL</EerpOut>
      <Level>2.0</Level>
      <OdetteID>OEVOE</OdetteID>
      <Restart>>true</Restart>
      <UseDescAsFilename>>false</UseDescAsFilename>
      <VdsnCharset>ODETTE</VdsnCharset>
    </Oftp>
    <Type>OFTP</Type>
  </Protocol>
  <Sid>EVOV</Sid>
</StationVirtual>
</rvsStationConfig>

```

Lines starting with (<!--) and ending with (-->) are interpreted as comments.

Edit this file if you have to change mandatory parameters (ODETTE ID, TCP/IP_Basic) or if you wish to assign values to other optional parameters such as Contact.

Use a text editor (e.g. Edit, TextPad) to do so. Please make sure to save your XML files as valid XML documents after editing as rvsEVO otherwise will not be able to read them and may fail to start correctly.

The changes in the GUI will be visible immediately (after saving) in the XML station configuration file; the changes in the XML station configuration file will be visible in the GUI only after a new start of the rvsEVO (command startGUI).

Note: For correct TCP/IP communication you must ensure that the IP ports for RMI (1099) and for Odette (e.g. 3305) are free.

For the description of the parameters in the `rvsStationlist.xml` file please refer to the description of the GUI parameters.

`updateStation-`
`List`

Additionally you have the possibility to set up the stations with a command tool at run-time. For this a XML file must be generated, which contains all informations about the stations you would like to modify. The content of this file is described in a XML pattern (`$RVS_HOME/system/data/xsd/stationList.xsd`). For each station, an additional XML element (`function`) is necessary.

Possible values:

- `insert`: add a new station
- `update`: modify existing stations
- `delete`: delete a station (the own station cannot be deleted)

This functionality is actuated with the following tool:

```
updateStationList -f <xml filename> -u <username> -p  
<password>
```

3.3 Customizing the JobStarts

The JobStart configuration comprises rules that allow special programs to be launched when appropriate files are being sent or received or when a job is failed.

Hint: Don't save changed scripts and your own one in `$RVS_HOME\bin` directory or subdirectories of `$RVS_HOME\bin` because these will be overwritten during update installations.

JobFilter If more than one jobFilter applies to the send job, receive job or failed job, the job is started whose jobFilter applies most exactly (eg while using wildcards: `testdoc*` is more exactly than `testd*`).

It is possible to customize JobStarts via the GUI or via the JobStart configuration file.

Hint: Within rvsEVO installation already configured jobstarts are saved in `$RVS_HOME\bin\jobstart` directory. The description of the Jobstart scripts can be found at the end of this chapter.

3.3.1 Customizing via GUI

At first you should open the Administration window selecting the Aadmin icon in the function bar. Then select the item Jobstart in the Administration tree on the left hand side. How to start GUI read please in the chapter 2.6.

It is possible to choose between Jobstarts in receive direction, Jobstarts in send direction and Jobstarts after failure. A Jobstart in receive direction is equivalent to a resident receive entry in rvs[®]. A Jobstart in send direction is equivalent to a Jobstart after send attempt in rvs[®]. From version 5.3 onwards it is possible to choose Jobstarts after processing error.

A new Jobstart will be created with a right-click on a Jobstart after receive, a Jobstart after send attempt or a Jobstart after fail in the Administration tree (**Add new entry**). To select the already existing Jobstart, double-click the appropriate line of the appropriate JobStart in the right-hand window.

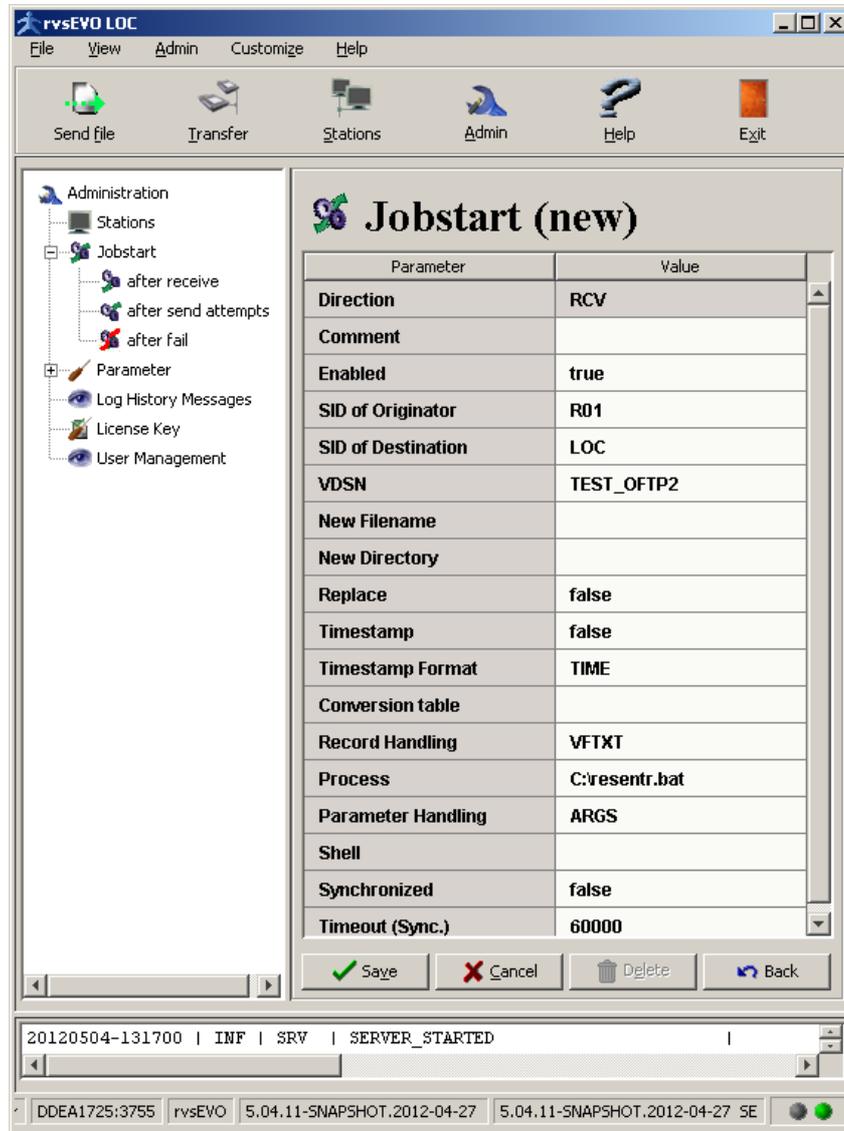
The following JobFilters' entries are possible:

- all JobStarts: Comment, Enabled, Sid of Originator, Sid of Destination, VDSN, Process, Parameter Handling, Shell, Synchronized and Timeout (Sync.)
- JobStart after receive: New Filename, New Directory, Replace, Timestamp, Timestamp Format, Conversion table and Recordhandling
- JobStart after send attempts: Send Attempts

Please refer to the table of the JobFilters for a detailed description.

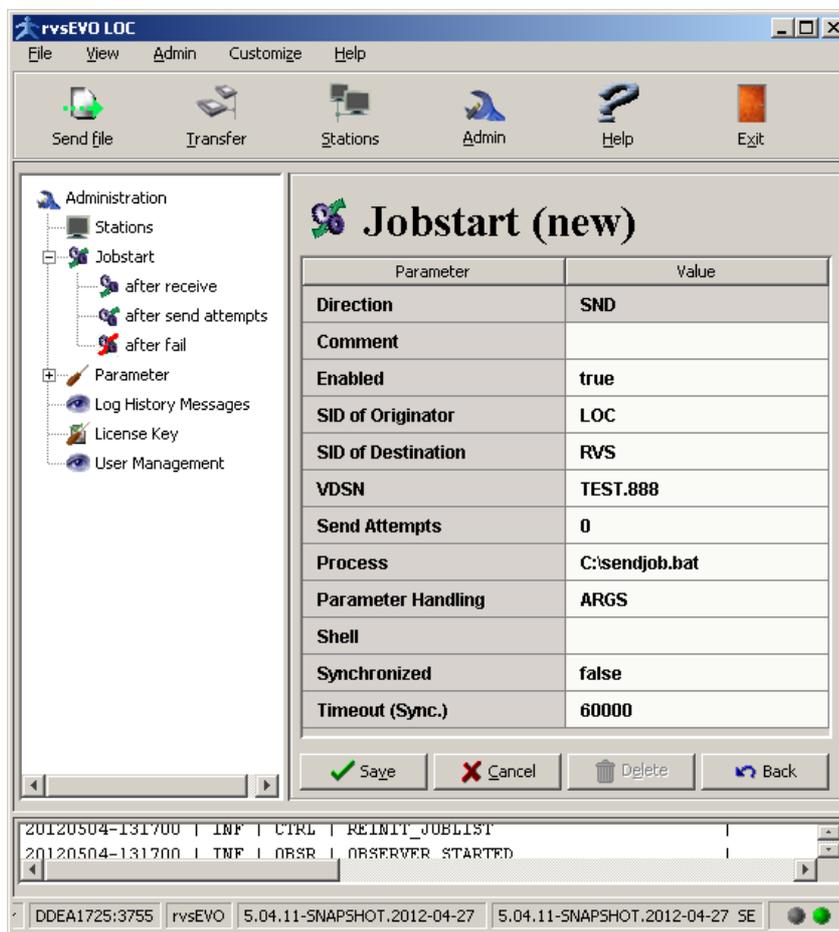
Examples (GUI) **1st example: JobStart after receive**

If rvsEVO receives a file with the virtual file name TEST_OFFTP2 from a routed station R01 the program C:\resentr.bat will be started



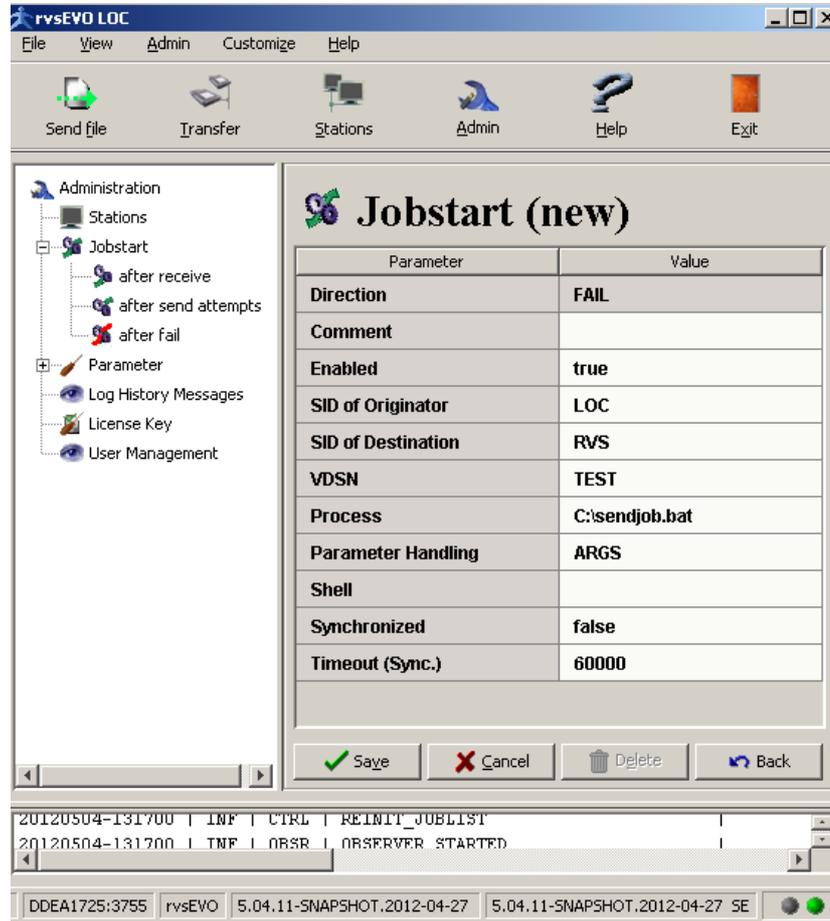
2nd example: JobStart after send attempts:

If rvsEVO sends successfully (`SendAttempts=0`) a file with the virtual file name `test.888` to the neighbour station RVS, the program `C:\sendjob.bat` will be started.



3rd example: JobStart after fail:

If rvsEVO receives an error message after send attempt a file with the virtual file name TEST to the neighbour station RVS, the program C:\sendjob.bat will be started.



Note: A JobStart after fail is launched if a job was changed to FAILED status or if active jobs were changed into an error status which requires an operator intervention (e.g. Restart). A JobStart after fail is started also after a NERP was received.

The following situations activate a JobStart after fail:

Send transmission:

- preprocessing fails (status: SP_FAILED)
- partner station is missing
- JobStart after send attempts fails (ENDED_WITH_JS_ERROR)
- SFNA (Start File Negative Answer) after connection establishment (e.g. unencrypted file transmission while the neighbour station demands encryption)
- FATAL_ERROR after SFNA (e.g. the file does not exist)

- JobStart fails after successful send attempt (ENDED_WITH_JS_ERROR)
- EFNA error (**End File Negative Answer**) after transmission
- receipt of NERP (Negative-End-to-End-Response)

Receive transmission:

- dispatch of SFNA
- dispatch of EFNA (e.g. the byte count of the received file is not conform to the byte count of the transmitted file)
- subsequent treatment fails; e.g. missing private key for decryption (SP_FAILED)
- JobStart after receive fails (ENDED_WITH_JS_ERROR)

3.3.2 Customizing via XML configuration file `rvsJobstart.xml`

Like most of the other rvsEVO configuration files this file is in the XML format as well.

```
<jobstarterData>
  <jobfilters>
    <jobfilter>
      <vdsn></vdsn>
      <direction>SND</direction>
      <sidOrig>LOC</sidOrig>
      <sidDest>RVS</sidDest>
      <sendAttempts>0</sendAttempts>
      <process>C:\jobstart.bat</process>
      <processingClass/>
      <enabled>true<enabled>
      <conversionTable>ANSI-IBM037</conversionTable>
    </jobfilter>
    <jobfilter>
      ...
    </jobfilter>
    ...
  </jobfilters>
</jobstarterData>
```

jobFilter elements

This file comprises any number of jobFilter elements. The table below gives a detailed description of individual jobFilter sub elements.

jJobFilter elements

Parameter	Description
Comment	free text

Conversion table	<p>For ASCII(ANSI) - EBCDIC conversion the following conversion tables are available: ASCII-IBM037, ASCII-IBM273, ANSI-IBM073, ANSI-IBM273.</p> <p>For EBCDIC - ASCII(ANSI) conversion the following conversion tables are available: IBM037-ASCII, IBM273-ASCII, IBM037-ANSI, IBM273-ANSI.</p> <p>For using your own conversion table, please read the note for conversion tables on page 85</p>
Direction	<p>Defines filter rules for the communication direction. Possible values: SND (when sending files), RCV (when receiving files) and FAIL (after processing error).</p>
Enabled	<p>With this parameter you decide whether the job is to be started or not. Possible values: <code>true</code>: to start the job (default) <code>false</code>: the job is not started</p>
New Filename	<p>With this parameter you can save the received file with an other name.</p>
New Directory	<p>With this parameter you can save the received file in an other directory than Inbox directory.</p>
Parameter Handling	<p>With this parameter you can decide how to transmit the job data to the process. Possible values: <code>ARGS</code>: job data are passed as arguments (default) <code>ENV</code>: job data are set as environment variables <code>REPLACE</code>: for compatibility with rvs® portable. For further information see chapter "Parameter handling" on page 82</p>

Process	<p>Program to be started when all filter conditions apply. A defined set of parameters is passed to the programs.</p> <p>Parameters:</p> <ol style="list-style-type: none"> 1 JobId 2 Station ID (of sender or recipient) 3 File name of the file sent or received 4 VDSN 5 Date of the job 6 Time of the job 7 Number of attempts to send. 8 Format of a file 9 Record length of a file 10 Transmitted Bytes <p>For explanation of parameters, when sending a file see chapter 4.5. How these parameters should be used in a script, see the example at the end of this chapter (file \$RVS_HOME/bin/jobstart/jobstart_detailed.bat). (not for Remote GUI; please see process (Server))</p>
Process (Server)	only for Remote GUI (instead of process)
Record Handling	<p>With this parameter you decide, whether each line of a received file in F or V format is terminated by line feed. Possible values:</p> <p>VFTXT: Files in F or V format are saved in text mode. Line feed is inserted at the end of each record (default)</p> <p>VTXT_FBIN: Files in V format are saved in text mode; files in F format are saved in binary mode</p>
Replace	<p>With replace you decide how to handle receiving files which own the same name like existing files.</p> <p>Possible values:</p> <p>true (Yes) : replace the existing file</p> <p>false (No) : create new data set with unique name; timestamp is added (default)</p>
Send Attempts	Number of failed attempts to send. Successful file transmission is indicated by "0" here.
Shell	Command Shell for executions of the program e.g. ksh, csh, ... on Unix systems
SID of Destination	Station ID of target station.
SID of Originator	Station ID of source station.

Synchronized	With setting <code>true</code> : the OFTP session between sender station and receiver station will be kept alive waiting until the process is finished. Default: <code>false</code>
Timeout (Sync.)	If setting Synchronized = <code>true</code> : Time Out after which the communication program the connection closes.
Timestamp	With this parameter you define rules for timestamp creation. Possible values: <code>true</code> (yes) : timestamp is to be generated generally <code>false</code> (No) : timestamp is to be added only if necessary (default)
Timestamp Format	Format of the timestamp. Possible values: <code>TIME</code> : time of creation of the job and counter; format: <code>hhmmssccc</code> (default) <code>DATETIME</code> : date and time of creation of the job and counter; format: <code>YYMMDDhhmmssccc</code> <code>SFID_DATETIME</code> : Odette timestamp of SFID; date, time and counter in format <code>YYMMDDhhmmssccc</code> <code>COUNTER</code> : 000000 - 999999. If the counter is not sequential, the gap is filled first before the counter is counted up.
VDSN	Virtual file name (regular expression as filter).

Parameter handling

Hint: Subsequent you can find a detailed description of the alignment **parameterHandling=ENV** and **parameterHandling=REPLACE**.

parameterHandling=ENV: job data are set as environment variables. The following environment variables are specified (in angle brackets: name in XML file, in brackets: GUI name):

- **RVS_COMPRESSION** <compression> (Compression)
- **RVS_CONVERSION_TABLE** <conversionTable> (Code table)
- **RVS_CREATED_AT** <creationDate> (Created at)
- **RVS_DATE:** Date of <creationDate>
- **RVS_DESCRIPTION:** <fileDescription> (File desc.)
- **RVS_DIRECTION:** <direction> (Direction)
- **RVS_DISPOSITION:** <disposition> (Disposition)
- **RVS_ENCRYPTION:** <encryption> (Encryption)
- **RVS_ENCRYPTION_ALGORITHM:** <encryptionAlgorithm> (Encryption Algorithm)
- **RVS_ERROR_ID:** <errorID> (error number)

- **RVS_ERROR_TEXT:** <errorText> (description)
- **RVS_EXTERNAL_JOBID:** <externalJobId> (External JobID)
- **RVS_FILENAME:** <filename> (Filename)
- **RVS_FILENAME_SRC:** <filenameSrc> (Source file name)
- **RVS_JOB_ID:** <jobNumber> (ID)
- **RVS_LABEL:** <label> (Label)
- **RVS_LAST_CHANGE:** <lastStateChange> (Last Change)
- **RVS_NERP_CREATOR_ODETTE_ID:** <nerpCreatorOdetteId> (Nerp Creator OdetteId)
- **RVS_NERP_REASON_CODE:** <nerpReasonCode> (Nerp Reason Code)
- **RVS_NERP_REASON_TEXT:** <nerpReasonText> (Nerp Reason Text)
- **RVS_RECORD_FORMAT:** <recordFormat> (Format)
- **RVS_RECORD_LENGTH:** <recordLength> (Record length)
- **RVS_RESTART_POSITION:** <restartPos> (Restart Position)
- **RVS_SECURITY_FEATURE_SET:** <securityFeatureSet> (Security Feature Set)
- **RVS_SEND_ATTEMPTS:** <sendAttempts> (Send attempts)
- **RVS_SERIALIZE:** <serialisation> (Serialisation)
- **RVS_SID_DESTINATION:** <sidDestination> (SID Destination)
- **RVS_SID_ORIGINATOR:** <sidOriginator> (SID Originator)
- **RVS_SIGNATURE:** <sign> (File signature)
- **RVS_SIGNATURE_EERP:** <signEERP> (request signed EERP/ NERP)
- **RVS_STATUS:** <status> (Status)
- **RVS_TIME:** Time of <creationDate>
- **RVS_TIME_START_FILE:** <timeStartFile> (Time Start File)
- **RVS_TRANSMITTED_BYTES:** <transmittedBytes> (Bytes transmitted)
- **RVS_VDSN:** <VDSN> (VDSN)

The table “JobInfoList: Explanation of the job attributes:” on page 121 shows the description of the environment variables.

parameterHandling=REPLACE: A copy of the stated file is created. The placeholders of the scripts are substituted for job data. The placeholders are encased by ? (e.g. "move ?DSN? /home/rvsevo/incomming" becomes "move /home/rvsevo/rvsEVO/files/inbox/TEST.TXT /home/rvsevo/incomming")

Placeholder of all jobstarts:

- **?DSN?:** name of local data set, where received information has been stored
- **?VDSN?:** virtual data set name under which the data set was transmitted
- **?DTAVAIL?:** date, when the data set was available for sending; Format TT/MM/JJ HH:MM:SS
- **?FORMAT?:** record format of data set

- **?BYTES?**: number of transmitted bytes
- **?RECORDS?**: number of transmitted records for **F** and **V** format data sets; always zero for **T** and **U** format data sets.
- **?DTRCV?**: date, when data set was delivered to local user in format TT.MM.JJ HH:MM:SS
- **?UID?**: UserID
- **?SID?**: StationsID
- **?DSNTEMP?**: name of temp. data set

Placeholder of jobstart type SND:

- **?MAXRECL?**: The meaning of this field depends upon the record format of the transmitted data set:
 - **F** format: length of each record
 - **V** format: maximum length a record may have
 - **T** und **U** format: always 0 (zero)
- **?LABEL?**: string if the send command contained a **LABEL** parameter. Can be used to identify the send command.
- **?SECN?**: rvsEVO JobID (number of send command **SE** in rvs® portable)
- **?SKCN?**: rvsEVO JobID (number of send command **SK** in rvs® portable)
- **?SIDORIG?**: StationID of originator
- **?SENDATT?**: number of unsuccessful attempts after which the program is to be started

Placeholder of jobstart type RCV:

- **?MAXRECL?**: The meaning of this field depends upon the record format of the received data set:
 - **F** format: length of each record
 - **V** format: maximum length a record may have
 - **T** und **U** format: always 0 (zero)
- **?CNQS?**: rvsEVO JobID (command number of EERP (End-to- End-Response) for received file in rvs® portable).
- **?CNIE?**: rvsEVO JobID (command number of IE for received file in rvs® portable).
- **?CNIZ?**: rvsEVO JobID (command number of IZ for received file in rvs® portable).

conversion tables

Note: The next section is a short explanation of the conversion tables, that are offered by rvsEVO.

Text files are stored on most systems in one of two computer codes, namely ASCII (American National Standard Code for Information Interchange) or EBCDIC (Extended Binary Coded Decimal Interchange Code).

ASCII is the standard code for UNIX and DOS/Windows systems. EBCDIC was developed for IBM Mainframe computers.

- **ASCII**: US-ASCII ISO 646; the ASCII character set defines 128 characters (0 to 127 decimal). This character set is a subset of many other character sets with 256 characters, including the ANSI character set of MS Windows.
- **ANSI**: Windows ANSI, Values 0 to 127 are the same as in the ASCII character set, values 128 to 255 are similar to the ISO Latin-1 character set.
- **EBCDIC 037**: support characters, which are used in the following countries: Australien, Brasilien, Kanada, Neuseeland, Portugal, Südafrika, USA.
- **EBCDIC 273**: supports characters (especially umlauts), which are used in the following countries: Germany, Austria and Switzerland.

How to add your own conversion table:

In order to use your own conversion table, please follow the procedure given below:

- create a new conversion table. The content of this file is described in the following XML schema:

```
<conversiontable codein='ASCII ISO 646' codeout='EbcDic 037'>
  <description>Table to converts signs from US-ASCII ISO 646 to IBM037
  (EbcDic 037)</description>
  <char value="0">0</char>
  <char value="0">0</char>
  .
  .
  .
</conversiontable>
```

Note: char value is equivalent to the source code and the value in <> is equivalent to the target code.

- store the XML file in the directory
\$RVS_HOME\conf\conversiontables
- to add your table to the list of conversion tables, edit the XML file
\$RVS_HOME\conf\conversiontables\characterSetConverters.xml.

Next you find an abridged version of the file characterSetConverters.xml:

```
<characterSetConverters coding="">
  <!-- -->
  <converter cid="ANSI-IBM037">conversion_ANSI_IBM037.xml</converter>
  <converter cid="IBM273-ASCII">conversion_IBM273_ASCII.xml</
converter>
  .
  .
  .
</characterSetConverters>
```

In the example above cid="ANSI-IBM037" (cid="IBM273-ASCII") is the used name in rvsEVO and

`conversion_ANSI_IBM037.xml`
(`conversion_IBM273_ASCII.xml`) is the name of your XML file.

- Please restart rvsEVO after your complement.

Scripts for Jobstarts:

The `$RVS_HOME\bin\jobstart` directory contains jobstart tools as batch files, delivered by rvsEVO. You have the possibility to conform the files to your requirements or to generate new batch files.

delaftersend.bat

This tool deletes a file after successful send attempt.

handleMangement.bat

This Tool refers to Central Administration. For further information please read chapter 13 "rvsEVO Central Administration".

jobstart.bat and **jobstart_detailed.bat**

This tools write information about the sended file and transmission in `$RVS_HOME\log\jobstart.out` file. With `jobstart_detailed` the dump in `jobstart.out` is formatted, with `jobstart` the dump is unformatted.

journal.bat

This script starts the `sendJournal` tool. `sendJournal` creates a journal and sends this to CentralJournal.

Benötigter Parameter:

-f <filename> Name of request file from CentralJournal.
Default: ZJREQUEST

For detailed information please see the user manual of CentralJournal..

sendback.bat

This tool sends back the received file to the originator. A protocol of this procedure is written in `$RVS_HOME\log\jobstart.out` file.

Jobstart after Aborted Send Attempts

It is possible to define a number of aborted send attempts after those a job shall be startet. Therefor set the parameter `Send attempt`. If several jobs are to be startet after a different number of send attempts this is handled as follows.

Example: Program 1 (P1) shall be started after one send attempt and program 2 (P2) shall be started after three send attempts, then:

1. send attempt -> P1
2. send attempt -> P1
3. send attempt -> P2
4. send attempt -> P1
5. send attempt -> P1
6. send attempt -> P2

Delete a Jobstart

If you want to delete a JobStart, select it at first with a double-click. Now you can remove it with the **Delete** button.

Jobstart Processes

Jobstart scripts With the installation of rvsEVO you get some examples of jobstart processes. They are saved in the `$RVS_HOME\bin\jobstart` directory. You can use or edit this files or create new batch files for your jobstarts. Please find in the following a short description of the scripts delivered by rvsEVO.

delaftersend.bat

This script can be launched in a jobstart after send attempt. After a successful send attempt the sended file is deleted.

handleMangement.bat

This script belongs to Central Administration. Please read chapter 13 "rvsEVO Central Administration" for more information.

jobstart.bat and **jobstart_detailed.bat**

These scripts can be lauched in a jobstart after send attempt or a jobstart after receive. A protocol with information about the sended file and data of the transfer is saved in `$RVS_HOME\log\jobstart.out` file. The display of `jobstart_detailed` is formatted not so the display of `jobstart`.

journal.bat

With this script the tool `sendJournal` is started. `sendJournal` creates a journal and sends this to CentralJournal.

Required parameters:

-f <filename> Name of request file from CentralJournal.
 Default: ZJREQUEST

Please read the user manual of CentralJournal for detailed information.

sendback.bat

With this script the received file is to be sent back to the originator. The settings are equal to the receiving transmission. Information about the file and data of the transfer is saved in `$RVS_HOME\log\jobstart.out` file.

4 Working with rvsEVO

Batch files The present chapter describes all programs available for everyday use of rvsEVO. These programs are located as batch files in the `$RVS_HOME\bin` directory or available via the rvsEVO GUI.

Note: To launch an rvsEVO program you must change to the `$RVS_HOME\bin` directory.

4.1 Starting the rvsEVO server

`startServer` Use the `startServer` program to start rvsEVO Server. rvsEVO Server will be also started by the program `startGUI`. `startGUI` starts GUI and then the rvsEVO server (please read the chapter 2.6).

Example:

```
startServer
```

It is not possible to specify any parameters.

A successful start is indicated as follows:

```
*  
* rvs Server has started.  
*
```

Note: The RMI port 1099 is necessary for the RMI registry, so if this port is occupied, rvsEVO Server will not start successfully.

4.2 Stopping the rvsEVO server

`stopServer` Use the `stopServer` program to stop rvsEVO.

Usage:

```
stopServer -m <mode> [-verbose]
```

All parameters are optional

Parameter	Description
-m <mode>	Time for jobs to terminate before rvsEVO is stopped. Possible values: 0 (default; 120 seconds), 1 (60 seconds), 2 (30 seconds), 3 (20 seconds), 4 (10 seconds),
-verbose	Verbose message output.
-help	Requests help information.
-?	Requests help information.

Example:

```
stopServer
```

Result: The server stops after 120 seconds.

Example:

```
stopServer -m 3
```

Result: The server stops after 20 seconds.

```
*
* rvs Server has stopped.
*
```

4.3 Displaying messages

rvsEVO messages and warnings are saved in the log files in directory \$RVS_HOME\log.

The monitor.log file contains the Monitor messages, the rvs.log file contains the messages from rvsEVO Server and the rvsClient.log file the messages from the command prompt and the rvsEVO clients.

4.3.1 Displaying Monitor messages

Log Messages are normally displayed in the bottom of the GUI window. You can view old Monitor messages via the GUI or via the program showMonitorLog.

`monitor.log` The Monitor messages are saved in the file `$RVS_HOME\log\monitor.log`. A new `monitor.log` file is generated daily and the old `monitor.log` file is renamed in `monitor.log` plus date pattern plus counter. Also a new file is generated if the maximum file size is reached.

The maximum file size and maximum number of log files can be defined in the element `appender name="monlog"` in the `$RVS_HOME\conf\rvsLogger.xml` file.:

– Maximum size: use the function `MaxFileSize`

Syntax: `<param name="MaxFileSize" value="10MB"/>`

– Maximum number of log files: use the function `MaxRollFileCount`

Syntax: `<param name="MaxRollFileCount" value="50"/>`

GUI

Log messages After a new start of the GUI the last 25 log messages are shown. Set the parameter `rvs_evo.monlog.initial_buffer_size` in the file `$RVS_HOME/conf/rvs-system.properties` to change this setting. Delete `#` at the beginning of the line, because otherwise the modification will be ignored. In the following example the last 50 log messages are shown after a new GUI start.

Example:

```
rvs_evo.monlog.initial_buffer_size=50
```

You can configure the display of the Monitor messages. This function is controlled by the properties file

`$RVS_HOME/conf/rvs-system.properties` with the following definition: `rvs_evo.monitorlog.client_layout`.

Example:

```
rvs_evo.monitorlog.client_layout=%1$-11s * %3$-4s *
%4$-36s * %8$-s
```

Explanation of the example:

In the first term `(%1$-11s *)1$` is the parameter, `-` means left-aligned (default is right-aligned), `11` is the column width and `*` is the separator. The last parameter has no column width. Each term begins with `%` and ends with `s`.

The following table shows the possible parameters.

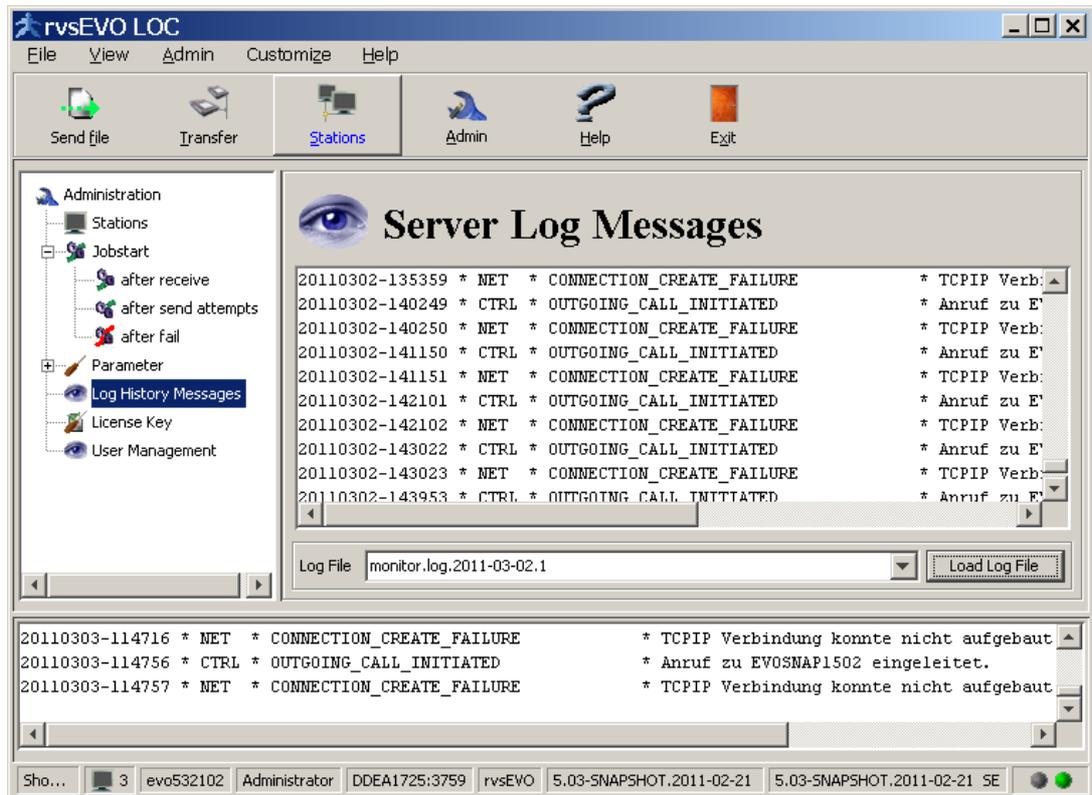
Parameter	Beschreibung
1\$ (time)	Date and time of the entry (format <code>yyyyMMdd-HHmms</code>)

2\$ (level)	Kind of message. Possible values: INF: Information WRN: Warning / important hint ERR: ERROR
3\$ (module)	Module of rvsEVO, which transmitted the message. Possible values: SRV: rvsEVO Server CTRL: controller NET: network OFTP: ODETTE protocol CONF: configuration PERS: persistence ACX: automatic exchange of certificats SP: Service Provider OBSR: Observer
4\$ (messageKey)	Message key
5\$ (sessionID)	ID of OFTP session
6\$ (stationID)	SID of neighbour station
7\$ (jobID)	rvsEVO JobID
8\$ (localized message)	Text message from message key (GUI) or parameter for text message (Log-Datei)

Hint: Using an Oracle database, the Monitor Messages can be saved in the operator control position in MONITOR_MESSAGE table from rvsEVO version 5.04 on. This functionality is to configure in rvsLogger.xml file in \$RVS_HOME/conf directory. Please delete <!-- at the beginning of the batch and --> at the end for activating this function.

History messages You can also view old Monitor messages. Select the Admin icon in the function bar of rvsEVO GUI. The Administration window opens with the Admin tree and the sub-entry **Log History Messages**. How to start of rvsEVO GUI please read the chapter 2.6.

Choose a log file in the selectbox on the right hand side of the window and press the button **Load log file** for viewing the history messages.



showMonitorLog **Command Line**

Use the `showMonitorLog` program to trace the current Monitor messages and to analyze error messages.

Usage:

```
showMonitorLog [-verbose]
```

Optional parameters:

-verbose	Verbose message output.
-help	Displays a description of the current command.
-?	Requests help information.

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

showMonitorLog- The showMonitorLogFile program is similar to the
File showMonitorLog program: use showMonitorLogFile to view a
Monitor log file.

Syntax:

showMonitorLogFile -i <filename>

Required parameter:

-i <filename>	Path and Name of the Monitor log file
----------------------------	---------------------------------------

Optional parameters:

-?	Requests help information.
-help	Displays a description of the current command
-i	Monitor Log filename

4.3.2 Messages from rvsEVO Server

The error messages of rvsEVO Server are saved in \$RVS_HOME\log\rvs.log file.

The maximum file size and maximum number of log files can be defined in the element appender name="rvslog" in the \$RVS_HOME\conf\rvsLogger.xml file.:

- Maximum size: use the function maxFileSize
Syntax: <param name="maxFileSize" value="2097152"/>
- Maximum number of log files: use the function maxBackupIndex
Syntax: <param name="maxBackupIndex" value="10"/>

4.3.3 Messages from command prompt and rvsEVO clients

The warnings and error messages of the command prompt and the rvsEVO clients are saved in \$RVS_HOME\log\rvsClient.log.

The maximum file size and maximum number of log files can be defined in the element appender name="Standard" in the \$RVS_HOME\conf\rvsClientLogger.xml file.:

- Maximum size: use the function maxFileSize
Syntax: <param name="maxFileSize" value="2097152"/>
- Maximum number of log files: use the function maxBackupIndex
Syntax: <param name="maxBackupIndex" value="10"/>

4.4 Activating a station

`activateStation` Use the `activateStation` program to activate the neighbour station in the command line. How to activate the neighbour station via the GUI, please see chapter 3.2.3 "Setting up of a neighbour station".

Note: You cannot activate a routed station. You can activate only a direct neighbouring node (neighbour station).

Usage:

```
activateStation -s <sid> [-verbose]
```

Optional parameters:

-verbose	Verbose message output.
-help	Displays a description of the current command.
-?	Requests help information.

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

Heed messages in the command prompt window starting with message or error if a station activation fails (e.g. due to an incorrect IP address).

Correct the configuration file with the station table when the IP address is incorrect. You must stop and restart rvsEVO each time you have saved the configuration file.

4.5 Sending a file

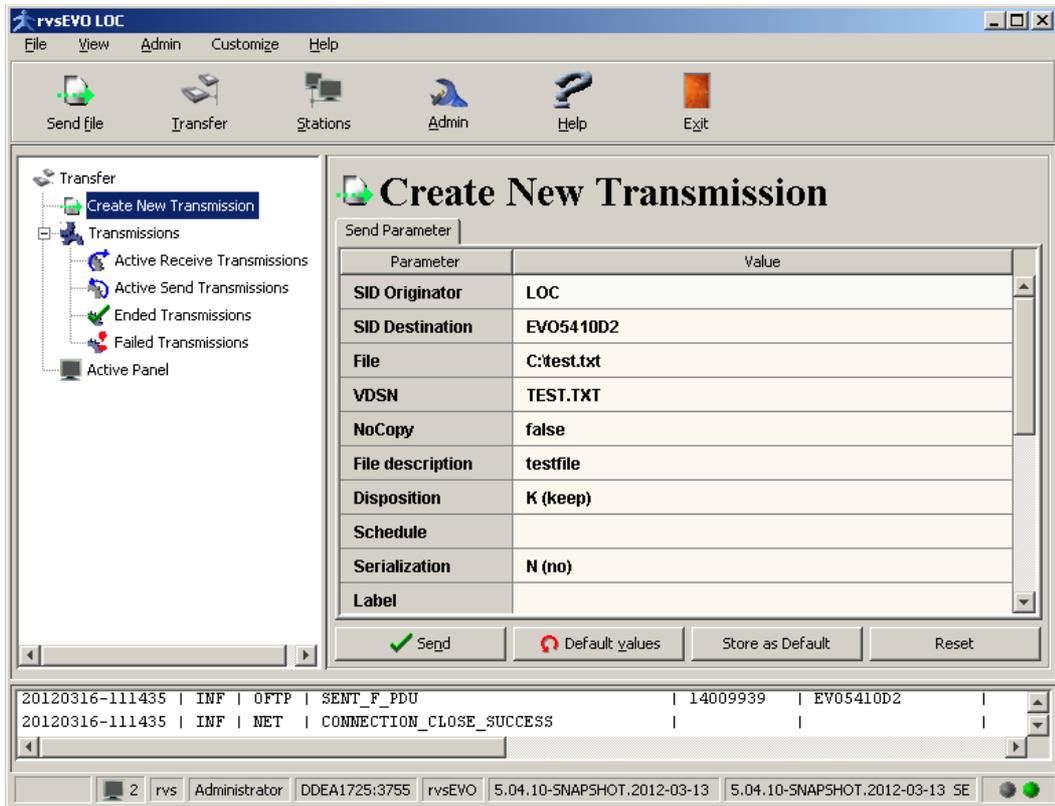
It is possible to send a file via the GUI or via the command line. How to start GUI, please read the chapter 2.6.

GUI

Select the icon **Transfer** in the function bar and thereafter the sub-entry **Create New Transmission** in the Transfer tree. An other way to open the **Create New Transmission** window is to select the **Send file** icon in the function bar.

In the **Create New Transmission** window you can type or select the send parameters and launch the sendjob with a click on the **Send** button.

Use the **Store as Default** button to save the settings and the **Default values** button to access the saved settings if required. By pressing the **Reset** button you can reset the settings to system default.



Send parameters Required parameters:

SID Destination	Recipient station ID.
File	File name of the file to be sent.
VDSN	Virtual file name; the length of the file name used for ODETTE transfer must not exceed 26 characters.

Optional send parameters:

-?	Requests help information.
-----------	----------------------------

Conversion table	<p>For ASCII - EBCDIC conversion the following conversion tables are available : ASCII-IBM037, ASCII-IBM273, ANSI-IBM073, ANSI-IBM273.</p> <p>For EBCDIC - ASCII conversion the following conversion tables are available : IBM037-ASCII, IBM273-ASCII, IBM037-ANSI, IBM273-ANSI.</p> <p>For using your own conversion table, please read the note for conversion tables on page 85</p>
Disposition	<p>Disposition of local file after successful send attempt. Possible values:</p> <ul style="list-style-type: none"> - K (KEEP) file will not be deleted after sending - D (DELETE) file will not be deleted after sending
Encryption	<p>File encryption; active only in connection with Security Feature Set=2, 3 oder 4;</p> <ul style="list-style-type: none"> - Y (Yes) file will be sent encrypted - N (No) file will be sent without encryption
Encryption Algorithm	<p>active only in connection with OFTP 2.0 (CMS) (see parameter Security Feature Set in this table. The following algorithm are possible:</p> <ul style="list-style-type: none"> - None - DES_EDE3_CBC (Triple DES) - AES256_CBC
File description	<p>comment, free text, available only in connection with OFTP version 2. If you partner does not support OFTP version 2, this field will be ignored.</p>
File signature	<p>active only in connection with OFTP 2.0 (CMS) (see parameter Security Feature Set in this table. Possible values:</p> <ul style="list-style-type: none"> - Y (yes): file will be signed. - N (no): file will not be signed
Format	<p>Format of the file to be sent:</p> <ul style="list-style-type: none"> - T=text file; a stream of ASCII characters - U=unstructured (binary); byte stream - V=variable; variable record length - F=fixed; fixed record length <p>Please see also the parameter Record Mode.</p>

Label	Name of group of serialized send jobs. User specified (descriptive) label for this job. If you do not specify this parameter the VDSN will be user as a label
MaxRecl	maximal record length for the files in format F or V, please see also the parameter Record Mode .
NoCopy	With this paramter you can decide whether the file which should be transmitted is to copy to Outbox directory or not. Possible values: true: no copy false: copy is saved in Outbox dorectory (default) Note: not with format U and code conversion
Offline Compression	Compression; active only in connection with Security Feature Set=2, 3 oder 4 . – Y (Yes) ODETTE compression – N (No) no ODETTE compression
Record Mode	In this parameter you indicate the mode of files in format 'F' or 'V'. Possible values: TXT: textfile BIN: binary file Textfile in format 'F': If the record length deviate from the specified one (Parameter MaxRecl) the line will be filled with blanks or will be cut up to defined value. Textfile in format 'V': longer records will be cut up to defined value. Binary files: The length of all records is the max. record size except the length of the last one. Binary files in format 'F': If the size of the last record is shorter than the specified one the line will be filled to the defined value.

Request signed EERP	<p>active only in connection with OFTP 2.0 (CMS) (see parameter Security Feature Set in this table). In OFTP 2.0 there is a possibility to request the signature of the EERP or of the NERP. Your partner must support OFTP 2.0 in this case. Possible values:</p> <ul style="list-style-type: none"> – Y (yes) – N (no) <p>In case that you request signed EERP or NERP and you partner does not send it to you, the rvsEVO job will end in the <code>FAILED</code> status.</p>
Security Feature Set	<p>This parameter applies to the format of encryption. According to the selected kind of the encryption, the groups of available parameters will be activated.</p> <p>Following values are possible:</p> <ul style="list-style-type: none"> – 1 / None: no encryption – 2 / ComSecure (V1) – 3 / ComSecure (V2) – 4 / OFTP 2.0 (CMS) <p>rvsEVO supports two formats of encryption: ComSecure (an appropriate format) und CMS (based on X.509 certificates).</p> <p>CMS format is only in with OFTP version 2 possible (your partner has to support OFTP version 2.0, too).</p> <p>ComSecure (V1) means version 1 (supported by the ComSecure product versions 1.1 and 1.2). ComSecure (V2) means version 2 (supported by the ComSecure product version 1.3). V2 is needed for the encryption of the files bigger than 4 Gbyte. Your partner must support this ComSecure version, too.</p>
Schedule	<p>With this parameter you can decide, what time the SendJob should be launched. The following entry formats are possible:</p> <p><yyyy-mm-dd hh:mm:ss> date and time</p> <p><yyyy-mm-dd> date: the job will be started at zero o'clock</p> <p><hh:mm:ss> time: the job will be started at the same or the next day.</p> <p>If the value is "h" the job is created with <code>HOLD</code> status.</p>

Serialization	Y (Yes)/N (No). If you set Serialization=Y, the files will be sent in the same order, as the send jobs were created. The next job will only be sent, if the previous is completely finished. All send jobs for the serialization must have the same LABEL. In the GUI the VDSN will be used as label. That means, that all jobs for the same serialization group must have the same virtual data set name (VDSN).
SID Originator	station ID of originator; local station (default) or virtual station

Command Line

`createSendJob` Use the `createSendJob` program to send a file to the partner station via the command line.

Usage:

```
createSendJob -d <filename> -s <receiver sid>
-v <vdsn>
[-t <conversion table>] [-F <format>] [-M
<length>] [-S <serialisation> -l <label>] [-desc
<description>] [-D <disposition>] [-C] [-Y] [-sfs
<set-id>] [-sif] [-rsr]
[-Yalg <encryption algorithmen>] [-Ycin <issuer
name>] [-Ycsn <serial no>] [-sifcin <issuer
name>] [-Ycsn <serial no>] [-h?]
[-verbose] [-Sd <init date>] [-St <init time>]
[-Sh] [-idf <filename2>] [-xid <external job id>]
[-so <originator sid> -rm <record mode>] [-su
<user> -sp <password> -sh <host:port>]
```

Required parameters:

-d <filename>	File name of the file to be sent.
-s <receiver sid>	Recipient station ID.
-v <vdsn>	Virtual file name; the length of the file name used for ODETTE transfer must not exceed 26 characters.

Optional send parameters:

-C <compression>	ODETTE compression
-------------------------------	--------------------

-D <disposition>	<ul style="list-style-type: none"> – K (KEEP) file will not be deleted after sending – D (DELETE) file will be deleted after sending
-desc <description>	comment, free text; see the description of the parameter Description for GUI.
-F <format>	<p>Format of the file to be sent:</p> <ul style="list-style-type: none"> – T=text file; a stream of ASCII characters – U=unstructured (binary); byte stream – V (variable); variable record length – F (fixed); fixed record length. <p>Please see also the parameter -rm <record mode>.</p>
-h	Requests help information
-help	Requests help information
-idf <filename2>	rvsEVO-JobID, which rvsEVO writes in a file (filename) after the file was sent successfully (return code = 0).
-j <start job>	without function
-l <label>	Name of group of serialized send jobs. User specified (descriptive) label for this job. If you do not specify this parameter the VDSN will be user as a label.
-M <length>	maximal record length for the files in format F or V. Please see also the parameter -rm <record mode> .
-nocp	By default with creating a send entry the files to be send are copied to Outbox directory of rvsEVO. With option -nocp the original file will be deleted after transmission was successfull. (Not if the format of the file to be send is U

-rm <record mode>	<p>In this parameter you indicate the mode of files in format 'F' or 'V'. Possible values:</p> <p>TXT: textfile BIN: binary file</p> <p>Textfile in format 'F': If the record length deviates from the specified one (Parameter -M <length>) the line will be filled with blanks or will be cut up to defined value. Textfile in format 'V': longer records will be cut up to defined value.</p> <p>Binary files: The length of all records is the max. record size except the length of the last one. Binary files in format 'F': If the size of the last record is shorter than the specified one the line will be filled to the defined value.</p>
-rsr	See the description of the parameter Request signed EERP for GUI.
-S <serialize>	Y (Yes)/N (No). If you set the option <code>-S=Y</code> , the files will be sent in the same order as the send jobs were created. The next job will only be sent, if the previous is completely finished. All send jobs for the serialization must have the same LABEL. In the GUI the VDSN will be used as label. In the command line you can specify your own label.
-Sd <init date>	Date, at which the transmission is to be executed. Format: <yyyy-mm-dd>
-sfs <set-id>	See the description of the parameter Security Feature set for GUI; the possible values are 1-4, too.
-Sh	Create a send entry with status <code>HOLD</code>
-sif	See the description of the parameter File signature for GUI.
-sifcin <issuer name>	certificate issuer name for file signature (only for sfs 4)
-sifcsn <serial number>	certificate serial number for file signature (only for sfs 4)
-so <originator sid>	station ID of originator
-St <init time>	Time, at which the transmission is to be executed. Format: <hh-mm-ss>

-t <conversion table>	name of your own conversion table: For ASCII(ANSI) - EBCDIC conversion: ASCII-IBM037, ASCII-IBM273, ANSI-IBM037, ANSI-IBM273. For EBCDIC - ASCII(ANSI) conversion: IBM037-ASCII, IBM273-ASCII, IBM037-ANSI, IBM273-ANSI. For using your own conversion table, please read the note for conversion tables on page 85
-verbose	Verbose message output.
-xid <external job id>	Parameter for an external JobID, which can refer to several rvsEVO-JobIDs. The external JobID is a string of alphanumeric characters.
-Y <encryption>	file will be sent encrypted
-Yalg <encryption algorithm>	See the description of the parameter Encryption Algorithm for GUI. Possible values: 3DES, AES.
-Ycin <issuer name>	certificate issuer name for encryption (only for sfs 4)
-Ycsn <serial number>	certificate serial number for encryption (only for sfs 4)
-?	Requests help information.

Optional remote parameters

-rd <file name>	remote file name of the file to be sent. The file must be in the Outbox directory
-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

Examples:

```
createSendJob -d C:\text.txt -s RVS -v test
```

In this example the C:\text.txt file is sent to the station RVS with a virtual name test33. The virtual file name parameter (option -v) is mandatory.

Examples:

```
createSendJob -d C:\text.txt -s RVS -v OFTP_TEST  
-F F -M 80
```

In this example the C:\text.txt file is sent to the station RVS with a virtual name OFTP_TEST, this file is a text file of which each line has a length of 80 characters without CR/LF (-F F -M 80).

```
createSendJob -d C:\TEMP\part.txt -s RVS -v PART  
-S Y -l AUTO
```

In this example the C:\TEMP\part1.txt file is sent to station RVS with a virtual name PART, this file belongs to the serialized group of files with the label (for the whole group) AUTO.

```
createSendJob -d C:\text.txt -s RVS -v test -sfs 4 -C  
-Y -rsr -sif
```

In this example the C:\text.txt file is sent to station RVS with a virtual name test, this file is encrypted with CMS format, compressed, signed and the signature of EERP/NERP is requested.

The message createSendJob exited with return code 0 appears when a send job was successfully created.

If the jobs are successfully terminated (e.g. after an EERP has been received) they will be in status ENDED directory. The FAILED directory holds jobs that could not be successfully processed.

4.5.1 States for Send Jobs and Receive Jobs

This chapter provides an overview of the possible States.

States for Send Jobs and Receive Jobs:

- RESTART (wait after interruption of transmission to send file again)
- SP_PROCESSING (Service provider is processing job)
- SP_ENDED (Service provider has processed job successfully)
- SP_FAILED (Service provider failed during processing of job)
- ENDED (Job has successfully ended)
- ENDED_WITH_JS_ERROR (Job run into error during call of jobstart)
- FATAL_ERROR (Job run into fatal error)
- DELETED (Job got deleted)

States for Send Jobs:

- CREATED (Job was created, but not sent)
- WF_SFID_ANSWER (Sendjob waits for answer to already sended SFID)
- WF_CDT (Sendjob waits for credit renewal)
- WF_EFID_ANSWER (Sendjob waits for answer to already sended EFID)
- WF_EERP (Sendjob waits for EERP)
- WF_EERP_ROUTING (Wait for EERP during Routing)
- SYNCHRONIZE_ERROR (Sendjob run into synchronize error)
- HELD (Sendjob got holded)
- RELEASED (Sendjob got released)
- FAILED_WITH_NERP (NERP received)

States for Receive Jobs:

- RESTART_AFTER_EFNA (Transmission with error; EFNA send; waiting for retransmission)
- RESTART_AFTER_EFPA_FAILURE (File already delivered but EFPA failed; assuming partner will send file again)
- RECEIVING (Receiving file data after EFPA)
- EERP_HELD (File was completely received; EERP is on hold and needs to get released)
- EERP_RELEASED (User released EERP, but it is still not send)
- EERP_DELETED (User deleted EERP; job will stop)
- NERP_RELEASED (User released NERP, but it is still not send)
- NERP_RELEASED_JS (NERP released by JS, but it is still not send)
- FAILED_WITHOUT_NERP (transmission failed; no NERP received)
- ROUTING_HELD_DURING_SHUTDOWN (Couldn't create routing send job, because of shutdown)
- FAILED_WITH_INVALID_EERP_SIGNATUR (Received EERP with invalid signature)
- FAILED_WITH_INVALID_NERP_SIGNATUR (Received NERP with invalid signature)

Note: Please see chapter 15.1 "ODETTE Protocol" for ODETTE protocol sequences (such as EFNA, SFID, EFID, EERP, ...).

4.6 Synchronization of Send Jobs

The Odette file transfer is asynchronous. That means: If you create a send job, a file will be only provided to be sent. The sending of the file will not possible, only after a connection to a partner station has been established.

In automated business with a lot of processes is desirable to react directly, if the sending of a file was not successfully in a certain period of time.

As a solution for this problem rvsEVO offers on client-side a functionality for synchronized file transfer. This program remains active, until the transfer is successfully finished or an error occurs. It is possible to define the number of send attempts or the time in which a file transfer has to be successfully done, or the whole transfer will be count as an error.

Note: A transfer is regarded as having been successfully completed when the ODETTE acknowledgement EERP (End-to-End Response) for this transfer has been received.

The `convertAndSend` program offers you the synchronization of send jobs functionality. It is similar to the `createSendJob` program: it has additionally the options for the synchronized file transfer and for the conversion of an EDI message with the EDI converter WEDICnv (see WEDICnv User Manual).

Usage:

```
convertAndSend -d <filename> -s <receiver sid>
-v <vdsn>
[-t <conversion table>] [-j <start job>] [-F
<format>] [-M <length>] [-S <serialisation> -l
<label>] [-za <attempts>] [-zt <timeout>]
[converterparms] [-h?] [-verbose]
[-Sd <init date>] [-St <init time>] -[Sh]
[-idf <filename2>] [-xid <external job id>]
[-so <originator sid> -rm <record mode>]
```

Required parameters:

-d <filename>	File name of the file to be sent.
-s <receiver sid>	Recipient station ID.
-v <vdsn>	Virtual file name; the length of the file name used for ODETTE transfer must not exceed 26 characters.

Optional send parameters:

-?	Requests help information
-----------	---------------------------

-F <format>	<p>Format of the file to be sent:</p> <ul style="list-style-type: none"> - T=text file; a stream of ASCII characters - U=unstructured (binary); byte stream - V (variable); variable record length - F (fixed); fixed record length. <p>Please see also the parameter -rm <record mode> .</p>
-help	Requests help information
-idf <filename2>	rvsEVO-JobID, which rvsEVO writes in a file (filename) after the file was send successfully (return code = 0).
-l <label>	Name of group of serialized send jobs. User specified (descriptive) label for this job. If you do not specify this parameter the VDSN will be used as a label.
-M <length>	maximal record length for the files in format F or V. Please see also the parameter -rm <record mode> .
-rm <record mode>	<p>In this parameter you indicate the mode of files in format 'F' or 'V'. Possible values:</p> <p>TXT: textfile BIN: binary file</p> <p>Textfile in format 'F': If the record length deviates from the specified one (Parameter -M <length>) the line will be filled with blanks or will be cut up to defined value. Textfile in format 'V': longer records will be cut up to defined value.</p> <p>Binary files: The length of all records is the max. record size except the length of the last one. Binary files in format 'F': If the size of the last record is shorter than the specified one the line will be filled to the defined value.</p>
-S <serialize>	Y (Yes)/N (No). If you set the option -S=Y , the files will be sent in the same order as the send jobs were created. The next job will only be sent, if the previous is completely finished. All send jobs for the serialization must have the same LABEL. In the GUI the VDSN will be used as label. In the command line you can specify your own label.

-Sd <init date>	Date, at which the transmission is to be executed. Format: <yyyy-mm-dd>
-Sh	Create a send entry with status <code>HOLD</code>
-so <originator sid>	station ID of originator
-St <init time>	Time, at which the transmission is to be executed. Format: <hh-mm-ss>
-t <conversion table>	name of your own conversion table: For ASCII(ANSI) - EBCDIC conversion: ASCII-IBM037, ASCII-IBM273, ANSI-IBM037, ANSI-IBM273. For EBCDIC - ASCII(ANSI) conversion: IBM037-ASCII, IBM273-ASCII, IBM037-ANSI, IBM273-ANSI. For using your own conversion table, please read the note for conversion tables on page 85
-verbose	Verbose message output.
-xid <external job id>	Parameter for an external JobID, which can refer to several rvsEVO-JobIDs. The external JobID is a string of alphanumeric characters.
-za <attempts>	count of send attempts for synchronized transmission.
-zt <timeout>	time-out for synchronized transmission in seconds

Optional convert parameters:

-cf	format description for converter step 1
-cf2	format description for converter step 2 (optional)
-ct	stylesheet for converter step 1, optional
-cd	converter direction for converter step 1, default: EDI2XML
-cd2	converter direction for converter step 2, default: XML2EDI.
-cl	log level for converter, default: 0

-cs	line feed behaviour for the EDI-message output; 0 - no LF as segment separator, 1 - LF as segment separator (default).
-cl	indentation for XML output; 0 - no indentation; 1 - indentation on (default).
-ce	encoding for the XML output; default: UTF-8.

Optional remote parameters

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

Examples:

```
convertAndSend -d C:\teil56.txt -s RVS -v TEILE
-za 4
```

In this example the C:\teil56.txt file is sent to the station RVS with a virtual name TEILE, count of send attempts is limited to 4.

```
convertAndSend
-d C:\INTEGRATION\test.txt -s RVS -v TEST
-cf C:\rvsET\system\fmtDesc\fw.kanban.ineas.xml
-ct C:\rvsET\system\stylesheets\ineas2deljit.xslt
-cf2 C:\rvsET\system\fmtDesc\edifact.97.orig.xml
```

In this example the C:\INTEGRATION\test.txt file is sent to the station RVS with a virtual name TEST. The test.txt file was also converted with the EDI converter **WEDIConv** (installed in the directory C:\rvsET\); first from the inhouse format test.txt to the XML format (fw.kanban.ineas.xml) and then to the EDIFACT message (edifact.97.orig.xml). In the first step was used a stylesheet ineqas2deljit.xslt for the special presentation.

Note: If you use convertAndSend to send files, you must pay attention to the fact, that with this feature a new send job after an occurred error can cause a double transmission (e.g. if the file is already transmitted, but the in the -zt parameter defined time out period passed without receiving the EERP. The transmission can be still active independent from the own local station. In this case, the new sending of the file can cause double transmission. The application above rvs® must

be able to handle this particular situation. A possible solution is: the file name must be unique or it must get a unique counter (a counter stamp)).

4.7 Active Panel

This tool gives you the additional details of the status of an active transfer.

In the GUI, you reach the active panel using the symbol transfer.

The following information is available in the active panel:

Parameter	Description
- Neighbour	station-ID of the neighbouring station
- State	Initiator (own station is an active side of the communication process); Responder (own station is the passive side of the communication process)
- Command	Create Session: creation of session Start File: beginning of the transfer TransmitFile: transfer of the file End File: end of the transfer
- Originator	SID of the originator
- Destination	SID of the destination
- Direction	Sending Receiving
- File Name	virtual file name
- Progress	percentage closed content of the transfer
- Line Speed	transfer rate in kB/sec
- Start File	time of the beginning of the file-transfer
- SessionID	is automatically signed for each transfer
- Start Session	Time of the session start
- Line Type	network type

Filter Using the button filter in the area of the window below, it is possible to restrict the display of the transfer status to particular networks.

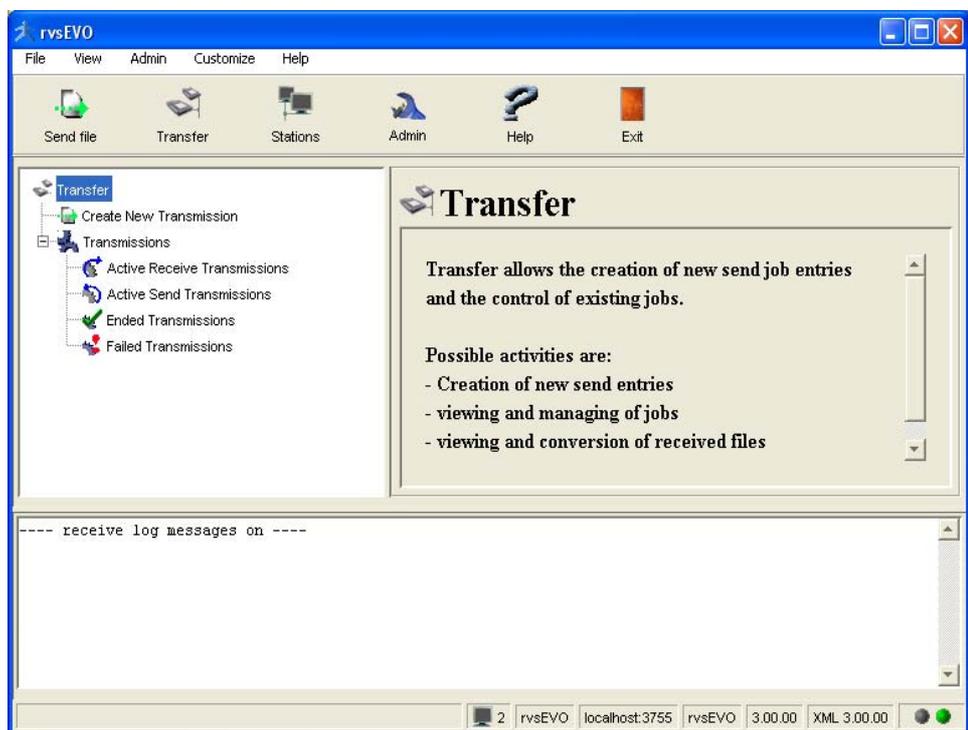
End session In order to cancel the current transfer, mark the corresponding session and press the button Terminate Session.

4.8 Listing of all receive and send jobs

The most important information about the send and receive jobs can be shown via the GUI or via the command line. How to start rvsEVO, please read the chapter 2.6.

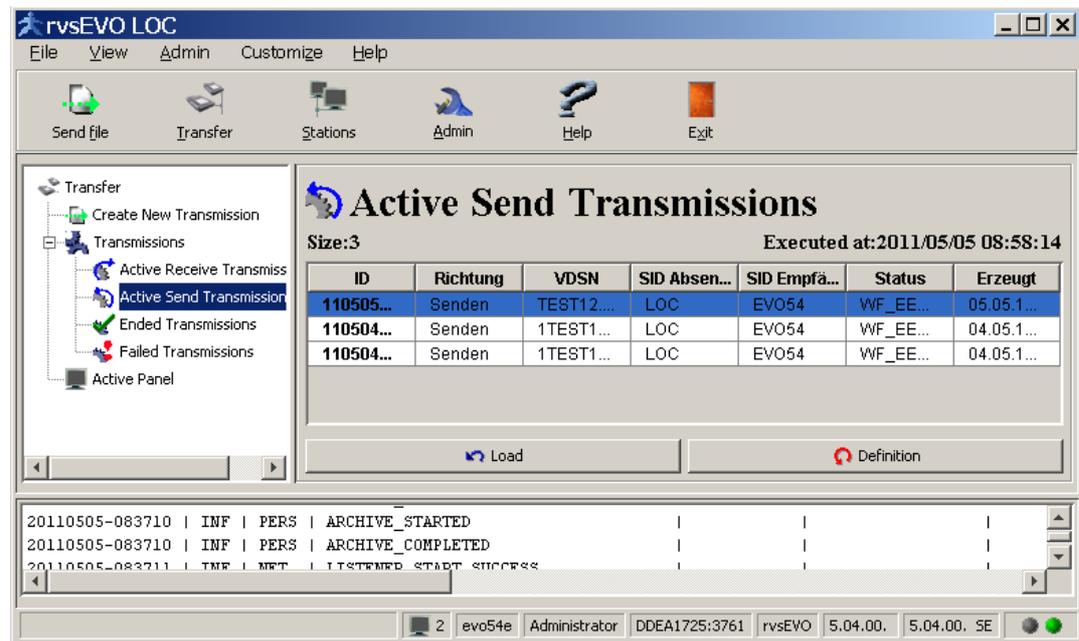
GUI

Select the icon **T**ransfer in the function bar. The Transfer window opens with the sub-entry **Transmissions** in the Transfer tree. Now you can select between the Active Receive/Send Transmissions, Ended Transmission and Failed Transmissions.

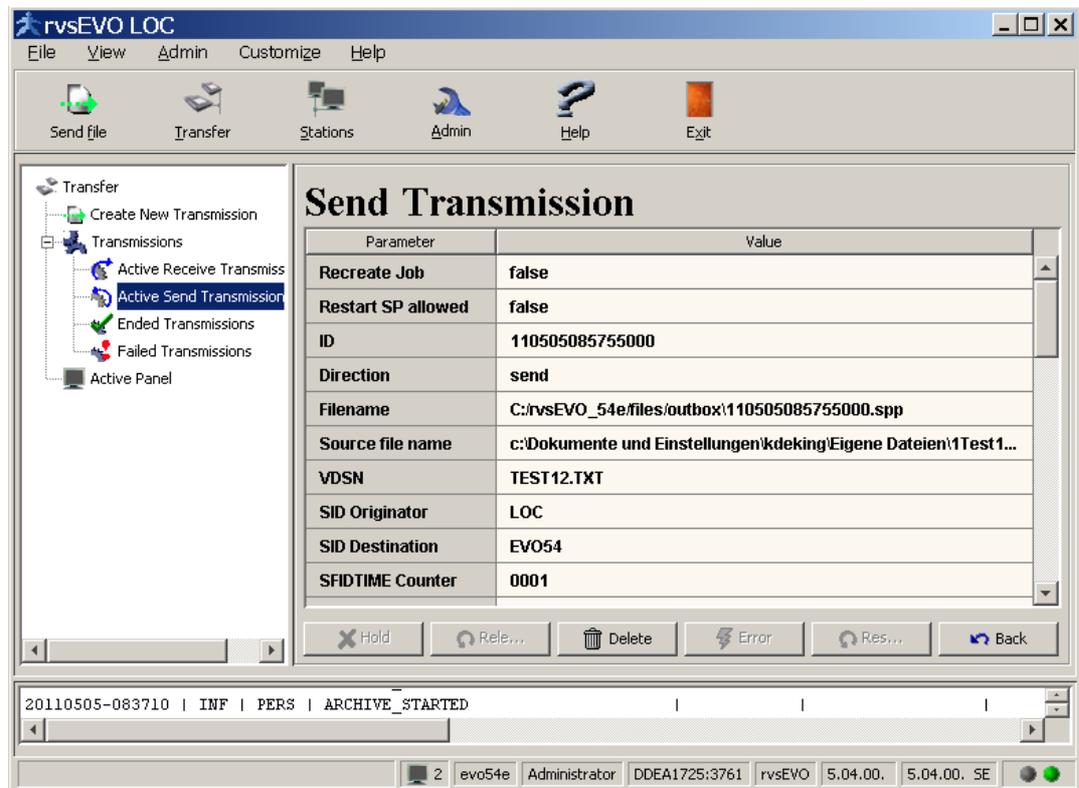


Note: A transfer is regarded as having been successfully completed when the ODETTE acknowledgement EERP (End-to-End Response) for this transfer has been received.

Single-click on selected job sub-ordner gives you an overview of all jobs in this ordner in the right-hand window.



By double-clicking on a job line in the right-hand section of the window you can obtain a detailed view of the relevant job.



Active Receive Transmissions: If you click on the line Active Receive Transmissions and then double-click on a particular job line, you will see all details about the particular receive job in transmission. The buttons on the right-hand side of the window allow several activities:

Release EERP The Button **Release EERP** releases the EERP_OUT in status HOLD (status=EERP_HOLDED).

Release NERP The Button **Release NERP** releases the NERP_OUT in status HOLD (status=EERP_HOLDED).

Delete EERP The Button **Delete EERP** deletes the EERP_OUT.

The same functionalities are available by the program `handleEERP` (see chapter 4.9).

Show With a click on the **Show** button the content of the received file is shown.

Back A click on the button **Back** shows you again the overview of Active Receive Transmissions.

Hint: Operations can also be activated by a right click on a particular job line. A context menu opens and the following functions can be chosen: Release EERP, Delete EERP, Release NERP Restart SP or Delete forced (allows to delete a current transfer. The originator gets the information that the job was canceled and the file is transmitted again with a new jobID but the same timestamp).

Active Send Transmissions: If you click on the line Active Send Transmissions and then double-click on a particular job line, you will see all details about the particular send job in transmission. The buttons on the right-hand side of the window allow several activities:

Hold Use the button marked **Hold** to pause the active job. The same functionality is available by the program `holdJob`.

Release The button **Release** releases a job in status HOLD. The same functionality is available by the program `releaseJob`.

Delete The button **Delete** deletes a job. If the job is active, you must pause it first. The same functionality is available by the program `deleteJob`.

Back A click on the button **Back** shows you again the overview of Active Send Transmissions.

Hint: Operations can also be activated by a right click on a particular job line. A context menu opens and the following functions can be chosen: Hold Transmission, Release Transmission, Delete Transmission, Restart SP or Delete forced (allows to delete an active job without pausing it before).

The following information is for encrypted transmission only:

Jobs which can not be decrypted or encrypted by the Service Provider were changed to the `SP_FAILED` status. Use the button **Error** to display the error number and the description. After bug fixing you can activate the job with a click on button **Restart SP**.

With **Ended Transmissions** the following functionalities are available:

Show File, Show Error and Recreate (recreates a transmission - only for Send Transmissions)

With **Failed Transmissions** the following functionalities are available:

Show Error and Recreate (recreates a transmission after a NERP was received - only for Send Transmissions)

Command Line

`holdJob` Use `holdJob` to pause the job.

Usage:

```
holdJob -n [-su <user> -sp <password> -sh <host:port>]
```

Required parameters:

-n	the jobID of the job to be held
-----------	---------------------------------

Optional parameters:

-help	prints help
-verbose	Verbose message output.
-?	Requests help information.

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

`deleteJob` Use `deleteJob` to delete the job

Usage:

```
deleteJob -n<jobID> [-f] [-su <user> -sp <password> -sh <host:port>]
```

Required parameters:

-n	the jobID of the job to be deleted
-----------	------------------------------------

Optional parameters:

-f	this parameter allows to delete active jobs without pausing the jobs before
-help	prints help
-verbose	Verbose message output.
-?	Requests help information.

Optional remote parameters:

-f	forced
-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

`releaseJob` Use `releaseJob` to release a job in status `HOLD`

Usage:

```
releaseJob -n [-su <user> -sp <password> -sh <host:port>]
```

Required parameters:

-n	the JobID to be released
-----------	--------------------------

Optional parameters:

-help	prints help
-verbose	Verbose message output
-?	Requests help information

Optional remote parameters:

-su <user>	client user for remote login
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

`restartJob` Use `restartJob` to restart a job in `FAILED_WITH_SFNA` status or `SP_FAILED` status.

Usage:

```
restartJob -n [-su <user> -sp <password> -sh
<host:port>]
```

Required parameters:

-n	the JobID to be restarted
-----------	---------------------------

Optional parameters:

-help	prints help
-verbose	Verbose message output
-?	Requests help information

Optional remote parameters:

-su <user>	client user for remote login
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

`getJobList` Use the `getJobList` program to list all jobs.

Usage:

```
getJobList [-a] [-e] [-f] [-verbose]
```

Optional parameters:

-a	Detailed information on jobs currently being processed (in addition display of creation date, SID of neighbour station, and VDSN).
-ac S	additional information on original filename
-e	Information on terminated jobs.
-f	Information on failed jobs.
-h	Requests help information.
-verbose	Verbose message output.
-?	Requests help information.

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client

-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server
------------------------------	--

If you call `getJobList` without parameter the display contains a list of all jobs currently being processed.

Example:

```
getJobList -e
```

Result:

```
*****
job 051208162928000 (RCU): state=ENDED
job 051208162930000 (RCU): state=ENDED
*****
List ended. size=2
*****
```

```
getJobList -a
```

Result:

```
C:\rvsEVO_U2\bin>getJobList -a
*****
Job/Transmission      Status      CreatedAt      SID:UDSN
*****
100823143944000 (RCU) RECEIVING      23.08.10 14:39:42  EU052:TEST1
100823143707000 (SND) WF_CDT      23.08.10 14:39:42  EU052:TEST2
*****
List ended. size=2
*****
getJobList exited with return code 0
C:\rvsEVO_U2\bin>
```

Information on a job entry

`getJob` Use the `getJob` program to retrieve information on a particular send or receive job.

Usage:

```
getJob -n <jobid> [-a] [-verbose]
```

Required parameters:

-n <jobid>	Information on a send or receive job with ID <jobid>.
-------------------------	---

Optional parameters:

-a	All available job information is given.
-help	Requests help information.
-verbose	Verbose message output.

-?	Requests help information.
-----------	----------------------------

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

Example:

```
getJob -n 040329173456000
```

Result: job 040329173456000 (RCV): state: ENDED

4.9 Deleting or releasing EERPs

`handleEERP` Use the program `handleEERP` to delete or release receipts (EERP_OUT). It is also possible to delete or release EERPs via the GUI (Please see the chapter 4.8).

Usage:

```
handleEERP -r|-d <num> [-verbose]
```

Required parameters:

-d <num>	ID of the job for which the EERP is to be deleted.
-n <num>	ID of the job for which the NERP (Negativ-End-to-End-Response) is to be send.
-r <num>	ID of the job for which the EERP is to be released.

Optional parameters:

-help	Requests help information.
-verbose	Verbose message output.
-?	Requests help information.

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

4.10 Create an Info File for an External JobID

`getJobInfoList` Use the program `getJobInfoList` to create an XML-file for an external JobID. The detailed information of all jobs which refer to the external JobID are stored in this file.

Usage:

```
getJobInfoList -xid <external JobID> [-f <filename>]
[-su <user>] -sp <password> -sh <host:port>]
```

Required parameters:

-xid <external JobID>	ID of the external job. The ID can be stated with creating a transmission. (See chapter 4.5 "Sending a file")
------------------------------------	---

Optional parameters:

-f <filename>	filename
----------------------------	----------

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

The XML schema `JobInfoList.xsd` is stored in the directory `$RVS_HOME/system/data/xsd`

JobInfoList: Explanation of the job attributes:

compression	Compressing; possible values: true(yes) or false(no). Standard: false
conversionTable	path to conversion table
creationDate	Date and time (from SFID) of creating a dispatch order in the form DD.MM.YY HH:MM:SS.
deleteAllowed	This parameter gives information as to whether deleting a job is dependent on its status. Possible values: true (Ja) oder false (Nein). Standard: false
direction	Direction; possible values SND (sending) or RCV(receiving).
disposition	Whereabouts: This parameter decides whether the file to be sent remains as it is, or is deleted locally after sending. Possible values: K (Keep): File remains as it is, after sending;
encryption	Encoding; possible values: true(yes) or false(no). Standard: false.
encryptionAlgorithm	Encoding algorithm; possible values: blank (no algorithm), DES_EDE3_CBC (3DES) or AES256_CBC..
encryptionCertificateIssuerName	Publisher name of the certificate. A value is displayed, if a certificate is indicated for use explicit. It is not displayed which certificate is used in fact.
encryptionCertificateSerialNumber	Serial number of the certificate. A value is displayed, if a certificate is indicated for use explicit. It is not displayed which certificate is used in fact.
errorID	If the error type is transmission, these error codes are identical with the ones from ESID, EFNA, SFNA of the Odette-log.
errorText	If the error type is transmission, these error texts are identical with the ones from ESID, EFNA, SFNA of the Odette-log.
errorType	Type of error; possible values: Internal, Transmission, MissingReceipt, Undefined.
externalJobId	JobID for external applikation
fileDescription	File-description. Text comment, possible after the OFTP version 2. If your destination device is not OFTP 2.0, this field is ignored.

filename	The entire path of the file, which is sent from the rvs-EVO-outbox-directory. While sending, the file is copied initially from the original-directory to the outbox-directory and then it is sent.
filenameSrc	The entire path of the original file; this file is copied in the rvsEVO-outbox-directory.
filePos	Number of the read bytes of the dispatch file.
holdAllowed	This parameter gives information as to whether the stopping a job, depending on the status, is allowed. Possible values: true(yes) or false(no).
jobNumber	rvsEVO-JobID
lable	Label for serialization. All the files which have been sent in the same group, should have the same ID (label). If this parameter in case of serialization=Y(true) is not set, VDSN is used as a lable
lastByteRead	Value of the last read byte of a file to be sent in decimal representation as a character string. Standard: -1.
lastByteSend	Value of the last sent byte of a file to be sent, in decimal representation, as a character string. Standard: -1
lastStateChange	Time of the last change in the job status in the format DD.MM.YY HH:MM:SS.
lengthOriginFile	Length of the original file while sending in bytes, before processing through the service provider. This parameter is set to 0 in case of receiving.
oidDest	OdetteID of the target station.
oidNeighbor	OdetteID of the neighboring station.
oidOrig	OdetteID of the sending station (Originator)
recCount	Number of the data records sent (records).
recordFormat	Format of the file to be transferred: T (Text): a consequence of ASCII-characters, F (Fixed): fixed record length, V (variable): variable record length, U (unstructured): binary file.
recordLength	Maximum record length.
releaseAllowed	This parameter gives information as to whether releasing a jobs depending on the status is allowed. Possible values: true(yes) or false(no). Standard: false.

restartPos	restart position, in case of which a new transfer of the file should be started..
scheduleDateTime	Planned time for the job start in the form DD.MM.YY HH:MM:SS
securityFeatureSet	Sicherheitsmerkmale; mögliche Werte: 1 (keine), 2 (ComSecure V1), 3 (ComSecure V2) oder 4 (OFTP 2.0)
sendAttempts	Number of faulty dispatch attempts. If there is a "0", it means that this is a successful file dispatch.
serialisation	This option means that your files are there in a serial order. All the files which should be sent in the same group, should have the same ID (Label). Possible values: true(yes) or false(no). Standard value: false
SFIDTIME Counter	sequential counter from SFID, if several send jobs were created at the same time.
SID	SID depending on the direction: While sending: SID of the recipient; While receiving: SID of the Originator station.
sidDestination	SID of the target station
sidOriginator	SID of source station
sign	Signature; possible values: true(yes) or false(no). Standard: false.
signCertificateIssuerName	Publisher name of the signature A value is displayed, if a certificate is indicated for use explicit. It is not displayed which certificate is used in fact.
signCertificateSerialNumber	Serial number of the signature A value is displayed, if a certificate is indicated for use explicit. It is not displayed which certificate is used in fact.
signEERP	Requesting signed EERP; possible values: true(yes) or false(no). Standard: false.
status	Job status. For possible status values, please see the rvsEVO-user manual (chapter 4.5 dispatching a file).
timeStartFile	Time of the transfer receipt (system time in milliseconds)
transferFileLength	Sending: Size of the file to be sent. Receiving: Size of the file to be sent.

transmittedBytes	Number of the actually transferred bytes
user	Number of the actually transferred bytes
VDSN	Virtual file name for Odette-transfer.
virtualSID	For future applications.
waitTime (TransmissionFailWaitTime)	Time in milliseconds, for a new start of transfer, after a failure (TransmissionFailWaitTime). Should be set in case of an SFNA(Start File Negative Answer) or EFNA (End File Negative Answer)..

Example:

```

<JobInfoList>
  <JobExtendedDetails>
    <JobDetails>
      <externalJobId>5006</externalJobId>
      <JobID>
        <direction>0</direction>
        <jobNumber>091222132218000</jobNumber>
      </JobID>
      <compression>>false</compression>
      <conversionTable></conversionTable>
      <creationDate>22.12.09 13:22:18</creationDate>
      <deleteAllowed>>false</deleteAllowed>
      <disposition>K</disposition>
      <direction>SND</direction>
      <encryption>>false</encryption>
      <encryptionCertificateSerialNumber></encryptionCertificateSerialNumber>
      <encryptionCertificateIssuerName></encryptionCertificateIssuerName>
      <fileDescription></fileDescription>
      <filename>C:\rvsEVO\files\outbox\rvsenv.dat.091222132218000</filename>
      <filenameSrc>C:\rvs\rvsenv.dat</filenameSrc>
      <filePos>789</filePos>
      <holdAllowed>>false</holdAllowed>
      <lastStateChange>22.12.09 13:22:20</lastStateChange>
      <lastByteRead>-1</lastByteRead>
      <lastByteSend>-1</lastByteSend>
      <lengthOriginFile>789</lengthOriginFile>
      <oidDest>OXP</oidDest>
      <oidNeighbor>OXP</oidNeighbor>
      <oidOrig>OGEDASEVO</oidOrig>
      <recordFormat>U</recordFormat>
      <recordLength>0</recordLength>
    </JobDetails>
  </JobExtendedDetails>
</JobInfoList>

```

```
<recCount>0</recCount>
<releaseAllowed>>false</releaseAllowed>
<restartPos>0</restartPos>
<scheduleDateTime></scheduleDateTime>
<sendAttempts>0</sendAttempts>
<serialisation>>false</serialisation>
<SID>XP</SID>
<sidOriginator>LOC</sidOriginator>
<sidDestination>XP</sidDestination>
<signCertificateSerialNumber></signCertificateSerialNumber>
<signCertificateIssuerName></signCertificateIssuerName>
<status>ENDED</status>
<timeStartFile>1261484539412</timeStartFile>
<transmittedBytes>789</transmittedBytes>
<transferFileLength>789</transferFileLength>
<user></user>
<VDSN>WEIH6644</VDSN>
<virtualSID></virtualSID>
<waitTime>0</waitTime>
<jobSecurityAndSign>
  <securityFeatureSet>1</securityFeatureSet>
  <encryptionAlgorithm></encryptionAlgorithm>
  <sign>>false</sign>
  <signEERP>>false</signEERP>
</jobSecurityAndSign>
</JobDetails>
```

4.11 Archiving the entries of processed send or receive jobs in the revision log

During the archiving process terminated and failed jobs are deleted from the rvsEVO database and the data is written into the RevisionLog.xml file in the \$RVS_HOME\archive directory. The name of revision file consists of „RevisionLog.“, timestamp and a sequential counter initiating with „0000“. (Format: RevisionLog.xml.yyMMddHHmss_cccc). Use the rvsEVO environment parameters **MaxRevisionLogSize** and **MaxRevisionLogCount** to indicate maximal size and maximal number of revision file. By default every 24 hours any jobs older than 7 days will be saved. This configuration can be set up via the parameters **CleanupDays**, **CleanupInterval** and **CleanupTime** (please see chapter 3.1.1 "rvsEVO Environment").

You can also launch the archiving via the menu bar (Admin -> Archiving) under specification of the age of the send and receive jobs.

archiveJobs

As an alternative you can also start this function by launching the script archiveJobs at the command prompt.

Usage:

```
archiveJobs -d <date> [-r] [-verbose] [-help] [-?]
```

Required parameter:

-d <date>	This option needs a date in the following form: yyMMddHHmss or yyMMdd. The jobs, which are older than a date will be archived.
------------------------	--

Optional parameters:

-r	remote, availables the archiving, during rvsEVO is running.
-help	Displays a description of the current command.
-verbose	Verbose message output.
-?	Requests help information.

4.12 Delivering a certificate

With this tool, a certain certificate for file encryption (from \$RVS_HOME\system\data\keystore.p12) can be delivered to a partner station and imported to the key management. Virtual file name for ODETTE transfer is ODETTE_CERTIFICATE_DELIVER.

Syntax:

```
deliverCertificate -s <stationID>
[-i <keyIndex>]-r <stationID>
[-su <user> -sp <password> -sh <host:port>] [-help]
[-?] [-verbose|-v]
```

Required parameters:

-r <stationID>	StationID of the partner-station, to which the certificate should be sent.
-s <stationID>	StationID of the station, whose certificate should be sent

Optional parameters:

-i <keyIndex>	Index of the certificate, which should be sent, if several certificates are available for a station. If this parameter is not set, all the available certificates are sent.
-verbose	Verbose message output.
-?	Requests help information.

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

4.13 Requesting a Certificate

With this tool, a certificate for file encryption (from \$RVS_HOME\system\data\keystore.p12) can be requested from a partner station. Virtual file name for ODETTE transfer is ODETTE_CERTIFICATE_REQUEST.

Syntax:

```
requestCertificate -s <stationID>
[-su <user> -sp <password> -sh <host:port>] [-help]
[-?] [-verbose|-v]
```

Required parameters:

-s <stationID>	StationID of the station, from which the certificate should be requested.
-----------------------------	---

Optional parameters:

-help	Displays a description of the current command.
-verbose	Verbose message output.
-?	Requests help information.

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

4.14 Replacing a certificate

With this tool, a certificate for file encryption (from \$RVS_HOME\system\data\keystore.p12) can be sent to a partner station and replace an already existing certificate in this station. Virtual file name for ODETTE transfer is ODETTE_CERTIFICATE_REPLACE.

Syntax:

```
replaceCertificate -s <stationID>
[-i <keyIndex>]-r <stationID>
[-su <user> -sp <password> -sh <host:port>] [-help]
[-?] [-verbose|-v]
```

Required parameters:

-s <stationID>	StationID of the individual station, whose certificate should be sent and replaced.
-r <stationID>	StationID of the partner-station, to which the separate certificate (certificates) are sent. The old (belonging to the individual station in case of the partner) certificate (certificates) are replaced and are cancelled.

Optional parameters:

-i <keyIndex>	Index of the certificate, which should be sent and replaced. If this parameter is not sent, all the available certificates are sent. Finally, in case of the partner, all the old certificates are replaced by new ones.
-help	Displays a description of the current command.
-verbose	Verbose message output.

-?	Requests help information.
-----------	----------------------------

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

4.15 Display a certificate list

The tool `getCertificateList` shows the certificate list of a special stationID.

Syntax:

```
getCertificateList -sid <stationID> [-help] [-?]
```

Required parameters:

-sid <stationID>	StationID of which the certificates are to be shown
-------------------------------	---

Optional parameters:

-help	Displays a description of the current command.
-?	Requests help information..

4.16 Open the keystore file

The program `startKeyMgn` opens the keystore file. Please read Chapter 6.4 for more information about the keystore file.

Syntax:

```
startKeyMgn [-h] [-?]
```

Optional Parameter:

-h	Displays a description of the current command
-?	Requests help information.

4.17 Import a CRL

With this tool, a CRL (Certificate Revocation List) can be imported for a certain partner station.

Preconditions: The certificate of the partner station must be issued by a CA (Certification Authority). The path where to store the CRL must be written in the parameter location of the PKI configuration file: `$RVS_HOME/conf/PkiParameter.xml`. The tool `importCRL` downloads the CRL (the internet address is stored in the certificate) and saves the CRL. (For more information please see chapter 11.2.2 "PKI configuration file").

Syntax:

```
importCRL -s <stationID> [-i <keyIndex>]
[-help] [-?] [-verbose|-v]
```

Required parameters:

-s <stationID>	StationID of the partner-station, whose CRL should be imported.
-----------------------------	---

Optional parameters:

-i <keyIndex>	Index of the certificate, if several certificates are available for a station. If this parameter is not set, a CRL is imported for all the available certificates of the partner station.
-help	Displays a description of the current command.
-verbose	Verbose message output.
-?	Requests help information.

Hint: Set your Proxy alignments in `$RVS_HOME/conf/rvs-system.properties` file with the following definitions:

- `http.proxyHost`: IP address or hostname of Proxy-Server
- `http.proxyPort`: Port of Proxy-Server
- `http.nonProxyHosts`: Hosts which should be connected directly and not through the proxy server
- `http.proxyUser`: Username
- `http.proxyPassword`: Password of user

4.18 Import a TSL

With this tool, a TSL (Trust Service Status List) can be imported into the keystore file (keystore).

Syntax:

```
importTSL -f <TSL filename> -k <keystore filename>
[-help] [-?] [-verbose|-v]
```

Required parameters:

-f <TSL Filename>	path of the file which includes the TSL.
-k <keystore filename>	path of the keystore file

Optional parameters:

-help	Displays a description of the current command.
-verbose	Verbose message output.
-?	Requests help information.

4.19 Terminate a Session

With this tool, an active session can be terminated.

Syntax:

```
terminateSession -s <session_id>
[-su <user> -sp <password> -sh <host:port>]
[-help] [-?] [-verbose|-v]
```

Required parameters:

-s <session_id>	ID of the session which should be terminated.
------------------------------	---

Optional parameters:

-help	Displays a description of the current command.
-verbose	Verbose message output.
-?	Requests help information.

4.20 Create and send a journal

With the tool `sendJournal` you can create a journal and send this to `CentralJournal`.

Syntax:

```
sendJournal -d <date> -f <file name> [-su <user> -sp
<password>] [-sh <host:port>] [-verbose] [-help] [-?]
```

Required parameters:

-d <date>	Date in format yyMMddHHmm
-f <filename>	Name of request file from CentralJournal. Default: ZJREQUEST

Optional parameters:

-help	Displays a description of the current command
-?	Requests help information.
-verbose (-v)	Verbose message output

Optional remote parameters:

-su <user>	client user for remote login.
-sp <password>	password of the client
-sh <host:port>	host is the IP address or hostname of rvsEVO-Server; port is the port of rvsEVO-Server

For detailed information please see the user manual of CentralJournal.

4.21 Doing character set conversion

With the tool `convertFile` you are able to do character set conversions for files.

Required parameters:

-c <conversion reference>	The possible values are described in Conversion table parameter in the table "JobFilter elements" on page 79.
-s <filename>	path and name of source file
-d <filename>	path and name of converted file

Optional parameter:

-?	Requests help information.
-----------	----------------------------

4.22 Command line tools for internal use

The following programs are only for internal use:

```
login.bat
rvsEVOService.exe
setclientcp.bat
setcp.bat
userManagerClient.bat
```

5 Backing Up and Recovering rvsEVO Data

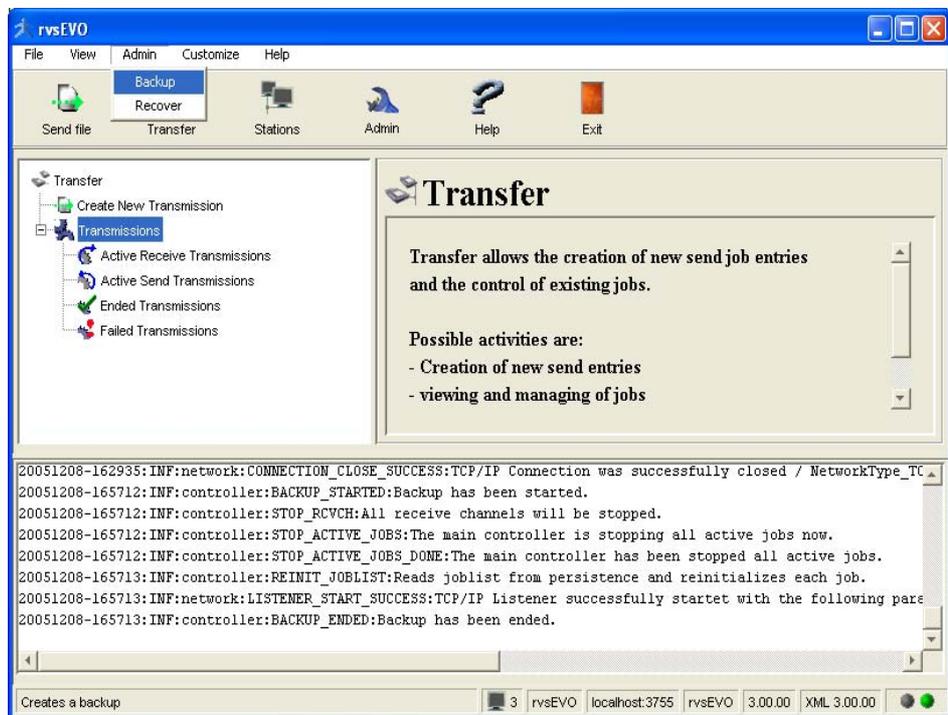
rvsEVO lets you back up all relevant data and recover them if necessary. This is particularly important when an error has occurred in rvsEVO and the user wishes to revert to the old status prior to the error.

5.1 Backup

The backup function should only be launched if the rvsServer is started.

All active transmissions will be terminated before the backup. After the backup process is finished the canceled transmissions will be activated again. A restart of rvsEVO server is not required.

To perform the backup, choose `Admin/Backup` menu item in the rvsEVO GUI window.



As an alternative you can also start this function by launching the script `createBackup` at the command prompt.

Syntax:

```
createBackup [-d <dir>] [-verbose] [-help] [-?]
```

All parameters are optional:

-d <dir>	Specifies the backup directory; without this option the backup data will be written to the \$RVS_HOME/archive directory.
-verbose	Displays detailed messages.
-help	Requests help information.
-?	Requests help information.

Note: You can use the BackupOnStartup parameter to specify an automatic backup to be performed each time you start rvsEVO. Set this parameter in the graphical user interface (Admin window) or in the rvsConfig.xml configuration file. Default value is Y (Yes).

5.1.1 What is backed up?

Backup covers the following files or directories:

- The entire \$RVS_HOME/conf directory containing the rvsEVO configuration files.
- backup of job data from rvsEVO database in the \$RVS_HOME/jobs directory.
- files from the \$RVS_HOME/system/data directory.

The backup data is written to a <BackupTime>.jar file. BackupTime is an automatically assigned file name, generated according to the YYMMDDHHMMSS format and featuring an additional 3-digit counter.

Example: The 051010112417000.jar is the backup file generated on 2005-10-10 at 11:24:17. At this time, there were no other backup files, resulting in the additional counter value of 000.

5.1.2 Redo Log

Starting at the backup time, any dynamic data is logged in a continuous log (Redo Log).

“Dynamic data” refers to information on send and receive jobs. This data is written in order to be able to recover incomplete transmission jobs at a later time.

The Redo Log is assigned the same time stamp as the pertaining backup file, bearing the Redo_ prefix and the .log extension and can thus be clearly assigned to a backup. Like the backup file, this file is written to the \$RVS_HOME/archive directory.

Example:

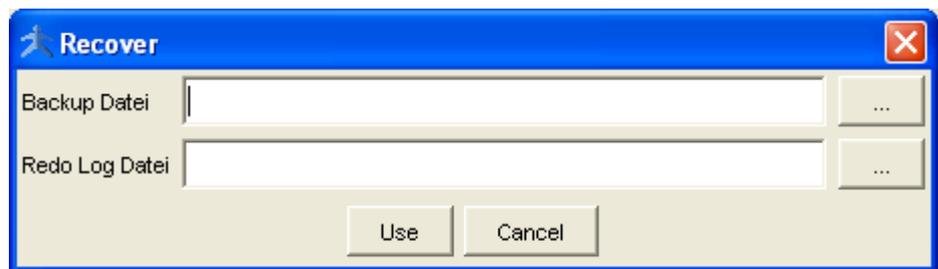
In addition to the `051010112417000.jar` backup file the pertaining `Redo_051010112417000.log` Redo Log file was generated.

A pertaining Redo Log file is generated as soon as a new backup was made. The `.log` extensions of all existing Redo Log files will be replaced by `.old`.

The Redo Log file is a text file. Each line comprises a `<Job>` XML element with all relevant job information such as ID, FileName, VDSN, SID.

5.2 Recovering the rvsEVO data

To recover rvsEVO data, choose the `Admin/Recovery` menu item in the rvsEVO GUI window. In the dialog that opens, you can specify the names for the backup file and the Redo Log. Backup file and Redo Log will be searched for in the `$RVS_HOME/archive` directory when no path has been specified.



As an alternative you can start this function by typing `$RVS_HOME/bin/doRecover.bat` at the command prompt.

Syntax:

```
doRecover -b <dir> -r <name> [-d <name>]
[-help] [-?]
```

-d <dir>	Mandatory parameters: Name of the backup file; you can specify the complete path with directory or just the file name without directory. In the latter case the backup file is expected in the <code>\$RVS_HOME/archive</code> directory.
-r	Name of Redo Log file. You can specify the complete path with directory or just the file name without directory. In the latter case the Redo Log file is expected in the <code>\$RVS_HOME/archive</code> directory.

-verbose	Displays detailed messages.
-help	Requests help information.
-?	Requests help information.

Note: Any send/receive processes as well as the Service Provider (encryption/compression) have to be terminated prior to recovery start.

Redo Log recovery is logged in the `$RVS_HOME/log/monlog.log` file.

Please restart rvsEVO after the recover function.

6 Encrypted transmission with rvsEVO

The present chapter describes the basics of encrypted transmission and key administration with rvsEVO.

6.1 Introduction: basics

rvsEVO offers two encryption formats: ComSecure and CMS.

Com-Secure is an appropriate format, which is used for rvs[®] portable, too. CMS (Cryptographic Message Syntax) is an Internet Standard (rfc 2630) for the syntax of an encrypted message. This standard is in the OFTP version 2 implemented.

encryption formats

The encryption in rvsEVO (for both formats: ComSecure and CMS) combines the benefits of symmetrical and asymmetrical techniques: the high speed of the symmetrical and the security level of the asymmetrical technique. rvsEVO uses the following techniques:

- **3DES** for ComSecure as symmetrical technique (length: 3x56 bits = 168 bits),
- **3DES** and **AES** for CMS
- **RSA** as asymmetrical technique for ComSecure and CMS (length: 768 to 2048 bits),

Electronic signature

For increased security, the encryption component uses an electronic signature. The signature ensures that data do not undergo any unnoticed changes during transmission.

OFTP version 2 enables the electronic signature also for the End-To-End-Response (EERP) and for the Negative-End-to-End-Response (NERP).

Note: In rvsEVO with OFTP version 2 features for the encryption on the session level the TLS (Transport Layer Security) will be used. Please read the chapter 3.2 "Customizing the station configuration" and the chapter 3.2.6 "How to configure a TLS receiver?", for information how to configure rvsEVO for TLS. The principles (public keys, private key) of the key management are the same for CMS, TLS and ComSecure.

6.2 System requirements

Hint: In rvsEVO 5.0 the following steps for installing JPS are done by the installation procedure (see the chapter 2).

Please note the following system requirements if using offline encryption and compression:

Using Unlimited Strength Jurisdiction Policy Files:

 rvsEVO uses the JCE (Java Cryptography Extension) of Sun Microsystems, Inc to implement the cryptographic features. Due to import control

restrictions of some countries, the JCE jurisdiction policy files shipped with the Java 2 SDK, v 1.5 allow "strong" but limited cryptography to be used. An "unlimited strength" version of these files indicating no restrictions on cryptographic strengths is available for those living in eligible countries (which is most countries). You can download this version and replace the strong cryptography versions supplied with the Java 2 SDK, v 1.5 with the unlimited ones.

Therefore you have to carry out the following steps:

- Download the JCE Unlimited Strength Jurisdiction Policy Files from this site:

http://java.sun.com/javase/downloads/index_jdk5.jsp

- Install the JCE Unlimited Strength Jurisdiction Policy Files
- Extract the contents of the downloaded archive file and copy it into the directory `$JAVA_HOME/jre/lib/security`.

Hint: If you use Java 2 SDK, v1.5 from IBM you need to install the unrestricted policy files from IBM. Which could be downloaded using the following URL.

<http://www-128.ibm.com/developerworks/java/jdk/security/50/>

6.3 Principle and sequence of rvsEVO encryption

The principles of the encryption, which will be explained in the following part, are the same for the all encryption formats, mentioned in this chapter.

Each participant in encrypted communication locally creates a key pair, comprising the **public key** and the **private key**.

Distributing the public key / safely storing the private key

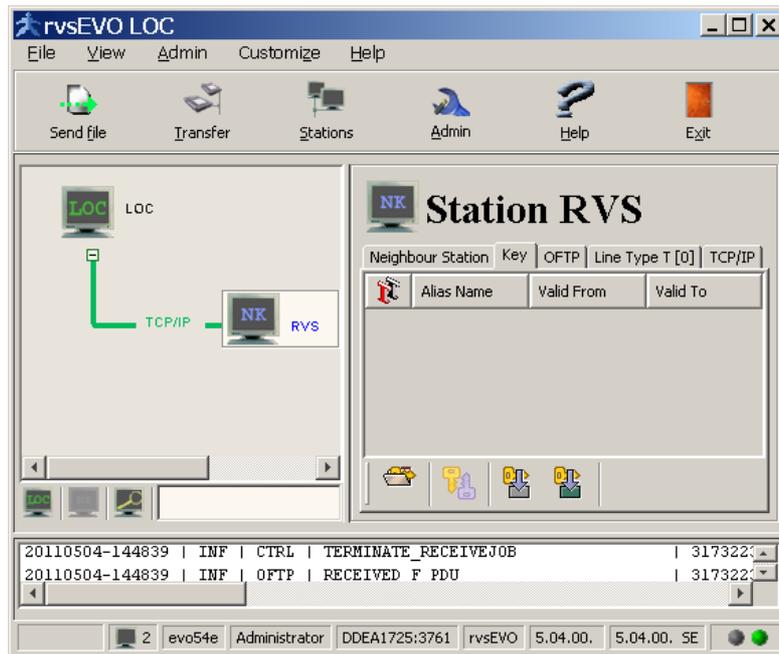
He provides the public key to each partner he expects files from. This allows the data to be exactly encrypted for the partner who sent this public key. You can safely distribute the public key since this key alone is not sufficient for decryption.

Each participant keeps his private key and stores it safely.

Three keys are required for decryption (the own key pair and the partner's public key). It will no longer be possible to decrypt files sent by the partner if one of the three necessary keys is lost.

For the key management you should open the GUI and choose the tab **Key** on the right side of the stations window.

The following window appears:



The following functions are available (symbols for some of them are visible as icons on the bottom of the key management window; and others are available with the context menu):

- Get Key and Certificate List
- Generate Key Pair
- Import Certificate
- Export Certificate (not available via Remote GUI)
- Export Certificate to ComSecure (not available via Remote GUI)
- Import ComSecure Public Key (not available via Remote GUI)
- Create a new Keystore
- Delete key
- Create Certification Request

In the next chapters we will describe the functions of the key management for the file encryption.

The following steps are necessary to perform, if you want to send encrypted files with rvsEVO:

- create an own key pair for encryption of files
- export an own public key in form of X.509 certificate
- send your own public key to the partner with whom you should exchange encrypted files (by E-Mail or rvsEVO)

- The partner has to import your X.509 certificate (public key) and send to you his X.509 certificate.
- You have to import the partners' X.509 certificate into your TrustManager key file.
- now try to activate the partner station and when OK send a test file with option Encryption Y.

How to send a file, please read in chapter 4.5 "Sending a file"

6.4 How do I create an own key pair?

Click on the button **Generate Key Pair**



in the key management window.

Select RSA as Key Algorithm and the default of 1024 as Key Size in the **Generate Key Pair** window.

Select SHA1withRSA as Signature Algorithm in the dialog that appears next. Validity (days) indicates the number of days the keys are valid.

The Common Name parameter applies in case of a connection to an existing PKI (Public Key Infrastructure) and if this parameter is required for the LDAP structure. This parameter is mandatory.

All other data concerns your organization. rvsEVO does not stipulate how to complete the fields.

Hint: In `$RVS_HOME/conf/rvs-system.properties` file via the definitions `rvs_evo.certificate.odette-id.include=LOC` (SID for ODETTE-ID) and `rvs_evo.certificate.dns-name.include=oftp2.t-systems.com` (DNS name) you have the possibility to insert the ODETTE-ID and DSN name into the certificate. This values are displayd in detailed information of the exported certificate in field Subject Alternativ Names.

Administration of own (private and public) and of partner keys (only public) occurs in a keystore file.

You must set the path to the keystore file to be used in the `$RVS_HOME/conf/cryptoParameter.xml` configuration file (XML element: `keyStoreParameter`; subelement: `fileName`). The default is: `$RVS_HOME/system/data/keystore.p12`.

Note: The extension `.p12` indicates that this keystore file is in the PKCS #12 format.

Certificate UsageDefinition

If you hold several own key pairs, please edit xml file \$RVS_HOME/conf/CertificateUsageDefinition for assigning activities to the private keys . The following activities can be chosen:

- F-ERP-SIGN => EERP/NERP signature (OFTP2)
- F-SIGN => file signature (OFTP2)
- F-COMSECURE-SIGN => file signature (ComSecure)

1st example:

Key with serial number 4D623B3B is to be used for file signature (OFTP2) by locale station LOCALSTATION.

```
<entry key="LOCALSTATION.F-SIGN.SERIALNUMBER">4D623B3B</entry>
```

2nd example:

Key with serial number 4D623B3B is to be used for EERP/NERP signature by virtual station VIRTUALSTATION.

```
<entry key="VIRTUALSTATION.F-ERP-SIGN.SERIALNUMBER">4D623B3B</entry>
```

6.5 How to import and export a certificate

rvsEVO enables to import trusted certificates from partners into the key management and to export already imported certificates from the key management into a file.

To import an X.509 certificate into a key management, press on the **Import Certificate** button:



In the next dialog **Import Trusted Certificate** you can select a file, which should be imported. Normally the certificate file has an ending `.cer`. After the acknowledgement with **OK** the certificate will be visible with the following symbol in the key management.

In the another direction it is also possible to export an certificate from the key management into a file. A right click on the marked line with the certificate symbol



opens a context menu with the function **Export Certificate**. It will be exported to the file with an ending `.cer`.

6.6 How to import and export ComSecure public keys

rvsEVO supports an appropriate format for the encryption, so it is necessary to convert a ComSecure public key into a X.509 certificate before importing it into the key management. X.509 is a standard format for digital certificates; certificate files end normally with `.cer`.

For the function you use the button: **Import ComSecure Public Key**



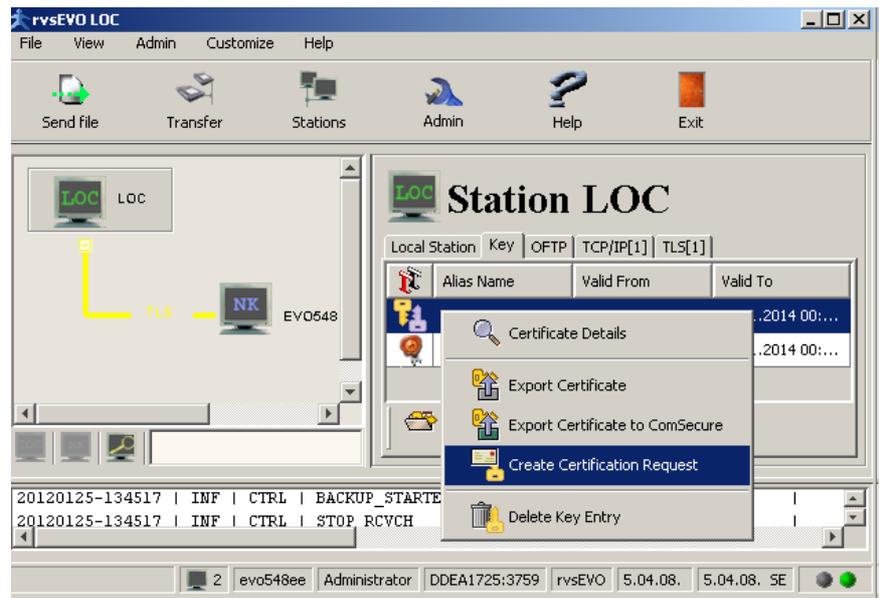
This function enables the import of a public key in a ComSecure format (normally a file with ending `pub`) as a X.509 certificate.

The second function **Export Certificate to ComSecure** supports the another direction: a X.509 certificate (`*.cer`) from the key management will be exported as a ComSecure public key (`*.pub`). A right click on the marked line with the certificate symbol opens a context menu with the function **Export Certificate to ComSecure**.

6.7 How to create a CSR (Certificate Signing Request) and import the CA certificates

This chapter describes the procedure for working with CA (Certificate Authority) certificates:

- Generate CSR
- Generate a key pair as described in chapter 6.4 "How do I create an own key pair?"
 - A right click on your key pair opens a context menu with the possibility to generate a Certification Request. Define name and directory where the request should be saved. By default the file is generated in PKCS#10 format.



Example of Certification Request:

```

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBWDCBwgIBADAbMRkwFwYDVQODDBB1dm81NDhkLW9yaS1maWx1MIGfMAOGCSqGS
A4GNADCBiQKBgQCczTte+HyTxy/1CCjtXWuADqO7H/LDDKfH7GZANKS2WqJ9+Ozw7:
Tc1k5JS3Re8L3q6xGBzDxucD7N6Qp3ikincYTxaE92yuy+rbK43YJJRLdgixKAu+a
YY/a2UjK0eGY+oVrMwJL1u+pbIs9VAsTmWIDAQAABMAOGCSqGSIb3DQEBBQUAA4GBA
p426CHVOREM6QzVWf2qNe4OR/mxhuKT118kP1xx9oG4K7pkPbHiMuT1SdNmzb/GDP:
tisBm185o+fCDOKaOpIKidjC9aTact16YtSoRdc4v7RIw5t1zprW+ANhwOwVwZMRg:
/pK79Mbm
-----END NEW CERTIFICATE REQUEST-----

```

- Send the request to the TrustCenter (CA). With Odette CA you can make your application online: <https://www.odetteca.com>. Please see <https://forum.odette.org/repository/odette-ca-help.pdf> for more information.

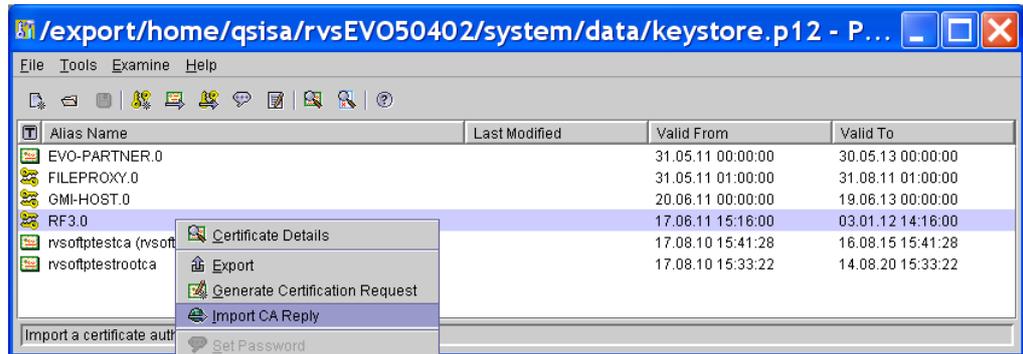
You receive your server certificate (your public key signed by CA), a CA root- and a CA certificate if all entries of your certificate order are correct.

Import of own certificates

You should import the certificates in a specific sequential arrangement into `$RVS_HOME/system/data/keystore.p12` file: at first the CA root-, thereafter the CA certificate and at last the server certificate.

- Launch the `startKeyMgn` program (see also chapter 4.16 "Open the keystore file"). The Portecle tool will open the `$RVS_HOME/system/data/keystore.p12` keystore file.
- Import the CA-Root certificate via function **Tools -> Import Trusted Certificate**
- Import the CA certificate via function **Tools -> Import Trusted Certificate**
- At last import the server certificate via function **Import CA Reply**. Highlight your key pair and after a right click the context menu offers

the option **Import CA Reply**.



You cannot see the CA certificates in the user interface of `keystore.p12` file but in the certificate details of your key pair (Certificate 1 of 3).

(Example: Key pair of RF3 -> right click -> Certificate Details)

- Save and close your keystore

At least you must send your CA certificates to your partner and import whose certificates.

Import certificates of partner

The procedure depends on the setting of **Certificate Validation Type** parameter. If **Certificate Validation Type** = `CERT_PATH`, please follow the procedure given below:

- Launch the `startKeyMgn` program (see also chapter 4.16 "Open the keystore file").
- Import the CA root certificate first, thereafter the CA certificate and at last the server certificate of your partner via function **Tools** -> **Import Trusted Certificate** in `keystore.p12` file.
- Save and close your keystore

With an other setting of **Certificate Validation Type** parameter you have to import the server certificate only. In this case, please follow the procedure given in chapter 6.5 "How to import and export a certificate".

Hint: For more information about certificate validation, please see chapter 11 "PKI Bindung".

6.8 How to receive files without decryption

Files, encrypted with ComSecure can be received decrypted and decompressed or encrypted and compressed.

This function is controlled by the properties file `RVS_HOME/conf/rvs-system.properties` with the following definition:

```
rvs_evo.servicepro-  
vider.receive_job.process_comsecure=false  
false = off (no decryption/no decompression)  
true = default / on (decryption / decpmpression)
```

After receipt not decrypted / decompressed files are stored in the directory `$RVS_HOME/files/inbox`.

6.9 How to delete a key entry

A right click on the key or certificate line opens a context menu with the function **Delete a Key Entry**. This function deletes a key pair and a appropriate certificate from the key management.

7 rvs® OFTP Proxy

This chapter describes the configuration and the functionality of rvs® OFTP Proxy implementation in rvsEVO.

7.1 Basics

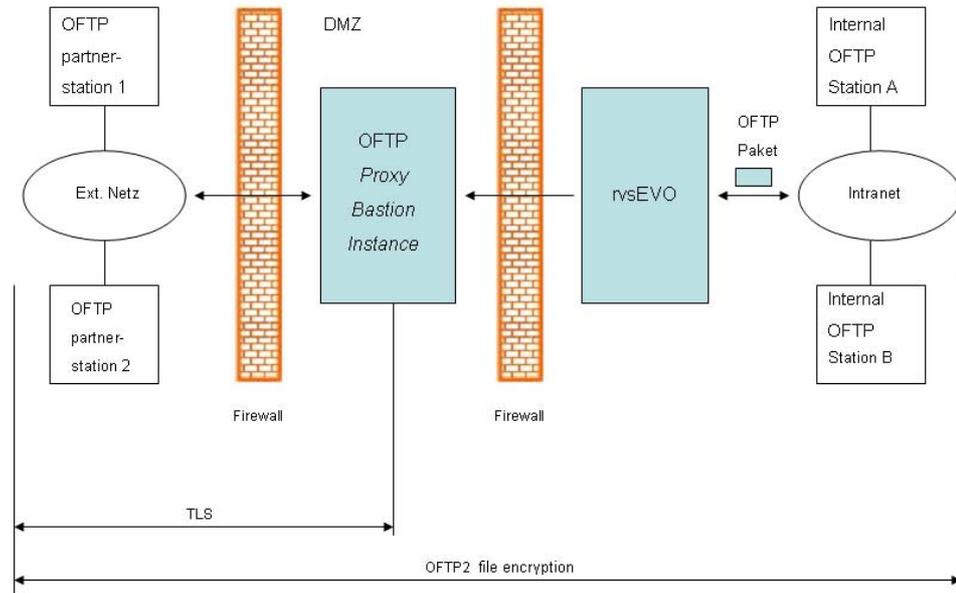
Internet Internet is increasingly used also for transferring the commercial and business-critical data of the content. Such advantages are deciding factors for this, such as its wide distribution, enormous available bandwidths and the reasonable transfer costs.

These advantages are accompanied by the risks to be considered seriously: in internet all data are susceptible to the attacks of their confidentiality and integrity. Their authentication is also not ensured without further ado. This may result into that the data is repudiated by its obvious sender. Such risk is encountered with suitable protocols, such as TLS and HTTPS. For CAD data and also for EDI and any other data, OFTP version 2 has been standardized as a secured internet-protocol.

Any further risk arises through malware, which is spread widely and automatically in internet. Virus, worms, Trojan and other harmful programs can damage the IT-infrastructure of the company considerably. In order to minimize this risk, network areas are developed which are separated from the corporate networks through the firewalls (also referred to as demilitarized Zone - DMZ). For example, http-proxies are set in DMZ, which transport the user data without allowing direct connections between internet and intranet. rvs® OFTP Proxy offers the similar solution, which can be implemented in DMZ in order to transfer the controlled data of OFTP protocol version 1 and 2 between intranet and internet.

7.2 rvs® OFTP Proxy Architecture

The rvs® OFTP Proxy is run with two instances: bastion instance and connection instance, which is integrated in rvsEVO. Following figure should illustrate these facts:



The connections are received from the external network using bastion instance. These external connections are forwarded to rvsEVO.

The other way round rvsEVO uses the bastion instance for establishing a connection to the OFTP station in the external network.

Communication between rvsEVO and bastion instance is executed exclusively through the connections built by rvsEVO. Communication is restricted to one port, the configurable RMI-port of the bastion instance.

Proxy Listener For being accessible for the external stations one or more Proxy listener are launched by rvsEVO. rvsEVO uses this Proxy listener to establish a connection to the partner station.

Altogether, all functional configurations, keys and certificates required for TLS communication are stored exclusively in rvsEVO.

The Bastion Instance is installed with the program rvs[®] OFTP Proxy. You can download rvs[®] OFTP Proxy using the following website:

<https://servicenet.t-systems.de/tsi/de/267072/Startseite/Business-Integration/rvs>

If it is not possible for you, please contact your sales partner:

phone from Germany: 0800 664 77 45
 phone from other countries: +375 606 19 902
 E-mail: rvs-service@t-systems.com).

We will send you the software also on a DVD.

Carry out the installation of the Bastion Instance, as explained in „User Manual rvs[®] OFTP Proxy“.

7.3 Configure Bastion Instance in rvsEVO

In order to be able to create a connection with the Bastion Instance, it is required to store the parameter in rvsEVO and to configure a proxy listener. A list of already created Bastion Instances can be displayed to you on the graphical user interface (Admin --> Parameter --> Proxy).

In order to add a Bastion Instance, right-click on the proxy and subsequently select `add proxy`. The parameters to be entered after that have already been stored during the installation of Bastion Instance. If a change in these values is required, e.g. after changing the hardware, it is to be considered, that these parameters are customised in the start script `$RVS_HOME/bin/bastion.cmd` (`$RVS_HOME/bin/bastion.sh`) as well as in case of corresponding Bastion Instance in rvsEVO.

You can find an explanation of the individual parameters in the following table.

Parameter

Parameter	Description
Instance	serial number, which is automatically assigned by rvsEVO for each Bastion Instance.
Command	directory, in which the Bastion Instance of rvs® OFTP proxy has been installed.
Server	server name or IP-address of the server, on which the Bastion Instance has been installed.
Service	Name of the Bastion Service. Standard: <code>service</code>
Port	Port of the Bastion Instance for the RMI-communication, on which a listener is started.

Configure Proxy Listener A receiver/listener must be defined for each Bastion Instance. For this purpose, please proceed, as described in chapter 3.2.2 "Configuring a local station"

You also have to add a neighbour station with Proxy TCP/IP or Proxy TLS network, for the communication via rvs® OFTP Proxy. Please read chapter 3.2.3 "Setting up of a neighbour station" for detailed information.

keep connection **Hint:** You can configure a time interval for sending heartbeats to prevent the interrupt of connection if there is no transmission. This functionality is to be set up in parameter `listener.controller-thread.sleep-time` and `listener.controller-thread.keep-alive-counter` in `$RVSPROXY_HOME\conf\RemoteListenerProperties.properties` file of rvs® OFTP Proxy program.

Use the `listener.controller-thread.sleep-time` parameter to set up the sleep time (in milliseconds) of the controller thread before it checks next time for status of listener.

Use the `listener.controller-thread.keep-alive-counter` parameter to define the number of checkups before the next heartbeat is sent.

8 File Service Module

8.1 Basics

The internet is increasingly used also for exchanging commercial and business critical data with any contents. The decisive factors for that are the wide distribution, the tremendous spread and the well-priced costs of exchange.

These advantages are opposed by seriously taken risks: On the internet all data are in danger for attacks on their confidentiality and integrity. As well their authenticity is not simply assured. It can lead to the disavowal of data by their evident sender (repudiation). These dangers are averting by applicable protocols as TLS and HTTPS. OFTP Version 2 is standardized for CAD data as much as EDI and any other data as a secure internet protocol.

There is other risk by malware (malicious software) which is automatically widespread on the internet. Viruses, worms, Trojans and other destructive programs can cause serious damage at the IT infrastructure of companies. To reduce the risk network arias are built separated from corporate networks by firewalls (DMZ). In the DMZ such as http proxys are implemented for transporting the reference data without permitting direct connections between internet and intranet. A similar solution is aspired for OFTP

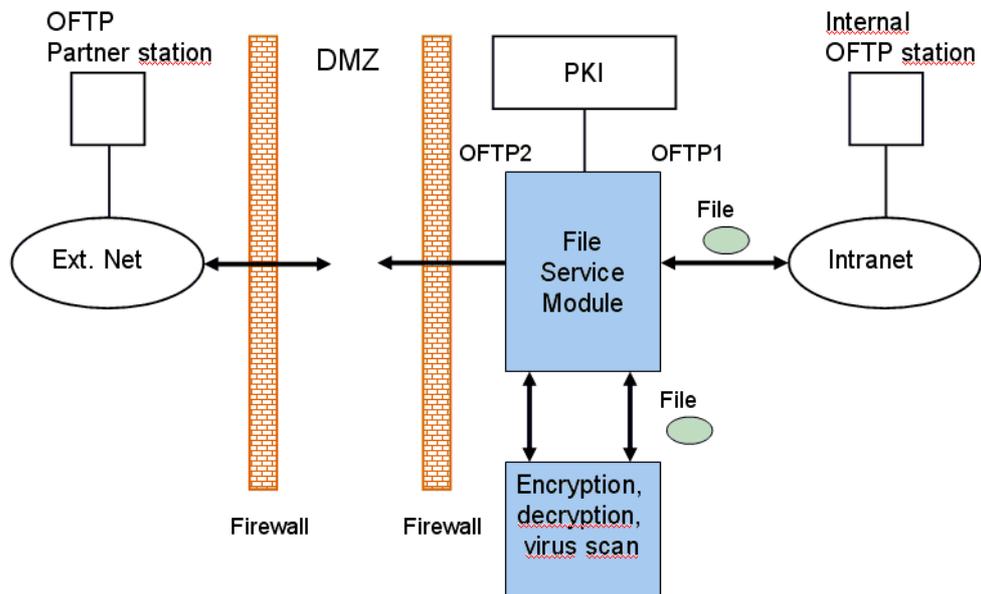
The File Service Module was developed for increasing security of data transfer on the internet. His task is as a OFTP router to de- and encrypt files accordingly OFTP2 for stations which do not support OFTP2 be itself. Also the File Service Proxy has to be able to proof files received over the internet for viruses after encryption.

The File Service Module is working on the level of the file services of OFTP and for that it needs the complete amount of features a "normal" OFTP router is providing. Therefore the File Service Module is seen as a rvsEVO installation which includes additional features.

8.2 Architecture of File Service Module

The File Service Proxy works as a OFTP router and de- and encrypted files for internal stations substitutional before he forwards them.

The following image shows the File Service Module architecture.



The following functions are available:

- The File Service Proxy works as an OFTP router
- Protocol alternation OFTP1 <-> OFTP2, all elements, even EERP
- En- and decryption of files acting for stations
- Virus scan of received files
- Connection to the PKI for downloads of certificates, OCSP check, download of CRL

8.3 Setting up of a Neighbourstation with File Service Module

In this chapter is described the adjustment of setting up of a neighbourstation for the OFTP File Service Proxy. Please read chapter 3.2 "Customizing the station configuration" for detailed description of station configuration.

The relevant parameters for File Service Module are:

File Service Proxy	<p>This parameter enables the File Service Module. Possible values:</p> <ul style="list-style-type: none"> – NONE: File Service Module is not needed (default) – INTERNAL: this partner station is an internal station (located in the intranet). When receiving an encrypted file which is routed to an internal station the mechanism of encryption at the File Service Module is automatically activated, depending on the encryption component used by the sending partner station. – EXTERNAL: this partner station is marked as an external station (located in the internet); for this option all encryption and compression features are available
--------------------	---

The next parameters are only available if **FileProxy Service** = EXTERN. The table Send parameters on page Seite 96 shows a description of the parameters.

- Security Feature Set
- Compression
- Encryption
- Sign
- Sign ERP
- Encryption Algorithm

Hint: Both direct neighbour stations and routing stations can be marked by File Service Module. Therewith the configured File Service Module function is station wide fixed and can not be overwritten while ceating a transmission. This corresponds with a forced encryption during activated encryption (**Security=FORCED**).

9 Remote GUI

In rvsEVO, it is possible from version 5.2 onwards to log in with a distant (remote) access to the rvsEVO GUI.

prerequisites The following prerequisites should be fulfilled for a remote connection:

- An rvsEVO client should be installed on the client-side (see chapter 2.4 "Fresh installation of rvsEVO").
- The client user is to be set up as a user, on the server side (see chapter 10.1.1 "Add user").
- The server should be started

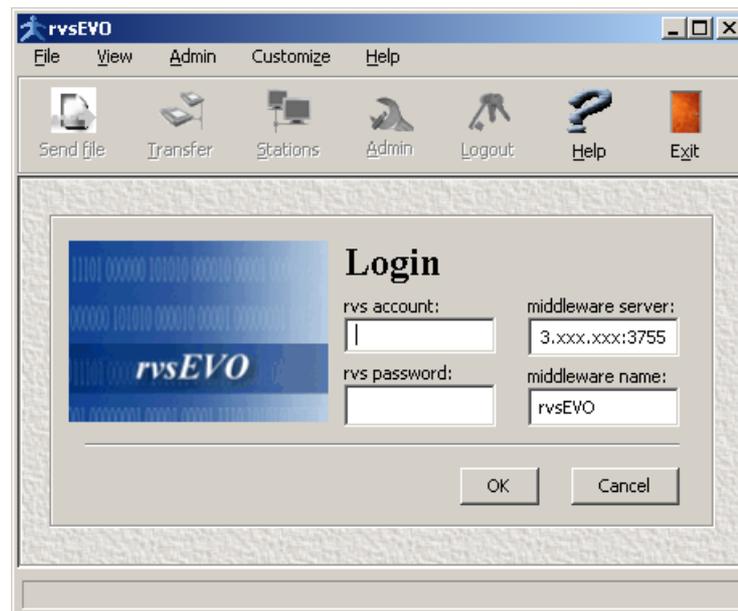
9.1 Starting the GUI from the remote computer

You can start rvsEVO with the help of the rvsEVO-program group:

Start -> All programs -> rvsEVO -> rvs GUI.

Alternatively, this function can be started in the command line, with the script `$RVS_HOME\bin\rvsGUI -r -u <user> -p <password>`.

The following window is opened and one can log in to the server:



Required specifications:

- `rvs account`: Login to rvsEVO (to the server) with the user set up in the user management (user ID)
- `rvs password`: Password of the user
- `middleware server`: Name or IP-address of the remote computer and port, to which the computer adheres to. The following syntax is to

be considered thereby: <RMI Server>:<Port>. 3755 is to be taken as a standard port.

- `middleware name`: Standard `rvsEVO`, this parameter can be found in the `rvsEVO` parameter-list, too (`Admin -> Param`).

9.2 Features of the remote GUI

The `rvsEVO` GUI on the client computer differentiates itself marginally from the `rvsEVO` GUI on the server computer.

The following differences exist:

- The functionalities **Export Certificate**, **Export Certificate to ComSecure** and **Import ComSecure Public Key** are not available.
- The encrypted part of the TLS connection cannot be configured.
- In the transfer window, you have the choice of creating the dispatch order remotely or locally. Thereby, the parameters for the file dispatch remain the same.
- Further, there is an option of uploading (transferring files to the server) or downloading (fetching / transferring files from the server) of files.

uploading In order to transfer a file to the server, you can select the menu item **uploading** in the transfer tree.

You can select the file to be sent, on the left side. In the field `File` on the right side, you can allocate a new name to the file. In the lower area, you have the option of specifying the format of the file to be sent.

With `execute`, you confirm the entries and starts the transfer. The file is stored on `rvsEVO` Server in the directory `$RVS_HOME\files\outbox`.

Hint: For the upload the server connects via a port, opened by the client. The port can be set up by the definition `<entry key="client.remotefileloader.port" value="3756" />` in the `$RVS_HOME/conf/rvsEvoClient.prefs` file.

downloading In order to receive a file from the server, you select the menu item **downloading** in the transfer tree.

In the area on the right, the files provided on the server (in the directory `$RVS_HOME\files\inbox`) for downloading are listed. One selects the file which is to be downloaded. On the left side of the window, you can select the directory, in which the file should be saved and in the field `File`, the file should be allocated a new name.

With `execute`, you confirm the entries and starts the transfer.

Command Tools The user ID and password of the user arranged with the user administration should be specified for all the command line calls which are started remotely.

Example:

```
updateStationList -f <xml-filename> -u <user> -p <password>
```


10 User Management

The functionality User Management is available starting from rvsEVO version 5.2. You reach it from the function bar of rvsEVO GUI (Admin -> User Management).

10.1 User Administration

Press the icon for user management to get the following window.



The following operations are available:

- **Add:** setup a new user
- **Delete:** removing an existing user
- **Edit:** editing an existing user

10.1.1 Add user

Press the button **Add** to add a new user.

In this template you can insert the following parameters:

- Surname (of the user)
- Name (of the user)
- Login
- Password
- Role (Administrator, Operater or User can be chosen in the selectbox, must be confirmed with **Add assignment**).

Press the button **OK** to setup the user.

Note: A basic **User** may use the rvsEVO program to send and receive files. An **Operator** has user rights and may also execute operator commands. A user with **Administrator**rights may in addition to this configure the rvsEVO system.

10.1.2 Delete a user

Select the user to be removed. To remove the user select the icon **Delete** and answer the query that appears with **Yes**.

10.1.3 Edit a user

Select a user to be edited. Execute the command **Edit**. Now you can edit all of the user data parameters. For changing the role you have to delete the active role first (highlight the role -> remove assignment -> OK). Following you can select the new role. Confirm your entries with **OK**.

Caution: You cannot edit the user data of the default user!

11 PKI Bindung

In this chapter the configuration and the functionality of the rvsEVO PKI binding will be described.

11.1 Introduction

In a PKI (Public Key Infrastructure) are stored public keys with their certificates. The access to the PKI normally occurs via LDAP. rvsEVO supports LDAP version 3.

Note: LDAP is a network protocol, which manages the communication between a LDAP Client and a LDAP Directory Server. This protocol offers the following features: login from client to the server, search queries for the information stored in the directory and if necessary: modification of information. This means relating to the PKI: in the LDAP Directory Server the public keys and certificates are administrated.

Following features are available for rvsEVO PKI binding:

- Access of public keys and certificates for a partner station via LDAP.
- certificate validation with OCSP (Online Certificate Status Protocol), CRT or CertPath.
- Transfer of information from a OCSP server by HTTP.

11.2 Configuration

The following steps must be done to configure the PKI binding in rvsEVO:

- configuration of stations, which should use PKI instead of a local key management file (see chapter 11.2.1).
- The configuration file `$RVS_HOME/conf/PkiParameter.xml` with the access parameters for the PKI binding must exist.

11.2.1 Configuration of stations

The PKI binding should be configured in the station configuration file `$RVS_HOME/conf/rvsStationlist.xml` with the XML element `<Pki>`. The Element `<Pki>` should be located in the XML element `<StationNeighbour>` and/or `<LocalStation>`.

Hint: Usually the PKI binding should be configured for a partner station (`<StationNeighbour>`). For the local station it can be interesting, only if you want to validate your own certificate.

Example (rvsStationlist.xml):

```
<Pki>
  <CertificateValidationType>NONE</CertificateValidationType>
  <PkiEnabled>true</PkiEnabled>
</Pki>
```

For the XML element <Pki> are possible two sub-elements : <PkiEnabled> and <CertificateValidationType>.

<CertificateValidationType>	This XML element describes the type of the certificate validation. The following 3 types are supported: OCSP (Online Certificate Status Protocol), CRL (Certificate Revocation List) and CertPath. The value None in this field means: No validation will be done.
<PkiEnabled>	Possible values: true or false. True: the PKI functionality with the LDAP access will be used. In this case the XML configuration file \$RVS_HOME/conf/PkiParameter.xml must have the valid LDAP access configuration. False: the certificates will be taken from the local key management file (Default: \$RVS_HOME/system/data/keystore.p12).
Parameter	Description

Hint: If you use as validation type OCSP or CRL, the configuration file \$RVS_HOME/conf/PkiParameter.xml with the valid configuration must exist.

Which validation method will be used is depending on security standards in the company. The strongest validation method is OCSP, then CRL following by CERT_PATH.

If you use OCSP as validation method, the OCSP service must be available (Internet, Intranet) and your station must be able to access it.

The validation with CRL can be done offline, too. The rvsEVO user/administrator must provide the list of certificate (CRL) and store it manually. The disadvantage of this way is, that certificate can expire between two updates.

11.2.2 PKI configuration file

The PKI access parameters must be configured in the PKI configuration file \$RVS_HOME/conf/PkiParameter.xml.

Example:

```

<pkiParameter>
<pkiProfile>
<pkiProfileId>default</pkiProfileId>
  <!-- ===== -->
  <!-- LDAP Einstellungen -->
  <!-- ===== -->
  <ldapParameter>
    <!-- ===== -->
    <!-- ldap server info -->
    <!-- ===== -->
    <ldapServer>
      <!-- server name or ip-address -->
      <name>localhost</name>
      <!-- ip-port -->
      <port>10389</port>
      <!-- LDAP version to use -->
      <version/>
    </ldapServer>
    <!-- ===== -->
    <!-- access info -->
    <!-- ===== -->
    <ldapSecurity>
      <user/>
      <password/>
    </ldapSecurity>
    <!-- ===== -->
    <!-- search pattern definition, how to retrieve X.509 Certificates -->
    <!-- ===== -->
    <ldapCertificateSearchPattern>
      <!-- root node (distinguishedName) in the X.509 directory,
      where search start from -->
      <root>ou=Prozesse,o=Volkswagen AG,dc=VW,dc=vwg,dc=com</root>
      <!-- common name pattern where to find the right certificate -->
      <commonName>cn=VW_OTFP </commonName>
      <!-- attribute which includes the X.509 certificate-->
      <certificateAttribute>userCertificate</certificateAttribute>
      <reverseOrder>false</reverseOrder>
    </ldapCertificateSearchPattern>
  </ldapParameter>
  <!-- ===== -->
  <!-- OSCP Einstellungen -->
  <!-- ===== -->
  <ocspParameter>
    <!-- URI of the OSCP Service -->
    <accessLocation>http://localhost:8080/ejb/publicweb/status/ocsp</accessLocation>
    <!-- ocspResponderDistinguishedName>CN=rvsRootCA,OU=rvs,O=tsystems
    </ocspResponderDistinguishedName-->
    <!-- issuerDistinguishedName></issuerDistinguishedName-->
    <!-- defines the handling of OSCP-failures -->
    <!-- ignoreMissingServer Define: if it should be ignored, when OSCP server is
    not reachable boolean value (true/false) if set to true, OSCP check returns
    VALID, when certificate will be checked, but OSCP server could not be
    connected -->
    <ignoreMissingServer></ignoreMissingServer>
    <!-- if set true, OSCP response signature and responder certificate will not be
    verified -->
    <skipOcpResponseValidation></skipOcpResponseValidation>
  </ocspFailureHandling>
</ocspParameter>
  <!-- ===== -->
  <!-- CRL Einstellungen -->
  <!-- ===== -->

```

```

<crlParameter>
  <!-- directory where the actual used CRLs are stored -->
  <location>g:\CRL\</location>
  <url/>
  <!-- defines the handling of CRL-failures -->
  <crlFailureHandling>
    <!-- if set true, CRL and CRL issuer are not verified -->
    <skipCRLValidation>>false</skipCRLValidation>
  </crlFailureHandling>
</crlParameter>
</pkiProfile>
</pkiParameter>

```

The parameters from the file `$RVS_HOME/conf/PkiParameter.xml` are described in the following table:

accessLocation	URL of OCSP service.
certificateAttribute	Defines the attribute, in which the user certificate is stored (e.g. userCertificate, userCertificate;binary).
commonName	Unique name, to distinguish between different certificates in the root directory. In the first version the ODETTE ID of the corresponding station will be added to this value. According to that, the DN has the following form: <root> + <commonName> + <ODETTE ID>.
crlFailureHandling	This parameter group describes the handling if the CRL validation fails.
crlParameter	This parameter group describes the parameters for certificate validation through CRL.
cspFailureHandling	describes the handling, if the OCSP query (request) fails.
ignoreMissingServer	Possible values: true or false. True means, that if the error occurs during the OCSP query, the certificate will be declared as valid, even though the OCSP service is not available. False: An error will occur, if the server is not available.
issuerDistinguished-Name	Optional. This parameter comprises the DN of the certificate issuer. Normally the name of the issuer should be declared in the certificate. For special cases, it is possible to set an another name in this field, however.

ldapCertificateSearch-Pattern	This parameter describes rules for the DN (distinguished name) structure of a certificate in the LDAP directory.
ldapParameter	In this group are described the parameters for the access to the LDAP directory.
ldapSecurity	Parameters for the LDAP authentication (only simple authentication will be supported). If these elements are not defined, the authentication will not be used (server must support ANONYMUS authentication).
ldapServer	IP configuration of the LDAP server.
location	The directory, where the CRLs are stored. The CRLs must be up to date and maintained by the user.
name	IP address or DNS name of the LDAP server.
ocspParameter	Defines parameters for certificate validation via OCSP.
ocspResponderDistinguishedName	Optional. DN of the OCSP responder certificate. The DN helps to find the OCSP responder certificate in the LDAP directory. This field is not necessary, if the responder certificate is transmitted through the OCSP response. In another case e.g. the OCSP response certificate will be signed through the certificate issuer. The DN of the certificate issuer is often part of the certificate.
Parameter	Description
password	Password
port	IP port of the LDAP server.
reverseOrder	defines the order of DN (default: <code>true</code>). For some LDAP directories the cn (common name) must be before or after the root. If the value of this parameter is <code>true</code> , the following order should be used: <code><commonName>+ODETTE-ID+<root></code> . If the value of this parameter is <code>false</code> the following order has to be applied: <code><root>+<commonName>+ODETTE-ID</code> .
root	LDAP root directory, in which all certificates are to be found.

<p>skipCrlValidation</p>	<p>Possible values: <code>true</code> or <code>false</code>. True: the validation of the CRL with its signature does not take place. False: the validation of the CRL with its signature will be performed. Hint: The validation of the signature includes the validation of the CRL issuer certificates with <code>CERT_PATH</code>, too.</p>
<p>skipOcspResponseValidation</p>	<p>An OCSP response with the signature must be validated. The validation of the signature means also, the validation of the responder certificate through <code>CERT_PATH</code>. Possible values: <code>true</code> or <code>false</code>. True: If this XML-Element has the value <code>true</code>, the validation will be skipped. False: The validation will not be skipped.</p>
<p>user</p>	<p>User name</p>

12 rvsEVO Database

From the rvsEVO version 5.0 the job data are written into a database and not anymore into the directory `$RVS_HOME/jobs` with the subdirectories ENDED, FAILED, SND or RCV.

Since version 5.02 also the data of user mangement are written into the rvsEVO database.

The following databases are available:

- Derby Embedded
- Oracle

12.1 Derby

Derby database is a free, java based relational database from Apache Foundation.

If you select Derby Embedded during the installation procedure, a database will be automatically (by rvsEVO) installed in the directory `$RVS_HOME/db`.

The following Derby databases were tested:

- Derby 10.3.1.4
- Derby 10.2.2.0

12.2 Oracle

The following preconditions must be fulfilled if you decided yourself for an Oracle database:

- an Oracle database user was set up with the user rights connect, resource, create session and create table.
- your Oracle configuration is accurate. If an Oracle client is installed on your machine, you can test the configuration with the following command:

```
sqlplus ORACLE-user@ORACLE-Network servername/password
```

Example:

```
sqlplus skk@RVS.TSYSTEMS.DE/skk
```

If this command could be executed successfilly (if a user exists and a database is available), you can start the rvsEVO installation.

The following Oracle databases were tested:

- Oracle 9i
- Oracle 10g

– Oracle 11g

While installation of rvsEVO (see Chapter 2.4 "Fresh installation of rvsEVO") the following Oracle connection parameters have to be set:

- jdbc.url
- jdbc.user and jdbc.password
- host connect string.

jdbc.url has the following syntax:

jdbc:oracle:thin:@<server>:<port>/<service_name>

Example:

jdbc:oracle:thin:@localhost:1527/rvsORA

The default port for Oracle is 1521.

Note: If Oracle is running on an external computer, this computer must be defined in the Oracle configuration file:

\$ORACLE_HOME/network/admin/tnsnames.ora.

Example (tnsnames.ora):

```
#TNSNAMES.ORA Network Configuration File: /opt/oracle/product/8.1.7/
network/admin/tnsnames.ora
# Generated by Oracle configuration tools.
```

```
RVS.TSYSTEMS.DE =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = rvsaix3)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = rvs)
    )
  )
```

In this example RVS.TSYSTEMS.DE is an Oracle network service name and rvs a service name. A network service name is needed to identify an Oracle database in a network. This name should not be mixed up with the global database name, although it has a similar syntax. The service name identifies a database instance, because several instances can run on the same computer.

jdbc.user is the Oracle user and jdbc.password is his password.

Note: The Oracle connection data from the installation window are stored in the file \$RVS_HOME/conf/jdbc.properties.

Connection Pooling

With Oracle database rvsEVO offers the functionality **connection pooling**. That means there is a pool which contains connections to the database. If a connection is needed, one connection of the pool can be

used and it is not necessary to establish a connection for every access. rvsEVO uses C3P0 (Hibernate) for connection pooling.

In order to counter firewall problems you should add C3P0 parameter to the file `$EVO_HOME/system/data/Oracle/hibernate.cfg.xml`.

Example of configuration:

```
<property
name="connection.provider_class">org.hibernate.
connection.C3P0ConnectionProvider</property>

<property
name="hibernate.c3p0.idle_test_period">300</property>
<property name="hibernate.c3p0.max_size">10</
property>
<property name="hibernate.c3p0.min_size">0</property>
```

The most important parameter is `hibernate.c3p0.idle_test_period`. With this parameter you define the time interval (in seconds) for considering the connections of the pool. The parameter `max_size` defines the maximum number of connections to the database and `min_size` defines the initial number of connections.

For further information please read the C3P0 documentation:

<http://community.jboss.org/wiki/HowToconfiguretheC3P0connectionpool>

12.3 How to Drop and Create the Database Tables?

Note: rvsEVO must be stopped before execution of database operations.

The database table with the job data is named `HIB_JOB_DATA`. If the job data are no longer needed, the database table `HIB_JOB_DATA` can be dropped with the `$RVS_HOME/system/DERBY/derby_drop` for a Derby database or `$RVS_HOME/system/ORACLE/oracle_drop/` script for an Oracle database.

After the deletion the database tables must be created again. Use the following scripts for the creation of the derby database tables:

- `$RVS_HOME/system/DERBY/derby_create`
- `$RVS_HOME/system/DERBY/derby_user`

and the following scripts for the creation of Oracle database tables

- `$RVS_HOME/system/Oracle/oracle_create`
- `$RVS_HOME/system/Oracle/oracle_user.`

12.4 How to view job data from a database?

With the following sql command you can view all data from a respective database table:

```
SELECT * from TABLE;
```

Example:

To view all job data parameter from the rvsEVO database table HIB_JOB_DATA the command will be:

```
SELECT * from HIB_JOB_DATA;
```

If you want to view a value of one job data parameter (e.g. JOBID) you should use the following command:

```
SELECT JOBID from HIB_JOB_DATA;
```

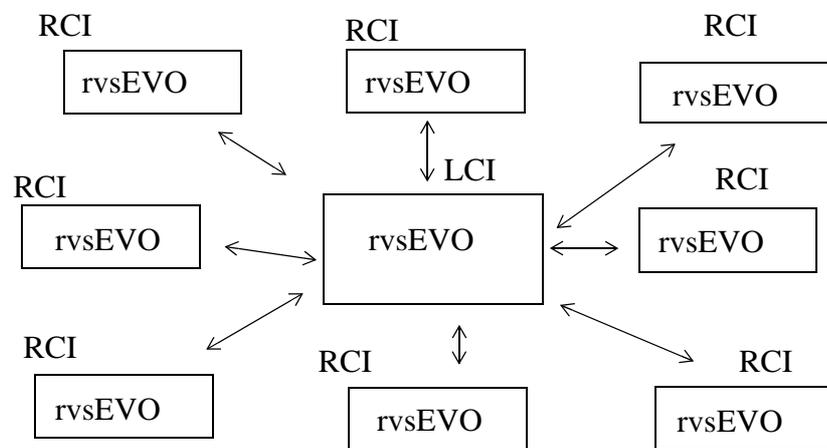
13 rvsEVO Central Administration

This chapter describes the powerful feature of rvsEVO: how to remotely administer other rvsEVO stations. It is essential for the network administrator being able to remotely administer all rvsEVO installations.

13.1 Introduction

The rvsEVO Central Administration enables the configuration of numerous rvsEVO installations by using special configuration files.

The configuration files will be sent from a rvsEVO station (we call it: Local Configuration Instance LCI) to the station, that should be administrated (we call it: Remote Controlled Instance RCI). The both instances are located in the rvsEVO star network (Please see the following picture).



To distinguish the configuration files for the central administration from the normal rvsEVO XML configuration files such as `rvsStationlist.xml`, the files for the central administration will be named **configuration container files**.

These are

- `cfg.req.jar` for configuration requests, sent by LCI to RCI and
- `cfg.rsp.jar` for configuration responses, sent by RCI to LCI.

The Local Configuration Instance LCI

Each rvsEVO installation can be used as LCI (please see the chapter 2.4 for the explanation how to install rvsEVO).

The LCI maintains a special directory where the configuration data of all RCI's (rvsEVO stations, that are to be administrated) are stored - the Configuration Repository CRep. The location of CRep is `$RVS_HOME/`

management. Please refer to the chapter 1.6 for the explanation of \$RVS_HOME.

The entries of CRep (\$RVSTINY_HOME/management) are:

- \$RVS_HOME/management/mgmt-datastore
This directory will be created only after a succesfull transmission and response to the request for the configuration file of the RCI. Please see the chapter 13.2 for the explanation how to make a configuration request.
- \$RVS_HOME/management/mgmt-log/activity-log This log file contains protocols of all configuration actions. Each entry comprises a timestamp, the type: **configuration request** or **configuration response** respectively, the SID of the administrated rvsEVO station (RCI) and a message text.
- \$RVS_HOME/management/mgmt-templates
This directory contains templates for management actions. It is not necessary to deal with this directory except for updates of the administration software itself.
- \$RVS_HOME/management/mgmt-workspace
This directory stores configurations of RCIs for editing. It contains subfolders, that are named after the RCI's SID (see the tool `prepareUpdateStation`, chapter 13.2 for an example).

The Remote Controlled Instance RCI

The RCI is a rvsEVO station, which should be administrated remotely.

When rvsEVO (RCI) receives a configuration request (a file `cfg.req.jar`) a dedicated job will be launched. This job starts the rvsEVO configuration process that handles the configuration request and generates the configuration response. The configuration response is sent back from the RCI to LCI as file `cfg.rsp.jar`.

13.2 Command Tools of the Central Administration

The whole process (all configuration and administration cases) is to be done by the following command tools. These tools are located in the directory `$RVS_HOME\bin` as batch files (please see the chapter 1.6 for the explanation of \$RVS_HOME.), so you must change to this directory to be able to execute them.

<pre>orderConfiguration.bat -s SID</pre>	<p>fetches the configuration of an RCI and stores it in CRep.</p> <p>Example: C:\rvsEVO\management\mgmt-datastore\TINYPW</p> <p>In this example TINYPW is the SID of RCI (of the station, that should be administrated).</p>
<pre>prepareUpdateStation.bat -s <SID></pre>	<p>gets a RCI configuration copy out of CRep to the WorkDir in order to serve as starting point for modifications.</p> <p>Example for the WorkDir:</p> <p>In the directory C:\rvsEVO\management\mgmt-workspace the following subdirectory will be created TINYPW\UPDATE_STATION_040826_114418\out.</p> <p>This is a subdirectory for the TINYPW station; the date of the creation is 2004-08-26, the time of the creation is 11:44:18.</p>
<pre>commitUpdateStation.bat -s <SID> -d <WorkDir></pre>	<p>sends a modified configuration from the WorkDir (without out subdirectory) to an RCI (RCI should be set with the option -s for stationID).</p> <p>Example:</p> <pre>commitUpdateStation.bat -s TINYPW -d C:\rvsEVO\management\mgmt-workspace\TINYPW\UPDATE_STATION_040826_114418</pre>

Note: The name of the WorkDir directory consists of CRep (see 13.1); RCI'SID (TINYPW in the table example), directory UPDATE_STATION with the timestamp (consisting of date and time; in the table example 040826_114418) and the directory out.

WorkDir contains all essential rvsEVO directories and files: By changing these files the following configuration actions for example are possible:

- Modify station configuration by modifying \$RVS_HOME/conf/rvsStationlist.xml.
- Modify jobstarts by modifying \$RVS_HOME/conf/rvsJobstart.xml.
- Update the software by replacing .jar files in the directory \$RVS_HOME/lib.

13.3 How to work with the central administration features?

The following steps will show you the typical configuration run. It may be used as a basic example when a rvsEVO network administrator deals with rvsEVO Central Administration. These steps are always necessary, independent of the fact, whether you want to make an update, change some rvsEVO parameter or exchange the license key file. More details about the particular administration tasks, you will find in the chapters 13.3.1, 13.3.2 and 13.3.3.

Note: The following commands are available as batch files in the `$RVS_HOME\bin` directory (please see the chapter 1.6 for the explanation of `$RVS_HOME`.), so you must change to this directory to be able to execute them.

- at first, get the configuration of the RCI (of the rvsEVO station, that should be administrated). Use the program `orderConfiguration` for this step.

Example: If you would like to administer the configuration of the rvsEVO station `TINY11`, launch the program `orderConfiguration` in the command line with the following command:

```
orderConfiguration -s TINY11
```

This command generates a configuration request file `cfg.req.jar` (see the chapter 13.1 for the explanation of the meaning of `cfg.req.jar`) and sends it via OFTP to the station `TINY11`. The transmission of the file `cfg.req.jar` may be watched in the rvsEVO GUI (Admin window). When the RCI (`TINY11`) receives the file `cfg.req.jar` the configuration process will be started. The configuration process stops rvsEVO (station `TINY11`), archives the actual configuration in a file `cfg.rsp.jar`, starts rvsEVO (`TINY11`) again and sends the configuration container file `cfg.rsp.jar` back to the LCI (rvsEVO station of the administrator). If this step was successful, you will receive a message „OrderConfiguration exited with return code 0“ in the console. All these steps are part of the program `orderConfiguration` and will be executed automatically.

The next step is to get a copy of the RCI's configuration, that arrived as the file `cfg.rsp.jar`. This should be done by the program `prepareUpdateStation` on the command line. This program will copy for you the arrived configuration file `cfg.rsp.jar` and store it to the `WorkDir` (see chapter 13.2 for the explanation of `WorkDir`).

Example:

```
prepareUpdateStation.bat -s TINY11
```

Result: The directory `C:\rvsEVO\management\mgmt-workspace\TINY11\UPDATE_STATION_040828_113315\out` with the complete configuration of the station `TINY11` will be created. If this action was successful, you will find a corresponding

message in the file `activity.log`. This log file is stored in the directory `$RVS_HOME/management/mgmt-log`.

- Now you can administrate the configuration of the rvsEVO station (e.g. TINY11). You can exchange a license key, modify the XML configuration files or substitute the appropriate `.jar` files to make an update of rvsEVO. Please read the chapters 13.3.1, 13.3.2 und 13.3.3 for more details about the particular configuration procedures.
- Send the modified configuration to the RCI (TINY11). You have to send the whole directory `C:\rvsEVO\management\mgmt-workspace\TINY11\UPDATE_STATION_040828_113315` with the command:

```
commitUpdateStation.bat -s TINY11 -d
C:\rvsEVO\management\mgmt-
workspace\TINY11\UPDATE_STATION_040828_113315
```

This command will store the whole modified directory and send it to the RCI (TINY11) again as a file `cfg.req.jar`.

After successfully receiving a file `cfg.req.jar` at the RCI (TINY11), rvsEVO will be stopped (it all happens with the process `commitUpdateStation`, you do not have to do any particular steps); the modified configuration will be updated; the update job checks, if all was correct and sends back a response as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

13.3.1 How to exchange a license key file?

This chapter describes the typical case in administrating rvsEVO, how to exchange an invalid license key. In this example the station TINY01 will administrate the station TINY02.

Prerequisites: The station TINY01 must have the station TINY02 in the station table as a neighbour station (please read the chapter for the explanation how to set up stations) and the station TINY02 must also have station the TINY01 as a neighbour station.

- To be able to replace a license key file of TINY02, TINY01 must at first get the configuration of TINY02 with the command:
`orderConfiguration -s TINY02`

If this step was successful, you will receive a message „OrderConfiguration exited with return code 0“ in the console and the file `cfg.rsp.jar` will be received. (see in the **Ended Transmissions**, Admin-window of the TINY01 GUI).

- The next step ist to get a copy of the TINY02 configuration, that arrived as the file `cfg.rsp.jar`. This must be done with the following command:

```
prepareUpdateStation.bat -s TINY02
```

Result: The directory `C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out` with the complete configuration of the station TINY02 will be created. If this action was successful, you will find the message in the file `activity.log`. This log file is stored in the directory `$RVS_HOME/management/mgmt-log`.

- Now you can rename the old license key from the directory `C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out\conf` to `licenseOLD.properties` and copy the new license key `license.properties` to the directory `C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out\conf`. How to obtain the new license key, please read the chapter 2.2.
- Send the modified configuration to TINY02 (you must send the whole directory `C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315`) with the command

```
commitUpdateStation.bat -s TINY02 -d
C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315
```

This command will store the whole modified directory and send it to the station TINY02 again as a file `cfg.req.jar`.
- After successfully receiving the file `cfg.req.jar` at the station TINY02, rvsEVO at TINY02 will be stopped (it happens all with the process `commitUpdateStation`, you do not have to do any particular steps); the modified configuration will be updated; the update job checks, if all was correct and sends back a response as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

13.3.2 How to change a station parameter?

This chapter describes a typical case in administrating rvsEVO, how to change a rvsEVO parameter e.g. the ODETTE-ID. It is the same procedure for changing any other rvsEVO parameter. In this example the station TINY20 will administrate the station TINY22.

Prerequisites: The station TINY20 must have the station TINY22 in the station table as a neighbour station (please read the chapter for explanation how to set up stations) and the station TINY22 must also have station TINY20 as a neighbour station.

- To be able to modify an rvsEVO parameter TINY20 must at first get the configuration of TINY22. it will be done with the following command:

```
orderConfiguration -s TINY22
```

If this step was successful, you will receive a message „OrderConfiguration exited with return code 0“ in the console and the file `cfg.rsp.jar` from the station TINY22 will be received as a response of this request; please see in the **Ended Transmissions**, Admin-window of the TINY20 GUI.

- The next step ist to get a copy of the TINY22 configuration, that arrived as the file `cfg.rsp.jar` to the WorkDir of TINY20. Please see the chapter 13.2 for the explanation of the WorkDir.

```
prepareUpdateStation.bat -s TINY22
```

Result: The directory `C:\rvsEVO\management\mgmt-workspace\TINY22\UPDATE_STATION_040829_133315\out` with the complete configuration of the station TINY22 will be created. If this action was successful, you will find a corresponding message in the file `activity.log`. This log file is stored in the directory `$RVS_HOME/management/mgmt-log`.

- Now you can edit the file `rvsStationlist.xml` from the directory `C:\rvsEVO\management\mgmt-workspace\TINY22\UPDATE_STATION_040829_133315\out\conf` and modify it e.g. parameter `ODETTE_ID` or TCP/IP address (parameter `IP_ADDR`) for the local station of TINY22 (`STATION_LOC`) or other stations.

- The next step is to send the modified configuration to TINY22 (you must send the whole directory (but without `out` directory)

```
C:\rvsEVO\management\mgmt-workspace\TINY02\UPDATE_STATION_040829_133315)
with the command
```

```
commitUpdateStation.bat -s TINY22 -d
C:\rvsEVO\management\mgmt-workspace\TINY22\UPDATE_STATION_040829_133315
```

This command will store the whole modified directory and send it to station TINY22 again as a file `cfg.req.jar`.

- After successfully receiving a file `cfg.req.jar` at the station TINY22, rvsEVO at station TINY22 will be stopped (it all happens with the process `commitUpdateStation`, you do not have to make any particular steps) and the modified configuration will be updated. Then the update job checks, if all was correct and sends back a response to the station TINY20 as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

13.3.3 How to make an update of rsvEVO?

This chapter describes a typical case in administrating rsvEVO, how to make an update of an another rsvEVO station. In this example the station TINY30 will administrate the station TINY33.

Prerequisites: The station TINY30 must have the station TINY33 in the station table as a neighbour station (please read the chapter for explanation how to set up stations) and the station TINY30 must also have station TINY33 as a neighbour station.

- To be able to make an update of rsvEVO at the station TINY30 you must get at first the configuration of TINY33.

```
orderConfiguration -s TINY33
```

If this step was successful, you will receive a message „OrderConfiguration exited with return code 0“ and the file `cfg.rsp.jar` will be received, see in the **Ended Transmissions**, Admin-window of the TINY30 GUI.

- The next step ist to get a copy to the WorkDir of the TINY33 configuration, that arrived as the file `cfg.rsp.jar`. Type the following command in the command line:

```
prepareUpdateStation.bat -s TINY33
```

Result: The directory `C:\rsvEVO\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315\out` with the complete configuration of the station TINY33 will be created. If this action was successful, you will find a corresponding message in the file `activity.log`.

- Now you must rename the old `.jar` file `rsv.jar` from the directory `C:\rsvEVO\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315\out\lib` to `rsvOLD.jar` and replace it with the new one. Please contact us to receive the actual files for the update (E-Mail: rsv-service@t-systems.com; Tel. +49 30 39971 777).
- Send the modified configuration to TINY33 station (you must send the whole directory `C:\rsvEVO\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315`) with the command

```
commitUpdateStation.bat -s TINY33 -d  
C:\rsvEVO\management\mgmt-  
workspace\TINY02\UPDATE_STATION_040830_113315
```

This command will store the whole modified directory and send it to TINY33 again as a file `cfg.req.jar`.

- After successfully receiving the file `cfg.req.jar` at the station TINY33, rsvEVO of TINY33 will be stopped (it all happens with the process `commitUpdateStation`, you do not have to do any

particular steps); the modified configuration will be updated; the update job checks, if all was correct and sends back a response as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

14 rvsbat Batch Interface

rvsbat rvsbat gives you an additional possibility for automated file transfer and for managing jobstarts.

The following rvsbat commands are supported by rvsEVO:

- SEND: create a send job
- RESENTR: create, delete, modify jobstarts after receive (in rvs® portable: resident receive entries)
- SENDJOB: create, delete, modify jobstarts after send attempt.
- FAILURE: create, delete, modify jobstarts after failure.

14.1 Starting rvsbat

rvsbat is to be started via command line:

Syntax:

```
rvsbat [/c] [/i<input file>] [/l<language>] [/q]
```

The command line parameters have the following meaning:

- **/C**: continue with rvsbat after an error occurred during execution of a utility command. By default, rvsbat will terminate after an error.
- **/I<filename>**: read commands from cmdfile. The command input file may contain the following elements:
 - Comment lines (starting with *)
 - Commands (may extend over several lines by specifying + as the last character in the line to be continued)
- **/L<language>**: use message language given by character language. Values: **E** (english), **D** (german)
- **/Q** execute user commands in quiet mode, i.e. do not echo them to standard output; feedback about success or failure of the operation will still be provided.

The rvsbat commands (SEND, RESENTR, SENDJOB und FAILURE) can be imported via input file or written into the command prompt. In the second case, rvsbat can be broken by **<STRG> C**. In both cases, the syntax of the command strings is identical.

Example:

```
SEND /C SIDORIG=LOC DSN=C:\docs\test.txt (SID=EVO54E
DSNNEW=TESTVDSN)
```

Import a command via input file:

Using the example of **SEND** command you see what steps are needed to perform a send job by using an input file:

- provide the send file (example: test.txt)
- provide the job file (contains send job parameters like address ID, OFTP file name, ...)

Example for a job file (input.txt):

```
SEND /C SIDORIG=LOC DSN=C:\docs\test.txt (SID=EVO54E
DSNNEW=TESTVDSN)
```

In the above example a send job ist created for sending the file C:\docs\test.txt from local station LOC to station EVO54E with VDSN TESTVDSN.

- call the rvsbat command.

Example:

```
rvsbat /LE /Ic:\docs\input.txt
```

Starting rvsbat without input of a command file:

In the following you find a description of starting rvsbat in the command prompt (expamle with **SEND** command):

- Start rvsbat in the command prompt

Example:

```
rvsbat
```

- Write the rvsbat commands in the cmd prompt after a successful start of rvsbat (message „Command 'START /USER' successfully executed).

Example:

- SEND /C SIDORIG=LOC DSN=C:\docs\test.txt (SID=EVO54E DSNNEW=TESTVDSN).

14.2 Create a Send Job with SEND Command.

Additional to the GUI and the createSendJob program you can use the SEND /CREATE (or SEND /C) command of rvsbat script, for sending files.

By rvsEVO supported CREATE /SEND parameters:

Required parameters:

DSN	path and name of local data set to be sent
SID	station ID of receiver (enclosed in parentheses)

Optional parameters:

CODEIN	Code of local file (A =ASCII, E =EBCDIC) Default: code of local system
---------------	--

DISP	Disposition of local file after successful send attempt. Possible values: <ul style="list-style-type: none"> – K=file will not be deleted after sending (default) – D=delete. (GUI name: Disposition)
FORMAT	Format of the file to be sent. possible values: <ul style="list-style-type: none"> – T=textfile; a stream of ASCII characters – U= unstructured (binary); byte stream – F=fixed; fixed record length – V=variable; variable record length Default: U
INITTIME	With this parameter you can decide, what time the send job should be created. Possible values: <ul style="list-style-type: none"> – date and time in format YYYY/MM/DD HH:mm:ss – H or HOLD: job is created with HOLD status. – N or NOW: job is started immediately (default) (GUI name: Schedule)
LABEL	User label (up to 20 characters) used to serialize on a proceeding send request (if SERIAL=Y) or which can be used for serialization by a subsequent send request
MAXRECL	Max. record length for the files in format F or V .
SERIAL	Y (Yes)/ N (No). If you set SERIAL=Y , the files will be sent in the same order, as the send jobs were created. The next job will only be sent, if the previous is completely finished. All send jobs for the serialization must have the same LABEL. In the GUI the VDSN will be used as label. That means, that all jobs for the same serialization group must have the same virtual data set name (VDSN). (GUI name: Serialization)
TSTAMP	No Function; for compatibility with <i>rvs® portable</i> . Possible values: <ul style="list-style-type: none"> – Y / N (Default)
VFTYP	In this parameter you indicate the mode of files in format 'F' or 'V'. Possible values: <ul style="list-style-type: none"> – X: textfile (record mode 'TXT' in the GUI) – U: binary file (record mode 'BIN' in the GUI) Please see parameter Record Mode in chapter 4.5 for further information. (GUI name: Record Mode)

Optional parameters (enclosed in parentheses)

ALG	Active only in connection with OFTP 2.0 (CMS). The following algorithms are possible: – None – 3DES – AES (GUI name: Encryption Algorithm)
CODEOUT	Desired code of data set at receiver; (A =ASCII, E =EBCDIC) Example: send /c dsn=C:\test22.dat CODEIN=A FORMAT=V(SID=RTT CODEOUT=E DSNNEW=FIX0GBE.TEXT)
CODETABLE	Defines the code table, which is to be used for the code conversion (see Parameter Conversion table in the table in chapter 4.5) . Indicate the alias name of conversion table. Example: send /c dsn=c:\programs\rvsEVO\files\outbox\ test22.dat FORMAT=V(SID=RTT CODETABLE=ASCII-IBM037 MAXRECL=80 DNSNEW=FIX0GBE.TEXT). (GUI name: Conversion table)
COMPRESSION	compression; active only in connection with sfs=2, 3 oder 4 . If Y (Yes) the file should be compressed before sending. This parameter has to be placed after SID parameter. Example: SEND /C DSN=\home\test\test11.txt (SID=RTZ COMPRESSION=Y). (GUI name: Offline Compression)
DSNNEW	Virtual file name; the length of the file name used for ODETTE transfer must not exceed 26 characters (GUI name: VDSN)
ENCRYPTION	File encryption; active only in connection with sfs=2, 3 oder 4 ; If Y (Yes) the file should be encrypted before sending. This parameter has to be placed after SID parameter. Example: SEND /C DSN=\home\test\test11.txt (SID=RTZ COMPRESSION=Y ENCRYPTION=Y).
FILEDESC	File description, active only in connection with sfs=4 (OFTP 2.0) (GUI name: File description)

SIDORIG	station ID of originator; local station (default) or virtual station (GUI name: SID Originator)
SFS	Security Feature Set ; this parameter applies to the format of encryption. Possible values: – 1 (no encryption; default) – 2 (ComSecure V1) – 3 (ComSecure V2) – 4 (OFTP 2.0 (CMS)) (GUI name: Security Feature Set)
SIGN	File signature is activated. Active only in connection with sfs=4 (OFTP 2.0). (GUI name: File signature)
SIGNRESP	Request signed EERP/NERP. Active only in connection with sfs=4 (OFTP 2.0). (GUI name: Request signed EERP)
XID	Parameter for an external JobID, which can refer to several rvsEVO JobIDs. The external JobID is a string of alphanumeric characters.

14.3 Managing Jobstarts via rvsbat

Jobstarts after receive (`RESENTR` command), after send attempt (`SENDJOB` command) and after processing error (`FAILURE` command) can be created, modified or deleted by `rvsbat`. Please read chapter 3.3 "Customizing the JobStarts" for further information about jobstarts.

14.3.1 The `RESENTR` command

Additional to the GUI and the XML configuration file `rvsJobstart.xml` you can use the `RESENTR` command of `rvsbat` script, for managing Jobstarts after receive (in accordance with resident receive entries in `rvs@portable`).

Command:

- `/CREATE` or `/C`: create a jobstart after receive
- `/UPDATE` or `/U`: edit a jobstart after receive
- `/DELETE` or `/D`: delete a jobstart after receive

Hint: The commands can be added in abbreviated or long version and with a blank or not. (`RESENTR /DELETE DSN="*.TENNIS.*" SID="*"` is the same as `RESENTR/D DSN="*.TENNIS.*" SID="*"`)

RESENTR parameters:

Required parameters: DSN, SID

Optional parameters: CODETABLE, CODETRANS, COMMENT, DSN, ENABLED, EXECSYNC, EXECTIMEOUT, JOB, NEWDIR, NEWNAME, PARAMHANDLING, REPLACE, SHELL, SIDDEST, TSTAMP, TSTAMPFORMAT, VFTYP

Parameters without function (for compatibility with `rvs@ portable`): ACCOUNT, DISP, FLAGCOMP, FLAGCRYPT, LUID, UID

The table in chapter 14.3.4 shows the description of the parameters.

1. Example:

```
RESENTR/CREATE LUID="*" DSN="TENNIS" UID="*" SID="*"
SIDORIG="*" DSNNEW="" REPLACE="N" DISP="K" TSTAMP="N"
JOB="C:\rvsTest\scripts\resentr.bat"
```

After receiving files with the virtual file name "TENNIS" the Job "C:\rvsTest\scripts\resentr.bat" is to be started. Existing files which own the same name are not replaced, a timestamp is to be added only if necessary. The parameters **LUID**, **UID** and **DISP** are without function.

2. Example:

```
RESENTR /D DSN="Test.txt" SID="*" 
```

Jobstarts which are to be launched after receiving the file `Test.txt` are deleted.

14.3.2 The SENDJOB Command

Use the `SENDJOB` command to create, edit or delete a jobstart after send attempt.

Commands:

- /CREATE or /C: create a jobstart after send attempt
- /UPDATE or /U: edit a jobstart after send attempt
- /DELETE or /D: delete a jobstart after send attempt

SENDJOB parameters:

Required parameters: SID (= SIDDEST), VDSN

Optional parameters: ATTEMPTS, COMMENT, ENABLED, EXECSYNC, EXECTIMEOUT, JOB, PARAMHANDLING, SHELL, SIDSENDER = SIDORIG, UID

The table in chapter 14.3.4 shows the description of the parameters.

1. Example:

```
SENDJOB /C VDSN="Test.txt" SID="RVS" ATTEMPTS=0
SIDSENDER="LOC" JOB="C:\rvsTest\scripts\sendjob.bat"
CODETRANS="" CODETABLE="" FLAGCOMP="N" FLAGCRYP="N"
```

After sending the file "Test.txt" from station LOC to station RVS the job "C:\rvsTest\scripts\sendjob.bat" is to be started. There is no code conversion; the **FLAGCOMP** and **FLAGCRYP** parameters are without function.

2. Example:

```
SENDJOB /D VDSN="Test.txt" SID="RVS"
```

Jobstarts which are to be launched after sending the file Test.txt to station RVS are deleted.

14.3.3 The FAILURE Command

Use the FAILURE command to create, edit or delete a jobstart after processing error.

Commands:

- /CREATE or /C: create a jobstart after processing error
- /UPDATE or /U: edit a jobstart after processing error
- /DELETE or /D: delete a jobstart after processing error

FAILURE parameters:

Required parameters: VDSN

Optional parameters: COMMENT, ENABLED, EXECSYNC, EXECTIMEOUT, JOB, PARAMHANDLING, SHELL, SIDDEST, SIDORIG

The table in chapter 14.3.4 shows the description of the parameters.

Example:

```
FAILURE/C VDSN="Test.txt" SIDORIG="LOC" SIDDEST="RVS"
JOB="C:\rvsTest\scripts\failure.bat"
```

After receiving an error message after send attempt of "Test.txt" file from station LOC to station RVS the job "C:\rvsTest\scripts\failure.bat" is to be started.

14.3.4 Jobstart Parameters

The following table shows the description of the jobstart parameters.

Jobstart parameters

Required parameters:

DSN	(only RESENTR) Virtual file name. Allowed: SID string and wildcard '*' at the end.. (GUI name: VDSN)
SID oder SIDDEST	(only SENDJOB) Station ID of target station. Allowed: SID string and wildcard '*' at the end. (GUI name: SID of Destination)

SID	(only RESENTR) Station ID of source station. Allowed: SID string and wildcard '*' at the end. (GUI name: SID of Destination).
VDSN	(SENDJOB and FAILURE) Path and name of the local file to send. Allowed: SID string and wildcard '*' at the end.

Optional parameters:

ACCOUNT	without function; for compability with <i>rvs@portable</i>
ATTEMPTS	Number of unsuccessful send attempmts. Successful file transmission is indicated by "0". (GUI name: Send Attempts)
CODETABLE	Alias of code table that is to be used if parameter CODETRANS = 'T' . See parameter Conversion table in table "jJobFilter elements" on page 79 Example: <code>RESENTR /C DSN="C:\files\test22.dat" CODETRANS="T" CODETABLE="ASCII- IBM037" DSNNEW="TEST22"</code>
CODETRANS	Defines whether a code translation has to be executed. Possible values: <ul style="list-style-type: none"> - no indication: no code translation - E: EBCDIC - ASCII conversion - A: ASCII - EBCDIC conversion - T: your own conversion table , that is indicated in CODETABLE parameter. (Please see also section "How to add your own conversion table:" on page 85)
COMMENT	free text;
DISP	no function; for compatibility with <i>rvs@portable</i> , value: K
DSNNEW	With this parameter you can save the received file with an other name and path. <ul style="list-style-type: none"> - no indication: the file is saved with current name in inbox directory - indication of new path and name: the file is saved in the indicated directory with the new name
ENABLED	With this parameter you decide whether the job is to be started or not. Possible values: <ul style="list-style-type: none"> - Y (Yes): to start the job (default) - N (No): the job is not started

EXECSYNC	If EXECSYNC = Y (yes) the OFTP session is kept alive until the process is finished. Default: N (No) (GUI name: Synchronized)
EXETIMEOUT	If setting EXECSYNC = Y : Time Out after the connection is closed by the communication program. (GUI name: Timeout (Sync.))
FLAGCOMP	no function; for compatibility with rvs® portable; Possible value: N / empty string
FLAGCRYPT	no function; for compatibility with rvs® portable; Possible value: N / empty string
JOB	Program to be started when all filter conditions apply. A defined set of parameters is passed to the programs. (GUI name: Process)
LUID	no function; for compatibility with rvs® portable; Possible value: N / empty string
NEWDIR	With this parameter you can save the received file in an other directory than Inbox directory: <ul style="list-style-type: none"> – no indication: file is saved in Inbox – indication of new path: the file is saved in the indicated directory (GUI name: New Directory)
NEWNAME	With this parameter you can save the received file with an other name. <ul style="list-style-type: none"> – no indication: VDSN is file name – indication of new name: the file is saved with the new name (GUI name: New Filename)
PARAMHANDLING	With this parameter you can decide how to transmit the job data to the process. Possible values: ARGS : job data are passed as arguments (default) ENV : job data are set as environment variables REPLACE : for compatibility with rvs® portable. For further information see chapter "Parameter handling" on page 82.
REPLACE	With REPLACE parameter you decide how to handle receiving files which own the same name like existing files. Possible values: <ul style="list-style-type: none"> – R oder Y (yes): replace the existing file – N (no): create new data set with uniqueness; timestamp is added (default) – I: no function; for compatibility with rvs® portable

SHELL	Command Shell for executions of the program e.g. ksh, csh, ... on Unix systems
SIDDEST	Station ID of target station. (GUI name: SID of Destination)
SIDORIG	Station ID of source station (local or virtual station) (GUI name: SID of Originator)
SIDSENDER	alternativ to SIDORIG
TSTAMP	With this parameter you define rules for timestamp creation. Possible values: <ul style="list-style-type: none"> - Y = timestamp is to be generated generally - N = timestamp is to be added only if necessary (default) (GUI name: Timestamp)
TSTAMPFORMAT	Format of the timestamp. Possible values: <ul style="list-style-type: none"> - TIME: time of creation of the job and counter; format: hhmmssccc (default) - DATETIME: date and time of creation of the job and counter; format: YYMMDDhhmmssccc - SFID_DATETIME: Odette timestamp of SFID; date, time and counter in format YYMMDDhhmmssccc - COUNTER: 000000 - 999999. If the counter is not sequential, the gap is filled first before the counter is counted up. (GUI name: Timestamp format)
UID	No function; for compatibility with rvs® portable; Possible value: empty string, '*'
VFTYP	In this parameter you indicate the mode of files in format 'F' or 'V'. Possible values: <ul style="list-style-type: none"> - X: textfile (is equivalent to record mode 'TXT' in 'Create Send Job') - U: binary file (is equivalent to record mode 'BIN' in 'Create Send Job') Please see Record Mode parameter in chapter 4.5 for further information. (GUI name: Record Mode)

15 Appendix

15.1 ODETTE Protocol

It is the purpose of the ODETTE File Transfer Protocol (OFTP) to ensure the reliable transfer of a data set. The OFTP enters a protocol session with the OFTP on the remote rvsEVO station which logically runs on top of the linedriver connection.

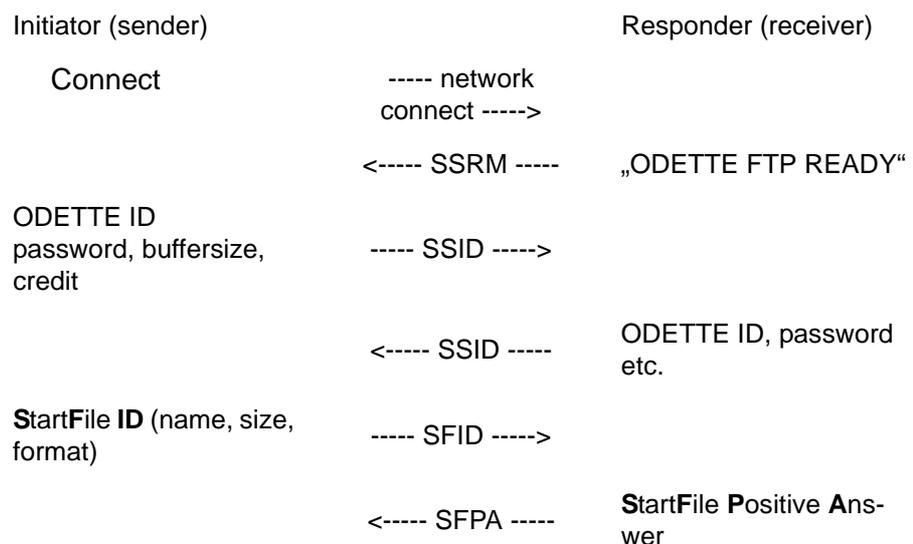
After the OFTP session has started, both sides exchange their ODETTE IDs and passwords, negotiate some parameters, like ODETTE exchange buffer size, ODETTE credit value (the number of buffers the sending side can send without waiting for a response), and exchange information about name, approximate size and format of the data set to be transferred.

During transfer, a compression and decompression of data is performed. After the data have been transferred, the byte count is checked between both sides. After the data set has successfully been stored, a receipt is sent to the sending station. If the transfer has been disrupted, for example by a link failure, the OFTP protocol provides a mechanism that allows to restart the transfer at the point of rupture.

'Change direction' feature allows the receiver to become sender and to send acknowledgments and data sets.

For protocol details kindly refer to the publications of the ODETTE and VDA groups: "ODETTE Specifications for File Transfer".

Shown below is the general but simplified message flow within an ODETTE session. The sending side acts as initiator, the receiving side as responder.



'n' data records	----- DATA -----> ----- DATA -----> ----- DATA -----> ...	
	<----- CREDIT -----	send credit value 'n'
send 'n' data records	----- DATA -----> ----- DATA -----> . .	
EndFile ID (byte count)	----- EFID ----->	
	<----- EFPA ----- or <----- EFNA -----	EndFile Positive Answer (if store successful and byte count correct)/ EndFile Negative Answer
Change Direction	----- CD ----->	
	<----- EERP ----- or <----- NERP -----	End-to-End Response/ Negative End-to-End Response (acknowledgment)
	<----- ESID -----	End-Session ID
network disconnect		network disconnect

A

Active Panel 111

B

bastion 147

Batch file 89

C

CA certificate 69, 142

certificate-properties 32

CertificateUsageDefinition 141

Client 155

Command Tools

activateStation 95

archiveJobs 126

commitUpdateStation 173

convertAndSend 106

convertFile 132

createBackup 133

createSendJob 100

deleteJob 115

deliverCertificate 127

doRecover 135

getCertificateList 129

getJob 118

getJobInfoList 120

getJobList 117

handleEERP 119

holdJob 115

importCRL 130

importTSL 130

login 132

orderConfiguration 173

prepareUpdateStation 173

releaseJob 116

replaceCertificate 128

requestCertificate 127

restartJob 116

rvsEVOService 132

rvsservice 29

sendJournal 131

setclientcp 132

setcp 132

showMonitorLog 93

showMonitorLogFile 94

startKeyMgn 129

startServer 89

startService 28

stopServer 89

terminateSession 131

updateStationList 74

userManagerClient 132

Comment 79

Configuration files 79

Connection type

ISDN 16

TCP/IP 17

CONTACT 73

Customizing configuration files 37

D

delaftersend.bat 87

Derby 167

E

EERP 59

EERP_OUT 119

Encryption

Electronic signature 137

encryption formats 137

principle and sequence 138

Environment Parameters

ARCDIR 38

BackupStartup 38

Browser 38

CentralJournalInstance 38

CleanupDays 38

CleanupInterval 38

CleanupTime 38

ConnSetupFailWaitTime 39

DB 39

Description 39

EngdatConfigFile 39

HelpFile 39

HostAllowFile 39

HostDenyFile 39

INBOX 39

JobstartConfigFile 39

JournalFilenamePrefix 39

LOGDIR 39

LooptestNeighbourSID 39

ManagementConfigFile *40*
MaxMonLogCount *40*
MaxMonLogSize *40*
MaxRevisionLogCount *40*
MaxRevisionLogSize *40*
MaxSessions *40*
OFTPTimeout *40*
PersistenceArchive *40*
RedoLog *40*
RMIServiceHost *40*
RMIServiceName *40*
RMIServicePort *41*
RvsStartScript *41*
SendJournalInterval *41*
SessionAliveTimeout *41*
SessionWaitTime *41*
StationsConfigFile *41*
TEMP *41*
Timestamp *41*
Traceltem *42*
TransmissionFailWaitTime *42*

G

GUI *26*

H

handleEERP *59*
handleMangement.bat *87*

I

importComSecureKeyPair *31*
ISDN connection *16*

J

jobFilter *75*
JobStart *75*
JobStart parameters
 Conversion table *80*
 Direction *80*
 Enabled *80*
 New Directory *80*
 New Filename *80*
 Parameter Handling *80*
 Process *81*
 Processing Class *81*
 processingClass *81*

Record Handling *81*
Replace *81*
Send Attempts *81*
sendAttempts *81*
Shell *81*
SID of Destination *81*
SID of Originator *81*
Synchronized *82*
Timeout (Sync.) *82*
Timestamp *82*
Timestamp Format *82*

jobstart.bat *87*
jobstart_detailed.bat *87*
JobstartConfigFile *43, 44*
journal.bat *87*

K

Kommandozeilentools
 rvsbat *181*
 startGUI *28*

M

Migration *30*

N

network parameters
 Card number *54, 62*
 client authentication *53*
 Dial Retry Count *62*
 Dial Retry Wait Time *62*
 enabled *51, 53, 54, 56*
 IP Address *51, 53*
 ISDN Address *54, 62*
 ISDN Facilities *54, 62*
 ISDN Protocol *54, 62*
 ISDN Terminal Identifier *55, 62*
 ISDN Userdata *62*
 Keystore file name *53*
 Local IP address *56*
 Local port *56*
 max. incoming sessions *51, 53*
 max.incoming sessions *55, 57*
 OrdinalNumber *55, 57*
 Port *51, 53*
 RCV timeout *55, 57*
 Receiver Number *52, 53*

ReceiverNumber *62, 64*
restart timeout *52, 53*
Router IP address *57, 64*
Router Port *57, 64*
SDN Userdata *55*
timeout *52, 53, 55, 57, 62, 64*
trusted certs keystore file name *54*
Type *55, 62*
X.25 Address *55, 57, 62, 64*
X.25 Closed User Group *55, 63*
X.25 DBit *55, 57, 63, 64*
X.25 Facilities *56, 57, 63, 64*
X.25 Modulo *57, 64*
X.25 PacketSize *56, 57, 63, 64*
X.25 Userdata *56, 57, 63, 64*
X.25 Window Size *56, 58, 63, 65*

O

ODETTE parameters
Authentication *58*
Certificate Validation Type *59*
Compression *59*
Encryption *60*
Encryption Algorithm *60*
End to End Response in *59*
End to End Response Out *59*
Exchange Buffer Credit *59*
Exchange Buffer Size *58*
File Service Proxy *60*
Odette ID *60*
OFTP Version *60*
PKI *60*
Receive Password *60*
Restart *60*
Security *61*
Security Feature Set *60*
Send Password *60*
SFIDDESC as Filename *61*
Sign *61*
Sign ERP *61*
VDSN charse *61*
Odette port *74*
OFTP Proxy *147, 151*
OUTBOX *40*

R

Receive *60*
Remote GUI *155*
RMIServiceHost *48*
RMIServiceName *48*
\$RVS_HOME *14*
RVS_HOME *30*
rvsbat
 CREATE /SEND parameters *182*
 FAILURE Command *187*
 Jobstart Parameters *187*
 RESETR command *185*
 SEND Command *182*
 SENDJOB Command *186*
rvsConfig.xml *47*
RVSENV *30*
rvsEVO version *27*
rvsStationlist.xml *49*
\$RVSTINY_HOME *14*

S

sendback.bat *88*
SNMP *42*
SNMP parameters
 Active *42*
 Interval *43*
 IP address *43*
 Log-Level *43*
 Port *43*
StationsConfigFile *43, 44, 46*

T

Typographic conventions *14*

U

User Management *159*

V

VDSN *82*
virtual station *66*

W

What is rvs® *9*
What rvs® is not *9*