System Administrators Manual

Bartrack 6

Prevas

Prevas AB
Box 1909
S-651 19 Karlstad

Phone: +46 54 147400
Fax: +46 54 147499

This manual was produced by Prevas AB
KS001b02/en v3

Contents

Introduction 7

Security 11

Server 17

Web Server 37

Introduction

About This Manual

Which Version

This manual covers the 6.0 release of the Bartrack system. If you do not know which version you have, do the following:

1. Start Bartrack (either web or VT session)
2. From web: Choose the menu item About/About Bartrack
   From VT: Press the **F10** key
3. A display box shows which version of Bartrack that is installed.
4. Click on the button OK (web) or press the key **DO** (VT)

The Intended Audience

This manual is written for a reader with prior knowledge of OpenVMS. Some of the commands shown in this manual might be unsafe to execute. If you have the slightest hesitation or feel insecure; don't.

A faulty or command or wrong parameters might stop Bartrack from functioning.

Scope of the Manual

This manual does not cover:

1. Normal non-Bartrack system administrator tasks.
2. Usage of the Bartrack program. See the User's Guide - KS001B01 for information on using Bartrack.
3. Non-standard error conditions. In which case you should contact the Prevas Support (according to your support agreement).

You can reach the helpdesk by phone:

Tel: +46 54 147444
Depending on your support agreement with Prevas, this service is open 24 hours a day, seven days a week.

You can also send an e-mail to support.sfc@prevas.se

Conventions

In order to keep the manual as easy to read as possible, the following conventions are used:

A request to press a key on the keyboard is written like this:
Press the **F10** key.

A request to choose a menu or menu item is written like this:
Choose the menu item System/About.

A request to click on an on-screen button is written like this:
Click on the button **Product structure**.

A sequence of instructions to be carried out in order is written like this:

1. This is step 1
2. This is step 2
3. etc.

A single instruction to be performed is written like this:

- This is a single action

A list of items is written like this:

- This is an item
- This is another item
- etc.

About Bartrack

What Does It Do?

Bartrack is a traceability system for individuals produced in factories specialized in electronics manufacturing. The purpose of using Bartrack is to:

- Create serial numbers for the individuals
- Keep track of the assembled parts of an individual
- Print out bar-code labels
- Keep master traceability systems updated
- Monitor the product structure

Every produced individual with traceability demands is assigned a serial number by Bartrack. To this unique number, assorted information about the individual is connected, for example product number, structure information, manufacturing week.

When an individual is ready to leave the factory, its information in Bartrack is deleted. But since Bartrack is one of the sources of traceability information, it first sends the information to the master traceability database, such as Tracy.

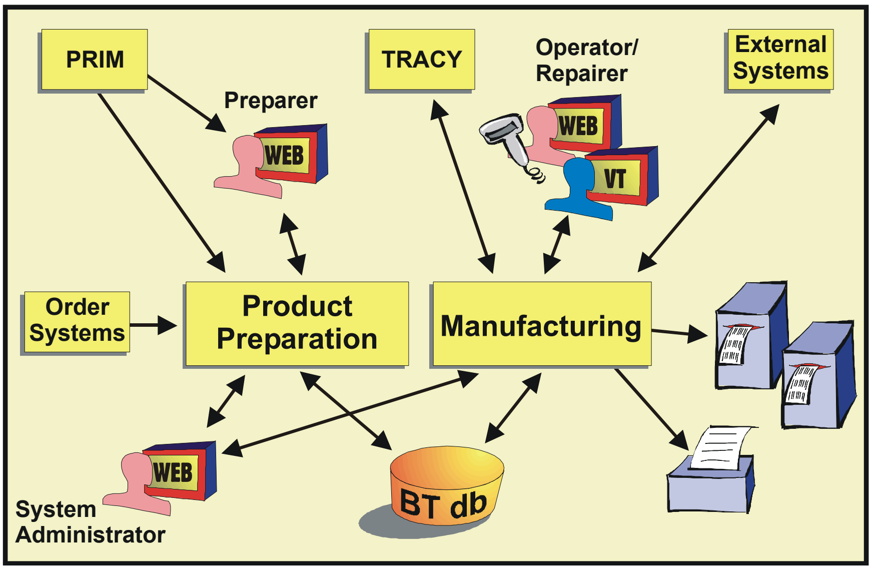
System overview

The Bartrack system consists mainly of two units:

- The Prepare/Administration unit
- The Manufacturing unit

Both of these units have access to the Bartrack Rdb-database.

The communication between Bartrack and other systems all go through these two units, except for the direct access the System Administrator has to the Rdb-database.



A system overview

There are four main user categories that use Bartrack:

- Operator
- Repairer
- Preparer
- System Administrator

They put in information, request tasks and read output information.

In addition to the real users, there are a number of virtual users, for example the Tracy interface or the PRIM interface.

Security

Accessing Bartrack

There are a number of safety devices against unauthorised access to Bartrack. The details are discussed in separate sections of this manual, since they belong to different parts of the Bartrack system.

This is a schematic view of the access paths to Bartrack:

Access path to Bartrack for web client

Web		Bartrack	
Client	**Intranet**	**User Administration**	**User Profile**
The web browser containing the Bartrack web application must be reachable	The Bartrack server must be reachable	The user must exist as a user in Bartrack and know the password	The user must have the privilege to the Bartrack functions

Access path to Bartrack for VT client

OpenVMS	Bartrack	
UAF	**User Administration**	**User Profile**
The user must have an account, and must have either the `BARTRACK_VT` or `BARTRACK_ADMIN` identifier	The user must exist as a user in Bartrack	The user must have the privilege to the Bartrack functions

Users

It is recommended that all users in Bartrack should have unique user-IDs and no collective logins should be allowed. This will make it possible to track each user's doings in Bartrack, as well as having different privileges for different users.

VT-users log in via their normal OpenVMS-login, which is used to determine their rights to the Bartrack executable and which privilege profile to use.

Web-users start their browser and click on the link to Bartrack. Then they will have to log onto Bartrack the normal way. The Bartrack username is used to determine which privilege profile to use.

Every access to the Bartrack database is controlled by OpenVMS and Oracle Rdb. Bartrack uses different privilege profiles to allow different menu entries for each user.

Since every user accesses Bartrack from their own username, it is easy to control access and processing.

Files

OpenVMS Server

Bartrack executable files are protected by normal OpenVMS operation.

The users who have the identifier **`BARTRACK_VT`** have execute access to the file **`BAR_EXE:UIV_MAIN.EXE`** that is run by the command file pointed to by the symbol **`BARTRACK`** when a Bartrack VT-session is started.

This file in its turn has the identifier **`BAR_DB_SUBSYSTEM`** that gives it access to the Bartrack database.

If you, for any reason, changes the physical disk on which this file reside, you have to manually issue the following command:

```
$ SET VOLUME/SUBSYSTEM <disk name>
```

Where **`<disk name>`** is the name of the new disk. This enables the protected subsystem for the disk.

The users who have the identifier **`BARTRACK_ADMIN`** have access to all the Bartrack files.

Web Client

For the web interface to function, the java.policy file must contain the path to the prevas.policy file found at the Bartrack web server. This path is needed for the Java Run-time Environment to be able to verify that the Bartrack web client files are authentic.

Bartrack relies on a special file. The file, .java.policy, must be copied from the web server the first time a user tries to start Bartrack. The file contains the security settings for the Java client, which enables Bartrack to access the clients disk. This is needed for the barcode interface.

To set this path, you will have to execute the Policyinstaller.exe found in the Bartrack directory of the web server.

The policyinstaller will prompt for the URL of the Bartrack website, where a file called prevas.key is located. The prevas.policy and the prevas.key in cooperation will ensure that only approved code will be executed on the client.

If the user hasn't copied the file before, it must be done before Bartrack can function. This is done by clicking on a link which will prompt the user for a name and a location of the file. The file should always be copied to the user's own directory, and should always be called .java.policy.

N.B. This file will grant all Bartrack web applications read/write access to the client's local disks. No other web applications will be granted read/write access.

Database

The database used by Bartrack is an Rdb database from Oracle.

Bartrack uses the OpenVMS protected subsystems to guard the database against unauthorised access. The only allowed accesses are from files having a special key, an identifier, to the database.

Files that should access the database need to be located on a disk having the **BAR_DB_SUBSYSTEM** identifier.

If a user has the **BARTRACK_ADMIN** identifier, or system privileges, the database can also be accessed.

Monitoring and Reliability - SFQPS

Bartrack uses the SFQPS (Shop Floor Quality Process Surveillance) to keep Bartrack up at any cost. The program is essentially two processes guarding each other, the watchdog (SFQ_SSSWDOG) and the supervisor (SFQ_SSSSUPV). The supervisor also guards any processes it has stored in its database. Should a guarded process cease to function, it will be started by the supervisor again. In order for this procedure to work, the **BAR_PROCESS** batch queue must be operating.

Every twenty seconds, a question is sent to the monitored processes, and within twenty seconds, a response is sent back. If there is no response, the process will be re-started.

The SFQPS relies on mailboxes for this communication. If custom processes are to be added to the SFQPS monitor list, they will have to comply with the OpenVMS mailbox protocol.

The following Bartrack processes are monitored by SFQPS:

BAR_LOGGING	A process for Bartrack logging
BAR_SFT_STR	A process handling structure requests
BAR_SFT_UPD	A process handling update information
BAR_SFT_SHP	A process handling ship requests
BAR_SFT_IDI	A process handling individual requests
BAR_SFT_CRT	A process handling create requests
BAR_TIF	A process handling test system files
BAR_ORC	A process handling order system files

Should any of the processes above die or fail, the SFQPS will start them again, ensuring that the vital processes for Bartrack are alive. Any process restarted by SFQPS is logged in the directory **SFQPS_LOG**.

The SFQPS database is loaded with the processes to guard at start-up.

Logical names for SFQPS

There are several logical names that are used by the SFQPS during operation.

Table of logical names for SFQPS

Logical name	Value	Explanation
DISK_SFQPS	<disk name>:[SFQPS.]	The root of the SFQPS directory structure.
DISK_SFQPSDB	<disk name>:[SFQPSDBDIR.]	The directory for SFQPS database.
DISK_SFQPSAIJ	<disk name>:[SFQPSAIJ.]	The root of the SFQPS AIJ structure (not used).
SFQPS_AIJ	DISK_SFQPS:[AIJ]	The directory for SFQPS AIJ-files (not used).
SFQPS_COM	DISK_SFQPS:[COM]	The directory for SFQPS command files.
SFQPS_DB	DISK_SFQPSDB:[DB]	The directory for the SFQPS database.
SFQ_SSS_DB	DISK_SFQPSDB:[DB]SFQPS_DB	The database file.

SFQPS_DBCRE	DISK_SFQPS:[DBCRE]	Directory for SFQPS database creation scripts and commands.
SFQPS_EXE	DISK_SFQPS:[EXE]	The directory for SFQPS executables.
SFQPS_LOG	DISK_SFQPS:[LOG]	The directory for SFQPS log files..
SFQSSSSHR	DISK_SFQPS:[EXE]SFQSSSSHR.EXE	The SFQPS executable.

SFQPS processes

Process name	Surveyed by SFQ_SSSSUPV	Surveyed by SFQ_SSSWDOG	User name	Description
SFQ_SSSSUPV		Yes	SFQ	Supervisor
SFQ_SSSWDOG	Yes		SFQ	Watchdog
BAR_SFT_STR	Yes		BAR_SFT	Structure requests
BAR_SFT_UPD	Yes		BAR_SFT	Update requests
BAR_SFT_SHP	Yes		BAR_SFT	Ship requests
BAR_SFT_IDI	Yes		BAR_SFT	Information requests
BAR_SFT_CRT	Yes		BAR_SFT	Create requests
BAR_ORC	Yes		BAR_SFT	Order import
BAR_DTF	Yes		BAR_SFT	DMQ-to-file
BAR_TIF	Yes		BAR_SFT	Test system interface
BAR_LOGGING	Yes		BAR_SFT	Bartrack logging

Adding processes to SFQPS

To add processes that are to be surveyed, you have to run the following command:

```
$ @SFQPS_DBCRE:<insert command>
```

Where `<insert command>` is substituted with the appropriate `.com` filename. The command inserts the start command for that particular process into the SFQPS database, and SFQPS then monitors the process.

After a command is inserted into the SFQPS database, the SFQ_SSSSUPV process must be restarted. To stop and start SFQPS and SFQ_SSSSUPV, see next section.

Insert Commands

Logical name	Explanation
INS_SFQPS.COM	This command inserts the watchdog into the SFQPS database.
INS_SFTSTR.COM	This command inserts BAR_SFT_STR into the SFQPS database.
INS_SFTUPD.COM	This command inserts BAR_SFT_UPD into the SFQPS database.
INS_SFTSHIP.COM	This command inserts BAR_SFT_SHP into the SFQPS database.
INS_SFTIDINFO.COM	This command inserts BAR_SFT_IDI into the SFQPS database.
INS_SFTCRT.COM	This command inserts BAR_SFT_CRT into the SFQPS database.
INS_ORC.COM	This command inserts BAR_ORC into the SFQPS database.
INS_DTF.COM	This command inserts BAR_DTF into the SFQPS database.
INS_TIF.COM	This command inserts BAR_TIF into the SFQPS database.
INS_LOGGING.COM	This command inserts BAR_LOGGING into the SFQPS database.

Removing processes from SFQPS

To remove processes that are not to be surveyed any more, you have to delete the process from the SFQPS database.

This is done using normal SQL on the SFQPS database **`SFQPS_DB`**. For each process there will be an entry in the **`SSS_PROCESS`** and the **`SSS_PROCESS_DATA`** table.

After a process is removed from the SFQPS database, the SFQ_SSSSUPV process must be restarted. To stop and start SFQPS and SFQ_SSSSUPV, see next section.

Starting and stopping SFQPS

Normally, SFQPS will be started from **`BARTRACK_STARTUP.COM`** during the computer's boot sequence. Should you want to start SFQPS manually, you have to be logged in as the user SFQ, then use the following command:

- **`$ SUBMIT/USER=SFQ SFQPS_COM:SFQPS_STARTUP.COM`**

To stop Bartrack on the server, you have to be logged in as the user SFQ, then use the following command to stop the SFQPS surveillance on the server:

- **`$ SUBMIT/USER=SFQ SFQPS_COM:SFQPS_SHUTDOWN.COM`**

Server

Specifications

For a full specification, read the “Target environment specification (TES)”, document number KS001f05/en.

The TES contains the minimum hardware and software specifications required in order to run Bartrack.

Licence handling

How it Works

Bartrack uses the standard OpenVMS Licence Management Utility. Each time a Bartrack client requests access to the Bartrack program on the server, Bartrack verifies that the registered licence is still valid. If so, the clients are allowed to run.

Licence Expiration

When the remaining time to the licence's expiration date is less than 30 days, each user of Bartrack will receive a notification about this fact when they log on. After the notification is shown, Bartrack operates in the normal way.

When the expiration date is reached, no user can log on to the Bartrack system, and an error message is shown.

Prolonging the Licence

Use the standard OpenVMS Licence Management Utility to enter the new licence into the system.

A new licence can be obtained by contacting Prevas (see page 7 for address and telephone number).

Queues

Batch Queues

There are two batch queues used by Bartrack:

- **`BAR_BACKUP_QUEUE`** The Bartrack backup batch queue

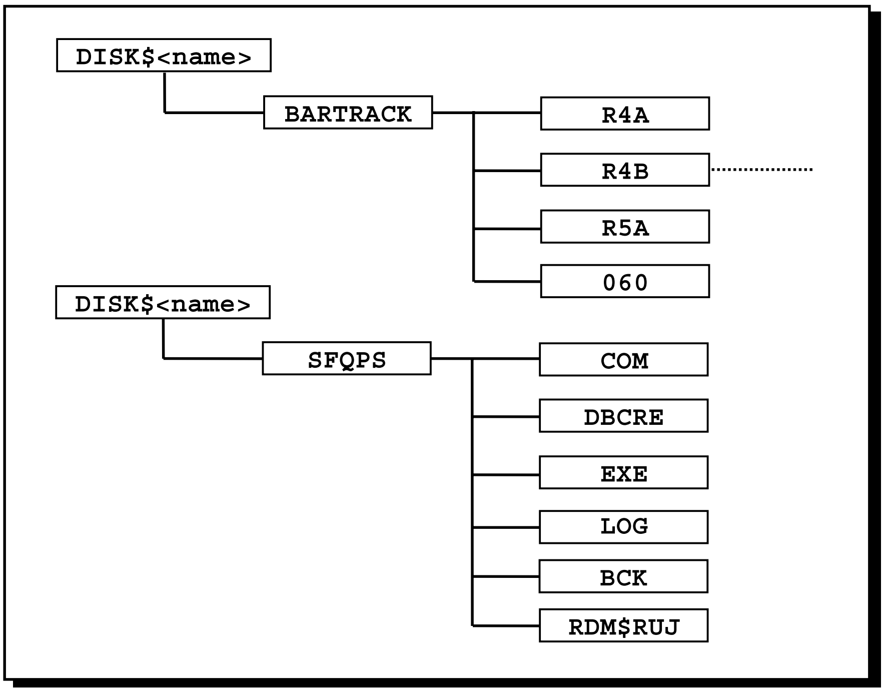
- **BAR_PROCESS** The Bartrack process batch queue

In **BAR_BACKUP_QUEUE** there will be only one entry during five minutes each night: **BAR_BACKUP**.

In **BAR_PROCESS** there may be up to five entries at normal operation: **TRACY_BAR**, **BAR_TRACY**, **BAR_DBI**, **PRIM_BAR** and **BAR_CHECK_BATCH**.

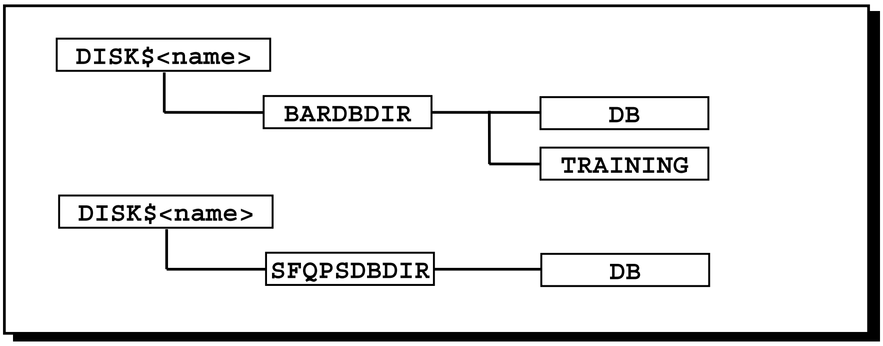
File Structure and Files

Bartrack is set up to be distributed over a number of physical disks, or on one disk. Every directory has its own logical name, giving direct access to it, regardless of its physical location.



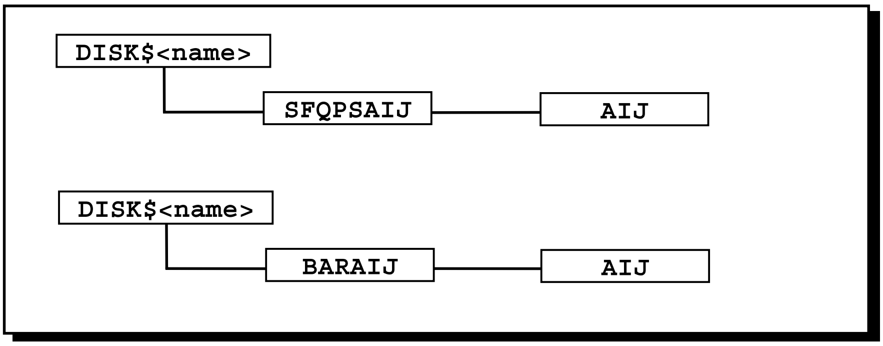
Server file structure: The Bartrack and SFQPS top directory structure

Bartrack is installed in its own directory named `BARTRACK`. This directory in its turn contains directories named after that particular release's version. Under the version `060` (for example) all the specific Bartrack files and directories related to the 6.0 release are located. The current Bartrack release is defined during start-up.



Server file structure: Databases

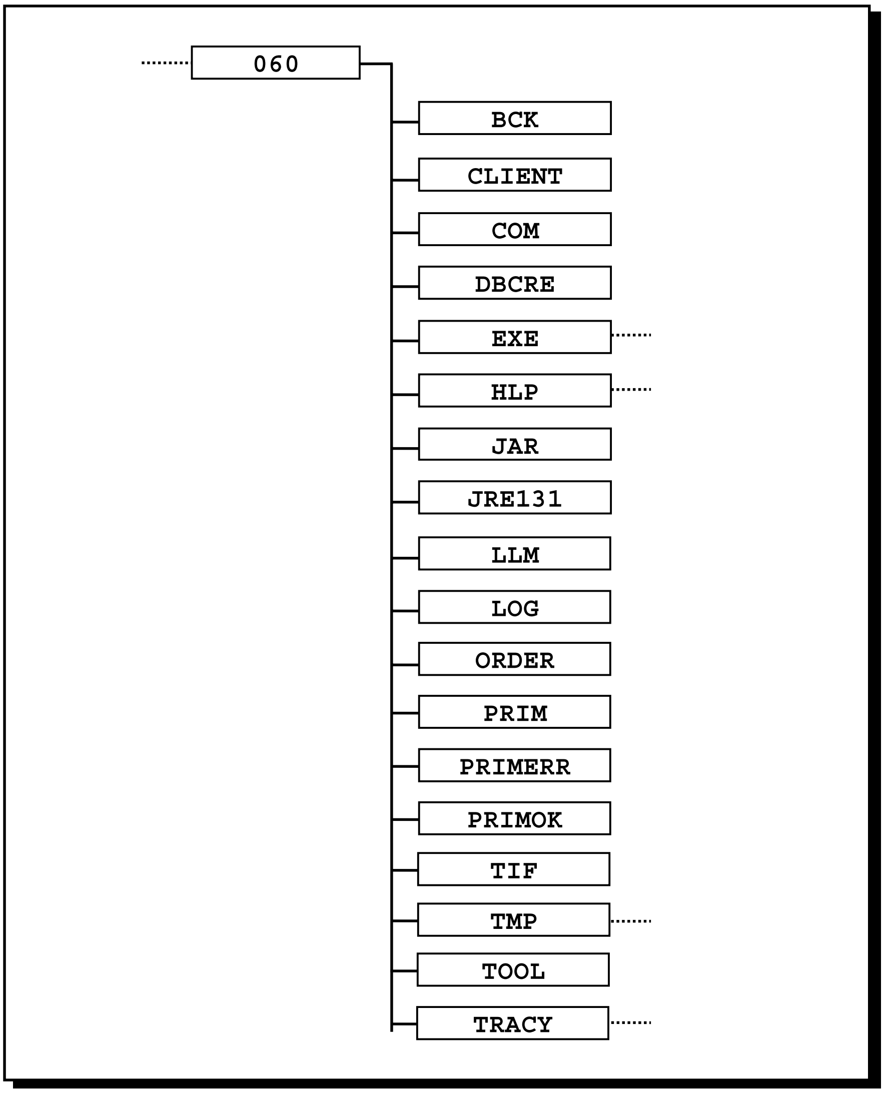
The databases are located in their own directories, separate from each other. The directory TRAINING does not contain a database, instead it contains the scripts to fill an empty database with exercises for training purposes.



Server file structure: AIJ files

The After Image Journals for the Bartrack database is stored in its own directory **BARAIJ:AIJ**.

The **SFQPSAIJ:AIJ** is not used, since no backups of that database is necessary.



Server file structure: the Bartrack 6.0 release

Servers directory contents

Logical name	Directory name	Explanation
DISK_BARAIJ	<disk name>:[BARAIJ.]	The disk where the Bartrack AIJ files reside.
DISK_BARTRACK	<disk name>:[BARTRACK.060.]	The current release. Many releases can exist in parallel, but only the one assigned to the logical name `DISK_BARTRACK` is used. This allows for easy change between current and previous releases.
DISK_BARDBDIR	<disk name>:[BARDBDIR.]	The disk where the Bartrack database resides.
BAR_AIJ	DISK_BARAIJ:[AIJ]	The AIJ files for the database backup.
BAR_DB	DISK_BARDBDIR:[DB]	This directory contains all the Rdb files that constitute the Bartrack Rdb database.
BDB	BAR_DB:BAR	The Bartrack database.
BAR_TRAINING	DISK_BARDBDIR:[TRAINING]	The scripts for the exercises.
BARTRACK_ PCTRANSFER	DISK_BARTRACK:[TIF.INFILES]	This directory contains files that are sent to Bartrack from a test system using file transfer.
BAR_BACKUP	DISK_BARTRACK:[BCK]	This directory contains the latests backup of the Bartrack database.
BAR_CLIENT	DISK_BARTRACK:[CLIENT]	This directory contains the Java code for the web server. The code is distributed to the clients.
BAR_COM	DISK_BARTRACK:[COM]	This directory contains all the command files.
BAR_DBCRE	DISK_BARTRACK:[DBCRE]	This directory contains all the SQL script files required to create a new, empty, database.
BAR_EXE	DISK_BARTRACK:[EXE]	This directory contain all the executable files
BAR_HLP	DISK_BARTRACK:[HLP]	This directory contains on-line help files.
BAR_HLP_VT	DISK_BARTRACK:[HLP]	This directory contains on-line help files.
BAR_JAR	DISK_BARTRACK:[JAR]	This directory contains the client Java code for the web client.
BAR_LABEL_DIR	DISK_BARTRACK:[TMP.LABEL]	This directory contains temporary files for printout of labels.
BAR_LAYOUT_DIR	DISK_BARTRACK:[LLM]	This directory contains layout files for labels.
BAR_LOG	DISK_BARTRACK:[LOG]	This directory contains some of the log files. User specific log files are located in the user's SYS$LOGIN directory.
BAR_ORD_ERROR_ DIR	DISK_BARTRACK:[ORDER.ERROR]	This directory contains all faulty order rows in a single file for each day.
BAR_ORD_IN_DIR	DISK_BARTRACK:[ORDER.IN]	This directory contains order files received and not yet imported.
BAR_ORD_SAVE_ DIR	DISK_BARTRACK:[ORDER.SAVE]	This directory contains completed order rows in a single file for each day.
BAR_REPORT_DIR	DISK_BARTRACK:[TMP.REPORT]	This directory contains temporary files for printout of reports.
BAR_SFQ_ TRANSFER	DISK_BARTRACK:[PRIM]	This directory contains delivered files from SPU.
BAR_SFQ_ TRANSFERERR	DISK_BARTRACK:[PRIMERR]	This directory contains erroneous files not read into Bartrack.
BAR_SFQ_ TRANSFEROK	DISK_BARTRACK:[PRIMOK]	This directory contains files correctly read into Bartrack.
BAR_TCY	DISK_BARTRACK:[TRACY]	This directory contains several directories for the Tracy communication.
BAR_TEMP	DISK_BARTRACK:[TMP]	This directory contains two directories for temporary

		printout information. There should be no files in this directory.
BAR_TOOL	DISK_BARTRACK:[TOOL]	Assorted site-specific COM-files. These are the only files available for modification by site personnel.

All installation files for the clients can be found in the directory **`CLIENT`**. The logical **`BAR_CLIENT`** points directly to this directory.

Bartrack Tools Directory

The **`BAR_TOOL`** directory contains files that are changed to suit the site's needs. Some files may be modified during installation, and some files may be modified when, for example, mail-addresses shall be changed.

It is recommended that any custom files made by the site be placed in another directory. This directory might change from version to version without notice.

Internal Interfaces

To Web client (RMI)

The Bartrack web interface is programmed in Java and distributed via TCP/IP and normal web technology. The technique used to handle the clients is called RMI (Remote Method Interface).

Set-up RMI for Bartrack

No special steps are required on RMI's set-up.

Set-up Bartrack for RMI

No special steps are required on Bartrack's set-up.

Logical names for RMI interface

There are several logical names that are used by RMI during operation.

Table of logical names for RMI

Logical name	Value	Explanation
BAR_JAR	DISK_BARTRACK:[JAR]	This directory contains the client Java code for the web client.
BAR_JRE_VERSION		The version number of the JRE instaled.
BAR_RMI_PROCESSES		The number of RMI process started on speculation.
HTTP_BASE_MST	WWW_SYSTEM:HTTP_BASE_SYSTEM	
WWW_DIRSERV_ACCES	<node>::BAR	
WWW_DIRSERV_OBJECT	WWWDIR	
WWW_JAVA_ACCES	<node>::BAR	
WWW_JAVA_OBJECT	WWWJAVA	
WWW_MAX_CGILIB_CONTENT	512000	
WWW_ROOT	DISK_BARTRACK:[HTTPSERVER.]	

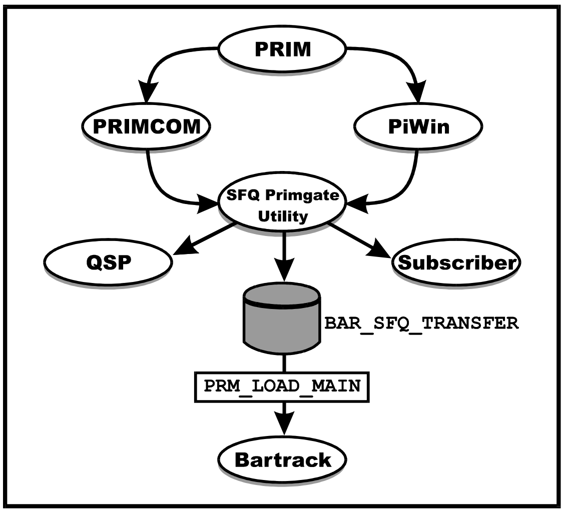
WWW_SERVER_PORTS	80	
WWW_SYSTEM	WWW_ROOT:[SYS TEM]	

External Interfaces

To PRIM (SFQ Primgate Utility)

PRIM is a product register used at Ericsson.

The SFQ Primgate Utility (SPU) converts PRIM data files, ordered from PRIMCOM or Piwin, into a text file format suitable for database loading. SPU delivers these text files to any application that has been set up as a subscriber. In the process, SPU can apply filters. In order to receive any PRIM information, Bartrack requires SPU to be installed on one node in your network (may be the same node as Bartrack).



Picture of PRIM import sequence via SPU

For more information about SPU, see the “SFQ Primgate Utility Installation and User Guide”, 198 17-LZY 236 26 Uen.

Set-up SPU for Bartrack

SPU must be configured to deliver PRIM text files to Bartrack in the BAR_SFQ_TRANSFER directory (see "File Structure and Files" on page 19). Two system logicals sets up the subscription list and controls the Primgate operation for each subscriber.

Logical names for SPU interface

There are several logical names that are used by SPU during operation. Some of them are relevant for Bartrack. For a complete list, refer to “SFQ Primgate Utility Installation and User Guide”, 198 17-LZY 236 26 Uen.

Table of logical names used by Bartrack

Logical name	Value	Explanation

BAR_SFQ_TRANSFER	DISK_BARTRACK:[PRIM]	This directory contains delivered files from SPU.
BAR_SFQ_TRANSFEROK	DISK_BARTRACK:[PRIMOK]	This directory contains files correctly read into Bartrack.
BAR_SFQ_TRANSFERERR	DISK_BARTRACK:[PRIMERR]	This directory contains erroneous files not read into Bartrack.

The following logicals are defined in the command file **SYS$STARTUP:SPU_STARTUP.COM**:

Table of some of the logical names for SPU

Logical name	Value	Explanation
DISK_SPU	<disk name>:[SPU.]	The root directory for SPU.
SPU_SUBSCRIBER	<subsc1,subsc2,subscn>	List of subscribers, separated with comma (,). One of them must be BAR.
SPU_BAR_DESTINATION	BAR_SFQ_TRANSFER:	Target directory for the PRIM text file.

Examples:

1. The SPU_SUBSCRIBER line before Bartrack set-up:

```
DEFINE/SYSTEM/NOLOG SPU_SUBSCRIBER "QSP"
```

   The same line after modification for Bartrack:

```
DEFINE/SYSTEM/NOLOG SPU_SUBSCRIBER "QSP,BAR"
```

2. The destination set-up for Bartrack:

```
DEFINE/SYSTEM/NOLOG SPU_BAR_DESTINATION
      "<Bartrack node>::BAR_SFQ_TRANSFER:"
```

Set-up Bartrack for SPU

All PRIM text files transferred from SPU are delivered in the Bartrack directory **BAR_SFQ_TRANSFER**. If SPU is installed on the Bartrack node, no further actions are required. If, however, SPU is installed on a remote node, that node must have write access to the **BAR_SFQ_TRANSFER** directory. This may be accomplished either by proxy access or by granting world write access to the directory.

Proxy example:

```
UAF> ADD/PROXY <SPU node>::SYSTEM BAR /DEFAULT
```

Operation

Bartrack queues the **PRIM_TRANSFER_MAIN** job onto the **BAR_PROCESS** batch queue. Every fourth hour, it checks the **BAR_SFQ_TRANSFER** directory for new files from SPU. A new file is read into the Bartrack temporary product database. Should anything go wrong (for example, some data not being loaded into Bartrack) the whole offending file is copied to **BAR_SFQ_TRANSFERERR**. If the transfer is OK, the file will be copied to **BAR_SFQ_TRANSFEROK**. In any case, the **BAR_SFQ_TRANSFER** directory is cleared of the read files.

The files stored in these three directories are normal ASCII text files, and thus they can be edited and read. If you edit an offending file (to clear out any problems) you can copy it back into the **BAR_SFQ_TRANSFER** directory. From this directory it is read once again when the next **PRIM_TRANSFER_MAIN** job starts.

The Preparer then decides which products to transfer from the temporary product database to the live Bartrack product database for further preparation.

BEAmessageQ (BmQ)

The BEAmessageQ (formerly known as DECmessageQ or DmQ) is used for three purposes;

- to communicate with test systems
- to allow for external systems to request information or actions from Bartrack
- to send internal logging information

The T95_BAR_01 queue is owned by the receiving part (Testnet), all other queues are owned by Bartrack.

Communicating with test systems

Bartrack can receive individual statuses after test from a test system via BmQ. While reading the BmQ queue Bartrack updates the individuals test status in the database.

External systems

Bartrack can be controlled by an external system or application. A message is sent to Bartrack containing instructions on what Bartrack should do. Bartrack can reply back to the external system.

It is also possible to use the BarAPIx (Bartrack Auxiliary Programming Interface) to let PC-applications use Bartrack.

Logging

The Bartrack logs are all stored in the **`BAR_LOG`** directory (some exceptions exists. Please refer to "Logging" on page 63 for a full description of logging). This is accomplished by sending BmQ messages to a logging server, which converts the messages to files.

Message Recovery Service

All the out queues used by Bartrack has the Message Recovery Service (MRS) enabled. This will ensure that a message is delivered to the receiver in case of transport errors.

Multi-Reader Queues (MRQ)

Bartrack uses Multi-Reader Queues (MRQ) for some of the incoming queues from external systems. This makes it possible to add additional polling processes if a single process is unable to cope with the message load.

Incoming message queues

There are five MRQ queues used for receiving request messages from another system:

- BAR_TST_UPD_01 This queue is used for sending test status updates on individuals to Bartrack.
- BAR_TST_STR_01 This queue is used for sending structure requests to Bartrack.
- BAR_TST_SHP_01 This queue is used for sending ship requests to Bartrack.
- BAR_TST_IDI_01 This queue is used for sending individual information requests to Bartrack.
- BAR_TST_CRT_01 This queue is used for sending create individual requests to Bartrack.

The five queues are each read by a dedicated processes, which communicate with Bartrack: **BAR_SFT_UPD** , **BAR_SFT_STR** , **BAR_SFT_SHP** , **BAR_SFT_IDI** and **BAR_SFT_CRT**.

There is one queue used for internal logging.

- BAR_BAR_LOG_01 This queue is used for receiving alarms and logs meant for the logging server.

It receives messages from Bartrack and converts them to log files. This is done by the process **BAR_LOGGING** .

Receiving message queues

There may be several queues used for sending messages from Bartrack to other systems. When Bartrack is to send a message to another system, Bartrack first attaches to a temporary queue, and then sends the message.

There is one queue used by Bartrack to convert DmQ messages to files:

- BAR_FIL_01 This queue is used for sending information to a system that cannot use DmQ.

The BAR_FIL_01 queue is read by the dedicated processes **DmqToFile** , which converts the messages to files.

For more information about DmQ/BmQ, see the DECmessageQ documentation set.

Set-up DmQ/BmQ for Bartrack

Refer to the “Bartrack Installation Guide” (KS001f04) and the DmQ/BmQ manual for instructions on how to set up the DmQ/BmQ environment.

Set up Bartrack for DmQ/BmQ

Refer to the “Bartrack Installation Guide” (KS001f04) and the DmQ/BmQ manual for instructions on how to set up the DmQ/BmQ environment.

Logical names for DmQ/BmQ

There are several logical names that are used by DmQ/BmQ during operation.

Table of logical names for DmQ/BmQ

Logical name	Value	Explanation
BAR_ALA_DMQ	BAR_BAR_LOG_01	The queue name for DmQ/BmQ alarm messages.
BAR_LOG_DMQ	BAR_BAR_LOG_01	The queue name for DmQ/BmQ logging messages.
BAR_DMQ_EXE_DIR	<DISK>:[DMQ$V50.EXE]	The directory containing executables for DmQ/BmQ.
BAR_DMQ_BUS_NO	<your bus number>	The DmQ/BmQ bus number for the site.
BAR_DMQ_GROUP_NO	3022	The DmQ/BmQ group number for Bartrack.
BAR_SFT_CRT_DMQ_NAME	BAR_TST_CRT_01	The queue name for DmQ/BmQ create individual requests.
BAR_SFT_IDI_DMQ_NAME	BAR_TST_IDI_01	The queue name for DmQ/BmQ individual information requests.
BAR_SFT_SHP_DMQ_NAME	BAR_TST_SHP_01	The queue name for DmQ/BmQ ship requests.
BAR_SFT_STR_DMQ_NAME	BAR_TST_STR_01	The queue name for DmQ/BmQ structure requests.
BAR_SFT_UPD_DMQ_NAME	BAR_TST_UPD_01	The queue name for DmQ/BmQ update requests.

To Tracy (DEC/EDI)

Bartrack uses a DEC/EDI client to exchange data with Tracy. Bartrack can request individual information or structure from Tracy. Tracy can in its turn request from Bartrack to ship individuals.

DEC/EDI is a file transfer protocol, and all files are located in the directories below **`BAR_TRACY`** (see "File Structure and Files" on page 19).

Each file can contain several messages.

There are seven messages going from Bartrack to Tracy:

- Send individual data (B-record)
- Update individual data (C-record)
- Send additional number(s) (E-record)
- Send external serial number information (F-record)
- Send structure data (S-record)
- Request information about an individual (Y-record)
- Scrap an individual (Z-record)

From Tracy there are five messages:

- Individual information requested by Bartrack (B-record)
- Additional numbers, if any (E-record)
- Retail products, if any (F-record)
- Structure information requested by Bartrack (S-record)
- Request to ship individuals from Bartrack (X-record)

There are some duplicated words when it comes to Tracy. The following table explains the Bartrack meaning and the Ericsson equivalent.

Bartrack terminology	Ericsson/Tracy terminology	Description
Undefined product	External serial number	A product that has not been prepared for use. These products can be registered automatically at assembly or manually.
Additional number	Synonym number	Extra numbers that can be used to identify the individual. For example MAC-address, IPEI, suppliers serial number.

If a message is to be sent or not is decided for each product in the **Product Preparation** in Bartrack. Only necessary messages are sent, that is, even if a product has been defined to send information to Tracy at shipping, Bartrack can decide not to. But only if Bartrack knows that no information is changed since the last time Tracy knew the information.

For more information about EDI, see the DEC/EDI documentation set.

Set up DEC/EDI for Bartrack

Refer to the DEC/EDI-manuals on how to set up the EDI environment.

Logical names for Tracy

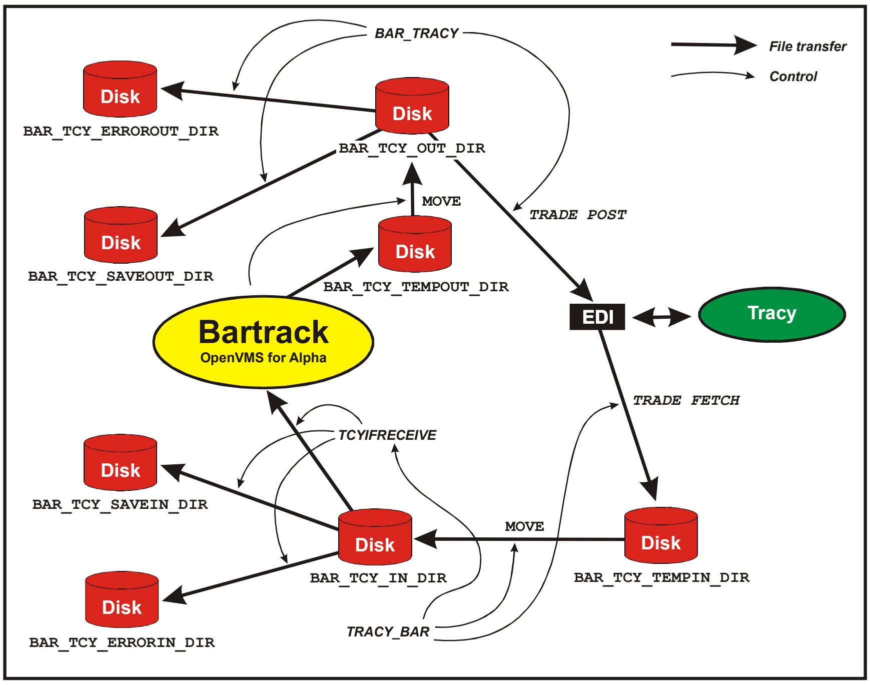
There are several logical names that are used by the Tracy interface during operation.

Table of logical names for Tracy

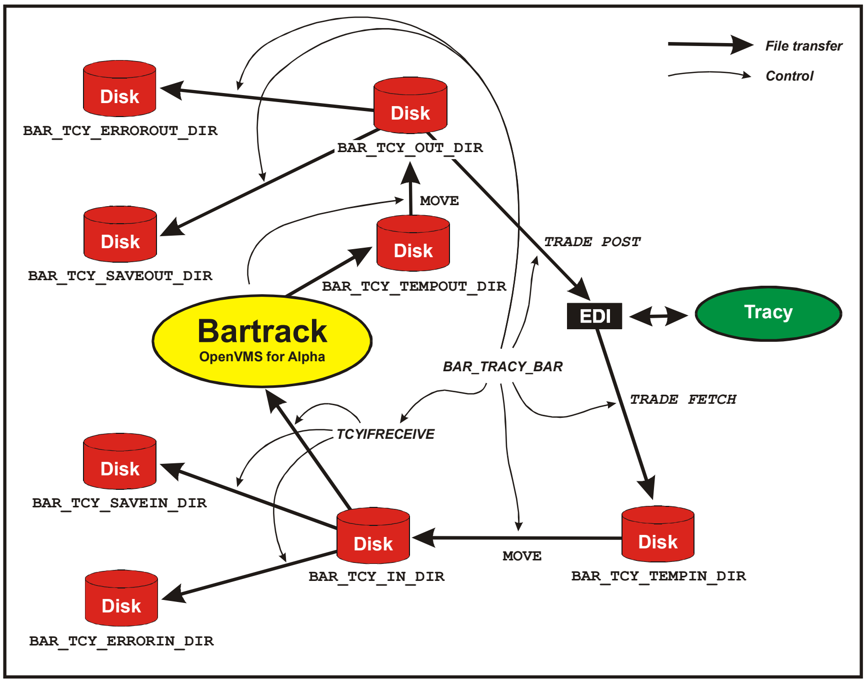
Logical name	Value	Explanation
BAR_TCY	DISK_BARTRACK:[TRACY]	The root of a directory structure, with six directories used for communication with Tracy. There should be no files in this directory.
BAR_TCY_ERRORIN_DIR	DISK_BARTRACK:[TRACY.ERRORIN]	Individuals from Tracy that could not be registered in Bartrack are stored here.
BAR_TCY_ERROROUT_DIR	DISK_BARTRACK:[TRACY.ERROROUT]	Files that could not be sent to Tracy are stored here.
BAR_TCY_IN_DIR	DISK_BARTRACK:[TRACY.PRODIN]	Files received from Tracy are stored here before they are taken care of by Bartrack.
BAR_TCY_OUT_DIR	DISK_BARTRACK:[TRACY.PRODOUT]	Files sent to Tracy are stored here before they are taken care of by DEC/EDI.
BAR_TCY_SAVEIN_DIR	DISK_BARTRACK:[TRACY.SAVEIN]	Correct files from Tracy are stored here before they are read into Bartrack.
BAR_TCY_SAVEOUT_DIR	DISK_BARTRACK:[TRACY.SAVEOUT]	Correctly transmitted files sent to Tracy are stored here.
BAR_TCY_TEMPIN_DIR	DISK_BARTRACK:[TRACY.TEMPIN]	Temporary location for files from Tracy.
BAR_TCY_TEMPOUT_DIR	DISK_BARTRACK:[TRACY.TEMPOUT]	Temporary location for files to Tracy.
BAR_TCY_UTPRODNAME_INDIVIDS	BAR_TCY_OUT_DIR:EDIINDIVIDS.UT	File used during Tracy file transfer.
BAR_TCY_UTPRODNAME_REQUESTS	BAR_TCY_OUT_DIR:EDIREQUEST.UT	File used during Tracy file transfer.
BAR_TCY_INFILE	BAR_TCY_IN_DIR:EDIINPUT.IN	File used during Tracy file transfer.
BAR_TCY_INTEMPFILE	BAR_TCY_TEMPIN_DIR: EDIINPUT.IN	File used during Tracy file transfer.
BAR_TCY_INSAVEFILE	BAR_TCY_SAVEIN_DIR: EDIINPUT.IN	File used during Tracy file transfer.
BAR_TCY_INERRORFILE	BAR_TCY_ERRORIN_DIR: EDIINPUT.IN	File used during Tracy file transfer.
DISK_M2T	DISK_BARTRACK:[M2T.]	The disk containing the M2T directory.
M2T_COM	DISK_M2T:[COM]	The directory containing command files for M2T.
M2T_EXE	DISK_M2T:[EXE]	The directory containing executable files for M2T.
M2T_LOG	DISK_M2T:[LOG]	The directory containing log files for M2T.
M2T_TMP	DISK_M2T:[TMP]	Temporary storage for all other records than Y-records.
M2T_TMP_Y	DISK_M2T:[TMP_Y]	Temporary storage for Y-records.
M2T_WAIT	DISK_M2T:[WAIT]	This is where mail sent to Tracy are stored waiting for success or failure status. Each night, any messages that are not confirmed are re-sent.
M2T_ERR	DISK_M2T:[ERR]	This is where mail sent to Tracy that weren't successful is stored. Purged after 7 days.
M2T_SAVE	DISK_M2T:[SAVE]	This is where mail sent to Tracy that weren't successful is stored. Purged after 7 days.
M2T_MANAGER	<e-mail address>	The address to send logs and status messages.

Messaging to and from Tracy

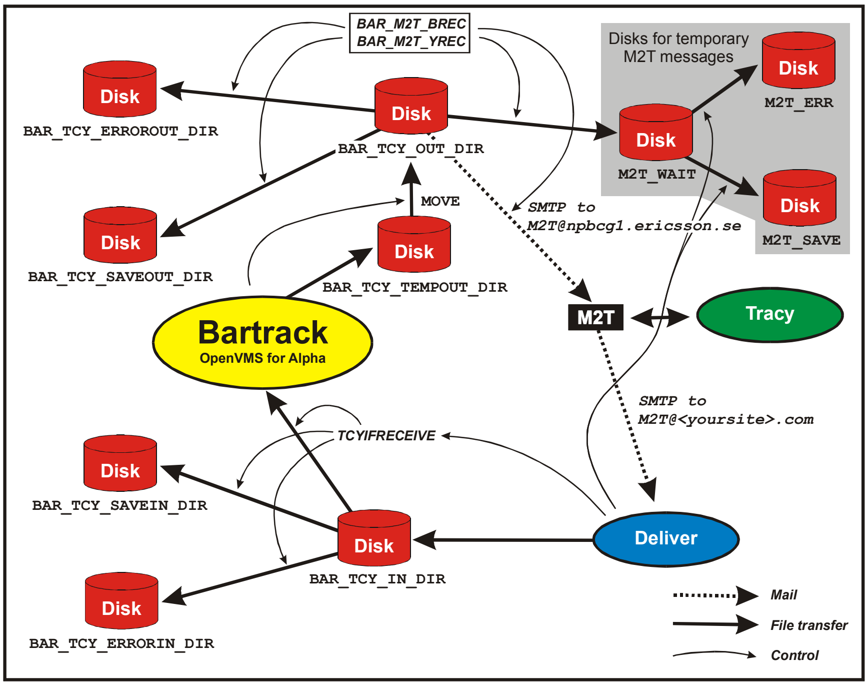
There are several logical names pointing to directories and files that are used when Bartrack exchanges information with Tracy (see previous section).



Bartrack-Tracy communication using EDI (normal timing)



Bartrack-Tracy communication using EDI (fast timing)



Bartrack-Tracy communication using mail

To Tracy Using EDI

The **BAR_TRACY** and **BAR_TRACY_BAR** processes are responsible for all actions on the “out” directories. They will scan the “out” directory **BAR_TCY_OUT_DIR**.

BAR_TRACY_BAR handles Y-records only (individual requests) and scans every minute.

BAR_TRACY handles all other record types and scans every 30 minutes.

When a message from Bartrack is sent to Tracy, a file is created in the **BAR_TCY_TEMPOUT_DIR** directory. When the file is complete (including the **<EOF>** character) it is moved to **BAR_TCY_OUT_DIR** . The file is then transferred to Tracy via DEC/EDI by one of the two scanning processes. If the transfer was successful, the file is moved and appended to a file with today's date in **BAR_TCY_SAVEOUT_DIR**. However, if the transfer was unsuccessful, the file is moved to **BAR_TCY_ERROROUT_DIR** .

When files including external serial numbers are sent, the serial numbers are stripped. The directory **BAR_TCY_SAVEOUT_DIR** may contain two files per day; the unaffected compilation of all serial numbers, and the filtered compilation of the serial numbers for that site. The extension for the filtered file is **.SENT** .

To Tracy Using Mail

The **BAR_M2T_BREC** and **BAR_M2T_YREC** processes are responsible for scanning the “out” directory **BAR_TCY_OUT_DIR**.

BAR_M2T_YREC handles Y-records only (individual requests) and scans every 20 seconds.

BAR_M2T_BREC handles all other record types and scans every 30 minutes.

When a message from Bartrack is sent to Tracy, a file is created in the **BAR_TCY_TEMPOUT_DIR** directory. When the file is complete (including the **<EOF>** character) it is moved to **BAR_TCY_OUT_DIR** . Then a mail is created and sent to Tracy by one of the two scanning processes. If the transfer was successful, the file is moved and appended to a file with today's date in

`BAR_TCY_SAVEOUT_DIR`. However, if the transfer was unsuccessful, the file is moved to `BAR_TCY_ERROUT_DIR` .

Messages to Tracy

The messages possible to send to Tracy are:

- Individual data (B-record)
- Individual data update (C-record)
- Synonym number (E-record)
- External serial number information (F-record)
- Structure data (S-record)
- Individual information request (Y-record)
- Scrap an individual (Z-record)

See "System Specification for Tracy" (EDT/R/IK-97:0106 Uen) for a full specification on the format of the messages on each record.

Messages sent to Tracy (EDI or Mail)

On event:	EDI-messages sent	Explanation
CREATE	B-record and [E-record]	The message contains the data of the individual. Additional numbers or synonym numbers for the individual. This message is only sent if there are additional numbers to send.
REGISTER (Reg Manually)	B-record and	The message contains the data of the individual.
REGISTER (Reg from Tracy)	Y-record	The message contains a request for individual information.
ASSEMBLE	[SA-record] or [FA-record] (add)	The message contains the serial numbers of the connected individuals. The message contains the data of the retail product. **Note:** The event ASSEMBLE occurs when the criteria for the **<Ready>**-button in the **Assemble** tab is met. For example, that the completeness check is performed.
CHANGE (Change Data) **Note:** This event only occurs if product number, R-state or exemption is changed.	C-record [EA-record] (add) or [ED-record] (delete)	The message contains updated information of the individual. Additional numbers or synonym numbers for the individual. This message is only sent if there are additional numbers to send. If the change will cause the additional numbers to be deleted for the individual, then no E-record is sent.
CHANGE (Change Structure) **Note:** This event occurs when the criteria for the **<Ready>**-button in the **Change Structure** tab is met. For example, that the completeness check is performed.	[SA-record] (add) [SD-record] (delete) [FA-record] (add) or [FD-record] (delete)	The message contains the serial numbers of all individuals in a structure. This message is only sent when children are added to the structure. The message contains the serial numbers of removed individuals. This message is only sent when children are removed from the structure. The message contains the data of the retail products. This message is only sent if the structure contains retail products.

STORE	B-record or	The message contains the data of the individual.
	C-record	The message contains updated information of the individual. Only used if the destination is changed.
	[SA-record] (add)	The message contains the serial numbers of all individuals in a structure. This message is only sent when a structure is stored.
	[FA-record] (add)	The message contains the data of the retail products. This message is only sent if the structure contains retail products.
	[EA-record] (add)	Additional numbers or synonym numbers for the individual. This message is only sent if there are additional numbers to send.
SHIP	B-record or	The message contains the data of the individual.
	C-record	The message contains updated information of the individual. Only used if the destination is changed.
	[SA-record] (add)	The message contains the serial numbers of all individuals in a structure. This message is only sent when a structure is shipped.
	[FA-record] (add)	The message contains the data of the retail products. This message is only sent if the structure contains retail products.
	[EA-record] (add)	Additional numbers or synonym numbers for the individual. This message is only sent if there are additional numbers to send.
SCRAP	Z-record	The message contains a request to scrap an individual.
REACTIVATE	Z-record	The message contains the data of the individual.

Note: Any number of messages can be mixed within a file, or occur as a single message in a single file.

From Tracy Using EDI

The `TRACY_BAR` process is responsible for all actions on the “in” directories. It will scan the “in” directory `BAR_TCY_TEMPIN_DIR`.

`TRACY_BAR` handles all record types and scans every minute.

When Bartrack is ready to receive from Tracy, the received file is created in the `BAR_TCY_TEMPIN_DIR` directory. When the file is complete (including the `<EOF>` character) it is moved to `BAR_TCY_IN_DIR` by the scanning process. The file is then read into Bartrack by the batch job `TCYIFRECEIVE`. If the read transaction was successful, the file is moved to `BAR_TCY_SAVEIN_DIR` . However, if the read transaction was unsuccessful, the file is moved to `BAR_TCY_ERRORIN_DIR` .

From Tracy Using Mail

The Deliver software is responsible for all actions performed on incoming mail from Tracy.

When a mail arrives at the Bartrack node, the Deliver software will decode its contents and act on it. The subject of the mail contains instructions to Deliver, which will move files, generate files and so on. The files intended to feed into Bartrack is stored in `BAR_TCY_IN_DIR` . These files are then read into Bartrack by the batch job `TCYIFRECEIVE`. If the read transaction was successful, the files are moved to `BAR_TCY_SAVEIN_DIR` . However, if the read transaction was unsuccessful, the files are moved to `BAR_TCY_ERRORIN_DIR` .

Messages from Tracy

The messages possible to receive from Tracy are:

- Individual information requested by Bartrack (B-record)
- Additional numbers if any (E-record)
- Structure information requested by Bartrack (S-record)
- Retail products if any (F-record)
- Request to ship individuals from Bartrack (X-record)

See “System Specification for Tracy” (EDT/R/IK-97:0106 Uen) for a full specification on the format of the messages on each record.

Messages sent from Tracy (EDI or Mail)

On event:	EDI-messages sent	Explanation
Individual information requested	B-record and	The message contains the data of the individual.
	[E-record]	Additional numbers or synonym numbers for the individual. This message is only sent if there are additional numbers to send.
	[S-record]	The message contains the serial numbers of the connected individuals.
	[F-record]	The message contains the data of external individuals.
Request to ship an individual	X-record	The message contains the serial number of an individual that Tracy want to ship from Bartrack.

To External Systems (DmQ/BmQ)

See "BEAmessageQ (BmQ)" on page 25 for extra information on DmQ/BmQ and a listing of the queues.

Bartrack uses a DECmessageQ interface to external systems. An external system might be any system capable of handling DMQ. A typical example of an external system is a test system, such as Testnet. For each action or request there is a corresponding queue. Each queue is uni-directional.

See “External interface specification: Bartrack - Shop Floor Transaction” (KS001B03/EN) for a full specification on the format of the messages on each queue.

N.B. The messages to external systems come in two versions. One is the same as in Bartrack R5A, and the other is a new format for Bartrack 6.0. The selection of either of these two versions can be done for each external system separately. Please refer to the "External interface specification: Bartrack - Shop Floor Transaction” (KS001B03/EN) interface description for details.

Messages sent from Bartrack to external systems by DmQ/BmQ

Event	DmQ/BmQ-message for R5A	DmQ/BmQ-message for 6.0	Explanation
CREATE	MsA_CreIDInfo (transaction code SERCRT)	NewInfo (transaction code SERCRT)	The message contains the data of the individual
REGISTER (only Reg Manually)	MsA_CreIDInfo (transaction code SEREXT)	NewInfo (transaction code SEREXT)	The message contains the data of the individual

ASSEMBLE	MsA_AsmIDInfo_2 (transaction codes SERPCA,SERPCD)	AsmID (transaction codes SERPCA,SERPCD)	The message contains the serial numbers and the product numbers **Note:** The event ASSEMBLE occurs when the criteria for the **<Ready>**-button in the **Assemble** tab is met. For example, that the completeness check is performed.
CHANGE (only Change Data)	Msa_UpdIDInfo (transaction code SERREV)	UpdInfo (transaction code SERREV)	The message contains the data of the individual **Note:** This message is only sent if a change in R-state has occurred.
STORE	Msa_UpdIDInfo (transaction code SERSTO)	UpdInfo (transaction code SERSTO)	The message contains the data of the individual
SHIP	Msa_UpdIDInfo (transaction code SERSHP)	UpdInfo (transaction code SERSHP)	The message contains the data of the individual
SCRAP	Msa_ScpIDInfo (transaction code SERSCP)	ScpInfo (transaction code SERSCP)	The message contains scrap information.

From External Systems (DmQ/BmQ)

If a test system is in use at the site, it is possible for the test system to send a message to Bartrack containing the test status for tested individuals.

The BAR_TST_UPD_01 queue used for these updates is read by a dedicated process, which communicate with Bartrack: `BAR_SFT_UPD`.

Messages sent from external systems to Bartrack by DmQ/BmQ

Event	DmQ/BmQ-message	Explanation
An individual has been tested	SERTST	The message contains the status of the test (pass or fail)

From External System by File

Bartrack can receive file-based messages from an external system. A special process, `BAR_TIF`, is dedicated to read the `BARTRACK_PCTRANSFER` directory.

The messages can be either a ship request or test status.

Please refer to the "Interface Description, Bartrack-Test System File Interface" (doc. no 155 19-CXA 110 038 Uen).

Messages sent from an external system to Bartrack by file

Event	Message	Explanation
An individual has been tested	<file name>.V01	The message contains the test status for an individual: F: Failed. The test status is set to "Failed" in Bartrack. P: Passed. The test status is set to "Passed" in Bartrack. The individual is also shipped from Bartrack.
An individual needs to be shipped	<file name>.V02	The message contains the test status for an individual or a request to ship the individual: F: Failed. The test status is set to "Failed" in Bartrack. P: Passed. The test status is set to "Passed" in Bartrack. S: Ship. The individual is shipped, but test status is not updated.

To Test System by File

Bartrack can send file-based messages to an external system. A special message (MsgToFile) in the BarSFT interface exists to create a file in a given directory. This message is used to create the four different files in the **`BARTRACK_PCTRANSFER`** directory.

N.B. Not all of the events in Bartrack can generate a file.

The messages can be of four different categories. Please refer to the "Interface Specification DmQToFile (DTF)" (doc. no 155 19-CXC137 101 Uen).

Table of messages sent from Bartrack to an external system by file

Event	File name	Explanation
CREATE (DmQ-message MsA_CreIDInfo_2)	<serial number>.S01	The message contains the data of the individual
REGISTER	N/A	No message is sent
ASSEMBLE (DmQ-message MsA_AsmIDInfo_2)	<serial number>.S17	The message contains the serial numbers and the product numbers of the structure. **Note:** The event ASSEMBLE occurs when the criteria for the **<Ready>**-button in the **Assemble** tab is met. For example, that the completeness check is performed.
CHANGE DATA (DmQ-message MsA_UpdIDInfo_2)	<serial number>.S06	The message contains the data of the individual **Note:** This event occurs when the criteria for the **<Save>**-button in the **Change Data** tab is met.
CHANGE STRUCTURE (DmQ-message MsA_AsmIDInfo_2)	<serial number>.S17	The message contains the serial numbers and the product numbers of the structure. **Note:** This event occurs when the criteria for the **<Ready>**-button in the **Change Structure** tab is met.
STORE	N/A	No message is sent
SHIP	N/A	No message is sent
SCRAP (DmQ-message MsA_ScpIDInfo)	<serial number>.S08	The message contains scrap information.

To a Financial System

It is possible to connect Bartrack to a financial system capable of handling orders, such as SAP/R3. The order system will send order information to Bartrack, which will prepare the structure of the order automatically.

Overview

The order system sends messages to the Bartrack system. The messages contain details about orders and products.

The messages can be transferred either as a file or as a BmQ message. If a file is used, it has to be placed in the directory **`BAR_ORDER_DIR`** with the name **`<date>.ORD`**. If a BmQ message is used, it has to be sent to the **`BAR_SFT_DTF_01`** queue and contain a reference to the directory and filename.

A dedicated process called **`BAR_SFT_DTF`** monitors the BmQ queue and converts incoming messages to a file in the appointed directory.

The logical name **BAR_DTF_BUFFERSIZE** sets the size of the buffer in the **BAR_SFT_DTF** process. If a message is received larger than this size, it will be thrown away and no file created.

The default value for **BAR_DTF_BUFFERSIZE** is 64.000.

The **BAR_SFT_DTF** process is monitored by the SFQPS to make sure it is always up and running.

Interface Description

For a full description of the message formats, please refer to the KS001b05 “Interface Description - Order Information”.

Web Server

Specifications

For a full specification, read the “Target environment specification”, document number KS001f05.

The TES contains the minimum hardware specifications required in order to run Bartrack.

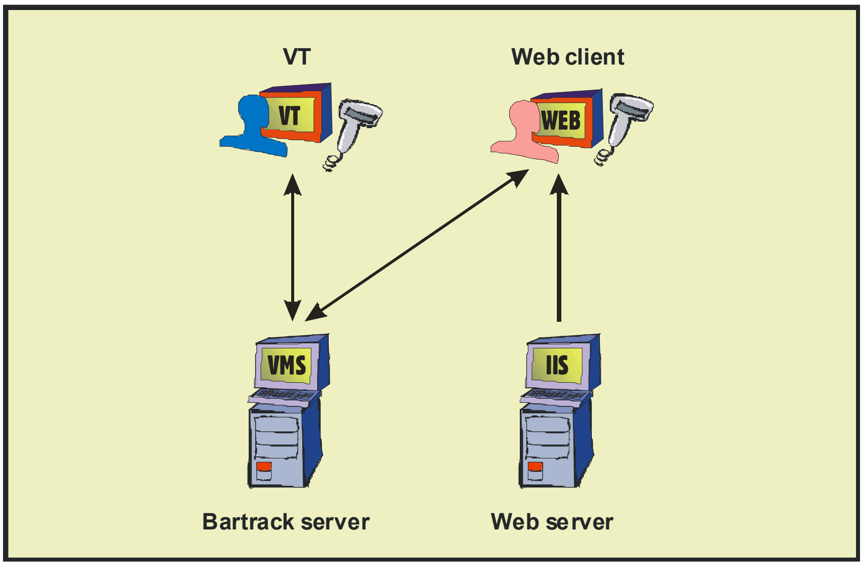
Distributing Bartrack Using the Web

One of the advantages of using the web interface is that the program files can be sitributed using normal web technology.

How It Works

Bartrack does not use the web server after the web client has started. The web server is only used to distribute the required files. When the files have been downloaded to the client, the client communicates directly with the Bartrack server.

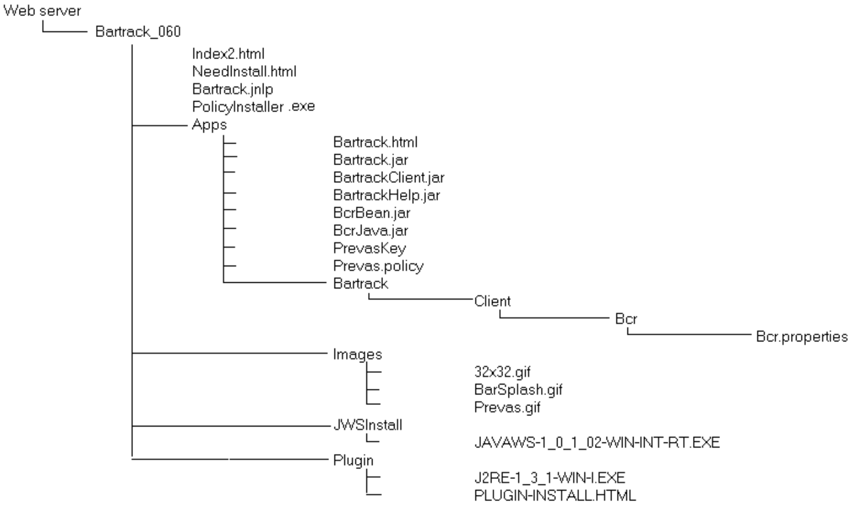
This means that the web server load caused by Bartrack Is limited to start-up of the client only. If you have problems with long start-up times, the web server should be upgraded ur tuned. If you are having problems with response times, the network might be the cause of the problems, and not the web server.



A system overview

File Structure and Files

All files needed by the web server is located in the **BAR_CLIENT** directory in an executable zip-file called **BARTRACK_WEB.EXE**. If you execute this file on the web server, all necessary files will be extracted into their correct directories.



Contents in ../Bartrack_060 on the web server

File name	Explanation
Index2.html	For starting Bartrack from a web page and running via Java Web Start.
NeedInstall.html	Web page for installing Java Web Start.
Bartrack.jnlp	This file is the start up file for the Bartrack client in a Java Web Start environment.
PolicyInstaller.exe	Running this application gives the Bartrack client permission to write to local disks and for caching files.

Contents in ../Bartrack_060/Apps on the web server

File name	Explanation
Bartrack.html	For starting Bartrack from a web page and running it via a web site.
Bartrack.jar	Communication objects for the communication between client and server.
BartrackClient.jar	Graphics and logic for the web client.
BartrackHelp.jar	Help texts for the web client.
BcrBean.jar	Java bean for barcode readers connected to the serial port.
BcrJava.jar	Resource files for the barcode reader.
Prevas.policy	This policy file will allow the web application to run on the client.
Prevaskey	This key will ensure that only signed JAR files will be able to execute on the client.

Contents in ../Bartrack_060/Images on the web server

File name	Explanation

32x32.gif	The Bartrack symbol
BarSplash.gif	A Bartrack picture shown at application start up.
Prevas.gif	Prevas logotype.

Contents in ../Bartrack_060/JWSInstall on the web server

File name	Explanation
JAVAWS-1_0_1_02-WIN-INT-RT.EXE	The Java Web Start installation program.

Contents in ../Bartrack_060/Plugin on the web server

File name	Explanation
J2RE-1_3_1-WIN-I.EXE	The JRE installation program.
PLUGIN-INSTALL.HTML	HTML file that triggers the JRE installation program.

There are two sub-directories called `com` and `plugin`. The `plugin` directory might contain the client JRE package needed to run the web client. The `com` directory contains two sub-directories with configuration files for Bartrack and serial port connected barcode readers.

Sometimes it is necessary to have several different types of barcode readers, or to use two different Bartrack servers within the same site. In these cases you will have to duplicate the structure in the `com` directory.

Each duplicate will allow you to have different settings for barcode reader and/or server. All you have to do to use the different settings is to distribute the web address of the `Bartrack.html` file (or whatever name you choose to use) in the `com` directory structure, and that particular branch of the settings will be used.

N.B. Some web servers are case sensitive (typically Unix systems) and it might be important what case you use for your file names.

VT-Terminal

Specifications

For a full specification, read the “Target environment specification”, document number KS001f05/en.

The TES contains the minimum hardware specifications required in order to run Bartrack.

Adding a VT-Terminal to the System

Other than the set-up below, no other action has to be done in order to add a VT-terminal to the Bartrack environment.

Set-up

In addition to the normal set-up needed for a VT-terminal to function properly, the following steps must be done for Bartrack to operate correctly:

- The terminal must be set to VT300 mode (or greater)
- The terminal must be set to VT320 ID (or greater)

Continue with “Adding a VT user” on page 67 if that is what you are doing.

External Programs

Label Design Software

You can use your favourite application for designing labels. This manual does not cover this topic. All information about designing labels can be found in the User Guide.

Web-client

Specifications

The web-client can be either an Internet Explorer 4 (or later) browser or a Netscape 4 (or later) browser. Any computer capable of running these browsers can use Bartrack. For a full specification, read the “KS001f02/en Target Environment Specification”.

The TES contains the minimum hardware specifications required in order to run Bartrack.

Adding a Web-Client to the System

This section deals with Web-clients. A PC-client is not a web-client.

When you are adding a Web-client, you must have administrator privileges on that machine.

These are the tasks to perform in order to connect a web-client to the Bartrack environment:

1. The client must comply with the specifications above. In short, it must be able to run Java Run-time Environment, and have a network connection.
2. The client must have access through TCP/IP to the following two locations:
   - The Bartrack server
   - The web server containing the web client
3. Execute the PolicyInstaller.exe found in the Bartrack directory of the web server. This file will add the Bartrack policy file to the policy file already installed by JRE.
   The policy installer will prompt for the URL of the Bartrack website.
4. Start a web browser and enter the URL (address) of the Bartrack web server. There might be different URL:s depending on the type of barcode reader that should be used.

Now it should be possible to start the Bartrack client.

If the communication fails, see the chapter Troubleshooting in this manual (page 71).

Parameters and Variables

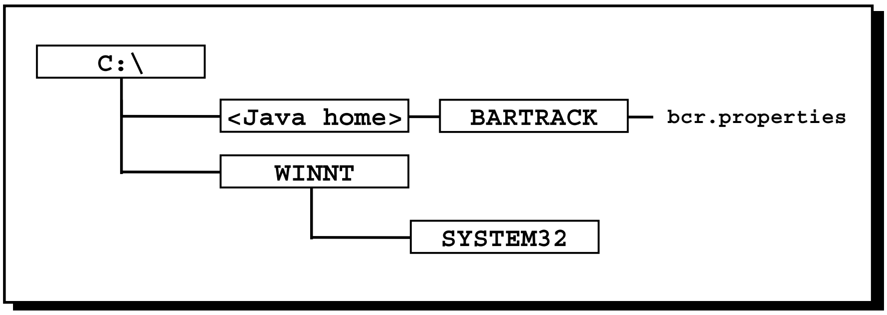
There is only one setting applicable to the web client. It is stored in the java.policy file, and contains the path to the prevas.policy file found at the Bartrack web server. This path is needed for the Java Run-time Environment to be able to verify that the Bartrack web application files are authentic.

To set this path, you will have to execute the PolicyInstaller.exe found in the Bartrack directory of the web server.

The policy installer will prompt for the URL of the Bartrack website, where a file called prevaskey is located. The prevas.policy and the prevaskey in cooperation will ensure that only approved code will be executed on the client.

Files and File Structure

All files needed by the web-client will be downloaded and stored automatically by the browser according to normal browser operation. The only exception to this rule is the .java.policy file. Please refer to the "Adding a Web-Client to the System" section on page 45 for information on this file.



Windows-based web browser - client file structure

Contents in <Java home>

File name	Explanation
bcr.properties	Contains the settings for the serial-port connected barcode reader. Copied here the first time such a barcode reader is used.

Contents in System32

File name	Explanation
jspWin.dll	Used by the serial-port connected barcode reader. Copied here the first time such a barcode reader is used.
jspWinNm.dll	Used by the serial-port connected barcode reader. Copied here the first time such a barcode reader is used.
jspWinRni.dll	Used by the serial-port connected barcode reader. Copied here the first time such a barcode reader is used.
jspWinRnia.dll	Used by the serial-port connected barcode reader. Copied here the first time such a barcode reader is used.

Internal Interfaces

To a Barcode Reader

Bartrack can use a barcode reader connected to either the web-client or the VT-terminal. When a barcode is scanned, its data is transferred to the correct field in Bartrack. When a PDF-code is scanned with a serial port scanner, all the data is written in the correct fields.

The web-client has a special way of handling the settings for a barcode reader. Please refer to the Bartrack user Guide on how to select barcode reader.

It is possible to connect barcode readers to a Bartrack web client in two ways; either via the serial port or via a keyboard connector. The option to choose is given during installation of the client.

Keyboard-connected barcode reader

A computer with a keyboard-connected reader can not differentiate between typed keys or scanned barcodes. It is therefore necessary to always be in the right field when scanning a barcode, that is, have the cursor where you want the scanned data to appear.

Different techniques can though be used to navigate between fields. For a VT-terminal for example, the cursor keys and the **DO** key can be used as sequences to navigate from one field to another when scanning a barcode shortcut on a separate paper. For a web-client, the access keys (for example **ALT-D**) can be used as barcode shortcuts to jump between fields

Any data identifiers in the barcode (for example **S** for serial number) must be stripped before the contents are used. See next section for more information on data identifiers.

Serial port connected barcode reader

If you have a barcode reader connected to the serial port, you have to edit the **BAR_BCRINIFILE** file to fit the brand of the reader. This fiel can be found in the web server directory for Bartrack.

The initialisation file can contain several settings for different barcode readers. They are separated by a row like this:

```
[barcodereader_intermec_9510]
```

Then come the different settings applicable to the specific barcode reader.

In order for Bartrack to find the correct initialisation file, and the correct barcode reader section for each web-client, it is possible to select from the menu in Bartrack which barcode reader to use.

It is also possible to transfer settings (programming) from the initialisation file to the barcode reader automatically. See the sample entries in the initialisation file for examples on different barcode readers and settings.

Fact Prefix

The data in a barcode or PDF-code can be preceded with different codes for the type of information. These codes are referred to as Fact Prefixes.

There are several different Fact Prefixes supported by Bartrack:

- **S** Serial number
- **6D** New standard: Any unspecified date

- **11D** Manufacturing week in the format YYYYWW
- **12D** Manufacturing date in the format YYYYMMDD
- **1P** Product number or Product number and R-state
- **21P** R-state
- **22P** Product name
- **3C** MAC-address
- **22S** Electronic serial number (ESN)
- **K** Order number
- **1K** Order number
- **14K** Order number

Note: There is an old version of the standard, where product number and R-state were defined like this:

- **1P** Product number
- **2P** R-state
- **6D** Manufacturing week in the format YYYWWW

Any other data or data with other identifiers will be mapped to the least significant fields, for example **Description**, **Comments** or **Free text**.

When PDF codes are used, all Fact Prefixes can be used simultaneously in one barcode. Any number of data elements can be used in a PDF-code, although at least one has to be present. The maximum number of data elements is limited by the PDF-code security level.

A PDF-code with just one data element looks like this:

```
])>RS06GS<data>RSEOT
```

A PDF-code with several data elements looks like this:

```
])>RS06GS<data>GS<data>GS<data>RSEOT
```

Where

`<data>` is the required data fields (for example SA240000001)
`RS` is the ASCII character Record Separator (value 30)
`GS` is the ASCII character Group Separator (value 29)
`EOT` is the ASCII character End Of Text (value 04)

This is an example of data to print as a PDF-code:

Serial number: A240000001
Product number: ABC1234 567
R-state: R2C

The corresponding PDF-code:

```
])>RS06GSSA240000001GS1PABC1234567GS2PR2CRSEOT
```

Database

Backup and Restore

Operation

The Bartrack database is set up to backup itself at 00.10 AM every day.

Should you want to alter the backup settings, please consult helpdesk.

The backup files are stored in the directory reached by the logical name **BAR_BACKUP** (See "File Structure and Files" on page 19). The backup's name will be **BAR_YYMMDD.RBF**, where '**YY**' is the year, '**MM**' is the month and '**DD**' is the day.

The three journal files are stored in the directory reached by the logical name **BAR_AIJ** (See "File Structure and Files" on page 19). Each journal file is 100.000 page quotas in size. When the first journal file is full, the next one is automatically enabled and the first one is automatically backed up. When the second file is full, the third one is enabled and the second one is automatically backed up. When the third file is full, the first one is enabled again, and so on.

Starting the Backup

The backup procedure is started during installation.

Every night at 00.00 the **BAR_CHECK_BATCH** is started. It queues the **BAR_BACKUP** job on the **BAR_BACKUP_QUEUE** queue scheduled to start at 00.10 AM.

If you have cancelled a backup and want to start it, the command to start the scheduled backup again is:

```
$ @BAR_TOOL:BAR_CHECK_BATCH
```

The command, among other things, queues itself for execution the next night at 00.00 AM and queues the **BAR_BACKUP** job on the **BAR_BACKUP_QUEUE** queue.

When **BAR_BACKUP** executes, it backs up the entire Bartrack Rdb database. It does not back up the SFQPS database.

Cancelling a Backup

To cancel a scheduled backup, you have to remove or stop the queued **BAR_BACKUP** job from the **BAR_BACKUP_QUEUE** queue.

You have to do this before the job has started (that is before 00.10) but after it has been queued (that is after 00.00).

If you want to permanently disable the backup procedure, you have to edit the **`BAR_TOOL:BAR_CHECK_BATCH.COM`** file.

Restoring From a Backup

The after image journalling (AIJ) is enabled during installation.

It is only possible to restore the Bartrack database to the earliest point of when the AIJ was enabled.

To restore a database, refer to the RdB documentation set. All backup files are located in the **`BAR_BACKUP`** directory.

All journal files are located in the **`BAR_AIJ`** directory.

Logical names for database

There are several logical names that are used for the database and backup.

Table of logical names for the Bartrack database

Logical name	Value	Explanation
BAR_DB	DISK_BARTRACK:[DB]	This directory contains all the Rdb files that constitute the Bartrack Rdb database.
BDB	BAR_DB:BAR	This is the name of the Bartrack database.
BAR_AIJ	DISK_BARAIJ:[AIJ]	The AIJ files for the database backup.
BAR_BACKUP	DISK_BARAIJ:[BCK]	The backup files for the database backup.

Creating an empty database

Note: Extreme caution must be taken if the Bartrack database is to be created. Do not attempt to do this unless you are very confident that you know what you are doing.

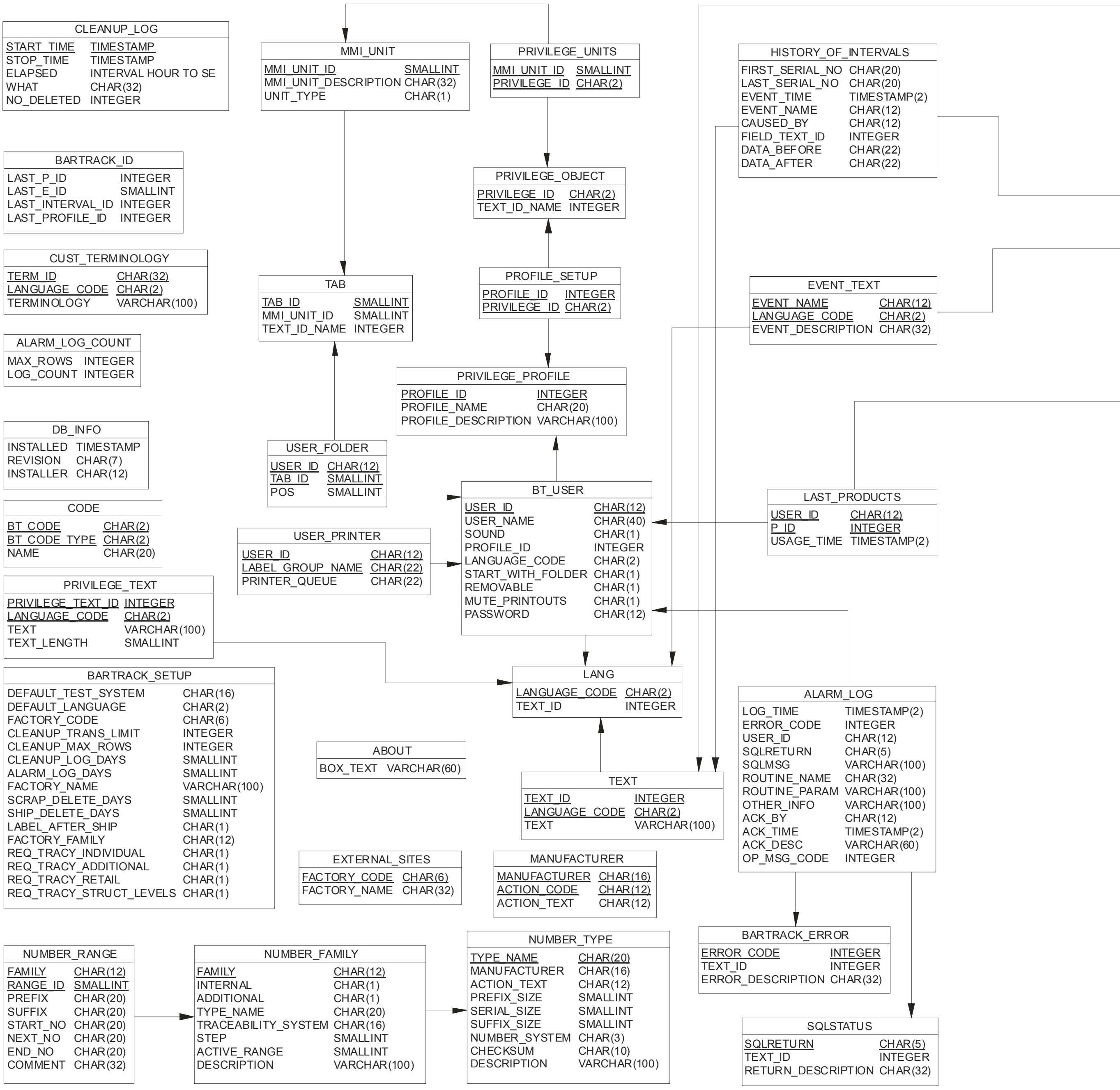
There are occasions when an empty database is required, for example for training purposes. The necessary steps to create an empty database are:

1. Log on as the user BAR.
2. Point the logical name **`BAR_DB`** to where you want the database to be created.
3. Set the **`DBCRE`** directory as default:
   ```
   $ SET DEFAULT BAR_DBCRE
   ```
4. Execute the command **`DB_CREATE`**:
   ```
   $ @DB_CREATE
   ```
   This command will take a while to execute.
5. An empty database is created in the directory pointed to by **`BAR_DB.`**

Refer to the “KS001f04/en Bartrack Installation Guide” for more information on creating databases.

Map

This is a map of all the tables in the relational database. There are arrows connecting the tables, and the direction of an arrow indicates the relation. The table from where the table starts need something to be defined in the table where the arrow ends.



CLEANUP_LOG

START_TIME	TIMESTAMP
STOP_TIME	TIMESTAMP
ELAPSED	INTERVAL HOUR TO SE
WHAT	CHAR(32)
NO_DELETED	INTEGER

MMI_UNIT

MMI_UNIT_ID	SMALLINT
MMI_UNIT_DESCRIPTION	CHAR(32)
UNIT_TYPE	CHAR(1)

PRIVILEGE_UNITS

MMI_UNIT_ID	SMALLINT
PRIVILEGE_ID	CHAR(2)

HISTORY_OF_INTERVALS

FIRST_SERIAL_NO	CHAR(20)
LAST_SERIAL_NO	CHAR(20)
EVENT_TIME	TIMESTAMP(2)
EVENT_NAME	CHAR(12)
CAUSED_BY	CHAR(12)
FIELD_TEXT_ID	INTEGER
DATA_BEFORE	CHAR(22)
DATA_AFTER	CHAR(22)

BARTRACK_ID

LAST_P_ID	INTEGER
LAST_E_ID	SMALLINT
LAST_INTERVAL_ID	INTEGER
LAST_PROFILE_ID	INTEGER

PRIVILEGE_OBJECT

PRIVILEGE_ID	CHAR(2)
TEXT_ID_NAME	INTEGER

CUST_TERMINOLOGY

TERM_ID	CHAR(32)
LANGUAGE_CODE	CHAR(2)
TERMINOLOGY	VARCHAR(100)

TAB

TAB_ID	SMALLINT
MMI_UNIT_ID	SMALLINT
TEXT_ID_NAME	INTEGER

PROFILE_SETUP

PROFILE_ID	INTEGER
PRIVILEGE_ID	CHAR(2)

EVENT_TEXT

EVENT_NAME	CHAR(12)
LANGUAGE_CODE	CHAR(2)
EVENT_DESCRIPTION	CHAR(32)

ALARM_LOG_COUNT

MAX_ROWS	INTEGER
LOG_COUNT	INTEGER

PRIVILEGE_PROFILE

PROFILE_ID	INTEGER
PROFILE_NAME	CHAR(20)
PROFILE_DESCRIPTION	VARCHAR(100)

DB_INFO

INSTALLED	TIMESTAMP
REVISION	CHAR(7)
INSTALLER	CHAR(12)

USER_FOLDER

USER_ID	CHAR(12)
TAB_ID	SMALLINT
POS	SMALLINT

BT_USER

USER_ID	CHAR(12)
USER_NAME	CHAR(40)
SOUND	CHAR(1)
PROFILE_ID	INTEGER
LANGUAGE_CODE	CHAR(2)
START_WITH_FOLDER	CHAR(1)
REMOVABLE	CHAR(1)
MUTE_PRINTOUTS	CHAR(1)
PASSWORD	CHAR(12)

LAST_PRODUCTS

USER_ID	CHAR(12)
P_ID	INTEGER
USAGE_TIME	TIMESTAMP(2)

CODE

BT_CODE	CHAR(2)
BT_CODE_TYPE	CHAR(2)
NAME	CHAR(20)

USER_PRINTER

USER_ID	CHAR(12)
LABEL_GROUP_NAME	CHAR(22)
PRINTER_QUEUE	CHAR(22)

PRIVILEGE_TEXT

PRIVILEGE_TEXT_ID	INTEGER
LANGUAGE_CODE	CHAR(2)
TEXT	VARCHAR(100)
TEXT_LENGTH	SMALLINT

BARTRACK_SETUP

DEFAULT_TEST_SYSTEM	CHAR(16)
DEFAULT_LANGUAGE	CHAR(2)
FACTORY_CODE	CHAR(6)
CLEANUP_TRANS_LIMIT	INTEGER
CLEANUP_MAX_ROWS	INTEGER
CLEANUP_LOG_DAYS	SMALLINT
ALARM_LOG_DAYS	SMALLINT
FACTORY_NAME	VARCHAR(100)
SCRAP_DELETE_DAYS	SMALLINT
SHIP_DELETE_DAYS	SMALLINT
LABEL_AFTER_SHIP	CHAR(1)
FACTORY_FAMILY	CHAR(12)
REQ_TRACY_INDIVIDUAL	CHAR(1)
REQ_TRACY_ADDITIONAL	CHAR(1)
REQ_TRACY_RETAIL	CHAR(1)
REQ_TRACY_STRUCT_LEVELS	CHAR(1)

LANG

LANGUAGE_CODE	CHAR(2)
TEXT_ID	INTEGER

ALARM_LOG

LOG_TIME	TIMESTAMP(2)
ERROR_CODE	INTEGER
USER_ID	CHAR(12)
SQLRETURN	CHAR(5)
SQLMSG	VARCHAR(100)
ROUTINE_NAME	CHAR(32)
ROUTINE_PARAM	VARCHAR(100)
OTHER_INFO	VARCHAR(100)
ACK_BY	CHAR(12)
ACK_TIME	TIMESTAMP(2)
ACK_DESC	VARCHAR(60)
OP_MSG_CODE	INTEGER

ABOUT

BOX_TEXT	VARCHAR(60)

TEXT

TEXT_ID	INTEGER
LANGUAGE_CODE	CHAR(2)
TEXT	VARCHAR(100)

EXTERNAL_SITES

FACTORY_CODE	CHAR(6)
FACTORY_NAME	CHAR(32)

MANUFACTURER

MANUFACTURER	CHAR(16)
ACTION_CODE	CHAR(12)
ACTION_TEXT	CHAR(12)

BARTRACK_ERROR

ERROR_CODE	INTEGER
TEXT_ID	INTEGER
ERROR_DESCRIPTION	CHAR(32)

NUMBER_RANGE

FAMILY	CHAR(12)
RANGE_ID	SMALLINT
PREFIX	CHAR(20)
SUFFIX	CHAR(20)
START_NO	CHAR(20)
NEXT_NO	CHAR(20)
END_NO	CHAR(20)
COMMENT	CHAR(32)

NUMBER_FAMILY

FAMILY	CHAR(12)
INTERNAL	CHAR(1)
ADDITIONAL	CHAR(1)
TYPE_NAME	CHAR(20)
TRACEABILITY_SYSTEM	CHAR(16)
STEP	SMALLINT
ACTIVE_RANGE	SMALLINT
DESCRIPTION	VARCHAR(100)

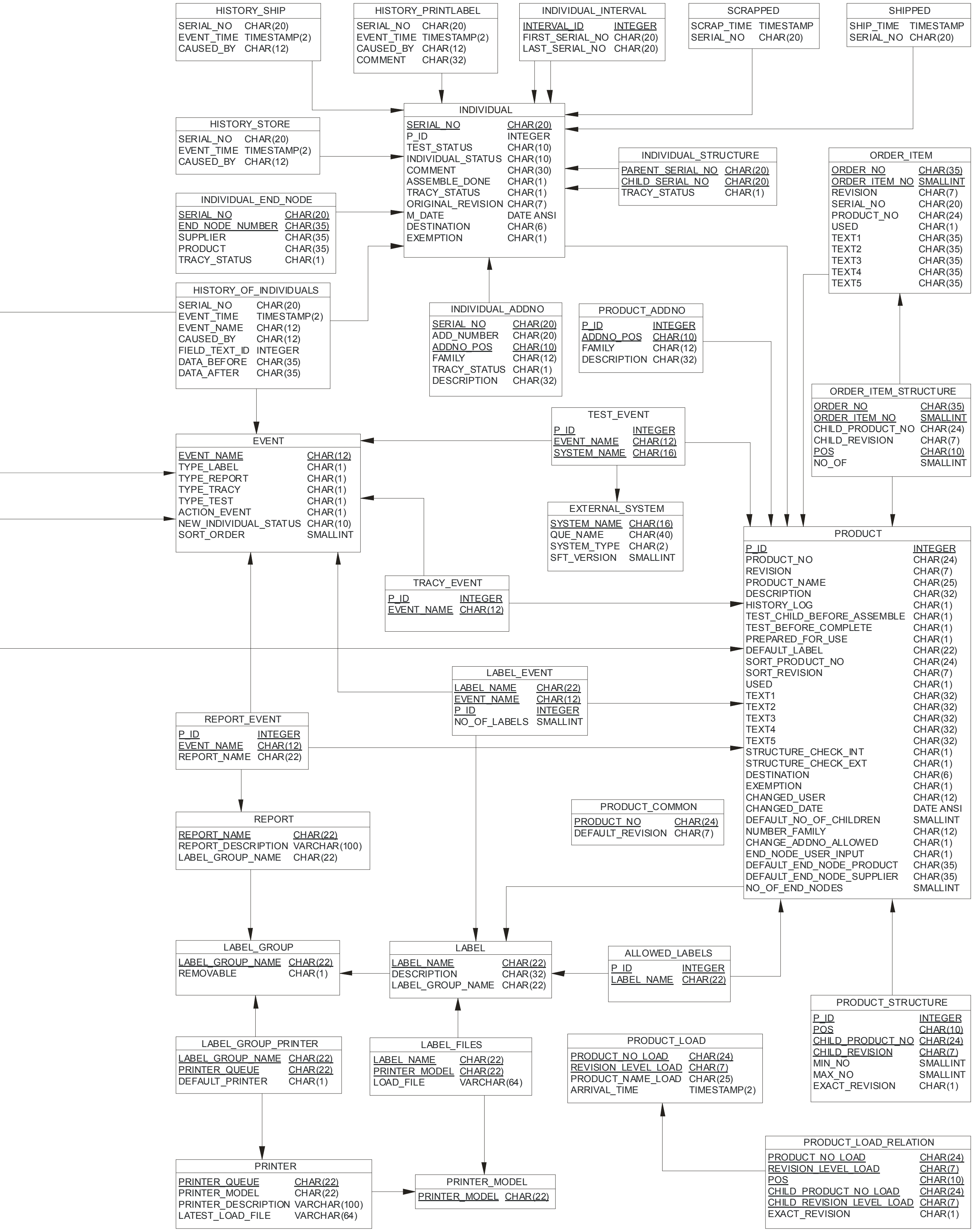
NUMBER_TYPE

TYPE_NAME	CHAR(20)
MANUFACTURER	CHAR(16)
ACTION_TEXT	CHAR(12)
PREFIX_SIZE	SMALLINT
SERIAL_SIZE	SMALLINT
SUFFIX_SIZE	SMALLINT
NUMBER_SYSTEM	CHAR(3)
CHECKSUM	CHAR(10)
DESCRIPTION	VARCHAR(100)

SQLSTATUS

SQLRETURN	CHAR(5)
TEXT_ID	INTEGER
RETURN_DESCRIPTION	CHAR(32)

Map of Bartracks Rdb database



HISTORY_SHIP	
SERIAL_NO	CHAR(20)
EVENT_TIME	TIMESTAMP(2)
CAUSED_BY	CHAR(12)

HISTORY_PRINTLABEL	
SERIAL_NO	CHAR(20)
EVENT_TIME	TIMESTAMP(2)
CAUSED_BY	CHAR(12)
COMMENT	CHAR(32)

INDIVIDUAL_INTERVAL	
INTERVAL_ID	INTEGER
FIRST_SERIAL_NO	CHAR(20)
LAST_SERIAL_NO	CHAR(20)

SCRAPPED	
SCRAP_TIME	TIMESTAMP
SERIAL_NO	CHAR(20)

SHIPPED	
SHIP_TIME	TIMESTAMP
SERIAL_NO	CHAR(20)

HISTORY_STORE	
SERIAL_NO	CHAR(20)
EVENT_TIME	TIMESTAMP(2)
CAUSED_BY	CHAR(12)

INDIVIDUAL	
SERIAL_NO	CHAR(20)
P_ID	INTEGER
TEST_STATUS	CHAR(10)
INDIVIDUAL_STATUS	CHAR(10)
COMMENT	CHAR(30)
ASSEMBLE_DONE	CHAR(1)
TRACY_STATUS	CHAR(1)
ORIGINAL_REVISION	CHAR(7)
M_DATE	DATE ANSI
DESTINATION	CHAR(6)
EXEMPTION	CHAR(1)

INDIVIDUAL_STRUCTURE	
PARENT_SERIAL_NO	CHAR(20)
CHILD_SERIAL_NO	CHAR(20)
TRACY_STATUS	CHAR(1)

ORDER_ITEM	
ORDER_NO	CHAR(35)
ORDER_ITEM_NO	SMALLINT
REVISION	CHAR(7)
SERIAL_NO	CHAR(20)
PRODUCT_NO	CHAR(24)
USED	CHAR(1)
TEXT1	CHAR(35)
TEXT2	CHAR(35)
TEXT3	CHAR(35)
TEXT4	CHAR(35)
TEXT5	CHAR(35)

INDIVIDUAL_END_NODE	
SERIAL_NO	CHAR(20)
END_NODE_NUMBER	CHAR(35)
SUPPLIER	CHAR(35)
PRODUCT	CHAR(35)
TRACY_STATUS	CHAR(1)

HISTORY_OF_INDIVIDUALS	
SERIAL_NO	CHAR(20)
EVENT_TIME	TIMESTAMP(2)
EVENT_NAME	CHAR(12)
CAUSED_BY	CHAR(12)
FIELD_TEXT_ID	INTEGER
DATA_BEFORE	CHAR(35)
DATA_AFTER	CHAR(35)

INDIVIDUAL_ADDNO	
SERIAL_NO	CHAR(20)
ADD_NUMBER	CHAR(20)
ADDNO_POS	CHAR(10)
FAMILY	CHAR(12)
TRACY_STATUS	CHAR(1)
DESCRIPTION	CHAR(32)

PRODUCT_ADDNO	
P_ID	INTEGER
ADDNO_POS	CHAR(10)
FAMILY	CHAR(12)
DESCRIPTION	CHAR(32)

ORDER_ITEM_STRUCTURE	
ORDER_NO	CHAR(35)
ORDER_ITEM_NO	SMALLINT
CHILD_PRODUCT_NO	CHAR(24)
CHILD_REVISION	CHAR(7)
POS	CHAR(10)
NO_OF	SMALLINT

TEST_EVENT	
P_ID	INTEGER
EVENT_NAME	CHAR(12)
SYSTEM_NAME	CHAR(16)

EVENT	
EVENT_NAME	CHAR(12)
TYPE_LABEL	CHAR(1)
TYPE_REPORT	CHAR(1)
TYPE_TRACY	CHAR(1)
TYPE_TEST	CHAR(1)
ACTION_EVENT	CHAR(1)
NEW_INDIVIDUAL_STATUS	CHAR(10)
SORT_ORDER	SMALLINT

EXTERNAL_SYSTEM	
SYSTEM_NAME	CHAR(16)
QUE_NAME	CHAR(40)
SYSTEM_TYPE	CHAR(2)
SFT_VERSION	SMALLINT

PRODUCT	
P_ID	INTEGER
PRODUCT_NO	CHAR(24)
REVISION	CHAR(7)
PRODUCT_NAME	CHAR(25)
DESCRIPTION	CHAR(32)
HISTORY_LOG	CHAR(1)
TEST_CHILD_BEFORE_ASSEMBLE	CHAR(1)
TEST_BEFORE_COMPLETE	CHAR(1)
PREPARED_FOR_USE	CHAR(1)
DEFAULT_LABEL	CHAR(22)
SORT_PRODUCT_NO	CHAR(24)
SORT_REVISION	CHAR(7)
USED	CHAR(1)
TEXT1	CHAR(32)
TEXT2	CHAR(32)
TEXT3	CHAR(32)
TEXT4	CHAR(32)
TEXT5	CHAR(32)
STRUCTURE_CHECK_INT	CHAR(1)
STRUCTURE_CHECK_EXT	CHAR(1)
DESTINATION	CHAR(6)
EXEMPTION	CHAR(1)
CHANGED_USER	CHAR(12)
CHANGED_DATE	DATE ANSI
DEFAULT_NO_OF_CHILDREN	SMALLINT
NUMBER_FAMILY	CHAR(12)
CHANGE_ADDNO_ALLOWED	CHAR(1)
END_NODE_USER_INPUT	CHAR(1)
DEFAULT_END_NODE_PRODUCT	CHAR(35)
DEFAULT_END_NODE_SUPPLIER	CHAR(35)
NO_OF_END_NODES	SMALLINT

TRACY_EVENT	
P_ID	INTEGER
EVENT_NAME	CHAR(12)

LABEL_EVENT	
LABEL_NAME	CHAR(22)
EVENT_NAME	CHAR(12)
P_ID	INTEGER
NO_OF_LABELS	SMALLINT

REPORT_EVENT	
P_ID	INTEGER
EVENT_NAME	CHAR(12)
REPORT_NAME	CHAR(22)

PRODUCT_COMMON	
PRODUCT_NO	CHAR(24)
DEFAULT_REVISION	CHAR(7)

REPORT	
REPORT_NAME	CHAR(22)
REPORT_DESCRIPTION	VARCHAR(100)
LABEL_GROUP_NAME	CHAR(22)

LABEL_GROUP	
LABEL_GROUP_NAME	CHAR(22)
REMOVABLE	CHAR(1)

LABEL	
LABEL_NAME	CHAR(22)
DESCRIPTION	CHAR(32)
LABEL_GROUP_NAME	CHAR(22)

ALLOWED_LABELS	
P_ID	INTEGER
LABEL_NAME	CHAR(22)

PRODUCT_STRUCTURE	
P_ID	INTEGER
POS	CHAR(10)
CHILD_PRODUCT_NO	CHAR(24)
CHILD_REVISION	CHAR(7)
MIN_NO	SMALLINT
MAX_NO	SMALLINT
EXACT_REVISION	CHAR(1)

LABEL_GROUP_PRINTER	
LABEL_GROUP_NAME	CHAR(22)
PRINTER_QUEUE	CHAR(22)
DEFAULT_PRINTER	CHAR(1)

LABEL_FILES	
LABEL_NAME	CHAR(22)
PRINTER_MODEL	CHAR(22)
LOAD_FILE	VARCHAR(64)

PRODUCT_LOAD	
PRODUCT_NO_LOAD	CHAR(24)
REVISION_LEVEL_LOAD	CHAR(7)
PRODUCT_NAME_LOAD	CHAR(25)
ARRIVAL_TIME	TIMESTAMP(2)

PRODUCT_LOAD_RELATION	
PRODUCT_NO_LOAD	CHAR(24)
REVISION_LEVEL_LOAD	CHAR(7)
POS	CHAR(10)
CHILD_PRODUCT_NO_LOAD	CHAR(24)
CHILD_REVISION_LEVEL_LOAD	CHAR(7)
EXACT_REVISION	CHAR(1)

PRINTER	
PRINTER_QUEUE	CHAR(22)
PRINTER_MODEL	CHAR(22)
PRINTER_DESCRIPTION	VARCHAR(100)
LATEST_LOAD_FILE	VARCHAR(64)

PRINTER_MODEL	
PRINTER_MODEL	CHAR(22)

Map of Bartracks Rdb database

Tables

Note: Extreme caution must be taken if the Bartrack database tables are to be read or written. Do not attempt to do this unless you are very confident that you know what you are doing.

To attach yourself to the database:

```
SQL> ATTACH 'FILENAME BDB';
```

Before you access the database, you must set the transaction to read only, if you intend to read the tables. Otherwise, you will lock up the whole database:

```
SQL> DECLARE TRANSACTION READ ONLY;
```

This is a selection of some of the tables that can come in handy when you work with Bartrack. Each table has its own listing, and a brief explanation of the field contents.

Field names in table BT_USER

Field name	Length	Explanation
USER_ID	CHAR(12)	Unique ID of users known to Bartrack. (KEY)
USER_NAME	CHAR(40)	Free text describing the user.
SOUND	CHAR(1)	Y or N for enabling audible alerts.
PROFILE_ID	INT	A number corresponding to the privilege profile for the user
LANGUAGE_CODE	CHAR(2)	EN or SV for English or Swedish texts.
START_WITH_FOLDER	CHAR(1)	Y or N for starting with personal folder.
REMOVABLE	CHAR(1)	Y or N for disabling deletion of the user.
MUTE_PRINTOUTS	CHAR(1)	Y or N for disabling printouts for the user.
PASSWORD	CHAR(20)	Encrypted password for the user. 12 characters, but in encrypted format 20 characters.

Field names in table INDIVIDUAL

Field name	Length	Explanation
SERIAL_NO	CHAR(20)	Unique serial number. (KEY)
P_ID	INT	Short, unique reference to the product.
TEST_STATUS	CHAR(10)	The individuals current test status.
INDIVIDUAL_STATUS	CHAR(10)	The individuals current Bartrack status.
COMMENT	VARCHAR(100)	Free text field, for comments.
ASSEMBLE_DONE	CHAR(1)	Y or N for telling if the individual is ready.
TRACY_STATUS	CHAR(1)	N: No data or structure sent to Tracy D: Data sent to Tracy S: Structure and data sent to Tracy C: C-record sent to Tracy
M_DATE	DATE ANSI	Manufacturing time of the individual.
DESTINATION	CHAR(6)	The destination of the individual when it is stored or shipped. Only values from the DESTINATION table are allowed.
EXEMPTION	CHAR(1)	Y or N for telling if the individual is manufactured with an exemption. U: The exemption state of the individual is unknown.

		N.B. This status is only used for individuals manufactured before the exemption was introduced in Bartrack.

Field names in table PRODUCT

Field name	Length	Explanation
P_ID	INT	Short, unique reference to a product. (KEY)
PRODUCT_NO	CHAR(22)	The product number, with spaces.
REVISION	CHAR(7)	Revision number (R-state).
PRODUCT_NAME	CHAR(25)	The product name.
DESCRIPTION	VARCHAR(32)	The products description.
HISTORY_LOG	CHAR(1)	Character Y or N for enabling the logging of transactions for this product.
TEST_CHILD_BEFORE_ ASSEMBLE	CHAR(1)	Character Y or N for demanding passed individuals before assembly.
TEST_BEFORE_COMPLETE	CHAR(1)	Character Y or N for demanding passed individuals in assemblies.
PREPARED_FOR_USE	CHAR(1)	Character Y or N for enabling the use of the product in manufacturing.
DEFAULT_LABEL	CHAR(22)	The name of the preferred label.
SORT_PRODUCT_NO	CHAR(22)	A copy of PRODUCT_NO optimised for sorting and searching.
SORT_REVISION	CHAR(7)	A copy of REVISION optimised for sorting and searching.
USED	CHAR(1)	Character Y or N. Used for telling whether or not the product has been used to create serial numbers.
TEXT1 to TEXT5	CHAR(32)	Free text field. Used for any site-specific data.
STRUCTURE_CHECK_INT	CHAR(1)	Character Y or N for demanding of structure check.
STRUCTURE_CHECK_EXT	CHAR(1)	Character Y or N for demanding of structure check.
DESTINATION	CHAR(6)	The factory code of the factory where this individual is being sent.
EXEMPTION	CHAR(1)	Character Y or N to indicate exemption from specification.
CHANGED_USER	CHAR(12)	The user-ID of the user that created or changed this product.
CHANGED_DATE	DATE ANSI	The date when the user created or changed this product.
DEFAULT_NO_OF_ CHILDREN	SMALLINT	A default value of the number of children for this product. Displayed in the Assemble tab.
NUMBER_FAMILY	CHAR(12)	The number family to use when individuals from this product is created.
END_NODE_USER_INPUT	CHAR(1)	Character N, A or M: N: [N]o, don't need user input. Use the default values. A: Yes, [a]sk the user. Display the default values. M: Yes, [m]andatory. The user must not leave blank values. Default values are displayed.
DEFAULT_END_NODE_ PRODUCT	CHAR(35)	A default 'Product number' value for retail products for this product. This value is displayed to the user when a retail product is assembled.
DEFAULT_END_NODE_ SUPPLIER	CHAR(35)	A default 'Supplier' value for retail products for this product. This value is displayed to the user when a retail product is assembled.
NO_OF_END_NODES	SMALLINT	The number of retail products for this product. Exactly this amount must be assembled if the structure is to be considered complete.

Field names in table BARTRACK_SETUP

Field name	Length	Explanation
DEFAULT_TEST_SYSTEM	CHAR(16)	The default test system for all new (not copied) products.
DEFAULT_LANGUAGE	CHAR(2)	The default language for all new (not copied) users.
FACTORY_CODE	CHAR(6)	The installation's factory code. Used as 'Sending unit' when Bartrack is communicating with Tracy.
CLEANUP_TRANS_LIMIT	INT	Internal use.
CLEANUP_MAX_ROWS	INT	Internal use.
CLEANUP_LOG_DAYS	SMALLINT	Internal use.
ALARM_LOG_DAYS	SMALLINT	Internal use
FACTORY_NAME	VARCHAR(100)	The name shown in the Main Menu and on reports.
SCRAP_DELETE_DAYS	SMALLINT	The number of days a scrapped individual is retained in Bartrack.
SHIP_DELETE_DAYS	SMALLINT	The number of days a shipped individual is retained in Bartrack.
LABEL_AFTER_SHIP	CHAR(1)	Character Y or N for allowing printout after an individual is shipped.
FACTORY_FAMILY	CHAR(12)	The default number family for all new (not copied) products.

Field names in table EXTERNAL_SYSTEM

Field name	Length	Explanation
SYSTEM_NAME	CHAR(16)	The name of the external system. (KEY)
QUE_NAME	CHAR(40)	The DmQ/BmQ queue name.
SYSTEM_TYPE	CHAR(2)	The type of the system. TE=Test system

Field names in table INDIVIDUAL_STRUCTURE

Field name	Length	Explanation
PARENT_SERIAL_NO	CHAR(13)	The serial number of the parent. (KEY)
CHILD_SERIAL_NO	CHAR(13)	The serial number of the child connected to the parent. (KEY)
TRACY_STATUS	CHAR(1)	The Tracy status of this assembly: D: This assembly is sent to Tracy. N: This assembly is not sent to Tracy.

Field names in table PRODUCT_STRUCTURE

Field name	Length	Explanation
P_ID	INT	Short, unique reference to a product. (KEY)
POS	CHAR(10)	The position of the child in the structure. (KEY)
CHILD_PRODUCT_NO	CHAR(22)	The product number of the child. (KEY)
CHILD_REVISION	CHAR(7)	The revision level of the child. (KEY)
MIN_NO	SMALLINT	The minimum number of connections allowed.
MAX_NO	SMALLINT	The maximum number of connections allowed.

EXACT_REVISION	CHAR(1)	Does the revision level have to be exact?
		Y=Yes
		N=No

Field names in table PRINTER_MODEL

Field name	Length	Explanation
PRINTER_MODEL	CHAR(22)	The name of a printer model. These models make it possible for Bartrack to send different label files to different printers. A model might be 'Zebra 600dpi'.

Field names in table EXTERNAL_SITES

Field name	Length	Explanation
FACTORY_CODE	CHAR(6)	The factory code of factories that should be displayed as 'Destination' in Bartrack (KEY).
FACTORY_NAME	CHAR(32)	The name of the factory.

Field names in table INDIVIDUAL_END_NODE

Field name	Length	Explanation
SERIAL_NO	CHAR(20)	The Bartrack serial number. Reference to an existing serial number in the INDIVIDUAL table (KEY).
END_NODE_NUMBER	CHAR(35)	The manufacturer of the retail product's own serial number (KEY). **N.B.** This number must be unique within each SUPPLIER. Two different suppliers may have the same number though.
SUPPLIER	CHAR(35)	The name of the supplier for this retail product.
PRODUCT	CHAR(35)	The retail product's manufacturer's own product number or product name.
TRACY_STATUS	CHAR(1)	The Tracy status of this individual: Y: This individual is sent to Tracy. N: This individual is not sent to Tracy.

Field names in table INDIVIDUAL_ADDNO

Field name	Length	Explanation
SERIAL_NO	Char 20	The Bartrack serial number. Reference to an existing serial number in the INDIVIDUAL table (KEY).

ADD_NUMBER	Char 20	The additional number for an individual.
POS	Smallint	The position where this additional number is stored. Up to 128 different positions may be used.
FAMILY	Char 12	The number family of the additional number.

Field names in table ORDER_ITEM

Field name	Length	Explanation
ORDER_NO	Char 35	The order number (KEY).
ORDER_ITEM_NO	Smallint	The order item number. A qualifier to the order number. The value -1 indicates that the order item is not in use. Any other value indicates the actual order item number (KEY).
REVISION	Char 7	The R-state of the product for the order.
SERIAL_NO	Char 20	The serial number of the unit being produced for the order.
PRODUCT_NO	Char 24	The product number of the order.
USED	Char 35	Y: The order has been started N: The order is not yet started.
TEXT1	Char 35	A customer specific text, received from the order system. Se the table CUST_TERMINOLOGY for the meaning of these fields in your particular factory.
TEXT2	Char 35	-"-
TEXT3	Char 35	-"-
TEXT4	Char 35	-"-
TEXT5	Char 35	-"-

Field names in table ORDER_ITEM_STRUCTURE

Field name	Length	Explanation
ORDER_NO	Char 35	The order number (KEY).
ORDER_ITEM_NO	Smallint	The order item number. A qualifier to the order number. The value -1 indicates that the order item is not in use. Any other value indicates the actual order item number (KEY).
CHILD_PRODUCT_NO	Char 24	A child product number for an order.
CHILD_REVISION	Char 7	A child R-state for an order.
POS	Char 10	The position to use in the product structure (KEY).
NO_OF	Smallint	The quantity of units on this position.

Destinations

N.B. Extreme caution must be taken if the Bartrack database tables are to be read or written. Do not attempt to do this unless you are very confident that you know what you are doing.

In Bartrack, there is a list of all approved destinations. One of these destinations can be chosen when an individual is shipped or stored. To add or delete a destination from this list, you will have to use SQL.

To list, add or delete a destination, you have to start SQL and then open the correct database:

1. Start SQL by typing the command `SQL`. If this does not work, try the command `MC SQL$`. If none of these commands does not work, please consult the Oracle Rdb documentation.
2. Type the following command to open the Bartrack database:
   ```
   ATTACH 'FILENAME BDB';
   ```
3. Verify that you are in the right database by typing the following commands:
   ```
   SET TRANSACTION READ ONLY;
   SELECT * FROM BARTRACK_SETUP;
   ROLLBACK;
   ```

A list of parameters should appear. The name of the database should be somewhere in the text. The name could be something like "**Acme Industries, production database**".

If this is the correct database, you can continue to add or delete destinations. Continue with one of the three different "step 4" below.

5. To exit SQL, type the command:
   ```
   EXIT
   ```

The normal VMS prompt will be displayed, and your changes will be available in Bartrack immediately.

List all destinations in Bartrack

To display a list of all the destinations in Bartrack:

4. Type the following commands:
   ```
   SET TRANSACTION READ ONLY;
   SELECT * FROM EXTERNAL_SITES;
   ROLLBACK;
   ```

A complete list of all the factory codes will be displayed.

Add a destination to Bartrack

To add a destination to Bartrack:

4. Type the following commands:
   ```
   SET TRANSACTION READ WRITE
   RESERVING EXTERNAL_SITES FOR SHARED WRITE;
   INSERT INTO EXTERNAL_SITES
        VALUES('<factory_code>',
               '<factory_name>);
   COMMIT;
   ```

Where `<factory_code>` is the 6-character destination you want to add.

Where `<factory_name>` is a plain text description of the factory.

N.B. The `SET TRANSACTION` and `RESERVING` rows needs to be typed in a single row.

If the destination is not in the list, SQL will reply with `1 row inserted`. Otherwise you will receive an error message stating that the destination or factory code already exists.

Delete a destination from Bartrack

To delete a destination from Bartrack:

4. Type the following commands:

```
SET TRANSACTION READ WRITE
RESERVING EXTERNAL_SITES FOR SHARED WRITE;
DELETE FROM EXTERNAL_SITES
WHERE FACTORY_CODE = '<factory_code>';
COMMIT;
```

Where `<factory_code>` is the 6-character destination you want to delete.

If the destination was in the list, SQL will reply with `1 row deleted`. Otherwise you will receive an error message stating that the destination or factory code does not exist.

Customer Terminology

N.B. Extreme caution must be taken if the Bartrack database tables are to be read or written. Do not attempt to do this unless you are very confident that you know what you are doing.

In Bartrack, there are some headings or concepts that may be customized for each factory. The definition of these are stored in the **CUST_TERMINOLOGY** table.

Each entry has an ID and a language code, which makes it possible to have different terminology in the two different languages available.

To list or change a definition, you have to start SQL and then open the correct database:

1. Start SQL by typing the command `SQL`. If this does not work, try the command `MC SQL$`. If none of these commands does not work, please consult the Oracle Rdb documentation.
2. Type the following command to open the Bartrack database:

```
ATTACH 'FILENAME BDB';
```

3. Verify that you are in the right database by typing the following commands:

```
SET TRANSACTION READ ONLY;
SELECT * FROM BARTRACK_SETUP;
ROLLBACK;
```

A list of parameters should appear. The name of the database should be somewhere in the text. The name could be something like "**Acme Industries, production database**".

If this is the correct database, you can continue to list or change definitions. Continue with the two different "step 4" below.

When you are finsihed you must exit SQL:

5. To exit SQL, type the command:

```
EXIT
```

The normal VMS prompt will be displayed, and your changes will be available in Bartrack immediately after the users logged in have refreshed their screens (by logging off and on again, or by switching between folders in Bartrack).

List all definitions in Bartrack

To display a list of all the definitions in Bartrack:

4. Type the following commands:
```
SET TRANSACTION READ ONLY;
SELECT * FROM CUST_TERMINOLOGY;
ROLLBACK;
```

A complete list of all the definitions will be displayed.

Change a Definition

To change a definition in Bartrack:

4. Type the following commands:
```
SET TRANSACTION READ WRITE
RESERVING CUST_TERMINOLOGY FOR SHARED WRITE;
UPDATE CUST_TERMINOLOGY
     SET TERMINOLOGY='<your new definition>'
     WHERE TERM_ID='<the term_id>'
     AND LANGUAGE_CODE=<the language>;
COMMIT;
```

Where `<your new definition>` is the 100-character definition you want to use instead of the old definition.

Where `<the term_id>` is the label that identifies the entry. List all definitions to find out the term_id you want to change.

Where `<the language>` is the 2-character language code.

N.B. The `SET TRANSACTION` and `RESERVING` rows needs to be typed in a single row.

If the dfinition is successfully updated, SQL will reply with `1 row updated`. Otherwise you will receive an error message stating that the combination does not exist.

Delete a destination from Bartrack

You should not delete a definition. If you do, the user will not see the header for the definition's fields on the screen.

Logging

Alarms

Some alarms or errors generated by external systems (such as Tracy or PRIM) are stored in the **Alarm Log**. Please refer to the Bartrack User Guide, section "Alarm Log", for more details on handling of alarms.

Server

There are several log files generated by Bartrack. Some of them may not be created, due to different configurations of Bartrack and, off course, the errors that have occurred.

All logs generated by the Bartrack server are stored as files in the directory **`BAR_LOG`**.

All logs generated by the process surveillance system (SFQPS) are stored as files in the directory **`SFQPS_LOG`**.

Server logical names for logging

Logical name	Value	Explanation
BAR_LOG	DISK_BARTRACK:[LOG]	This directory contains log files.
SFQPS_LOG	DISK_SFQPS:[LOG]	The directory for SFQPS log files..

Server log files that can be found in BAR_LOG

Log file	Contents
BAR_BACKUP	
BAR_CHECK_BATCH	
BAR_DB_VERIFY	
BAR_LOGCXCS_START	
BAR_LOGGING	Errors from the BAR_LOGGING process
BAR_RMI	Error messages from web-client transport software.
BAR_RMIREGISTRY	
BAR_RMIREGISTRY_STARTUP	
BAR_RMISTART	
BAR_RMI_STARTUP	
DMQTOFILE	Errors from the DmqToFile process

DMQ_SCRIPT	
ERROR	All error messages from Bartrack that has been displayed to the user and any process error.
SFTCRT	
SFTCRTS_START	
SFTIDINFO	
SFTIDINFOS_START	
SFTSHIP	
SFTSHIPS_START	
SFTSTR	
SFTSTRS_START	
SFTUPD	
SFTUPDS_START	
SSN	Error messages from the SSN Module
TIF	Error messages from the BAR_TIF process
TIFS_START	

VT-User

For VT users there are no logging except the normal server logging. That means that no user specific log files exist.

Web-user

For web users there are no logging except the normal server logging. That means that no user specific log files exist.

Maintenance

Database

Read the Rdb documentation set to find out what can be done, and how database performance can be optimised.

Files

Once in a while, you can delete or purge all logging files in the login directory for a user. Be sure to leave those logs that may be needed later. See chapter Logging for references as to where the logs can be found.

Processes

Bartrack processes are executed in OpenVMS accounts listed in the “User name” column in the table. These accounts are created during installation.

The processes that can be found during Bartrack operation are:

Bartrack processes, internal

Process name	Surveyed by SFQPS	User name	Batch job	Description
BAR_LOGGING	Yes	BAR		Logging.
BAR_SFT_UPD	Yes	BAR_SFT		SFT test status.
BAR_SFT_STR	Yes	BAR_SFT		SFT requests information.
BAR_SFT_IDI	Yes	BAR_SFT		SFT requests information.
BAR_SFT_SHP	Yes	BAR_SFT		SFT requests information.
BAR_SFT_CRT	Yes	BAR_SFT		SFT requests information.
BAR_ORC	Yes	BAR_SFT		Order import
BAR_RMI_xxxx		BAR		One process for each web client.
BAR_RMIREGISTRY		BAR		Web interface.
BAR_RMI		BAR		Web interface
BAR_DTF	Yes	BAR_SFT		Dmq-to-file converter
BAR_TIF	Yes	BAR_TIF	Yes	File interface.
BAR_DBI		BAR	Yes	DB cleaning.
BAR_PRIM		BAR_PRIM	Yes	PRIM interface.
BAR_TRACY		BAR_TRACY	Yes	Tracy interface.

TRACY_BAR		BAR_TRACY	Yes	Tracy interface handling any other records than individual requests by EDI.
BAR_TRACY_BAR		BAR_TRACY	Yes	Tracy interface handling individual requests by EDI.
M2T_BREC		M2T	Yes	Tracy interface handling any other records than individual requests by mail.
M2T_YREC		M2T	Yes	Tracy interface handling individual requests by mail.

In addition to these processes and batch jobs, there might be additional, temporary, processes and batch jobs. For example the **TCYIFRECEIVE** batch job is only visible during the reading of Tracy files into Bartrack.

The BAR_DBI process

The BAR_DBI process is by default set up to execute once every 24 hours. It scans the database looking for scrapped or shipped individuals. A found individual is timestamp is compared to the retention period. If the retention period has expired, the individual is deleted from the Bartrack database. No undo information other than normal Rdb operation is kept.

The BAR_DBI process also cleans up the Alarm Log. It deletes confirmed messages first and then the oldest messages should the log run out of space.

The Bartrack Scrapping Utility

The command **BAR_SCRAP** in the **BAR_TOOLS** directory will allow you to scrap specified individuals from the Bartrack database without using the user interface.

The command will need some criteria that let you specify exactly which individuals to scrap. Only individuals that match these criteria will be scrapped.

N.B. It is important to have an up-to-date backup before using this utility.

N.B. It is important to make a test-run on a copy of the database on a test machine. Some searches may take very long time, depending on the search criteria and the size of the database.

To scrap individuals from Bartrack:

1. Run the command:
   ```
   @BAR_TOOLS:BAR_SCRAP
   ```
2. First you will be shown some information about the severity of using this command.
3. Then you can specify any of six different criteria:
   ```
   Product number:
   R-state:
   Starting pattern:
   First manufacturing date:
   Last manufacturing date:
   Individual status:
   ```
4. After the criteria have been specified, you will get the option to verify your input, and start over again.
5. Then you will be asked to perform the scrapping of individuals matching your criteria.

N.B. All individuals that match the criteria will be scrapped. It may be impossible to get these individuals back.

Users

Identifiers

All VT-terminal users must have the identifier `BARTRACK_VT` granted.

All web-client users must have the identifier `BARTRACK_WEB` granted.

Adding a VT User

To add a Bartrack VT user, you have to do the following:

1. Create a normal OpenVMS user account with the settings of your choice. Set page file quota to greater than 100.000. Do not use the UIC [31,*] since this UIC is reserved for Bartrack.
2. Grant the identifier `BARTRACK_VT` to the user:
   ```
   UAF> GRANT/IDENTIFIER BARTRACK_VT <USERID>
   ```
3. The user must have permission and access to the file `BAR_COM:BARTRACK.COM` and the symbol `BARTRACK`.
4. Add the user as a user in Bartrack. Use the User Administration tab, and be sure to enter the VMS user name as the Bartrack user name.

Resources

Each active (logged in) VT-user requires the following resources:

- One process
- More than 8 MB RAM for the user's server process
- More than 100.000 page file quota

Adding a Web User

To add a web user, all you have to do is add the user in the Bartrack **User Administration** folder.

Every Web-user shares the Bartrack VMS system account, `BAR`. For every user there will be a process called `BAR_RMI_xxxx`, where `xxxx` is a hexadecimal value, starting from 7FFF and counting downwards for each process.

Resources

Each active (logged in) Web-user uses the **BAR** account, and the settings for the **BAR** account applies to each web user.

System Users

There are a number of users that are not real users, that is, they are virtual users used by Bartrack:

- BAR_PRIM
- BAR_SFT
- BAR_TIF
- BAR_TRACY
- BAR_SUBSC (not used)

There are two accounts that can be logged in to:

- BAR – The main account for Bartrack processes
- SFQ – The main account for SFQ processes

The last two users are also used when Bartrack commands shall be given, such as start, stop and restoring.

Starting, Stopping and Upgrading

Starting Bartrack

Normally, Bartrack will be started during the computer's boot sequence. Should you want to start Bartrack manually, you have to be logged in as a system administrator (having the rights to submit in other user's names), then use the following command to start the Bartrack application on the server:

- `$ SUBMIT/USER=BAR SYS$STARTUP:BARTRACK_STARTUP.COM`

The command executes the `BAR_CHECK_BATCH` command, which in its turn starts all relevant images among other things.

VT

Use the following command to start a VT-session to the application:

- `$ BARTRACK`

Web

Use the following procedure to start a web-session to the application:

- Click on the link to Bartrack on the Bartrack homepage The location of the homepage is defined locally at each site.

Stopping Bartrack

To stop Bartrack on the server, you have to be logged in as a system administrator (having the rights to submit in other user's names), then use the following command to stop the Bartrack application on the server:

- `$ SUBMIT/USER=BAR SYS$STARTUP:BARTRACK_SHUTDOWN.COM`

If you want to disable the use of Bartrack temporarily, you can define the system logical `BAR_DOOR` to either `CLOSED` or `OPEN`.

`CLOSED` means that no new VT users can log on to Bartrack.

`OPEN` means that Bartrack operates normally.

If you want to, you can define the `BAR_DOOR` as a local logical name to override the system logical. This enables you to run Bartrack for testing purposes, without Bartrack being available to other users.

The `BAR_DOOR` logical is set to `OPEN` during Bartrack start-up.

Upgrading Bartrack

Please contact helpdesk for information regarding upgrading. See page 7 for contact information.

Troubleshooting

If you encounter any problems or errors that you cannot deal with yourself, please contact our helpdesk. The contact information can be found at page 7 of this manual.

Server

Problem:

A restored database (from backup) is not accessible.

Cause:

The protected subsystem identifier has changed. It is probably caused by trying to restore the database on another computer, with different identifier. The subsystem identifier is stored in the backed up database.

Solution:

Check the protected subsystem identifiers.

Web

Problem:

It is not possible to connect to Bartrack using the web interface. Connection is refused or nothing happens.

Cause:

There are no available RMI processes because the RMI process handler has crashed.

Solution:

Restart the RMI handler:

Enter the following command at the VMS prompt. You must have Bartrack privileges.

```
>$ @BAR_COM:STOP_BAR_RMI S
```

The 'S' at the end means [S]tart the handler after it has been stopped. If you issue the command without the 'S' the RMI handler will be shut down and not restarted.

Problem:

It is not possible to connect to Bartrack using the web interface. Connection is refused or nothing happens.

Cause:

It is not possible to access the Bartrack web server and/or Bartrack server using the network.

Solution:

Investigate if it is possible to access Bartrack from the client:

- Ping the Bartrack web server
- Ping the Bartrack server

Problem:

It is not possible to connect to Bartrack using the web interface. Connection is refused or nothing happens.

Cause:

The version of JRE (Java Runtime Environment) is not correct.

Solution:

Verify which version of JRE is installed on the client. Compare it to the version stated in the TES (Target Environment Specification) or Prevas website:

www.prevas.se/systems

Go to Bartrack and log in to the customer pages to see if more information on JRE versions is available.

Problem:

The error message “4999” is displayed when trying to start a Bartrack session.

Cause:

The database is not accessible.

Solution:

Check the **`BAR_LOG:ERROR.LOG`** for more details about the problem.

Check the access rights, subsystems and identifiers for the user.

VT-Session

Problem:

The error message “4999” is displayed when trying to start a Bartrack session.

Cause:

The database is not accessible.

Solution:

Check the access rights, subsystems and identifiers for the user.

Problem:

It is not possible to start Bartrack.

Cause:

The user account is not correct.

Solution:

The user must have the identifier `BARTRACK_VT`.
The user must not have the UIC [31,*].
The user must exist in Bartrack. Use the **User Administration** tab.

Problem:

When you start Bartrack, you get

```
Enable error. FormsStatus = 3
```

Cause:

The terminal session is not correctly defined, or you are using an older VT-terminal.

Solution:

If you are using a VT-terminal:

- The terminal mode must be set to VT300 (or greater)
- The terminal ID must be set to VT320 (or greater)

If you are using a terminal emulator (such as KEA, Reflection or xCursion):

- The terminal mode must be set to VT300 (or greater)
- The terminal ID must be set to VT320 (or greater)

The command to set the terminal mode is:

```
$ SET TERMINAL/DEVICE_TYPE=VT300
```

Glossary of Terms

BEA	BEA Systems.
BmQ	BEAmessageQ – Message oriented middleware from BEA Systems. The same as DmQ.
Data identifier	Special codes in the beginning of a barcode. Used for navigating between fields
DEC	Digital Equipment Corporation
DECforms	Tool for developing and running UI applications on VT terminals
DmQ	DECmessageQ – Message oriented middleware from BEA Systems. The same as BmQ.
EDI	Electronic Data Interchange – A common protocol, used by Bartrack to exchange data with Tracy
IIS	Internet Information Server. Microsoft's web server
MRQ	Multi-Reader Queue. A special DmQ queue allowing several readers.
OBB	ObjectBroker – Middleware from BEA Systems, used by Bartrack to communicate with the PC-clients
PC	Personal Computer. A desktop computer with Windows.
PRIM	Ericsson's central product register.
Rdb	Oracle Rdb – Relational database used by Bartrack
RMI	Remote Method Invocation. This is a communication feature used by Java and the web browser.
SFQ	Shop Floor Quality - The collection of Prevas's software for manufacturing
SFQPS	SFQ Process Surveillance - Bartracks surveillance utility
SPU	SFQ Primgate Utility - A subscription utility
Tracy	Ericsson's global traceability database where all individuals produced and delivered are stored
UI/GUI	User Interface / Graphical User Interface - The screen layout and procedures used seen by the user
VMS	OpenVMS. The operating system that Bartrack uses. VMS = Virtual Memory System.
VT	Video Terminal. A "dumb" black/white terminal without computing power.

Indexes

Lists

List of pictures

List of tables