Major New Features in TRIP 7

TRIPsystem
Product Documentation

End User License Agreement

All rights to this software, its documentation and logotypes of the TRIP product family and software (altogether “Software”) supplied by Smaser AG (Smaser) are exclusively owned by Smaser.

The transfer of this Software, solutions or parts thereof requires the prior written agreement of Smaser. Furthermore, the customer has the right to use licensed Software and / or process solutions supplied by Smaser to the extent specified in his contract with Smaser.

The free-to-use non-commercial version doesn’t require a prior written agreement with Smaser but such customers, organizations and/or third parties agree by using the software and / or solution of Smaser to be strongly obliged to keep all rights to this software, documentation and logotypes of the TRIP product family absolutely uninfringed and protected.

Table of Contents

Introduction

This document describes briefly the major new features of TRIP versions 7.0, 7.1 and 7.2. More information can be found in the specific documentation.

TRIP 7.0

- Configuration file
- TRIPclassic UI on Windows
- Command line tool interface changes
- Event monitoring
- Synchronous indexing

TRIP 7.1

- Graph databases
- Separate indexing enhancement
- Unicode conversion with Packit

TRIP 7.2

- New license format
- Performance improvements for clustered search and display
- Removal of the limit of max 250 open databases
- Execute access to procedures owned by other users
- Database file name verification tool

TRIP 7.0

Configuration File

Changed Name and Location

From TRIPsystem 7.0, the configuration file has been renamed and relocated. In older TRIP versions, the configuration file was located under:

- Windows: `C:\TRIPrcs`
- Unix/Linux: `/.TRIPrcs`

The new location is:

- Windows: `%TDBS_HOME%\conf\tdbs.conf`
- Unix/Linux: `/usr/local/trip/sys/conf/tdbs.conf`

TDBS_HOME refers to the TRIPsystem installation directory. This is not an environment variable, although its value is can be reported by the queryrcs tool on Windows.

PLEASE NOTE! We have updated our documentation to reflect this change, but if you should find the name TRIPrcs in some document please read it as tdbs.conf.

Effects on System Administration

This change may have consequences for existing system administrations and installation procedures. Custom scripts and tools that read or write the TRIPrcs file must be altered so that the tdbs.conf file is accessed instead.

Effects on Installation Procedures

If a custom installation procedure has been or will be implemented, the changed location of the configuration file must be taken into account. Writing a TRIPrcs file instead of the required tdbs.conf file when performing a custom installation of TRIPsystem 7.0 or later will result in undefined behaviour when attempting to run it, and failure to correctly install add-on products such as TRIPview, TRIPxml and TRIPsql.

The directory `/usr/local/trip/sys/conf` is actually a link to the installation directory. For example, if TRIPsystem is installed under `/opt/trip/sys/v700`, then there will be a link:

`/usr/local/trip/sys/conf` → `/opt/trip/sys/v700/conf`

If you are writing a custom installer, you must create this link.

The actual configuration file must ALWAYS be written under the `conf` directory in the TRIP installation directory.

TRIPclassic UI

In older versions of TRIPclassic the command prompt window always used black background with white text, but from v7 onwards the command prompt window keeps the background and text colors defined for the command prompt window.

You can also specify a color for the Bold text attribute by setting the environment variable TRIP_BOLD_COLOR=x in the TRIP config file (tdbs.conf). Supported values for x are:

B = Blue

C = Cyan

G = Green

M = Magenta

R = Red

Y = Yellow

When creating a desktop shortcut as described in the Installation Guide, it was previously setup to use the code page 1252. From version 7.0, this is now by default code page 858, resulting in pseudo graphics being used to create the “boxes” used by TRIPclassic to partition the window for searches, command

history and display. Other code pages supported are 437, 850 and 865. These will also use pseudo graphics.

The € (Euro) symbol is only supported by the Lucida Console font with the code pages 858 and 1252.

If you want to keep the old style you just set the code page to 1252 and choose black background and white text in the properties for the command prompt window started by the desktop shortcut. Command Prompts for TRIP saved in TRIP versions before version 7.0 will continue as before until you change them.

The behavior of the numeric keypad on the keyboard has also been changed from v7 and will now be as described in CCL_Command_Reference.pdf and TRIPclassic_User_Guide.pdf and it now behaves in the same way as on Unix platforms.

Command Line Tools

Changed Tools

Several command line tools have been enhanced with the capability to take arguments and options via the command line. In several cases, the command line interfaces for the tools have also changed.

The following command line tools have been changed:

- bafini
- exif
- index
- load
- loadix
- rebif
- setlock

Changes to bafini

The bafini program now takes the database name to reset the index status for as argument. It still will prompt for the database name if invoked without arguments.

Example:

```
bafini ALICE
```

Changes to exif

The exif program now can take all its required parameters as arguments. It will prompt for any arguments not specified, and thus remains backwards compatible with the previous mode of use.

Display valid options to the exif program using the “—help” or “-h” options to the program on the command line.

Changes to index

The format of the parameters to the index command line program has been altered.

Option	Description
-d <database>	Name of database to index
-r <recordid>	Record ID of single record to index.
--[no-]reindex	Re-index the database.
--defaults	When possible, use default values without prompting
-s <filename>	STO file name
-b <filename>	BUT file name

Example:

```
index –d ALICE ––reindex ––defaults
```

Note that it is still possible to invoke the index command line program as previously done by specifying the database name as the first argument, or to omit arguments altogether and have the index program prompt for the database name.

Changes to load

The argument list for the load program has been changed. Previous invocation syntax is no longer supported. The current syntax can be displayed using the "—help" or "-h" options to the program on the command line.
The current valid parameters to the load program are:

Option	Description
-d <database>	Name of database to load data into
-f <tfo-file>	TFORM file with data to load
-x <ldx>	LDX file name
-t <timestamp>	Starting time stamp (YYYY-MM-DD HH:MM:SS)
-r <maxcount>	Highest record number to be added
--[no-]sentence	Enable or disable sentence scanning
--[no-]tstamp	Enable or disable use of timestamps
--defaults	When possible, use default values without prompting

Example:

```
load –d mydb –f mydata.tfo ––defaults
```

Changes to the loadix script

The argument list for the loadix script has been changed. Previous invocation syntax is no longer supported. The current syntax can be displayed using the "—help" or "-h" options to the program on the command line.

The current valid parameters to the loadix script are:

Option	Description
-d <database>	Name of database to load data into and index

-f <tform>	TFORM file to load
-t <tstamp>	Timestamp (YYYY-MM-DD HH:MM:SS)
--[no-]sentence	Enable or disable sentence scanning
--[no-]reindex	Index incrementally or perform re-indexing
--defaults	When possible, use default values without prompting

Example:

```
loadix –d mydb –f mydata.tfo --defaults
```

Changes to setlock

The setlock program has been extended with the capability to take the license information on the command line. If no or insufficient arguments are specified, setlock will prompt for the required information, making it backward-compatible with the behaviour in previous versions.

The valid options to the setlock program are:

Option	Description
-d <date>	License expiration date in format YYYYMMDD
-o <options>	License options
-c <count>	Max number of concurrent users (or zero for no limit)
-r <count>	Max number of registered users (or zero for no limit)
--cpulock	License is CPU locked
-L <codes>	Space-delimited list of license codes to be installed

NOTE: Setlock was changed again in version 7.2, rendering this information obsolete. See below for more information.

Changes to rebif

The rebif program now can take all its required parameters as arguments. It will prompt for any arguments not specified, and thus remains backwards compatible with the previous mode of use.

Display valid options to the rebif program using the “—help” or “-h” options to the program on the command line. The valid options to the rebif program are:

Option	Description
-f <filename>	BIF or VIF file name
-o <filename>	New name for BIF or VIF file
-B <filename>	BUT file name
-w <entrywidth>	Entry width (prefix with +/- to modify)
-e <value>	Entry overflow limit percentage (0-500)
-d <value>	Data sub-item size limit (20-100)
-p <value>	Pointer sub-item size limit (50-200)
-g <value>	Size guideline for new segments (10-75)
--defaults	When possible, use default values without prompting
--[no-]but	Use BUT file? (implicit if -B is specified)

Removed Tools

The following tools have been removed:

- UNIX/Linux: load shell script– replaced with a binary executable
- UNIX/Linux: index shell script – replaced with a binary executable
- baffit shell script – replaced with the “load” binary executable
- baffre – integrated into the ‘index’ program
- resciff – obsolete, removed without replacement
- sciffex – obsolete, removed without replacement
- sciffit – integrated into the ‘index’ program

Effects on System Administration and Custom Installation Procedures

If you have custom batch scripts or other utilities that make use of the TRIP command line tools mentioned above, you must review the changes to the tools you use and adapt your scripts and utilities accordingly.

Event Monitoring

Overview

Event monitoring is a feature that allows the TRIP systems administrator or DBA to output events from TRIP sessions.

Events in this context are:

- Errors in the current session
- Changes to users (e.g. created, deleted)
- Changes and actions on databases (e.g. created, deleted, opened, closed)
- Submitted batch jobs (index jobs, print jobs and global updates)
- Session changes (login, logout)

How to Enable

Event monitoring is disabled by default. It is enabled by adding the following property to the tdbs.conf file:

Property name	Value description
TDBS_MONITOR_LIB	Fully qualified path to the monitor library file

It is also possible to generate performance measurements on query executions. This is not enabled per default, even if event monitoring is otherwise enabled, but may be enabled by setting the following property in the tdbs.conf file:

Property name	Value description
TDBS_MONITOR_QPERF	“Y” or “1” to enable query performance event logging

Example, with query monitoring enabled:

```
TDBS_MONITOR_LIB=${TDBS_HOME}/bin/libmonlog.so
TDBS_MONITOR_QPERF=Y
```

Monitoring is always enabled if the `TDBS_MONITOR_LIB` property is defined and refers to the `libmonlog` shared object or DLL file.

Parameters

Additional parameters to the event monitor can be given by adding the following properties to the tdbs.conf file:

Property name	Value description
TDBS_MONLOG_FLUSH	Determines if log statements should be forced to disk immediately as they are written, or if they are allowed to delay in the file system cache. Enable by specifying Y. Default is N (false).
TDBS_MONLOG_MONOLITHIC	Determines if the monitoring event log is monolithic or per session. When monolithic logging is enabled, all sessions share the same log file. Default is Y (true). Disable by specifying N.
TDBS_MONLOG_MONOLITHIC_PERIOD	Determines the time period of monolithic logs. A new log file is created for each new period. Valid values are: • HOUR • DAY • WEEK • MONTH • YEAR Default is DAY.
TDBS_MONLOG_TSTAMP	Determines if a year-to-second time stamp should be included in the log file for each log row. Default is N (false). Enable by specifying Y.

Event Log Output

The format of the event log file is a comma-separated values (CSV) file. The following tables describe the fields in the file.

Common fields for all message types:

Field nr	Value Type	Description
1	Process ID	The process ID of the process from which the event originated. This is typically the TRIP Daemon (tripd), the TRIP server (tbserver), and TRIP classic (trip), but also includes server-side TRIP process that directly uses the TRIP kernel.
2	Time stamp	Time stamp in the format “YYYY-MM-DD hh:mm:ss”. This field contains an empty value unless the TDBS_MONLOG_TSTAMP property is set to Y.
3	Hi-res time	A nano-second time offset from the start of the process from which the event originated. The format of this value is hh:mm:ss:mmm.uuu.nnn.
4	Message Type	Indicates the type of the information on the current row. The value types of the remainder of the fields are determined by this value.

The following fields apply to message type PROCINFO, which denotes basic process information. This information is sent when the monitored process is starting up.

Field nr	Value Type	Description
5	Process Type	The type of the process. This is “TRIP kernel” for a TRIP session or “Tieto TRIP daemon” for the TRIP Daemon (tripd).
6	User name	The name of the currently logged on user
7	Is Session	Indicates if the process currently hosts an active TRIP session, i.e. a logged on user.

The following fields apply to message type EVENT, the most common entry in the event monitoring log.

Field nr	Value Type	Description
5	Severity	The severity of the event. Possible values are INFO, WARNING, ERROR and CRITICAL.
6	Resource Type	The type of resource associated with the event. Typical values are USER and DATABASE. Additional possible values are SESSION, HOST, PROCESS and JOB.
7	Resource ID	The identity name of the resource associated with the event.
8	Event Type	The type of the event being signalled. Valid values are: • STARTED – e.g. a successful user login • STOPPED – e.g. a logout • OCCURRED – a generic event • CREATED – e.g. a database was created • DESTROYED – e.g. a database was deleted • ERROR – e.g. an error has been raised • WARNING – e.g. a non-critical error has occurred • FAILED – e.g. generic non-erroneous failure • MEASUREMENT – e.g. query performance
9	Event Name	Display name or title for the event. In case of errors and warnings, this indicates the code of the TRIP error.
10	Description	Human-readable event description. In case of errors and warnings, this is typically the TRIP error message.

Log File Location and Name

The event log is written to the directory indicated by the TDBS_LOG property in the tdbs.conf file.

The name of the file is determined by whether monolithic mode is enabled (see the description of the property TDBS_MONLOG_MONOLITHIC) and the period of the monolithic logging (see the description of the property TDBS_MONLOG_MONOLITHIC_PERIOD).

If monolithic logging is disabled, the log file name will be:

```
eventlog_<year><month><day><hour><min><sec>_<pid>.log
```

If monolithic logging is enabled, the log file names depend on the time period:

HOUR	`eventlog_<year><month><day><hour><min><sec>.log`
DAY	`eventlog_<year><month><day>.log`
WEEK	`eventlog_<year><dayofyear>.log`
MONTH	`eventlog_<year><month>.log`
YEAR	`eventlog_<year>.log`

Year is always a 4-digit number. The values for month, day, hour, minute and second are all zero-padded 2-digit numbers. The value for 'dayofyear' used for WEEK time period is a zero-padded 3-digit number that denotes the day of the year for the Monday in the current week.

Synchronous Indexing

Overview

Synchronous indexing is a way to index new and changed records in a database that is performed within the TRIP session itself and not submitted to the TRIP Daemon (tripd) as a batch job. A call to perform synchronous indexing does not return until the index operation is complete. For regular batch-oriented indexing, the call to perform indexing returns immediately and the index operation itself is then executed asynchronously, allowing the application to continue its processing.

Indexing Single Records

Synchronous indexing of single records can be performed via:

- The TRIP API function TdbIndex. It replaces an older, broken function with the same name.
- The index method on the TdbRecord class in version 3.0 of TRIPjxp and TRIPnxp.
- The index method on the Record class in TRIPjtk.
- The index method on the Record object in TRIPcom (TRIPclient).

If a database is frequently indexed in this manner, it is strongly recommended that a regular index job (not specific to a particular record) is performed regularly. This is because single record indexing is a much lighter weight operation, which – among other things - will not perform reorganization of the index files even if needed. Using an index file in need of reorganization will result in degraded performance for searching and indexing.

Multiple Records

TRIP supports synchronous indexing of multiple records. This operation is identical to regular batch-oriented indexing but is running entirely within the session and is not submitted to the TRIP Daemon (tripd) for batch (asynchronous) processing.

This can be done via a new overloaded index method in the TdbDatabaseDesign class in version 3.0 of TRIPjxp and TRIPnxp. It can also be done using the TdbIndex API function by passing 0 (zero) as number of the record to index.

Note that synchronous indexing of a database with a large number of new or altered records may take a long time and potentially causing a timeout in the client/server session.

TRIP 7.1

Graph Databases

Graph databases are supported in TRIPsystem from version 7.1. An overview of the capabilities of graph databases is found in the document "*TRIP as a Graph Database – Tutorial*" (GraphDatabaseTutorial.pdf), provided with the documentation set for TRIPsystem.

Separate Indexing Enhancement

Performance enhancements have been made to the indexer for separate indexed fields (index attribute "*create field-specific index*" in TRIPmanager). This affects DISPLAY orders on such fields and makes it more convenient to do on-the-fly generation of facets also in larger databases.

In order for this change to take effect on a database, it must be first be re-indexed.

Unicode Conversion with Packit

The Packit program can now be used to convert a non-Unicode database to Unicode as part of its operations. For detailed instructions, please refer to the Packit manual (Packit.pdf), provided with the documentation set for TRIPsystem.

TRIP 7.2

New license format

The format of the TRIP license has been renewed. The most visible effect is that licenses are entered as an opaque file instead of as a set of properties.

This affects the Unix/Linux installation procedure and the usage on all platforms of the gethw and setlock tools. Refer to the TRIPsystem_Installation_Guide_Unix.pdf and TRIP_license_key.pdf documents for more information.

Performance improvements for clustered search and display

When searching or using DISPLAY in a database cluster, TRIP will now perform the operation in parallel against the individual databases instead of serially like it was done in older versions. The number of worker threads is 16 by default, but can be customized in the tdbs.conf configuration file using the new TDBS_MAX_THREADS symbol.

Removal of the limit of max 250 open databases

The limit on the maximum number of open databases has been removed. It is no longer set to a hard limit of 250. The new limit is variable and depends mainly on two factors: what the license allows for, and what the operating system is capable of handling.

There are two feature levels in the license that govern the use of open databases: the number of open databases in total, and the max size of database clusters. Use the new showlic tool to inspect what your license allows for.

To use a large number of databases, the operating system may have to be configured to allow for additional file descriptors per process. On UNIX and Linux systems, this limit is fairly low by default, and may not be sufficient even for “only” 250 open databases. To calculate the needed number of open file descriptors allowed per process:

- If transaction log files **are not** used (the default): multiply the desired max number of open databases and thesauri by 3 and add 16.
- If transaction log files **are** used: multiply the desired max number of open databases and thesauri by 4 and add 16.

In order to make room for larger data sets, two standard kernel window formats have been adjusted to allow for higher record counts; the history window and the display result window. The last column is now two columns further to the right for the history window, and three columns further to the right for the display result window. **Applications that parse the contents of these windows may have to be adjusted**.

Execute access to procedures owned by other users

By defining the new symbol TDBS_RO_USER_PROCEDURES in the privileged section of the tdbs.conf file, private user procedures will be executable by other users. Other users will also be able to read procedure properties such as the comment when such information is requested via the API. Procedure lists such as can be seen in TRIPclassic and TRIPmanager are not affected by this and therefore behave like before.

This does not apply to procedures owned by SYSTEM. Such procedures remain strictly private.

Database file name verification tool

Introduced in version 7.2-1, the command line tool `chkdbfiles` is used to verify the paths, file names, and existence of the BAF, BIF and VIF files for databases. Usage of this tool is recommended especially when moving a TRIP installation from one machine to another. The following problems can be detected:

- Path does not exist for BAF, BIF or VIF files when specified with a fully qualified path.
- Logical name symbol (tdbs.conf entry) used to specify the directory of a BAF, BIF or VIF file does not exist.
- Logical name symbol (tdbs.conf entry) used to specify the directory of a BAF, BIF or VIF file does correspond to an existing directory path.
- The database does not contain any records
- The database index is not sufficiently up to date

The program must be started as a TRIP user with file manager rights. The program will perform an analysis of all databases owned by the specified user. If the SYSTEM user is used and the TDBS_SUPERMAN logical name is set, an analysis of all databases will be performed.

The valid options to the chkdbfiles program are:

Option	Description
-u <username>	Name of TRIP user with file manager rights

-p <password>	Password for the specified TRIP user
--prompt	Prompt for password
--[no-]empty	Enable/disable warning for empty databases (enabled by default)
--[no-]files	Enable/disable warning for missing files (enabled by default)
--[no-]logfiles	Enable/disable checking of LOG files (disabled by default)

Below is an example of an analysis that have found that the logical name used as storage location for a database does not correspond to a directory:

```
**** TRIP System Utility CHKDBFILES - Analyze database file names ****

                          infinIT Services GmbH

Checking database file existence as user DEMOFM.

DEMODB
   BAF is using symbol 'LOCAL_BASES' defined as
   : C:\Data\demo
   : ERROR: the directory does not exist

   BIF is using symbol 'LOCAL_BASES' defined as
   : C:\Data\demo
   : ERROR: the directory does not exist

   VIF is using symbol 'LOCAL_BASES' defined as
   : C:\Data\demo
   : ERROR: the directory does not exist

   WARNING: Database is empty
```