English

FUJITSU Software

openUTM V6.3

Messages, Debugging and Diagnostics in Unix Systems and Windows Systems

User Guide

Edition January 2015 - update May 2015

Comments… Suggestions… Corrections…

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to:
manuals@ts.fujitsu.com

Certified documentation according to DIN EN ISO 9001:2008

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2008.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright and Trademarks

Contents

1 Preface

Modern enterprise-wide IT environments are subjected to many challenges of rapidly increasing importance. This is the result of:

- heterogeneous system landscapes
- different hardware platforms
- different networks and different types of network access (TCP/IP, SNA, ...)
- the applications used by companies

Consequently, problems arise – whether as a result of mergers, joint ventures or labor-saving measures. Companies are demanding flexible, scalable applications, as well as transaction processing capability for processes and data, while business processes are becoming more and more complex. The growth of globalization means, of course, that applications are expected to run 24 hours a day, seven days a week, and must offer high availability in order to enable Internet access to existing applications across time zones.

openUTM is a high-end platform for transaction processing that offers a runtime environment that meets all these requirements of modern, business-critical applications, because openUTM combines all the standards and advantages of transaction monitor middleware platforms and message queuing systems:

- consistency of data and processing
- high availability of the applications (not just the hardware)
- high throughput even when there are large numbers of users (i.e. highly scalable)
- flexibility as regards changes to and adaptation of the IT system

An UTM application can be run as a standalone UTM application or sumultanously on several different computers as a UTM cluster application.

openUTM forms part of the comprehensive **openSEAS** offering. In conjunction with the Oracle Fusion middleware, openSEAS delivers all the functions required for application innovation and modern application development. Innovative products use the sophisticated technology of openUTM in the context of the **openSEAS** product offering:

- BeanConnect is an adapter that conforms to the Java EE Connector Architecture (JCA) and supports standardized connection of UTM applications to Java EE application servers. This makes it possible to integrate tried-and-tested legacy applications in new business processes.
- The WebTransactions member of the openSEAS family is a product that allows tried-and-tested host applications to be used flexibly in new business processes and modern application scenarios. Existing UTM applications can be migrated to the Web without modification.

1.1 Summary of contents and target group

The openUTM manual “Messages, Debugging and Diagnostics in Unix Systems and Windows Systems” is intended for users, administrators and programmers of UTM applications.

It describes how to debug an openUTM application under Unix systems and Windows systems, the structure of the openUTM dump, behavior in the event of an error, and all the openUTM messages and return codes output by openUTM.

Chapter 4 also describes the openUTM message system and the options for outputting messages in different languages for specific users or for modifying the messages supplied with openUTM for specific applications or redirecting messages to different destinations.

This manual assumes some familiarity with openUTM and Unix operating systems Windows operating systems respectively. A separate openUTM manual “Messages, Debugging and Diagnostics in BS2000 Systems” is available for the BS2000/OSD operating system.

Wherever the term Unix system or Unix platform is used in the following, then this should be understood to mean both a Unix-based operating system such as Solaris or HP-UX and a Linux distribution such as SUSE or Red Hat.

Wherever the term Windows system or Windows platform is used below, this should be understood to mean all the variants of Windows under which openUTM runs.

1.2 Summary of contents of the openUTM documentation

This section provides an overview of the manuals in the openUTM suite and of the various related products.

1.2.1 openUTM documentation

The openUTM documentation consists of manuals, the online help systems for the graphical administration workstation openUTM WinAdmin and the graphical administration tool WebAdmin, and a release note for each platform on which openUTM is released.

Some manuals are valid for all platforms, and others apply specifically to BS2000 systems, Unix systems or Windows systems.

All the manuals are available as PDF files on the internet at

http://manuals.ts.fujitsu.com

On this site, enter the search term “openUTM V6.3“ in the **Search by product** field to display all openUTM manuals of version 6.3.

The manuals are included on the Enterprise DVD with open platforms and are available on the WinAdmin DVD for BS2000 systems.

The following sections provide a task-oriented overview of the openUTM V6.3 documentation. You will find a complete list of documentation for openUTM in the chapter on related publications at the back of the manual on page 483.

Introduction and overview

The **Concepts and Functions** manual gives a coherent overview of the essential functions, features and areas of application of openUTM. It contains all the information required to plan a UTM operation and to design an UTM application. The manual explains what openUTM is, how it is used, and how it is integrated in the BS2000, Unix based and Windows based platforms.

Programming

- You will require the **Programming Applications with KDCS for COBOL, C and C++** manual to create server applications via the KDCS interface. This manual describes the KDCS interface as used for COBOL, C and C++. This interface provides the basic functions of the universal transaction monitor, as well as the calls for distributed processing. The manual also describes interaction with databases.
- You will require the **Creating Applications with X/Open Interfaces** manual if you want to use the X/Open interface. This manual contains descriptions of the UTM-specific extensions to the X/Open program interfaces TX, CPI-C and XATMI as well as notes on configuring and operating UTM applications which use X/Open interfaces. In addition, you will require the X/Open-CAE specification for the corresponding X/Open interface.
- If you want to interchange data on the basis of XML, you will need the document entitled openUTM **XML for openUTM**. This describes the C and COBOL calls required to work with XML documents.
- For BS2000 systems there is supplementary documentation on the programming languages Assembler, Fortran, Pascal-XT and PL/1.

Configuration

The **Generating Applications** manual is available to you for defining configurations. This describes for both standalone UTM applications and UTM cluster applications how to use the UTM tool KDCDEF to

- define the configuration
- generate the KDCFILE
- and generate the UTM cluster files for UTM cluster applications

In addition, it also shows you how to transfer important administration and user data to a new KDCFILE using the KDCUPD tool. You do this, for example, when moving to a new openUTM version or after changes have been made to the configuration. In the case of UTM cluster applications, it also indicates how you you can use the KDCUPD tool to transfer this data to the new UTM cluster files.

Linking, starting and using UTM applications

In order to be able to use UTM applications, you will need the **Using openUTM Applications** manual for the relevant operating system (BS2000 or Unix systems/Windows systems). This describes how to link and start a UTM application program, how to sign on and off to and from a UTM application and how to replace application programs dynamically and in a structured manner. It also contains the UTM commands that are available to the terminal user. Additionally, those issues are described in detail that need to be considered when operating UTM cluster applications.

Administering applications and changing configurations dynamically

- The **Administering Applications** manual describes the program interface for administration and the UTM administration commands. It provides information on how to create your own administration programs for operating a standalone UTM application or a UTM cluster application and on the facilities for administering several different applications centrally. It also describes how to administer message queues and printers using the KDCS calls DADM and PADM.
- If you are using the graphical administration workstation **openUTM WinAdmin** or the Web application **openUTM WebAdmin**, which provides comparable functionality, then the following documentation is available to you:
  - A **description of WinAdmin** and **description of WebAdmin**, which provide a comprehensive overview of the functional scope and handling of WinAdmin/WebAdmin. These documents are shipped with the associated software and are also available online as a PDF file.
  - The respective **online help systems**, which provide context-sensitive help information on all dialog boxes and associated parameters offered by the graphical user interface. In addition, it also tells you how to configure WinAdmin or WebAdmin in order to administer standalone UTM applications and UTM cluster applications.

For detailed information on the integration of openUTM WebAdmin in SE Server's SE Manager, see the SE Server manual **Operation and Administration**.

Testing and diagnosing errors

You will also require the **Messages, Debugging and Diagnostics** manuals (there are separate manuals for Unix systems / Windows systems and for BS2000 systems) to carry out the tasks mentioned above. These manuals describe how to debug a UTM application, the contents and evaluation of a UTM dump, the behavior in the event of an error, and the openUTM message system, and also lists all messages and return codes output by openUTM.

Creating openUTM clients

The following manuals are available to you if you want to create client applications for communication with UTM applications:

- The **openUTM-Client for the UPIC Carrier System** describes the creation and operation of client applications based on UPIC. In addition to the description of the CPI-C and XATMI interfaces, you will find information on how you can use the C++ classes to create programs quickly and easily.
- The **openUTM-Client for the OpenCPIC Carrier System** manual describes how to install and configure OpenCPIC and configure an OpenCPIC application. It describes how to install OpenCPIC and how to configure an OpenCPIC application. It indicates what needs to be taken into account when programming a CPI-C application and what restrictions apply compared with the X/Open CPI-C interface.
- The documentation for the **JUpic-Java classes** shipped with **BeanConnect** is supplied with the software. This documentation consists of Word and PDF files that describe its introduction and installation and of Java documentation with a description of the Java classes.
- The **BizXML2Cobol** manual describes how you can extend existing COBOL programs of a UTM application in such a way that they can be used as an XML-based standard Web service. How to work with the graphical user interface is described in the **online help system**.
- If you want to provide UTM services on the Web quickly and easily then you need the manual **WebServices for openUTM**. The manual describes how to use the software product WS4UTM (WebServices for openUTM) to make the services of UTM applications available as Web services. The use of the graphical user interface is described in the corresponding **online help system**.

Communicating with the IBM world

If you want to communicate with IBM transaction systems, then you will also require the manual **Distributed Transaction Processing between openUTM and CICS, IMS and LU6.2 Applications**. This describes the CICS commands, IMS macros and UTM calls that are required to link UTM applications to CICS and IMS applications. The link capabilities are described using detailed configuration and generation examples. The manual also describes communication via openUTM-LU62 as well as its installation, generation and administration.

PCMX documentation

The communications program PCMX is supplied with openUTM on Unix and Windows systems. The functions of PCMX are described in the following documents:

- CMX manual “Betrieb und Administration“ (Unix-Systeme) (only available in German)
- PCMX online help system for Windows systems

1.2.2 Documentation for the openSEAS product environment

The **Concepts and Functions** manual briefly describes how openUTM is connected to the openSEAS product environment. The following sections indicate which openSEAS documentation is relevant to openUTM.

Integrating Java EE application servers and UTM applications

The BeanConnect adapter forms part of the openSEAS product suite. The BeanConnect adapter implements the connection between conventional transaction monitors and Java EE application servers and thus permits the efficient integration of legacy applications in Java applications.

- The manual **BeanConnect** describes the product BeanConnect, that provides a JCA 1.5- and JCA 1.6-compliant adapter which connects UTM applications with applications based on Java EE, e.g. the Oracle application server.
  The manuals for the Oracle application server can be obtained from Oracle.

Connecting to the web and application integration

You require the WebTransactions manuals to connect new and existing UTM applications to the Web using the product **WebTransactions**.

The manuals will also be supplemented by JavaDocs.

1.2.3 Readme files

Information on any functional changes and additions to the current product version described in this manual can be found in the product-specific Readme files.

Readme files are available to you online in addition to the product manuals under the various products at *http://manuals.ts.fujitsu.com*. For the BS2000 platform, you will also find the Readme files on the Softbook DVD.

Additional product information

Current information, version and hardware dependencies, and instructions for installing and using a product version are contained in the associated Release Notice. These Release Notices are available online at *http://manuals.ts.fujitsu.com*.

Readme files under Unix systems

The Readme file and any other files, such as a manual supplement file, can be found in the *utmpath* under `/docs/`*language*.

Readme files under Windows systems

The Readme file and any other files, such as a manual supplement file, can be found in the *utmpath* under `\Docs\`*language*.

1.3 Innovations in openUTM V6.3

The following sections provide more detail on the innovations in the individual areas.

1.3.1 New server functions

Additional UTM system processes for internal tasks

In addition to the processes specified by means of the start parameters, UTM starts up to three additional processes that are reserved for internal openUTM tasks or privileged jobs issued by the administrator.

To permit this, both generation and administration interfaces have been extended:

- Generation, KDCDEF statement MAX
  - New operand PRIVILEGED-LTERM, used to identify a specific LTERM as privileged. When a user signs on with administration authorizations, all the user's jobs are considered to be privileged jobs.
  - TASKS operand: The maximum value has been reduced to 240 due to the additional system processes.
- KDCADMI administration interface
  - Data structure *kc_max_par_str*: New field *privileged_lterm* for the generated privileged LTERM.
  - Data structure *kc_tasks_par_str*: New fields *gen_system_tasks* and *curr_system_tasks* for the system processes.
  - Data structure *kc_curr_par_str*: New field *curr_system_tasks* for the system processes.

Higher resolution for output of used CPU time

The used CPU time is now output in microseconds for TACs and in milliseconds for USERs. The following interfaces have been changed to support this:

- KDCADMI
  - Data structure *kc_tac_str*: New field *taccpu_micro_sec* for the average used CPU time in microseconds.
  - Data structures *kc_user_str* and *kc_user_dyn1_str*: New field *cputime_msec* for the used CPU time in milliseconds.

- KDCADM command interface
  - KDCINF type=TAC: TACCPU outputs the average used CPU time in microseconds.
  - KDCINF type=USER: CPUTIME outputs the used CPU time in milliseconds.
- KDCEVAL lists
  - Some times are now output in microseconds in the KDCEVAL lists.

New trace functions

Additional traces can be enabled and disabled during live operation.

– ADMI trace, i.e. trace of the administration program interface (KDCADMI)
– X/Open traces (CPI-C, TX, XATMI)

The following interfaces have been extended to support this:

- Start parameters:

  New start parameters ADMI-TRACE, CPIC-TRACE, TX-TRACE and XATMI-TRACE for enabling traces.

- KDCADMI

  Data structure *kc_diag_and_account_par_str*: New fields *admi_trace*, *cpic_trace*, *tx_trace* and *xatmi_trace* for enabling and disabling traces.

KDCDEF input/output via LMS library elements

In BS2000 systems, it is possible to read KDCDEF statements from LMS library elements and, in the case of inverse KDCDEF, output them to LMS library elements. The following interfaces have been extended to support this:

- Generation
  - KDCDEF statement OPTION: New operand value LIBRARY-ELEMENT(...) in the DATA operand.
  - KDCDEF statement CREATE-CONTROL-STATEMENTS: New operand value LIBRARY-ELEMENT(...) in the TO-FILE operand.
- KDCADMI

  Data structure *kc_create_statements_str*: New fields *lib_name*, *elem_name*, *vers*, *type*, *stmt_type* and *file_error_code*.

- Messages

  New messages K234, K519 and K520 when reading KDCDEF statements from LMS library elements and outputting KDCDEF statements to LMS library elements.

Performance enhancements

- UTM cache

  The UTM cache has been optimized in order to improve performance during intensive use of the UTM cache (e.g. in the case of extremely extensive service data).

- UTM lock algorithm

  The Compare&Swap functionality offered by the operating system is used throughout on open platforms for concurrent access to internal UTM administration data.

- UTM network access

  The network access on open platforms has been improved so that delays no longer occur when sending data to UTM partner applications, in particular in low-load situations.

Other changes

- Messages
  - The message area for system messages has been increased and now comprises the range from K001 to K399 (previously up to K249). As a result, the following message areas have been moved:
    - The message numbers for messages exclusively output by KDCUPD now occupy the range K800 to K899 instead of K250 to K322.

      Messages output by KDCUPD and by online import are considered to be system messages and remain unchanged.
    - The message numbers for KDCCSYSL and KDCPSYSL messages now occupy the range K600 to K649 instead of K550 to K599.
  - New message K235 if name resolution for a computer takes too long.
  - The default message destinations for messages K162 and K163 have been changed.

- KDCADMI
  - The fields *auto_connect* in *kc_lpap_str* and *auto_connect_number* in *kc_osi_lpap_str* have the property GPD instead of PD, changes to these fields always have a global effect throughout the application. Any administrative change to the properties "automatic establishment of connection" in the case of LPAP and "number of connections" for OSI-LPAP remains effective beyond the end of the application.
  - New field *max_btrace_lth* in *kc_diag_and_account_par_str* for the maximum length of the recorded data when the BCAM trace function is activated.
- In the case of platforms on which UTM can run in 64-bit mode, KDCUPD makes it possible to migrate from a 32-bit application environment to a 64-bit application environment. At present, UTM only supports 64-bit mode on Unix platforms.
- The Oracle User ID can also be entered in lowercase in the KDCDEF statements DATABASE and RMXA.
- The InstallAware installation procedure is used on Windows systems. As a result, openUTM is supplied in the form of MSI files for Windows systems.
- New sample program ADJTCLT (ADJust Tac-CLass Table)

  Using the C program unit ADJTCLT, users can control how the processes are distributed to the TAC classes in the light of the current total number of processes and the current number of asynchronous processes. To do this, the user creates a table containing the desired settings. The settings must be chosen in such a way that there is always at least one process free to perform other tasks, such as end-of-transaction processing for distributed transactions for example.

1.3.2 Load simulation with "Workload Capture & Replay"

Thanks to the new Workload Capture & Replay function, it is possible to record UTM application communications with UPIC clients and then replay these in combination with adjustable load profiles. In this way, it is possible to test the behavior of the UTM application at high loads under real-life conditions.

Workload Capture & Replay consists of the following components:

- *UPIC Capture*: Records communication with the UPIC client.

  The trace function BTRACE (BCAM trace), which is present on all the server platforms, is used to record a UPIC session.
- *UPIC Analyzer*: Used to analyze the recorded communication.
- *UPIC Replay*: Used to replay the recorded UPIC session with different load parameters (speed, number of clients).

UPIC Analyzer and *UPIC Replay* are only available on 64-bit Linux systems and are supplied with openUTM Client (UPIC).

openUTM for Unix and Windows systems also comes with the utility program *kdcsort*. You can use *kdcsort* to sort the communication recorded by BTRACE over time if the UTM application ran with more than one process during the recording period and multiple process-specific files have therefore been generated.

1.3.3 New client function

On Windows systems, UPIC Client is available in both a 32-bit and a 64-bit variant.

1.3.4 New and modified functions for openUTM WinAdmin

- WinAdmin supports all the new features of UTM V6.3 relating to the administration program interface. These include, for example, the new trace functions, the writing of KDCDEF statements to library elements on Inverse KDCDEF runs in BS2000 or the display of a user's used CPU time in milliseconds.
- Introduction of a lifetime for statistical values in order to limit the number of statistical values stored in the configuration database.

1.3.5 New functions for openUTM WebAdmin

Additional functions

WebAdmin now provides additional functions that go beyond the functionality available in the KDCADMI administration interface and which were previously available only in WinAdmin:

- Display of message queues (DADM functionality)
- Administration of statistics collectors and tabular display of the associated values (including the new "Lifetime for statistical values" function).
- Depiction of statistics in graphical form (graphs)
- Execution of threshold actions for statistics collectors

Support for new features in openUTM V6.3

WebAdmin supports all the new features of UTM V6.3 relating to the administration program interface. These include, for example, the new trace functions, the writing of KDCDEF statements to library elements on Inverse KDCDEF runs in BS2000 or the display of a user's used CPU time in milliseconds.

Integration in SE Server

WebAdmin can be installed as an add-on in the management unit (SE Manager) of an SE Server. It then provides much the same range of functions as when operated outside of the SE Manager.

1.4 Notational conventions

Metasyntax

The table below lists the metasyntax and notational conventions used throughout this manual:

Representation	Meaning	Example
UPPERCASE LETTERS	Uppercase letters denote constants (names of calls, statements, field names, commands and operands etc.) that are to be entered in this format.	`LOAD-MODE=STARTUP`
lowercase letters	In syntax diagrams and operand descriptions, lowercase letters are used to denote place-holders for the operand values.	`KDCFILE=filebase`
lowercase letters in italics	In running text, variables, the names of data structures and fields and keywords (e.g. C commands, Unix files names and Windows file names, etc.) are indicated by lowercase letters in italics.	*utmpath* is the UTM installation directory
`Typewriter font`	Typewriter font (Courier) is used in running text to identify commands, file names, messages and examples that must be entered in exactly this form or which always have exactly this name or form.	The call tpcall
{ } and |	Curly brackets contain alternative entries, of which you must choose one. The individual alternatives are separated within the curly brackets by pipe characters.	`STATUS={ ON | OFF }`
[ ]	Square brackets contain optional entries that can also be omitted.	`KDCFILE=( filebase [, { SINGLE| DOUBLE} ] )`
( )	Where a list of parameters can be specified for an operand, the individual parameters are to be listed in parentheses and separated by commas. If only one parameter is actually specified, you can omit the parentheses.	`KEYS=(key1,key2,...key`*n*`)`

Representation	Meaning	Example
Underscoring	Underscoring denotes the default value.	`CONNECT= { A/YES | NO }`
abbreviated form	The standard abbreviated form of statements, operands and operand values is emphasized in boldface type. The abbreviated form can be entered in place of the full designation.	**T**RANSPORT**-SEL**ECTOR=c‘C‘

. . .	An ellipsis indicates that a syntactical unit can be repeated.
	It can also be used to indicate sections of a program or syntax description etc.
		`Start KDCDEF`
		`:`
		`:`
		`OPTION DATA=statement_file`
		`:`
		`END`

Other symbols

Indicates notes that are of particular importance.

Indicates warnings.

utmpath

On Unix and Windows systems, designates the directory under which openUTM was installed.

2 Debugging and error diagnosis

This chapter tells you

– how to test a UTM application
– how to proceed with error diagnosis
– which traces you can use for diagnostics

2.1 Debugging UTM applications

For most test purposes, it is sufficient to start the application perfectly normally. Only if you want to debug a program unit that executes automatically when the application is started, e.g. program units for a Start-Exit or the MSGTAC program for start messages, do you have to start the application in a special way: In both cases, the following applies:

– A range of test tools are available to you for testing. In Unix systems, for example, *dbx*, *sdb*, *adb*, *xdb*, *debug*, *gdb*, Cobol test tools, and in Windows systems, the debugger that is integrated in Microsoft Visual Studio. It is possible to connect to the application's work processes with the test tool or to start multiple work processes that are monitored by one and the same test tool.

– You do not need any special generation, i.e. you can also use the tested application program in production operation without having to recompile and relink it.

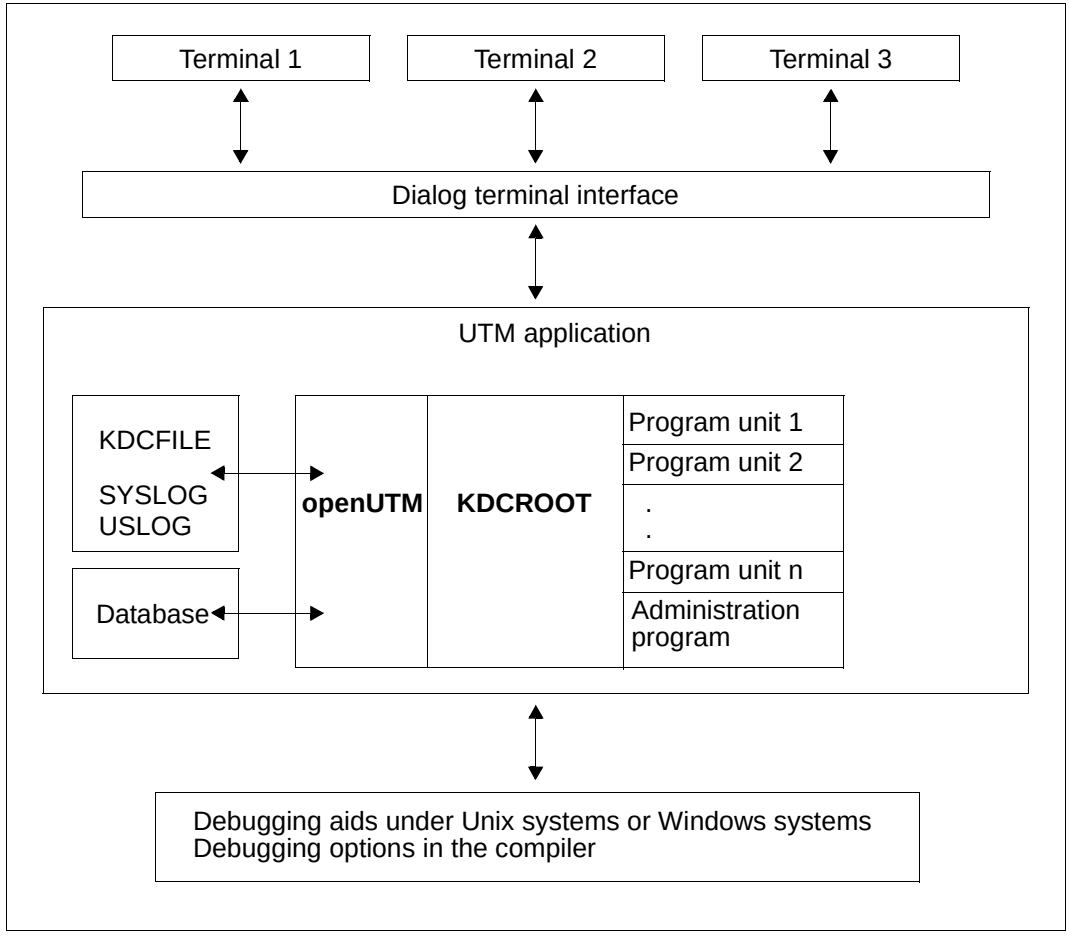
– The full functionality of openUTM under Unix systems and Windows systems can be tested even with distributed applications.

– As with a "real" production application, wait times are monitored by the timer process.

– You can also test output jobs to printers under Unix systems.

– In test operation, all the processes except for the utmwork processes are started in exactly the same way as for the production application. As a result, for example, full access is possible to and from the network.

The following diagram shows the structure of an interactive test.



In addition, access from the network is also possible.

2.1.1 Testing a UTM application under Unix sytems

In Unix systems you can use debugging tools such as *dbx*, *gdb*, *sdb*, COBOL debugger (Animator), etc. With the debugger, it is usually possible to attach to the individual processes of the running application. This means that the application can be started perfectly normally (with TEST STARTUP), see "Debugging a UTM application under Unix sytems" below.

A different procedure is used if you want to test programs that execute automatically when the application is started, e.g. Start Exit or MSGTAC, see section “Starting a UTM application under Unix systems with the debugger” on page 30.

Requirements for debugging Micro Focus COBOL programs with Animator

The following preparatory activities must be undertaken in order to use Animator to debug programs compiled with the Micro Focus COBOL compiler:

1. Set the compiler option *-a*
2. Prepare the environment to be started under Animator using the command:
   - `export COBSW=+A`

     Permits dynamic animation without explicitly calling Animator.
   - `export COBPATH=`*source-directory*

     Specifies the path under which Animator searches for the *.cbl* and *.int* files.

2.1.1.1 Debugging a UTM application under Unix sytems

To debug a program unit, proceed as follows:

1. Start the application normally in debug mode.
2. Start a debugger and attach to the work process(es).

   Examples

   - Solaris:

     `dbx` *pid* or
     `dbxtool` *pid*
   - Linux:

     `gdb` *pid* or
     `ddd` *pid*

   pid is the process ID of a work process.

2.1.1.2 Starting a UTM application under Unix systems with the debugger

If you want to debug a program unit that executes automatically when the application is started, e.g. program units for a Start-Exit or the MSGTAC program for start messages then you must start the application in a special way:

1. Start the *utmmain* process as a background process as with a UTM production application. However, you must also specify the TEST parameter.

   utmpath`/ex/utmmain` *filebase startparam-file* `TEST &`

   You must always specify the start parameter file as a fully qualified file name even if the default name is used.

   The main process *utmmain* then creates only the timer process and additional resources such as the pipe used to communicate with the work process. No work processes are created, however.

2. The main process requests you to start one or more work processes via message U244. Please note that if you are testing in dialog mode then the second work process is a UTM system process and that it may therefore be necessary to start three work processes if, for example, programs are to be tested with PGWT.

   You must start the first work process, *utmwork*, under the control of the debugging aids with the following arguments:

   utmwork version appliname filebase startpar pid id mode

   The entire statement must be entered **without** a carriage return, even if you need more than one line. The arguments have the following meanings:

version	openUTM version used, e.g. V06.3A00
appliname	Name of the application as generated in MAX APPLINAME.
filebase	Directory in which the application program is stored (fully qualified).
startpar	File in which the start parameters are defined (fully qualified).
pid	Process ID of the predecessor process for PENDER or a value of 0 for additional processes.
id	ID of the utmwork process. On the start of the first and an additional work process (Mode=Y or N), you must always specify the value 0. If Mode=E then the value from the corresponding U244 message must always be specified.
mode	Code for the start mode Y: First work process of the application N: Additional new work process E: Restarted after PENDER

Example for dbx

```
run V06.3A00 sample /home/utmbsp /home/utmbsp/startparameter 0 0 Y
```

You can take the parameters for starting the work processes from the associated message U244.

3. You should always restart work processes when the main process requests you to do so with message U244.

   Additional processes always require the value N as the last argument.

   utmwork version appliname filebase startpar pid id `N`

   Example for dbx

   ```
   run V06.3A00 sample /home/utmbsp /home/utmbsp/startparameter pid id N
   ```

   You cannot start more work processes than were specified in the start parameter file in TASKS= . The maximum number of work processes is defined in the generation in the MAX statement. If you attempt to start more work processes, start errors will occur.

4. After the UTM application has terminated, you must use the `kill` command to delete the main process, if not all the work processes have fully completed process termination processing.

   Example

   ```
   kill –9 utmmain
   ```

   where 1234 is the process ID of utmmain.

You will find a detailed description of the steps to be taken to start the application in the openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.

Behavior in the event of errors

If a work process creates a PEND ER dump, this work process is terminated. The main process does not then start a new work process. You must then restart the work process manually and specify E as the last parameter. The exact parameters can be taken from message U244. As *pid*, specify the process ID of the preceding work process that terminated with PEND ER.

utmwork version appliname filebase startpar pid id `E`

If a work process is terminated incorrectly, you should call the UTM tool KDCREM before the next application start. See also the corresponding section in the openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.

Terminating the application

As usual, you terminate the application using UTM administration functions or using the UTM tool KDCSHUT. If the application is not terminated normally, you should call the UTM tool KDCREM.

For further details, see the corresponding sections on KDCSHUT and KDCREM in openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.

2.1.2 Testing a UTM application under Windows systems

When testing applications under Windows systems, you use the debugger that is integrated in Microsoft Visual Studio 2010. With Visual Studio, it is usually possible to attach to the individual processes of the running application. This means that the application can be started perfectly normally (with TEST STARTUP), see "Debugging a UTM application under Windows systems".

A different procedure is used if you want to test programs that execute automatically when the application is started, e.g. Start-Exit or MSGTAC, see "Starting a UTM application under Windows systems with the debugger".

The following variant refers to the English variant of Visual Studio 2010.

2.1.2.1 Debugging a UTM application under Windows systems

To debug a program unit, proceed as follows:

1. Start the application normally in debug mode.
2. Open the application project in Microsoft Visual Studio.
3. In the menu bar under *Debug*, choose *Attach to Process*, select a *utmwork.exe* process and click *Attach*.

   Repeat this operation for each work process in your UTM application.

2.1.2.2 Starting a UTM application under Windows systems with the debugger

If you want to debug a program unit that executes automatically when the application is started, e.g. program units for a Start Exit or the MSGTAC program for start messages then you must start the application in a special way:

1. You start the main process just like for a UTM production application. However, you must also specify the TEST parameter.

To do this, open a Command Prompt window, change directories to the application directory and enter the following:

```
utmmain .  startparam-file  TEST 1>utmp-out.txt <nul
```

The PATH variable must be set accordingly when this is done. The output to *stdout* is redirected to a file here. nul corresponds to `/dev/null` in Unix systems.

The main process *utmmain* then creates only the timer process and additional resources such as the pipe used to communicate with the work process. No work processes are create, however. The main process requests with message U244 that you start a work process. U244 can appear as follows, for example:

```
U244 utmmain: Please start ./utmwork with arguments:
V06.3A00 sample01 . ./startp.std 0 0 Y
```

`You will need the second line later for the debugger. The argument 0` means that this process will be restarted, i.e. it does not replace any previously terminated work process. In addition, `Y` means that this is the first work process.

You may not close the window yet, otherwise the main process is terminated!

Please note that if you are testing in dialog mode then the second work process is a UTM system process and that it may therefore be necessary to start three work processes if, for example, programs are to be tested with PGWT.

2. You must start the first work process utmwork under the control of the debugger. To do this, proceed as follows:

   a) Open the application project in Microsoft Visual Studioand choose the menu item *Project - utmwork properties*.

   b) In the navigation area, click *Debugging* under *Configuration Properties*, enter the name of the *filebase* in the *work directory field* and specify the values from the U244 message in the *Command Arguments* field.
   These values have the following structure:

   utmversion appliname filebase startparameter-file `0 0 Y`

   c) Click OK to start the first work process.

   d) Click the *Start Debug* command in the *Debug* menu to start the first work process with the debugger.

3. You can start additional work processes after that. You should always restart work processes when the main process requests you to do so with message U244. Additional processes always require the value N as the last argument.

   Please note that if you are testing in dialog mode then the second work process is a UTM system process and that it may therefore be necessary to start three work processes if, for example, programs are to be tested with PGWT.

You can take the parameters for starting the work processes from the associated message U244.

version appliname filebase startparam-file pid id N

You cannot start more work processes than was specified in the start parameter file for TASKS= . The maximum number of work processes is defined in the generation in the MAX statement. If you try to start more work processes, then this will result in start errors.

4. If not all work processes have fully completed the scheduled end-of-process handling, you must explicitly terminate the main process after the work processes have been terminated by closing the associated window or terminating the process *utmmain.exe* using the Task Manager.

You must terminate the `utmmain` process when terminating the application by closing the window, for example.

You will find a detailed description of the steps to be taken to start the application in the openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.

Behavior in the event of errors

If a work process produces a PEND ER dump, then this work process is terminated and you are requested by message U244 to start a new work process.
To do this, repeat step 2 but replace the following values:

– replace the values 0 for *pid* and *id* with the values specified in message U244
– replace the value Y with E

2.1.3 Outputting messages when starting a process

To make it easier to diagnose errors that occur when a UTM application or follow-up process starts, all K messages from openUTM that appear in the start phase are always output to *stderr* and *stdout*.

2.1.4 Working at the terminal in test mode

Several terminals can sign on to the application. The dialog terminal process is started as in live operation by calling the `utmdtp` program.

`utmdtp` is located in the directory *utmpath*`/ex` and is started as follows:

`utmdtp [␣-S[`*username*`]][␣-A`*applname*`][␣-D][␣-P`*pterm-name*`]`

The entries in brackets represent options which may be specified but are not mandatory. These options have the following meanings:

-S[*username*]

You use this option to control the interactive authorization check (access check) which UTM performs after successful setup of a connection to the UTM application.

With this switch you control if a UTM user ID must be explicitly passed for the authorization check in the dialog or if a Unix or Windows ID is to be implicitly passed as a UTM user ID.
openUTM performs the authorization check (system access control) after the connection has been successfully established to the UTM application.
An authorization check is always performed regardless of whether or not the -S option was specified when the application was generated with user IDs (USER statement), but the LTERM partner was not assigned a connection user ID in the generation (LTERM USER=...).

If the terminal user starts the dialog terminal process **with** the -S option, then you must pass a UTM user ID to openUTM for the authorization check. You can specify the user ID with -S*username* (username = generated UTM user ID) directly when the dialog terminal process is started. If you only specify -S, then openUTM requests the UTM user ID in a dialog after the connection has been established.
If you specify a UTM user ID for which a password or an ID card check is generated, then openUTM queries the necessary data in the dialog.

If you start the dialog terminal process on the terminal **without** the -S option, then the dialog terminal process passes the Unix or Windows user ID implicitly as the UTM user ID for the authorization check. A password is not passed to openUTM. A password for the user ID may be assigned in the UTM application; the terminal user is then requested to enter this password just like in the case where the user ID is specified explicitly.

If the check of the implicitly passed user ID fails, then an explicit authorization dialog is executed just as if the -S option had been specified.

-A*applname*

With this option you specify on the terminal which application you wish to be connected to. The name of the application must be specified for *applname*.
If "-A*applname*" is **not** specified when the dialog terminal process is started, then the dialog terminal process asks for the application name in a dialog.

-D

With this option you determine how the dialog terminal process reacts to the DEL key (Unix systems) or the CTRL + C key combination (Windows systems).

If you specify the -D option, then the DEL key or the CTRL + C key combination is ignored by the dialog terminal process.

If you do **not** specify the -D option, then pressing the DEL key or the CTRL + C key combination signs you off from the application and terminates the dialog terminal process.

-P*pterm-name*

You sign on to the UTM application under the PTERM name *pterm-name*.

Unix systems: By default, openUTM uses the last part of the output from the tty command (the term following the last slash) as the *pterm-name*.
This corresponds to the output of the basename `tty` command. During the KDCDEF generation, an associated PTERM statement should be specified (under this *pterm-name*) for all local terminals and pseudoterminals.

In Unix systems, it can happen that the default assignment of the *pterm-name* by openUTM is not unique. Depending on the type of the networks to which the system is connected, two or more pseudoterminals can exist which do not differ in the last term of the tty (following the last slash). Only one of these terminals is then able to set up the connection to the application using this pterm name. The connection request of the second terminal is rejected by openUTM.

Windows systems: By default, openUTM assembles the *pterm-name* with the format `tty`*nnnnn*, where *nnnnn* is the PID of the `utmdtp` process in decimal format.

Under Unix systems you can also enter the `utmdtp` program directly as the start program in your own `.profile` or in the `/etc/passwd` file. The dialog terminal process is then started immediately after you have successfully logged on to Unix systems.

Under Windows systems you can also enter the `utmdtp` program in the `Startup group`. The program is then started automatically.

2.2 Error diagnosis

This section tells you

– which error codes the program interface supplies
– how UTM signals errors by means of messages
– which documentation has to be produced in the event of errors
– which traces you can use for diagnostic purposes

For a description of an openUTM dump and how to evaluate it, please refer to chapter “The UTM dump” on page 57ff.

2.2.1 Return codes at the program interface

Following each KDCS call (except for PEND), openUTM returns the following error codes and IDs in the return field of the communication area:

– the KDCS error code,

– the internal error code.

KDCS error code in the KCRCCC field

Please note the following points:

– If two or more errors occur at the same time, the ones in the highest category are displayed. The lowest return code is not always specified within a category (different to DIN 66 265).

– The precise meanings of the KDCS error codes for each KDCS call are described in the openUTM manual „Programming Applications with KDCS”. A summary of all KCRCCC error codes can be found on page 413ff.

Internal UTM error code in KCRCDC

The internal error code contains a more accurate specification of the error than the KDCS error code in KCRCCC. Usually, this error code is set with the KDCS error codes 40Z or 70Z (system or generation error). The precise meaning can be found on page 413ff.

The internal UTM error code is not part of the standardized KDCS interface.

2.2.2 openUTM messages in response to program errors

Program errors are errors in the programming of the KDCS interface (see the KDCS return code). openUTM then generates messages that are output by default to STDOUT, STDERR and generally also to SYSLOG (see section “Destinations of UTM messages” on page 393).

Abnormal termination of a service

If a dialog service is terminated abnormally, openUTM issues message K017. If an asynchronous service is terminated abnormally, it issues message K055.

These messages contain return codes that indicate the cause of the error:

- `K017 Service &TCVG terminated by UTM (&RCCC/&RCDC &RCF2A) – input please`

  The entries in the message have the following meanings:

  ```
  &TCVG  :   TAC with which the service was started
  &RCCC  :   KDCS return code in the KCRCCC field
  &RCDC  :   internal return code in the KCRCDC field
  &RCF2A :   (always 0)
  ```

- `K055 Asynchronous service &ATAC1 terminated by UTM; KCRCCC= &RCCC ; KCRCDC= &RCDC ; USER= &USER ; LTERM= &LTRM`

  The entries in the message have the following meanings:

  ```
  &ATAC1 :   TAC with which the asynchronous service was started
  &RCCC  :   KDCS return code in the KCRCCC field
  &RCDC  :   internal return code in the KCRCDC field
  &USER  :   user ID that created the asynchronous service
  &LTRM :    LTERM partner that generated the asynchronous service
  ```

Errors in the INPUT exit

- When there are errors in the INPUT exit, openUTM generally issues message K098 to the terminal. You will find an explanation of the error codes contained in K098 on page 251.

2.2.3 Diagnostic dump with defined messages/events

You can cause a diagnostic dump, known as a message dump, to be generated when a certain event occurs. The dump ID depends on the event type.

A message dump is only created by the task in which the event occurs. The UTM application is not terminated. In order to take such a message dump, you have to activate test mode for the application and define the event at which the message dump is to be taken. Both of these things can be done by means of start parameters or using the administration functions.

You can specify the following events:

– the output of a specific K message
– the occurrence of a specific KDCS return code (CC or DC) in a program unit run
– the occurrence of a specific SIGN status when a user signs on

Activating test mode

- using the relevant start parameter

  `.UTM TESTMODE = ON`

- using the relevant administration command

  `KDCDIAG TESTMODE = ON`

- via the administration program interface (e.g. via WinAdmin or WebAdmin)

  In the object type KC_DIAG_AND_ACCOUNT, specify:

  `testmode='Y'` (data structure *kc_diag_and_account_par_str*)

Activating and resetting the message dump function

You can activate and reset the message dump function by means of a start parameter or using the administration functions. The function is deactivated by default at application startup.

Activating the message dump function

- using the relevant start parameter

  `.UTM START DUMP-MESSAGE = (`*event-type,event)*

  This causes a message dump to be created as soon as the event occurs.

  You can only specify one event when you activate the function using a start parameter. It is not possible to specify inserts for a message.

- using the relevant administration command

  KDCDIAG DUMP–MESSAGE = (*event-type,event)*

  In the same way, you can use the parameters DUMP–MESSAGEx (where x = 1, 2, 3) to specify up to three different events for generating a message dump. In this case, DUMP–MESSAGE is synonymous with DUMP–MESSAGE1.

  You can specify up to three inserts as additional constraints for the event "output of a specific K message" (parameter INSERTx where x = 1,2,3). For further details, see KDCDIAG in the openUTM manual "Administering Applications".

- via the administration program interface (e.g. via WinAdmin or WebAdmin)

  In object type KC_DIAG_AND_ACCOUNT, specify the event and the event type in the *event* and *event_type* fields (data structure *kc_diag_and_account_par_str* with the corresponding substructures). You can also specify up to three inserts as a condition. At the program interface, you can specify one event per call. For further details, see the description of KC_DIAG_AND_ACCOUNT in the openUTM manual "Administering Applications".

event-type specifies the event type and *event* specifies a particular event for which the message dump is to be generated. You can specify the following events:

- Output of a specific K message (*event-type* = MSG)
  Specify the UTM message number *Knnn* as the *event*.
  A dump is generated each time the message number occurs until such time as you reset the message number. Only one dump is generated for the message numbers K043, K061, K062. The message number is then automatically reset.
- Occurrence of a particular compatible KDCS return code (*event-type* = RCCC)
  Specify the number of the compatible KDCS return code (KCRCCC) as the *event* e.g. 14Z.
  If the return code occurs during a KDCS call, only one dump is generated and the parameter DUMP–MESSAGE[x] is reset to *NONE.
- Occurrence of a particular incompatible KDCS return code (*event-type* = RCDC)
  Specify an incompatible KDCS return code (KCRCDC) as the *event*, e.g. KD10.
  If the return code occurs during a KDCS call, only one dump is generated and the parameter DUMP–MESSAGE[x] is reset to *NONE.

  Note
  In the case of all KDCS return codes ≥ 70Z and the associated incompatible KDCS return codes, when no PENDER dump is written (e.g. 70Z/K316), no message dump is created either.

- Occurrence of a particular sign-on status (*event-type* = SIGN)
  Specify the SIGNON status code in the form *xyy* as the *event* (e.g. U05):
  – *x* corresponds to the value in KCRSIGN1, with U, I, A or R being possible values.
  – *yy* corresponds to the value in KCRSIGN2
  If the status code occurs during a SIGN call, only one dump is generated and the parameter DUMP-MESSAGE[x] is reset to *NONE. This happens regardless of whether or not a sign-on service is generated in the application.

Deactivating the message dump function

- using the relevant start parameter

  ```
  .UTM START DUMP-MESSAGE = *NONE
  ```

- using the relevant administration command

  ```
  KDCDIAG DUMP-MESSAGE = *NONE
  ```

  The relevant parameter must be set to *NONE for events activated using the parameter DUMP-MESSAGEx.

- via the administration program interface (e.g. via WinAdmin or WebAdmin)

  In the object type KC_DIAG_AND_ACCOUNT, reset all the events by specifying *event_type=NONE* for each event.

2.2.4 Producing error documentation

The following information is required for error diagnosis:

- detailed description of the error situation
- information about current versions of software involved
- precise specification of the computer type

The error documentation provided should be as complete as possible. The following may serve as error documentation:

- UTM dumps from all work processes along with associated "gcores" under Unix systems or "mini dumps" under Windows systems. The "mini dumps" are by default located in the DUMP directory and have the extension DMP. These files should be provided as binary files, i.e. not in edited form
- the SYSLOG file(s) (see page 155)
- the *stdout* and *stderr* logs from all openUTM processes
- the *stdout*, *stdin* and *stderr* logs of the KDCDEF generation and the start procedure including the start parameters
- all linkage editor listings, compiling listings and compilation procedures
- In the case of errors which are associated with openUTM network connection, the following additional documentation can be produced:
  - messages from the openUTM network processes on *stdout* and *stderr*
  - CMX traces
  - OSS traces
  - dynamic openUTM trace
  - Mapped host name file
  - CHECKTNS file generated by means of the KDCDEF statement OPTION; this file is contained in the *filebase* directory and is called `def_tns`

  For a description of how to generate the traces listed, refer to page 44 (CMX trace), page 51 (OSS trace) and page 44 (dynamic openUTM trace).
- In the case of errors in UTM cluster applications, then the following documents are also required:
  - All files that are global to the cluster, log files (and DUMPs) for all node applications
  - the cluster configuration file and, in the case of administrative problems, all the administration journal files with the suffix JKAA, JRN1, JRN2.
  - in the case of problems caused by interactions between the node applications, the log files of all the other node applications
  - The start procedure and the procedures specified as EMERGENCY-CMD and FAILURE-CMD during generation

  - in the case of user problems (e.g. sign-on problems), also the cluster user file (i.e. the file with the suffix UTM-C.USER)

- (Unix systems only): the core files with the associated phases (utmwork) and shared objects. The shared objects can be determined using the command `ldd␣utmwork`.

You should attempt to reproduce the errors by using static libraries.

Procedure in the event of errors

- If the service/application is aborted, you should proceed as follows:

  1. Evaluate the UTM dump using the KDCDUMP tool - see page 60.
  2. Reproduce the error using suitable debuggers, for example: dbx, sdb, adb, xdb, gdb, debug under Unix systems or the debugger integrated in Microsoft Visual Studio under Windows systems.
  3. Determine the call hierarchy during core write with the aid of a debugger.
     (If you use the sample application, then you can display the call hierarchy under Unix systems using the `p/stack` shell script.)

- Abnormal termination with signals

  If a PEND ER dump occurred with 70Z/XT10 or XT11 or an application aborted with SIG010/SIG011 (signal SIGBUS/SIGSEGV), the openUTM signal handling facility should be deactivated with the start parameter START STXIT=OFF.

  The start parameter STXIT=OFF causes the system to automatically start the debugger (Windows systems) or generate a core dump immediately under Unix systems (without openUTM causing any delay) and terminate the process without a UTM dump after a faulty command is issued.

  Before the next new start you must, in any case, call KDCREM since openUTM does not perform any end-of-process handling in conjunction with STXIT=OFF.

- After a start error such as error number 32 or 40, the KDCREM tool must be called before restarting.

2.2.5 Traces

You can utilize the following traces and tools for diagnostic purposes for openUTM (in addition to the traces in the UTM dump):

– dynamic openUTM trace (using the UTMTRAC environment variable)
– tracing COBOL and C/C++ program unit runs
– BCAM trace in openUTM
– KTA trace in file
– OSS trace for OSI-TP
– ADMI trace, i.e. trace of the administration program interface (KDCADMI)
– creating a `core` when an application crashes (Unix systems only)
– KDCIPC tool
– KDCKAA tool

2.2.5.1 Dynamic openUTM trace via an environment variable

A dynamic trace can be activated by setting the UTMTRAC environment variable.

The environment variable is evaluated when the process is started.
Default: the trace is disabled.

Syntax

Unix systems:

```
UTMTRAC=prog1#trace1,trace2 [, ...][.file1][:prog2#trace1, ...][: ...]
export UTMTRAC
```

Windows systems

```
SET UTMTRAC=prog1#trace1,trace2 [, ...][.file1][:prog2#trace1, ...][: ...]
```

Meaning of the parameters:

prog *n* — Either `all`, when all openUTM programs are to be traced, or the openUTM program that is to be traced. The following programs are allowed:
`utmmain`, `utmwork`, `utmtimer`, `utmnet`, `utmdtp`, ...
`kdcdef`, `kdcrem`, ..., `kdcupd` and read and write accesses from `kdcupd`, `kdcrV...`, `kdcrWV62A`.

trace *n* — Either `all`, when all trace units are to be activated, or a number between 1 and UTM_MAX_TRACE_UNITS, inclusive, to activate the specified trace unit.

file *n* — Optional specification of the output file.
file *n* can contain %d (Unix systems) or %%d (Windows systems): this placeholder is replaced by the current *pid*.
If file *n* is not specified, the trace sends its output to *stderr*.

You can specify a list of programs in UTMTRAC with different specifications for the individual programs, trace units and output files, or you can enable the trace for all programs and trace units. You will find the definitions of the trace units in the `xidyntrc.h` header file supplied in the *utmpath*`/include` directory.

Example 1

```
UTMTRAC=all#all (Unix systems)
export UTMTRAC

SET UTMTRAC=all#all (Windows sysems)
```

All trace units are enabled for all processes. The trace output is sent to *stderr*.

Example 2

```
UTMTRAC=utmwork#1.wrkp.%d (Unix systems)
export UTMTRAC

SET UTMTRAC=utmwork#1.wrkp.%dd (Windows systems)
```

Trace unit 1 (KCXPIPE) is activated for utmwork. The trace is output to `wrkp.`*pid* (*pid*= current process ID of the `utmwork` process).

2.2.5.2 Tracing program unit calls

You activate the trace for COBOL and C/C++ program unit calls by setting the environment variable KDCS_C_DEBUG.

The environment variable is evaluated each time a work process is started. If the C/C++ programs have been programmed using the KDCS macros from the header file*kcmac .h* then all the KDCS calls from these C/C++ program units are also logged. By default, the traces are output to *stdout*. If the trace is to be output to another file, you have to set the KDCS_DEBUG_FP constant to the value of a global FILE variable before including kcmac.h. You supply this variable in the start exit.

Default: the calls are not traced.

2.2.5.3 BCAM trace in openUTM

The BCAM trace function of openUTM allows logging of all connection-related activities within a openUTM application.

Contents of the BCAM trace

The following types of trace record are written:

– Parameter block : The BCAM parameter blocks of the calls REQCON, ACCON, REJCON, DISCON and also the BCAM parameter blocks of the calls RECLET and SENDLET which supplied a return code.

– Announcement: All connection-related announcements

– Connection letter

– Communication via TS applications of socket type: parameters blocks for the `connection request`, `connection response` and `disconnect request` functions of the socket interface.

– Message: All input/output messages

– CMX record: The parameters which are used in the work process in the case of the CMX calls `t_conrq`, `t_conrs`, `t_event`, `t_datain`, `t_datarq` and `t_disrq`.

Every trace record contains the following entries in the sequence given:

1. Time stamp
2. BCAMAPPL/ACCESS-POINT name
3. PTERM/CON/TSEL name in the case of OSI-CON
4. PROCESSOR name
5. LTERM/LPAP name
6. USER name
7. Type of the trace record (see above):
   – announcement
   – parameter block
   – connection letter
   – TCP/IP record
   – message (input/output message)
   – CMX record
8. Up to 32767 bytes of data (depending on the record type and the value of the *length* operand in the start parameter BTRACE).

Activating/deactivating the BCAM trace

The BCAM trace can be activated and deactivated by means of a start parameter or an administration command. By default, the function is inactive on starting the application.

- Activation/deactivation by start parameter

  ```
  .UTM START,BTRACE = { ON/OFF                  }
                      { ( ON / OFF, length ) }
  ```

  In this way, the trace function is activated (ON) or left inactive (OFF) on starting the application.

  You can also specify the maximum length of the data to be recorded.
  Minimum: 32
  Maximum: 32767
  Default value: 256

  This maximum length can only be defined via start parameters.

  If you use the BCAM trace for the UPIC Capture function (see openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”) then it is advisable to use the maximum value.

- Activation/deactivation by administration command

  KDCDIAG BTRACE=ON/OFF[,LTERM=*lterm-name*/LPAP=*lpap-name* / *USER=user-name*]

  In this way, the trace function is activated/deactivated while the application is running.
  If an LTERM or LPAP name is specified, only the events associated with this connection are recorded.
  If an USER name is specified, only the events associated with this user id are recorded.

- Activation/deactivation via administration program interface (e.g. via WinAdmin or WebAdmin)

  In the object type KC_DIAG_AND_ACCOUNT, specify:

  bcam_trace=’Y’ or ’N’ (data structure *kc_diag_and_account_par_str*)

Each work process generates its own trace file in the form *filebase*/KDCBTRC.*pid*.
After activation of the BCAM trace, the trace file is created or opened in "append" mode, as the case may be.

After deactivation of the BCAM trace, the trace files are closed and can subsequently be evaluated.

The trace function is terminated if errors occur while accessing the trace files.

Evaluating the BCAM trace

You can use the tool KDCBTRC to prepare trace files. The tool KDCBTRC can only be used to evaluate trace files of the same UTM version.

Prior to evaluation, the trace files for the different work processes can be sorted in chronological order and entered in a file using the tool *kdcsort*:

utmpath`/ex/kdcsort` *btrcace_out btrcace-1 btrcace-2 ... btrcace-n* (Unix systems)

utmpath`\ex\kdcsort` *btrcace_out btrcace-1 btrcace-2 ... btrcace-n* (Windows systems)

For details, see openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.

The tool KDCBTRC is called as follows:

utmpath`/ex/kdcbtrc␣` *btrcace-file*␣[*argument-1 .... argument-n*] (Unix systems)

utmpath`\ex\kdcbtrc␣` *btrcace-file*␣[*argument-1 .... argument-n*] (Windows systems)

The optional arguments *argument-1, ... , argument-n* serve to control editing. If no arguments are specified, then the entire trace file will be edited. The result of the evaluation is written to *stdout*.

The possible arguments and their meanings are described below.

Arguments for the editing program

LT=ltn
: An LTERM name or LPAP name can be specified with this operand. As a result, only those trace records which contain the LTERM/LPAP name are edited.
Default: all trace records are edited.

PT=ptn
: A PTERM name, CON name or TSEL name (OSI-CON) name can be specified with this operand. As a result, only those trace records which contain the PTERM/CON/TSEL name (OSI-CON) name are edited. CMX=Y must be specified for the TSEL name (see below).
Default: all trace records are edited.

BC=bcn
: A BCAMAPPL name or ACCESS-POINT name can be specified with this operand. As a result, only those trace records which contain the BCAMAPPL/ACCESS-POINT name are edited. CMX=Y must be specified for the ACCESS-POINT name (see below).
Default: all trace records are edited.

PR=prn
: A processor name can be specified with this operand. As a result, only those trace records which contain the processor name are edited.
Default: all trace records are edited.

US=usr
: This operand allows you to specify a user ID (USER). This means that only those trace records are edited that contain the name of this user ID.
Default: all trace records are edited.

AN=Y/N
: When AN=Y is specified, those trace records which contain announcements are edited. If AN=N, editing is suppressed.
Default: AN=N

PB=Y/N
: When PB=Y is specified, those trace records which contain BCAM parameter blocks are edited. If PB=N, editing is suppressed.
Default: PB=N

CL=Y/N
: When CL=Y is specified, those trace records which contain connection letters are edited. If CL=N, editing is suppressed.
Default: CL=N

IN=Y/N
: When IN=Y is specified, those trace records which contain input messages are specified. If IN=N, editing is suppressed.
Default: IN=Y

OUT=Y/N
: When OUT=Y is specified, those trace records which contain output messages are specified. If OUT=N, editing is suppressed.
Default: OUT=Y

CMX=Y/N
: When CMX=Y is specified, those trace records are edited which contain traces of CMX functions. When CMX=N, editing is suppressed.
Default: CMX=N

SOCKET=Y/N
: When SOCKET=Y is specified, those trace records are edited which contain traces of functions for communication using the TCP/IP protocol. When SOCKET=N is specified, editing is suppressed.
Default: SOCKET=Y

2.2.5.4 KTA trace in file

In certain special error situations the trace entries to be found in the UTM dump will not be sufficient to ascertain the cause of an error. The KTA trace is available for such situations; this writes the KTA trace entries to a file whenever the trace area overflows.

The name of this file is made up of the base name *filebase* and the PID of the work process in question, i.e. the following file is created for each work process:

filebase.KTATRC.*pid* (*pid* max. 4 characters)

Activating/deactivating the KTA trace

The administration command:

```
KDCDIAG TESTMODE = FILE
```

creates and opens the UTM trace file. However, the trace entries are only written when the KTA trace area is full or when this work process is terminated normally.

This function is deactivated with the administration command

```
KDCDIAG TESTMODE = OFF
```

When this is done, the content of the KTA trace area is written to the file, which is then closed.

On starting the application the function is inactive. If the function was active on terminating the application, the remainder of the traces are written to file.

Evaluating the KTA trace

Unix systems: the `hd` shell command can be used to view the trace file and/or put it in a printable form.

Windows systems: you can evaluate the trace file using a program suitable for use with binary files.

2.2.5.5 OSS trace

The OSS trace comprises several types of trace records. Logging can be activated either for each type individually or for all types, i.e.

filebase.OSST.*nr.pid*

where *filebase* is the base name of the KDCFILE in the MAX statement, *no* is the serial number of the trace (0 - 9), and *pid* is the number of the work process.
openUTM monitors the size of the OSS trace file and switches to the next file when necessary. The numbers 0 though 9 are assigned cyclically, and trace file 0 is opened and overwritten once trace file 9 has been closed.

Contents of the OSS trace

The OSS trace comprises a number of different types of trace records. Logging can be activated for each type individually or for all types.

The following types of trace records exist:

SPI	The XAP-TP System Programming Interface is logged.
INT	Internal execution in the XAP-TP module is logged.
OSS	The OSS calls are logged.
SERV	The OSS-internal trace records of the type O_TR_SERV are logged.
PROT	The OSS-internal trace records of the type O_TR_PROT are logged.

Activating/deactivating the OSS trace

This trace can be activated and deactivated either by using a start parameter, i.e. on starting the application, or by means of an administration command.

- Activation/deactivation by start parameter:

```
[.UTM]   START    [,OTRACE=ON ]
                  [,OTRACE=( type1 [, type2 ], ... )
                  [,OTRACE=OFF]
```

  When OTRACE=ON is set, the OSS trace is activated on starting the application and all types of trace records are logged.

  With OTRACE=(type1 [, type2], ...), the OSS trace is activated for the specified types after the application is started (the types can be specified in any sequence).

  With OTRACE=OFF (default) the trace is inactive on starting the application.

- Activation/deactivation by administration command:

  ```
  KDCDIAG  OT[RACE]=ON | ( type1 [, type2 ], ... ) | OFF
  ```

  OTRACE = ON activates the OSS trace for all types of trace records, OTRACE=(type1 [, type2], ...) activates the OSS trace for all the specified types, OTRACE=OFF deactivates the OSS trace.

At the OSS interface, the parameters are set to the following values when the trace is activated:

```
o_trmode       =   O_TR_NEW
o_trsel        =   O_TR_USER + O_TR_PROT
o_traopt       =   0
o_mludata      =   32767
o_mldt         =   0
o_mltd         =   0
```

These parameters are described in detail in the manual "OSS(UNIX)".

Evaluating the OSS trace

The program STEP is shipped with openUTM for evaluating the OSS trace files that are produced. The program is available under *utmpath*`/oss`. Use of this evaluation program is described in detail in the manual "OSS(UNIX)".

2.2.5.6 ADMI trace

The ADMI trace logs all calls of the KDCADMI program interface.

The following data is written prior to the call:

– Content of the data area

The following data is written after the call:

– Addresses of the parameter area, identification area, selection area and data area

– Contents of the fields in the parameter area, e.g. operation code (opcode), object type, length of the data area, return code

– Content of the data area

The ADMI trace can be enabled via the start parameter ADMI-TRACE and enabled or disabled via WinAdmin, WebAdmin or the administration program interface (KDCADMI).

By default, the trace is written to the following file in the application directory (*filebase*):

`KDC.TRC.ADMI`*.appliname.hostname.pid*

Where *appliname* is the name of the UTM application (MAX APPLINAME), *hostname* is the name of the computer on which the application is running and *pid* is the number of the work process.

2.2.5.7 Creating a core when an application crashes

A core is created under Unix systems when an application crashes by setting the UTM_ABORT_WITH_EXCEPTION environment variable.
The debugger is activated when this happens under Windows systems.

It is recommended that you only set this environment variable when the start parameter STXIT contains the value OFF.

Default: the environment variable is not set.

CAUTION!
If you have specified STXIT=OFF, then no transactions will be rolled back when an error occurs.
A lock will remain until the next warm start of the application, and then the transaction is rolled back.

2.2.5.8 Suppressing gcore dumps

By setting the UTM_CORE_DUMP environment variable you can suppress the creation of core dumps (Unix systems) or mini dumps (Windows systems) in the processes of a UTM application.

If the environment variable set contains the value “NO”, no core dump/mini dump is created in the work process for a UTM dump or in the external processes if a process is terminated abnormally. If you do not set the environment variable or it does not contain the value “NO”, a core dump/mini dump is created in the above situations.

2.2.5.9 KDCIPC tool

If problems are experienced with internal openUTM inter-process communication, a dump of the "IPC shared memories" can be prepared while the application is running and output to *stdout*. No additional editing program is required in order to do this.

It is also possible, while the application is running, to activate or deactivate the IPC trace; this is done regardless of the value of the start parameter TESTMODE=.

The tool writes all information to *stdout*.

Calling KDCIPC

KDCIPC is called as follows:

utmpath`/ex/kdcipc␣`*filebase*`␣[T] [D] [,tron/troff]` (Unix systems)

utmpath`\ex\kdcipc␣`*filebase*`␣[T] [D] [,tron/troff]` (Windows systems)

filebase	Base name of the KDCFILE in the MAX statement.
T	This operand causes the buffer of the IPC trace area to be output in chronological order to *stdout*. In other words, the last record in the list is also the most recently generated. Default: no output of the IPC trace area to *stdout*.
D	This operand causes the entire IPC shared memory to be output in edited form to *stdout*. Default: no output of entire IPC shared memory.
tron/troff	"tron" activates the IPC trace mode; "troff" deactivates it. Default: the IPC trace mode is activated/deactivated according to the start parameter TESTMODE.

2.2.5.10 KDCKAA tool

As an aid to diagnosis, information from the KAA can be output in edited form to *stdout* while the application is running. No additional editing program is required in order to do this.

Calling KDCKAA

KDCKAA is called as follows:

utmpath`/ex/kdckaa␣`*filebase* (Unix systems)

utmpath`\ex\kdckaa␣`*filebase* (Windows systems)

where *filebase* is the base name of the KDCFILE in the MAX statement.

3 The UTM dump

In the event of serious errors, openUTM generates a UTM dump of all relevant data. Possible reasons for a UTM dump are as follows:

– a program unit has issued a PEND ER call

– a KDCS return code KCRCCC ≥ 70Z occurred in an application program because of a severe error in a KDCS call or an error during interoperation with a database.

– a diagnostic dump was requested (e.g. KDCDIAG)

– an error in the UTM system code or in other software components caused the application run to crash

In the last instance, the UTM dump contains data from all work processes of the application and in all other cases, only data for the work process concerned. All dumps are written in compressed form.

You can display the version of the operating system on your computer under Unix systems with the "`what <file> | grep UNAME`" command.

UTM dump in tabular form

The length of the individual lines of a UTM dump (80 or 132 characters) depends on the EDIT operand of the KDCDUMP utility (see also page 60) so you may have to specify the -pb3 option in the lpr command in order to print out the dump file. With the exception of the PROGRAM table, each line of an edited dump has the form (for 32-bit platforms):

I	AAAAAAAA	XXXXXXXXXXXXXXX	XXXXXXXXXXXXXXX	CCCCCCC	CCCCCCC

Where

I is the 4-digit index or sequence number; not all lines contain one.
A is the 12-character address.
X is the data in hexadecimal form, 64 characters in length (corresponds to 32 bytes); not included in the PROGRAM table.
C is the data in plain text, if it consists of printable characters; 32 characters in length (except in the PROGRAM table).

3.1 The files of the UTM dump

The dump files are located in the directory *filebase*/DUMP or *filebase*\dump, where *filebase* is the base name of the KDCFILE in the MAX statement.

If the dump is caused by an error in the UTM system code (system dump), a file generation group (FGG) is created (see below).

Otherwise, the name of the dump file has the following format:

RRRRRR.PPPP.NN

where

RRRRRR	ID identifying the cause of the memory dump (6 characters).
PPPP	are the last four digits of the process ID number (PID) under which the dump was produced.
NN	is the dump number.

In a system dump, a file generation group (FGG) is set up in the directory *RRRRRR.PPPP.NN*. *PPPP* are the last four digits of the process number of the process which discovered the system error. If two or more work processes are running in the application, one dump file is set up for each process. The FGG has the form:

/INFO	(FGG administration file)
/0001	(first dump file)
/0002	(second dump file, etc.; cf. USLOG-FGG)

.

If the UTM application was started with TESTMODE=OFF, and if a PEND ER occurs followed by one of the following KCRCDC codes, the UTM dump is suppressed.

```
FH01, K301, K302, K345, K601, K602, K603, K608,
KM01, KM02, KM03, KM04, KM05, KM07, KM08,
KR01, KR02,
KT01, KT02, KT04,
KU14
```

In this event, the work process is not terminated or restarted, i.e. the work process continues working for the UTM application. Exception: PGWT calls are permitted for the current TAC, and the call at which the error occurred was not a PEND call.

In the case of the KCRCDC code K316, no UTM dump is written, regardless of test mode. If PGWT calls are permitted for the current TAC, and the call at which the error occurred was not a PEND call, the program is loaded.

Reducing the volume of dump information with the start parameter DUMP-CONTENT

The start parameter DUMP-CONTENT allows you to specify whether openUTM is to reduce the volume of dump information or not. In this case, reduction means that process-independent memory areas (shared memories) are only included in the dump of the work process which caused the application to abort. Reducing the dump information means that the diagnostic documentation in the event of abortion of an application requires far less space. Costs for data storage and file transfer costs when forwarding the diagnostic documentation are far lower. For this reason, reduction of the dump information is activated by default. The start parameter DUMP-CONTENT can be used to deactivate or reactivate reduction of the dump information as required.

Syntax of the start parameter:

```
.UTM START DUMP-CONTENT={ STANDARD | EXTENDED }
```

STAN**D**ARD When UTM creates a dump file generation, process-independent memory areas are only contained in the dump for the first process (which caused abortion). This is generally sufficient for diagnostic purposes and is set by default.

EXTENDED Process-independent memory areas are contained in all the dumps of a dump file generation. You should only set this value when required to explicitly by the responsible Service staff.

3.2 The KDCDUMP tool

To edit a dump, you must call the KDCDUMP program. KDCDUMP offers two possible methods:

- Interactive evaluation:
  KDCDUMP loads a dump or part of a dump into memory, allowing you to process it interactively at the terminal. The dump is output to `stderr`. Interactive evaluation allows you, for example, to perform specific editing and search operations on particular tables in an extensive dump.
  You can specify which editor you want to use for editing (KDCDUMP command EDT) using the EDITOR environment variable.
  Default under Unix systems: `vi`.
  Default under Windows systems: WORDPAD.
- Editing of entire dump files in list form:
  KDCDUMP produces complete, edited dump files or file generation groups (FGGs) which you can view on the terminal by using an editor, for example.

Editing of the dumps is controlled by means of statements which are entered after starting KDCDUMP.

KDCDUMP is terminated by the control statement END.

When an error occurs in KDCDUMP during the evaluation, you should try to prepare the dump without summary information (INFO=DUMP).

If KDCDUMP and the dump file produced do not belong to the same openUTM version, then the dump is not output. KDCDUMP then outputs message K719. You can determine which version of openUTM the KDCDUMP and the UTM dump each belong to by reading this message.

Please note, the dumps from all the work processes of an application are required as when you are putting together diagnostic material.

3.2.1 Starting KDCDUMP

KDCDUMP is called with:

utmpath`/ex/kdcdump` or *utmpath*`\ex\kdcdump`

KDCDUMP reads the statements listed below from `stdin` and outputs messages and other output to `stderr`. You will find a list of the KDCDUMP messages in section “Messages of the UTM tool KDCDUMP” on page 349.

The program writes the edited dump files to the current directory and saves them under the name you specified using the OUTFILE operand of the FGG or FILE command. If you did not specify the OUTFILE operand, then KDCDUMP stores the output file under the name `dmplst`*/appl-name/RRRRRR/PPPP.NN* (see page 58 for a description of the letters).

3.2.2 KDCDUMP statements

The following table gives an overview of all the KDCDUMP statements and their meanings:

Statement	Meaning
!	Interrupt KDCDUMP and execute system command
!!	Execute most recently executed system command again
AFIND	Find address in UTM dump
scrolling statements	Position cursor in work area
DUMP	Read complete UTM dump or part of a UTM dump into memory
EDT	Call editor
END	Terminate KDCDUMP
FIND	Find and show table entry
FILE	Edit single dump file
FGG	Edit all files of an FGG (file generation group)
HELP	Display help for KDCDUMP statements
LIST	Edit table section
SFIND	Search for a pattern in the dump
SH | SYS	Start a Bourne shell from KDCDUMP under Unix systems or open a command prompt window under Windows systems (`cmd.exe`).
SYSLST	Activate/deactivate logging
TABLE	Show table

Entering KDCDUMP statements

KDCDUMP statements are read from `stdin`. A statement may comprise up to 256 characters; longer inputs result in error message K759.

Where statement names can be abbreviated, this is indicated in bold type. Thus, for example **AF**IND, indicates that you can abbreviate the statement name to AF.

With certain operands, one of the three input formats "C-string", "X-string" or "decimal" must be observed:

Designation	Input format / Example
C-string	[C]'This is a C-string'
X-string (hexadecimal)	X'AAAF' or X'aaaf' or X'AaAf'
Decimal	12345

If indices or displacements/offsets are specified, then these are always interpreted as being positive.

Note for openUTM under Unix systems and Windows systems on Intel processors

Hexadecimal inputs are always interpreted as arithmetic, i.e. indices or displacements which are read directly from the hexadecimal edited dump must be entered byte for byte in the reverse order, i.e. as arithmetic, for "little endian" machines (e.g. Intel). Generally, it is unnecessary to enter leading zeros.

Little endian examples

1. The 4-byte displacement X'00010203' is edited as X'03020100' in the hexadecimal section and must be entered as X'00010203'.
2. The 2-byte index X'FEAF' is edited as X'AFFE' in the hexadecimal section and must be entered as X'FEAF'.

In the following, "output to terminal" signifies the output to stderr.

The sections which follow describe the KDCDUMP statements in alphabetical order.

! Enter system command

This statement allows you to execute a Unix or Windows command during a KDCDUMP session.

```
!␣          command
```

command Unix or Windows command.

!! Repeat most recently executed system command

This statement allows you to repeat the most recent Unix or Windows command (executed with ! or !!) during a KDCDUMP session.

```
!!
```

AFIND Find address in dump

This statement serves to find a memory address in the UTM dump which was a valid address for a UTM table area while the dump was in progress.

The associated table entry is output at the terminal either as of the start of the entry or from the searched for location onwards. If the specified address does not represent a memory address or cannot be assigned to exactly one table entry then a message to this effect is output (K712 or K713).

```
AFIND␣           address [,F[ORMAT]=BE|LE]
```

address Desired address in memory; must be specified in hexadecimal form. Addresses in the dynamic table XAP-DYNM-AREA and the table group XAP-LOC-BUFF are not found.

FORMAT= In the case of "little endian" machines, specifies the format in which KDCDUMP expects the address specified in *address* ("big endian" or "little endian").

This parameter is ignored for "big endian" machines.

- BE KDCDUMP expects the address in "big endian" format , default value.
- LE KDCDUMP expects the address in "little endian" format.

Scrolling statements for interactive evaluation

If the required information cannot be displayed in one screen, the following statements can be used to scroll the display:

Statement	Meaning
+ ␣ Return only (blank entry)	Scroll forward one screen page.
+n	– With table elements which cannot be displayed in their entirety in one screen page, n is added to the start address of the information currently displayed. The result of this addition is the address starting at which the next information is displayed. – With tables where one element of the table can be displayed in its entirety in one screen page, n is added to the index of the first table element currently displayed. The result of this addition is the index of the element at which the continuation of the table display commences. If the end of the table is exceeded, the last table element is displayed; with trace tables, division into pages is performed automatically. n can be specified in decimal or hexadecimal form. n is ignored for the statement HELP TABLE-NAMES. Scrolling continues normally.
++	The end of the table or of the table entry is displayed. The trace tables are exceptions, where scrolling is performed to the chronological end. With the statement HELP TABLE-NAMES a normal "+" statement is executed.
- -n --	For backward scrolling, the same applies - with the opposite sign - by analogy as with forward scrolling. Scrolling is performed in the reverse direction no further than to the beginning of the table.
<	Only for the table TRCA: "<" can be used to scroll back to the beginning of the processing of the current (or previous) ANNOUNCEMENTs. In the case of the SFIND command, “<“ takes you to the previous hit even if the current hit is in a TRCA table.
<<	In the case of the SFIND command, "<<" takes you to the beginning of the hit list.
>	Only for the table TRCA: ">" can be used to scroll forward to the beginning of processing of the next ANNOUNCEMENT. In the case of the SFIND command, “>” takes you to the next hit, even if the current hit is in a TRCA table.
>>	In the case of the SFIND command, ">>" takes you to the end of the hit list.
X	The table display is aborted. Alternatively, the END command may also be entered, for example.

Screen support is offered in the form of a display indicating which statements are permitted in each case, for example:

"+/-/X" for all tables except TRCA
"+/</>/-/X" for the table TRCA

DUMP Read UTM dump into memory

This statement allows you to read into memory either a complete UTM dump file or part of a UTM dump file (not dump files in a directory). This statement must be issued before a UTM dump can be processed interactively at the terminal (for example, using the statements TABLE, LIST).

Particularly in the event of large UTM dumps, where the main memory available is insufficient to accommodate the entire uncompressed dump, it makes sense to read a number of individual dump areas into main memory one after another (using several DUMP commands) and to edit them.

If the UTM dump has not been written in full then completed dump areas can be read in and evaluated in turn.

You use the DOMAIN operand to specify the area of the UTM dump you wish to edit. If you then wish to edit other areas of the UTM dump, you must issue additional DUMP statements. If you specify FILE=*SAME in place of the filename, openUTM reads the new area from the UTM dump file that has already been uncompressed (suffix .T). This considerably accelerates the reading process.

Every time a DUMP command is issued, the area that had been read into main memory previously is removed and the new area is read in..

```
DUMP␣          { FILE = { filename | *SAME }

               DOMAIN = { ALL | KAA | KTA | CACHE | MPGP | SLOT |
                          XAPTP | ROOT | STACK | USERFILE |
                          JOURNALFILE-1 | JOURNALFILE-2 |
                          BUFSEGMENTS | GSSBFILE | LOCKFILE | CFGFILE |
                          ULSFILE } ]
```

FILE=

filename — Name of the UTM dump file

*SAME — Specify FILE=*SAME in place of the file name or link name if you wish to read a new area of the UTM dump which has already been uncompressed using DUMP into main memory (see DOMAIN operand).

DOMAIN= — This operand specifies the area of the UTM dump to be read into main memory.

ALL — The entire UTM dump is read into memory.
Default

KAA — Read the **K**DC **A**pplication **A**rea (global application system storage) into memory.

KTA — Read the **K**DC **T**ask **A**rea (task-specific system storage) into memory.

CACHE Read the UTM cache into memory.

MPGP Read the memory pagepool into memory.

SLOT Read slots (dynamically created tables) into memory.

ROOT Read KDC**ROOT** areas with the DIAGAREAs into memory.

XAPTP Read areas for the XAPTP module into memory.

STACK Read the STACK area into memory.

USERFILE
Read cluster user file into memory.

JOURNALFILE-1,JOURNALFILE-2
Read cluster administration journal file(s) into memory.

BUFSEGMENTS
Read segments of buffer management at local node level into memory.

GSSBFILE
Read cluster GSSB file into memory.

LOCKFILE
Read cluster lock file into memory.

CFGFILE
Read cluster configuration file into memory.

ULSFILE
Read cluster ULS file into memory.

EDT Call editor

This statement serves to call an editor. If the environment variable EDITOR is set to the name of an editor, KDCDUMP attempts to call this editor. If EDITOR is not set, then the "vi" editor is called under Unix systems, and WORDPAD is called under Windows systems.

```
EDT␣              [ filename ]
```

filename Name of the file to be read in.

END Terminate KDCDUMP

This statement serves to terminate KDCDUMP normally.

```
END
```

FGG Edit all files of an FGG

The FGG statement serves to edit all the files of an FGG jointly in a single output file. The name of the output file is specified with the OUTFILE operand. In command mode, the user is free to issue any number of FGG commands.

```
FGG␣        fgg-name
            [,EDIT = { PRINTER | TERMINAL } ]
            [,INFO = { LONG | DUMP | SHORT } ]
            [,OUTFILE = filename ]
```

fgg-name edits all dump files in the directory *fgg-name*. openUTM creates a separate FGG file for each work process in the application.

EDIT= This operand controls editing:

With EDIT=PRINTER (default), an output file is created which is intended for printing on the printer. The output contains feed control characters, page headers, with a maximum line length of 132 characters. The table of contents at the end of the output relates to print pages (default).

With EDIT=TERMINAL, the output file is edited in such a way that it can be evaluated on screen using an editor. The maximum line length is 80 characters. The table of contents at the end of the output relates to line numbers.

INFO= This operand controls output of the summary information. This is an extract of the complete dump information and contains the data frequently required for diagnosis.

INFO=LONG: editing with summary information (default).

INFO=DUMP: editing without summary information.

INFO=SHORT: KDCDUMP outputs only summary information.

OUTFILE= This operand allows you to specify that output is to be written to a file with the name *filename*.

If you do not specify OUTFILE, KDCDUMP assigns the default name `dmplst`*/appl-name/RRRRRR/PPPP.NN* (see page 58 for a description) and writes the output file to the current directory.

Notes

– When the FGG control statement is entered, a UTM dump file which was read in by means of the DUMP command is removed in its entirety from memory. Consequently, the UTM dump under examination prior to the FGG command will no longer be available for diagnostic purposes.

– The individual file generations of the FGG are read consecutively and removed from memory again after processing, so that after execution of this command none of the files remains in memory.

FILE Edit single dump file

This control statement edits a single dump file. The result of editing is written to an output file. The name of the output file is specified with the OUTFILE operand. If you do not specify OUTFILE, KDCDUMP assigns the default name.

```
FILE␣          [ dumpfile ]
               [,EDIT={ PRINTER | TERMINAL } ]
               [,INFO= { LONG | DUMP | SHORT } ]
               [,OUTFILE = filename ]
```

dumpfile — Name of the UTM dump file. This file may also belong to an FGG.

If the positional operand is omitted, it is assumed that a UTM dump file has already been read in with the DUMP statement. The FILE statement is then applied to this file. In this instance, the UTM dump file is not removed from memory.

EDIT= — This operand controls editing:

With EDIT=PRINTER (default), an output file is created which is intended for printing on the printer. The output contains feed control characters, page headers, with a maximum line length of 132 characters. The table of contents at the end of the output relates to print pages (default).

With EDIT=TERMINAL, the output file is edited in such a way that it can be evaluated on screen using an editor. The maximum line length is 80 characters. The table of contents at the end of the output relates to line numbers.

INFO= — This operand controls output of the summary information. This is an extract of the complete dump information and contains the data frequently required for diagnosis.

INFO=LONG: editing with summary information (default).

INFO=DUMP: editing without summary information.

INFO=SHORT: KDCDUMP outputs only summary information.

OUTFILE= — This operand allows you to specify that output is to be written to a file with the name *filename*.

If you do not specify OUTFILE, KDCDUMP assigns the default name `dmplst`*/appl-name/RRRRRR/PPPP.NN* (see page 58 for a description) and writes the output file to the current directory.

Note

– It should be noted that a UTM dump file which was read in by means of the DUMP command is deleted in its entirety from memory when the FILE control statement is used with the positional operand. After specification of a FILE statement with a file name, the file is deleted in its entirety from memory.

– In lists, hyphens in the table names (e.g.. CPTRT-NSR) are replaced by underscores (e.g. CPTRT_NSR).

FIND Find and show table entry

FIND serves to display the table entry for a UTM resource. The type of the display is the same as for the TABLE command.

FIND can only be executed if the current dump contains a KAA portion. Dumps with REASON PENDER or FMTERR, for example, do not contain a KAA portion.

```
FIND␣          { *ANY | type }
               [ {, *ANY | kdcdef-name } ]
```

*ANY — All UTM types specified for *type* in the KDCDEF statement are searched for.

type — Type of UTM resource. The following types can be specified:

type	Meaning	type	Meaning
AC	ACCESS-POINT	PO	(LTERM-) POOL
BC	BCAMAPPL	PR	PROGRAM
ED	EDIT	PT	PTERM, CON, OSI-CON
GB	GSSB	SB	KEYSET
LC	LTAC	TC	TAC
LM	LMOD	TL	TLS
LS	LSES	UL	ULS
LT	LTERM, LPAP, OSI-LPAP	US	USER

If the resource to be sought is a secondary storage area, the corresponding entry from the table DICN-NSR or DICN-SR is displayed. Otherwise, the entry in either the relevant NSR table or an SR table is output.

*ANY — All KDCDEF names are output, which were found for the specified *type*.

kdcdef-name — Name from the KDCDEF generation; must be entered as a C-string.

If KDCDUMP is unable to unambiguously identify the resource because, for example, "kdcdef-name" has not been specified, KDCDUMP then offers individual suggestions for selection. These must be acknowledged with "+" or ">":

\+ — displays the associated entry. Further scrolling is then possible with ">".

\> — continues the search

A new command can also be entered instead of “+” or “>”.

HELP Help about KDCDUMP

The HELP command provides information about the operation of KDCDUMP. This information is output to stderr.

```
HELP␣          [ { ALL | command-name | TABLE-NAMES } ]
```

HELP without operands simply outputs a list of all KDCDUMP statements. The operands have the following meaning:

ALL outputs an overview of the KDCDUMP statements. Default value.

command-name
Name of a KDCDUMP statement about which brief information is output.

TABLE-NAMES
outputs all valid table names. The letters following the table names are only of significance internally.

The dynamic table XAP-DYNM-AREA and the table group XAP-LOC-BUFF cannot be displayed with the TABLE command.

Table groups are indicated by an asterisk (*) prefixed before the name. The individual tables of the table group are listed according to the name of the table group and indicated by a plus sign (+) prefixed before the name.

If a dump is in memory, then the number of table entries is output (in hexadecimal form). This does not apply to the SLOT tables: In this case, the highest table entry index is output - the actual number of table elements is usually smaller.

As a consequence of negative results obtained during address validation when taking a dump it may happen that UTM areas or tables are not contained in the dump. Since no table entries are present in such situations, the tables are flagged as follows:

UA_ERROR: It was not possible to dump the UTM area containing the table.

TA_ERROR: Only the table could not be dumped.

LIST Edit table section

This control statement can be used to have a table section written to a file.

```
LIST␣          table-name,listfile
               [, START-INDEX = { FIRST | start } ]
               [, END-INDEX = { LAST | end } ]
               [, EDIT = { TERMINAL | PRINTER } ]
```

table-name — Name of the table from which a part is to be output. The valid names can be ascertained by means of HELP TABLE-NAMES.
You can also specify a table group name as the table name (these are indicated by an asterisk in the output from HELP TABLE-NAMES). In this case, all the tables belonging to the group are output (indicated by a prefixed plus sign (+)).

listfile — The table section is output to the file "listfile". An existing file will be overwritten without any warning.

START-INDEX= Entry in the table at which the output is to commence:

- FIRST — First entry in the table, default value.
- start — Table entry index at which the output is to commence. This value can be entered in either decimal or hexadecimal form.

END-INDEX= Last table entry which is to be output:

- LAST — Output continues to the end of the table, default value.
- end — Table entry index at which the output is to be terminated. This value can be entered in either decimal or hexadecimal form.

EDIT= Editing for printer or terminal; see description of the FGG statement on page 67.

Notes

– The LIST command does not support the table layout type "MEMORY LAYOUT" in the case of "bit tables".
– If you wish to have output a section of an existing SLOT table or of the RSBF table which is contained in the UTM dump and that section is not present, no message is output.

Example

You enter the following statement:

```
LIST S-S-S,V.LIST,S-I=2,E-I=3
```

If the table SLOT-SCB-STD exists, but neither entry 2 nor entry 3 exists, then the file V.LIST will be written containing only the header.

SFIND Search for a string

This statement allows you to search for a string in the UTM dump and output it on the terminal. However, you can only search for strings that are part of the user information in the dump. Information created by KCSDUMP in order to prepare the dump is not compared with the search string.

As far as the stacks are concerned, only the AUTOMATIC area is searched.

In addition, the strings are only searched for within a table entry. In other words, if the string you are searching for begins in entry 1 in the USRT-NSR table and ends in entry 2, for example, this does not count as a hit.

In some tables (MPV, NPROGRAM-TABL, ...), the information found is output in an edited form. In other words, if the table contains the string you are searching for, the table is displayed in a specially prepared way. Consequently, the string that has been found cannot always be seen immediately.

```
SFIND␣      search
            [, ALIGN = {1 | 2 |4 | 8}]

            [, HITS = {ALL | nr_max_hits}]
            [, DOMAIN = {ALL | KAA | KTA | AUTO | SLOT | ROOT |
                         CACHE | MPGP | XAPTP-LOCAL | XAPTP-GLOBAL |
                         USERFILE | JOURNALFILE-1 | JOURNALFILE-2 } ]
                         BUFSEGMENTS | GSSBFILE | LOCKFILE |
                         CFGFILE | ULSFILE } ]
```

searchThe positional operand stands for the string you are searching for, which cannot be longer than 190 bytes. It can be specified as a C-string or an X-string.

ALIGN = You can use this to specify the alignment limit of the search string.

- 1 Single-byte alignment

  This is the default.
- 2 2-byte alignment
- 4 4-byte alignment
- 8 8-byte alignment

HITS = You can use HITS to specify the number of hits after which the search is terminated.

- ALL The entire dump is searched.

  This is the default.
- nr_max_hits

The search is terminated after *nr_max_hits*. *nr_max_hits* can be any value from 0 to 32767. 0 has the same effect as ALL.

DOMAIN This allows you to specify which parts of the UTM dump are to be searched.

ALL The whole dump is searched.

This is the default.

KAA The **K**DC **A**pplication **A**rea is searched (global application system memories).

KTA The **K**DC **T**ask **A**rea is searched (process-specific system memories).

AUTO **AUTO**MATIC STORAGES are searched (working memory areas of the interrupted UTM modules).

SLOT Slots are searched (dynamically created tables).

ROOT KDC**ROOT** areas and DIAGAREAs are searched.

CACHE CACHE memory areas are searched.

MPGP The memory page pool is searched.

XAPTP-LOCAL
Local processor areas of the XAPTP module are searched.

XAPTP-GLOBAL
Global areas of the XAPTP module are searched.

USERFILE
Search through cluster user file

JOURNALFILE-1,JOURNALFILE-2
Search through cluster administration journal file(s).

BUFSEGMENTS
Search through segments of buffer management at local node level.

GSSBFILE
Search through the cluster GSSB file.

LOCKFILE
Search through the cluster lock file.

CFGFILE
Search through the cluster configuration file.

ULSFILE
Search through the cluster ULS file.

SH and SYS Interrupt KDCDUMP

The SH or SYS statement starts a Bourne shell under Unix systems or a command prompt window under Windows systems from within KDCDUMP, allowing you to enter commands.

```
{ SH | SYS }
```

You can return to the KDCDUMP program with the `exit` command. The shell started is terminated or the command prompt window is closed.

SYSLST Activate/deactivate logging

This statement serves to output the results of the statements AFIND, FIND, HELP TABLE-NAMES and TABLE to *stdout*.

The maximum length of an output line is 80 characters. No messages are written to *stdout*.

```
SYSLST␣          { ON | OFF }
```

ON	The output is directed to *stdout*.
OFF	No output to *stdout*. “SYSLST OFF” applies when KDCDUMP starts.

TABLE Show table

The TABLE statement serves to output part or all of a table from the currently processed UTM dump. The output can be in dump format or in symbolic form.

```
TABLE␣          table-name
                [, start-index
                   [, { END-INDEX = { SAME | LAST }    |
                        DISPL = displacement
                      }
                ]  ]
```

table-name Name of the table to be output, with a maximum length of 13 characters.

You can use the HELP TABLE-NAMES command to display a list of all table names.

The dynamic table XAP-DYNM-AREA and the table group XAP-LOC-BUFF cannot be displayed with the TABLE statement. They can only be displayed with the LIST statement.

The table names can be abbreviated as long as they remain unique. The following rules apply:

– There must be at least one character at the beginning and after a hyphen. This character must be the first character of the name.

– The individual name parts which begin with a hyphen and are delimited by the next hyphen or by the end can be omitted if the following name part - where one was present - is also removed.

Example
The table name SLOT-VGT-DYN can be abbreviated as
SL-V, S-VGT or S-V-D etc.

start-index This positional operand specifies the number of the entry at which the output is to commence.

The default value is 0.

If the operand is given the value 0, the entire table is output and all other operands have no effect. The SLOT tables and the STACK area are exceptions here:

– The second positional operand is *not* optional for SLOT tables, but must be assignable to a valid SLOT.

– For the STACK area, in the case of operand value 0 (default value), all interrupt addresses (absolute and relative - module name + internal module address) are displayed. If the operand value n is greater than 0, then the stack entry belonging to the interrupt address n is displayed in its entirety.

With both these exceptions, following operands have no validity and, if specified, have no effect.

The input can be in either hexadecimal or decimal form.

Notes

– In the case of the trace tables, with a null specification the current entry is displayed as the last item on the screen page.

– If, in the case of the tables PTRM-NSR, LTRM-NSR, USRT-NSR, TACT-NSR, POOL-NSR, EDIT-NSR and LTAC-NSR, a particular entry is to be displayed, then the generation name is also included in the output. For the first three of these tables, in the event of a connection existing at the time of the dump, cross-references to the other tables are given.

– For bit tables and tables whose entries are always two bytes in length, it is generally not possible to position precisely to the desired entry. But the entry is then contained in first line output with table information.

END-INDEX= This operand can be used to specify whether, apart from the entry specified under "start-index", any other table entries are to be displayed.

SAME Only the entry specified with "start-index" is displayed.

LAST The entire table is displayed starting with the entry "start-index".

DISPL=displacement

This facility can be utilized in order to output the table entry specified in the second positional operand with a displacement from the beginning of the table entry. This capability is supported only for tables in normal dump format, i.e. in hexadecimal representation.

The input format is decimal or hexadecimal (see example).

The default value applies for the optional operands END-INDEX and DISPL:
END-INDEX = SAME.

Notes

– If the selected table information does not fit in one screen page, scrolling statements (see relevant section) can be used to make further data visible (does not apply to symbolic editing).

– The TABLE command does not support the table layout type "MEMORY LAYOUT" in the case of "bit tables".

Example

1. `T KAA,1,D=100`

   Outputs the KAA structure in hexadecimal and print-edited form starting from offset 100.

2. `T ROOT,1,S=*YES`

   The ROOTDATA structure is output in full in symbolic form

3.2.3 Messages of KDCDUMP

KDCDUMP issues messages in the format K7nn. You will find the message texts and additional information on the messages later on in this document.

3.3 Contents of the UTM dump

The structure of a UTM dump is described below. A UTM dump edited with KDCDUMP can contain the following areas:

KAA the global application system storage.

KTA the process-specific system storage.

AUTOMATIC STORAGES
Automatic storage areas of the utmwork process call hierarchy.

KDCROOT with the KDCROOT tables and the DIAGAREAs.

User File, Journal Files, ...
Areas for UTM cluster applications.

SUMMARY A summary

Contents.

The terms "task" and "process" are used synonymously below.

If the dump was produced due to a PEND ER (either programmed by the user or produced internally after KCRCCC ≥ 70Z), then the dump only contains the KDCROOT area.

To make finding the table easier, the prepared dump contains a list of tables with page numbers at the end of the dump. Each of these prepared tables has a header that identifies the prepared table. The headers are also independent of whether the dump was prepared for the terminal or for printing.

The header starts with the corresponding storage area dumped (KAA TABLE / KTA TABLE or KDCROOT) and the name of the table.

Only the names of the prepared tables are listed in the following.
Example: (KAA TABLE) TRAN_SYN_NSR / Transfer Syntax NSR.

The part of the name after the slash has been omitted in the following if only the hyphens have been replaced by underscores or spaces in that part of the name.

Note

Storage areas edited in hexadecimal format must be analyzed according to machine type. The examples used in this chapter were produced on INTEL machines.

3.3.1 Global application system storage (KAA)

The KAA storage area (KDCS application control area) is application-specific, i.e. openUTM creates a KAA system storage area for every UTM application running.
This storage area is only contained in the dump of the first (process task) of the application (cause of the dump) by default. See the DUMP-CONTENT start parameter.
It consists of the following areas and tables (the suffix NSR stands for "not security-related“, SR for "security-related" and DSR for "dynamically security-related" in the following):

KAA	An area with a fixed structure. It mainly contains the parameters for the generation statement MAX and other non-security related parameters that openUTM requires to control the application run.
MPV	Memory page vector that contains the KAA-relative addresses for all UTM pages of the KAA.
PGPT-NSR	Page pool bit list.
PGBT-NSR	Page pool block table (bit list).
PGCT-NSR	Page pool block page counter table.
CATRT-NSR	Cache translation table.
CPTRT-NSR	Cache translation table for cluster page pool (NSR).
HASH-NSR	Hash table for DICN_NSR.
MPGP-HL-CHNT	MEMORY-Pagepool High-Level Chain-Table
MPGP-LL-CHNT	MEMORY-Pagepool Low-Level Chain-Table
DICN-NSR	Dictionary of names.
TSKT-NSR	Task table.
PTRM-NSR	Physical terminal table.
PTRM-DYN-NSR	Pool-relative addresses for dynamic physical terminal tables.
DCB-DYN-NSR	Pool-relative addresses for dynamic UTM-D control blocks.
LTRM-NSR	Logical terminal table.
TACT-NSR	Transaction code table.
USRT-NSR	User table.
QUEU-NSR	Table for temporary queues.
VGT-DYN-NSR	Pool-relative addresses for dynamic service tables.
POOL-NSR	Terminal pool table.
FPMM-NSR	Table for the administration of asynchronous jobs.

TASEQ-NSR	Table for the order of transactions.
LOG-CB-NSR	Logging-Exit Control-Block
CARD-NSR	internal administration table (NSR)
CDCONT-NSR	Pool-relative addresses of ID card information read in.
BCAT-NSR	BCAM application table.
ACPNT-NSR	OSI TP access point table.
CON-NSR-EXT	OSI TP extensions for physical terminal tables.
PRGT-NSR	Program table.
LMOD-NSR	Load module table.
LTAC-NSR	Local TAC table for UTM-D.
KSET-NSR	Table with key sets.
LPAP-NSR-EXT	Logical partner application table extension.
CACT-NSR	Cache control table.
CONS-ENTRIES	Consistency entries.
CL-CONS-ENTR	Consistency entries in the UTM cluster files (NSR)
SLOTMP-NSR	Slot memory pool control table.
UTMX-NSR	UTM-specific data (Unix systems/Windows systems).
MMOD-NSR	Message module table.
LIBRARY-TABLE	Table of libraries for KDCROOT (NSR).
AREA-TABLE	Area table for KDCROOT (NSR).
DB-NSR	Table containing the generation information for database systems.
EXIT-TABLE	Table of exits for KDCROOT (NSR).
NODE-NSR	Node table for OSI TP.
INST-NSR	Table of entities for OSI TP.
TRAN-SYN-NSR	Transfer syntax table for OSI TP.
AB-SYN-NSR	Abstract syntax table for OSI TP.
APL-CNTX-NSR	Application context table for OSI TP.
PRCP-NSR	Internal administration table for principal entries (NSR).
CLND-NSR	Address information for the nodes of a cluster.

SLOG-NSR	SYSLOG buffer area.
CLFB-NSR	Data of the cluster configuration file at the time of the last read access.
KAA-SR	Area for global runtime parameters.
KAA-SR-MAP	Bit list for valid KAA pages of the KDCFILE.
HASH-SR	Hash table for DICN_SR.
HASHQU-SR	Hash table for QUEU_SR.
DICN-SR	Dictionary of names.
LTRM-SR	Logical terminal table.
TACT-SR	Table of transaction codes.
USRT-SR	User table.
QUEU-SR	Table for temporary queues.
VGT-SR	Service table.
POOL-SR	Terminal pool table.
FPMM-SR	Table for the administration of asynchronous jobs.
RVGT-SR	Remote service table.
CHNT-SR	Chain table for the administration of the pagepool of the KDCFILE.
NODE-SR	Node table for OSI TP.
KAA-DSR	Table for global runtime parameters.
KAA-DSR-MAP	Bit list for valid KAA pages of the KDCFILE..
TSKT-DSR	Task table.
HASHPT-DSR	Hash table for PTRM_DSR.
HASHLT-DSR	Hash table for LTRM_DSR.
HASHTC-DSR	Hash table for TACT_DSR.
HASHUS-DSR	Hash table for USRT_DSR.
HASHPR-DSR	Hash table for PRGT_DSR.
HASHKS-DSR	Hash table for KSET_DSR.
HASHLC-DSR	Hash table for LTAC_DSR.
PTRM-DSR	PTERM table.
LTRM-DSR	LTERM table.

TACT-DSR	TAC table.
USRT-DSR	USER table.
PRGT-DSR	Program table.
LMOD-DSR	LMOD table
KSET-DSR	Key set table.
LTAC-DSR	Local TAC table for UTM-D.
RSAKEY-DSR	Table for RSA keys (DSR)
CPCH-DSR	Chain table for cluster page pool (DSR)

3.3.1.1 The CONS_ENTRIES table

The CONS_ENTRIES table provides space for 10 entries and is written cyclically, i.e. the 11th entry overwrites the 1st entry.

Entries are written in the table when the following events occur:

– when a KDCDEF generation is executed
– when KDCUPD is called
– each time the UTM application is started

A table entry has the following format:

type	r1	r2	f1	i1	i2	i3	f2	yymmdd	hhmmss	tttt	
4	1	1	2	1	1	1	1	6	6	4	Byte

type ’DEF ’ for KDCDEF run
’UPD ’ for KDCUPD run
’STRT’ for application start.

r1 and r2 designate the correction status of KDCDEF, KDCUPD and UTM system code (e.g. ’10’ for V6.3A10).

r1,r2 ’nm’ two digits for the source correction character
’00’ space for the first release of a version.

f1, f2 Filler.

For *type* ’STRT’ and ’DEF’: i1,i2 and i3 contain additional information.
For *type* = STRT

i1 ’C’ for UTM cold start
’W’ for UTM warm start

i2 ’P’ for production for *type* = STRT

i3 ’B’ for batch task
’D’ for dialog task

For *type* = DEF

i2 ’-’ when the KDCFILE was created successfully for *type* e= DEF
’W’ when the KDCFILE was created with errors.

yymmdd
Date (year, month, day).

hhmmss
Time (hours, minutes, seconds).

tttt PID of the UTM task for *type* = STRT.

3.3.1.2 CACHE buffer

CACHE-BUFFER	Cache memory

3.3.1.3 UTM SLOT POOLS

Dynamically created tables (slots):

SLOT-CARD	Contents of read ID cards
SLOT-PTRM-DYN	Dynamic, physical terminal tables
SLOT-DCB-DYN	Dynamic UTM-D control tables
SLOT-VGT-DYN	Dynamic service tables
SLOT-LOG-CB	Dynamic control table for logging exit

3.3.2 Global application system memory of XAP-TP

The storage area of XAP-TP contains global application tables and areas that XAP-TP sets up to control the application run. This area exists only if the UTM application is generated with OSI-TP communication. It contains only information that is not relevant for logging.

By default, the storage area is contained only in the dump of the first process (=task) of the application (initiator), see section “Reducing the volume of dump information with the start parameter DUMP-CONTENT” on page 59.

It consists of the following areas and tables.

XAP-SHAREMEM	Fixed part of the global storage area of the XAP-TP component; includes, for example, generation values for OSI-TP communication.
XAP-INST	Instance table for XAP-TP.
XAP-TRAN-SYN	Transfer syntax table for XAP-TP.
XAP-AB-SYN	Abstract syntax table for XAP-TP.
XAP-ACCPT	OSI-TP access point table for XAP-TP.
XAP-PARTNER	Partner table for XAP-TP.
XAP-ASSOC	Association table for XAP-TP.
XAP-CAF	Channel auxiliary facility table for XAP-TP.
XAP-DIAL	Dialog table for XAP-TP.
XAP-LOG-DAM	Log damage record table for XAP-TP.
XAP-RCH-GROUP	Recovery context handle group table for XAP-TP.
XAP-DYNM-AREA	Dynamic global area for XAP-TP; the size of this area is defined by the OSI-SCRATCH-AREA parameter of the MAX statement.

3.3.3 The process-specific system memory (KTA)

The system memory KTA (KDCS task-specific area) is displayed for each process (task) of an application. This memory consists of the following areas and tables:

KTA	Area with task-specific runtime parameters and working areas.
TACB	Transaction control block.
PIBF	Processing item buffer
FILT	File table.
BCAT	BCAM application IDs.
ADMP	Addresses of shared memory.
ILBF	Input letter buffer.
OLBF	Output letter buffer.
OSBF	OSI TP MSG buffer.
ILBF-SNA	Input letter buffer SNA.
OLBF-SNA	Output letter buffer SNA.
PPLT	PAM parameter list table.
SPB	Station Level Parameter Block.
VTCB	VTSU Control Block.
FCBP	File control blocks PAM.
FCBS	File control blocks SAM.
TPGA	Task program table.
MSGB	Message buffer.
MUXI	Input buffer for MUX protocol header.
MUXO	Output buffer for MUX protocol header.
LTERM-BUNDLE	Master LTERM bundle table
LPAP-BUNDLE	Master LPAP bundle table
FMGT	FMGT file management table
LCBT	Lock Control Block for CLUSTER (global locks).
ETPNDS-TPR	List of the components linked into the UTM subsystem together with their ETPNDs.
TRCA	Trace area.

USBF	In UTM cluster applications: user file buffer.
JFBF	In UTM cluster applications: journal file buffer.
CRBF	Confirmatory record buffer.
REST	Restart control area (only for dumps during the warm start).
RSBF	Restart buffer area (only for dumps during the warm start).

3.3.4 Process-specific system memory of XAP-TP

The storage area exists only if the UTM application is generated with OSI-TP communication.

It consists of the following areas and tables.

XAP-LOCALMEM	Area with process-specific runtime parameters and working areas.
XAP-TRACE	Trace area of XAP-TP.
XAP-LOC-ACCPNT	Process-specific table for OSI-TP access points.
XAP-LOC-BUFF	Dynamic local buffer containing a group of tables (XAPL-CONO, XAPL-CTRE, XAPL-CTAC, XAPL-SCRA, XAPL-OSSB, XAPL-UDAT). These areas cannot be edited interactively with the TABLE command.

3.3.5 The KDCROOT area

Tables and areas are printed out from KDCROOT; these can be used in many cases to diagnose application errors. With PEND ER dumps, only this data is made available.

The following areas are output:

CONTEXT-AREA / Context Area
: This area is only output if the dump was triggered by a signal. It contains the address of the interruption. This address is also output symbolically as a function + displacement.

PROGRAM-TABL / Program Table
: An entry contains, among other things, the program name and the start address of a program unit.

LOAD-MODULE / Load Module Table
: Area with information on the shared objects.

AREA/Area Table
: Area containing information on the generated areas.

EXIT / Exit Table
: Area containing information on the exits (contains indices in PRGT).

LIB / Library Table
: Library table.

MEMORY-POOL / Memory Pool Table
: Area containing information on the memory pools. If generation is performed without load modules, this area simply contains information from the generation and additional information (e.g. the address of the memory pool) can be found in the User Root area.

MSG-MODULE / Message Mod Table
: Area containing information on the message modules.

UTM-DIAGAREA / UTM Diagarea
: Area with diagnostic information on all KDCS calls.

KB / Communication area KB
: Communication area, consisting of a KB header, KB return information and KBPROG in the generated length.

SPAB / Standard primary working area SPAB
: Standard primary working area.

MPUT-BUFFER / MPUT Buffer
: Intermediate storage for MPUT messages.

FORMUSER-BUF / FORMUSER Buffer
: Buffer area with logical I/O messages.

RESTART-BUFF / RESTART Buffer
: Restart area for screen formatting (not relevant for Unix systems or Windows systems).

IO-BUFFER / IO Buffer
: Buffer area with physical I/O messages.

ROOTDATA / ROOTDATA
: Communication areas between KDCROOT and the UTM system modules.

ROOT-TRACE / Root Trace
: Area with trace records for ROOT execution.

FORM-USER-AR
: Interface parameters relating to the formatting system (not relevant for Unix systems or Windows systems).

HLL-USER-ARE / HLL User Area
: Parameter list of IUTMHLL.

IPC-HEADER
: Administration area for IPC shared memory.

IPC-FREE-QUEU
: Anchor for the list of free objects of IPC-ELEMENT, IPC-LETTER and IPC-ANNO.

IPC-TIMER-ID
: Identification for utmtimer jobs.

IPC-SEMA
: Semaphore table.

IPC-APPL-GLOB
: Global application information.

IPC-APPL
: Table of application names.

IPC-EXTP
: Table of external partners.

IPC-BRSE
: Bourse table.

IPC-ELEMENTS
: Area for IPC-ELEMENT objects (message type, length, etc.).

IPC-ANNOS
: Area for IPC-ANNO objects.

IPC-LETTER
: Area for IPC-LETTER objects (message contents).

IPC-VIRT_HOST
: Table containing the virtual host names from the Mapped Hostname file.

IPC-REAL_HOST
: Table containing the real host names from the Mapped Hostname file.

IPC-SM2-DATA
: Area with the measured values for openSM2.

IPC-TRACE
: IPC trace area.

VGM-AREA / VGM Area
: Area for the service memory for a connected database.

USER-ROOT / Root gen by user
: Area contains data from user-own ROOT module.

OSS-AREA / OSS Area
: OSS shared memory.

NLS-AREA / Environment Area
: Environment of the work process.

XA-AREA / XA Area
: Database area.

TIMER-AREA / Timer Area
: Timer administration area.

TABDESC-AREA / TABDESC-AREA / Table Descriptors
: Area containing data on the edited root tables (contains name, address, number of entries and length).

SHMPROT-AREA / SHMPROT Area
: Area for shared memory protection.

ADMI-DIAGAREA / Administration DIAGAREA
: Area containing trace records for all calls to the administration program interface from the program units.

ADMI-USERAREA / Administration USERAREA
: Area containing a trace record for the data passed from the program unit through the administration program interface.

LOGEXITBUFFER/LOG-EXIT-MESSAGE-BUFFER
: Area for logging exit

STRT-PAR
: Area containing the start parameters specified when the application was started.

ACCOUNTING-A / Accounting Area
: Area for accounting data.

TAM / TAM
: Transaction storage for a connected database.

TSKM / TSKM
: Task-specific storage for database communication.

DB-DIAGAREA / DB Diagarea
: Area containing diagnostic information for all database calls (only if a database has been generated).

DB-USER-AREA / DB User Area
: IUTMDB parameter list (only if DB generated.)

DB-INF-PROG / Db Info Program
: Area which contains data relating to the current program unit.

DB-INF-APPL / Db Info Application
: Area which contains data relating to the application.

DB-SUMMARY
: Area containing general information on the generated databases (e.g. number etc.).

DB-TABLE
: Table of generated databases

3.3.5.1 PROGRAM table

An entry has the following structure (decimal byte number) and meaning:

Entry	Meaning
1	Index of the entry in the ROOT program table
2	Program name
3	Language type of program as specified in statement PROGRAM... COMP= . Possible values: C = X'06' COB2 = X'0A' (COB2 is an alias name for all Micro Focus compilers) CPP= X'0C' NETCOBOL=X'0D'
4	Load mode of program as specified in statement PROGRAM .. LOAD= . If the application is generated with shared objects, this output corresponds to the value specified in the SHARED-OBJECTS statement (LOAD-MODE parameter). Meaning of values*: STATIC = X'00': Program linked statically to application STARTUP = X'01': Program is dynamically loaded on starting application POOL = X'03': Program is loaded into a memory pool
5	Exchange mode of program, always set to NOTCH = X'00' (not exchangeable).
6	Not relevant, always set to X'00'.
7	
8	
9	Load status of program, always set to LOADED = X'01' (program loaded).
10	Program address

3.3.5.2 LOAD-MODULE table

The entries have the following meanings:

Bytes hexadec.: 32-Bit (64-Bit)	Meaning
-	Index of this entry in the ROOT shared object table (LMOD)
24-43 (48-67)	shared object name
44 (68)	Load mode of shared object, possible values: X'00' = shared object linked statically to application X'01' = shared object is dynamically loaded on starting application X'02' = shared object is loaded when first called X'03' = shared object is loaded into a memory pool
45 (69)	not used
46 (6A)	Exchange mode of shared object, possible values: X'00' = shared object not exchangeable X'01' = shared object individually exchangeable X'02' = (not currently used) X'03' = shared object exchangeable only with entire application because loaded into local pool
47-5F (6B-83)	Version number of the shared object to be loaded (on exchanging)
60-61 (84-85)	Index of the context to which this shared object belongs
62-63 (86-87)	Index of the memory pool (MPOOL) into which this shared object is loaded
64-65 (88-89)	Index of the library (LIB) from which this shared object was loaded
66-67 (8A-8F)	not used
68-6B (90-97)	UTM internal Info
6C-6D (98-99)	Index of the first program in this shared object
6E-85 (9A-B1)	Version number of the current (=most recently loaded) shared object
86-87 (B2-B3)	UTM internal Info

Bytes hexadec.: 32-Bit (64-Bit)	**Meaning**
88-89 (B4-B5)	Index of the next shared object in the same context
8A-8B (B6-B7)	Index of the first AREA in this shared object
8C (B8)	Load status of shared object: X'00' = shared object not loaded X'01' = shared object loaded X'02' = shared object when a program which is linked into this shared object is loaded. X'03' = (not currently used)

If the new version number is the same as the old version number, this means that this shared object has not been exchanged since the last KDCDEF run.

3.3.5.3 UTM-DIAGAREA

The UTM DIAGAREA is a process-specific trace area in which all events are logged. This event thus contains all the events that occurred immediately before a service or application aborted.

The UTM DIAGAREA is written cyclically. Two cycles are separated by a dividing line comprising '=' characters and banks. The newest entry is above the dividing line and the oldest entry is below the dividing line. Every entry is 136 or 256 (64-bit) bytes in length. The total number of entries that can be accommodated by the UTM DIAGAREA depends on the generation parameter `MAX TRACEREC`.

The following types of entries are written to the UTM DIAGAREA:

- UTM records (type `KDCS`)

  UTM records are written if the following events occur:

  – a KDCS call occurs in a program unit or
  – an internal call is issued to the UTM system code or
  – openUTM issues an internal PEND ER call (system PEND ER) in response to a serious error.

  If a PEND ER is issued by the system, then the entry in bytes 22- 57 contains an error text.

  Additional trace information is written for calls from the administration interface (KDCS-opcode=ADMI), see ADMI-DIAGAREA page 114.

- UTM records to identify the service (type `VGID`)

  A VGID record is written to the UTM-DIAGAREA every time a program unit is started and when a PGWT call returns.

Header for the records in the UTM DIAGAREA

Every record starts with a header containing the following information:

Byte	Meaning
0 - 1	Count of the current entries in the DIAGAREAs (UTM and DB)
2 - 5	Type identifications (KDCS, VGID, FHCL, ITRC)
6 - 7	Currently not used (preset to '= =')
8 - 15	Time stamp

Structure of the header

As of byte 16, the content of the records depends on the record type.

Structure of the UTM-DIAGAREA for a KDCS call from a program unit

Byte: 32-Bit (64-Bit)	Field name and meaning					
16-19 (16-19)	KCOP:[1]	user operation code: INIT, MGET, MPUT, etc.				
		internal operation code: see table on page 112				
20-21 (20-21)	KCOM: operation modification					
22-23 (22-23)	KCLA: area length or queue level (in the case of QCRE) or KCLKBPRG: length of KB program area with INIT					
24-25 (24-25)	KCLM: message length or KCLPAB: length of standard primary work area with INIT KCWTIME: wait time in seconds in the case of DGET					
26-33 (26-33)	KCRN: reference name					
						MCOM call
34-41 (34-41)	KCMF: name of the abstract syntax KCLT: LTERM name of the LTERM partner or KCUS: user ID or KCPA: name of the partner application (with APRO call)					KCPOS: destination of positive follow-up message
42-43 (42-43)	KCDF: screen function, see table on page 102					KCNEG: destination of negative follow-up message
	DPUT/DADM	DGET	QCRE	PADM	APRO	
44 (44)	KCMOD: mode (“A”/”R”/”_”)	KCQTYP: destination type ("U"/"Q"/"T")	KCQMODE: mode ("S"/ "W"/bin. zero)	KCACT: action (ON/ OFF/CON/ DIS)	KCPI: service ID	
45-47 (45-47)	KCTAG: days	empty	empty			
48-49 (48-49)	KCSTD: hours			KCADRLT: new LTERM name of the printer		
50-51 (50-51)	KCMIN: minutes					KCCOMID: complex ID
52-53 (52-53)	KCSEC: seconds				KCOF: OSI TP functions	
54 (54)	KCQTYP: destination type ("U"/"Q"/bin. zero)					
55-57 (55-57)	empty			empty		

Structure of an entry of the UTM-DIAGAREA for a KDCS call

Byte: 32-Bit (64-Bit)	Field name and meaning
58-89 (58-89)	KCRFELD in KCKBC: KB return information area (see the table on page 102)
92-95 (96-103)	Return address to the program unit (the address points to a position after the KDCS call in the program unit)
96-99 (104-111)	Address of the user data area (2nd parameter of the KDCS call)
100-103 (112-119)	Service index
104-111 (120-127)	KCLOGTER in KCKBC: LTERM name
112-119 (128-135)	KCBENID in KCKBC: name of the current user ID

Structure of an entry of the UTM-DIAGAREA for a KDCS call

[1] If KCOP=INFO (bytes 16-19), the message area of the logged call is written to the next entry of the UTM DIAGAREA at a length KCPAC. In the event of operation modification KCOM=CK (bytes 20-21), this information is of interest for diagnostic purposes, as it logs the call to be checked.

If KCOP=INFO and KCOM=CK, the message area of the logged call is written to the next entry in DIAGAREA at the length specified by KCPAC without incrementing the counter.

For internal calls from openUTM, the following codes are entered in the KCOP field:

KCOP contents	Situation in which this entry was written	Entries in subsequent fields
STRT	Start of UTM application program, beginning of start handling in the UTM system code	No entry
WAIT	UTM task joins the UTM work bourse	No entry
CONT	Continuation in UTM system code after a DB action via KDCROOT or after the INPUT exit is called	KCRCCC, KCRCKZ and KCRCDC only. for INPUT exit: parameters, see table page 103
NOOP	Buffer area of MESSAREA must be emptied (only possible when event monitor is switched on)	- - -
ADMI	UTM administration action	UTM internal interface

Structure of KCRFELD

<table>
<tr><td>58-59</td><td colspan="3">KCRDF: screen function return
KCRWVG: number of waiting services with DGET</td></tr>
<tr><td>60-61</td><td colspan="3">KCRLM: actual length of the message</td></tr>
<tr><td></td><td colspan="2">INFO CK call</td><td>MGET call</td><td>SIGN ON call</td></tr>
<tr><td>62</td><td colspan="2" rowspan="3">KCRINFCC:
KDCS return code
of the KDCS
call checked</td><td>KCVGST:
service status</td><td>KCRSIGN1:
primary code</td></tr>
<tr><td>63</td><td>KCTAST:
transaction status</td><td rowspan="2">KCRSIGN2:
secondary code</td></tr>
<tr><td>64</td><td>empty</td></tr>
<tr><td>65</td><td>empty</td><td colspan="2">KCRMGT: message type</td><td></td></tr>
<tr><td>66-68</td><td colspan="3">KCRCCC: KCDS return code</td></tr>
<tr><td>69</td><td colspan="3">KCRCKZ: identifier: P (productive or live operation application),</td></tr>
<tr><td>70-73</td><td colspan="3">KCRCDC: internal return code</td></tr>
<tr><td>74-81</td><td colspan="3">KCRMF: return of the abstract syntax</td></tr>
<tr><td>82- 89</td><td colspan="3">KCRPI: service identification return
KCRUS: user ID in the case of SIGN ST or creator of the message with DGET
KCRQN: name assigned by openUTM to the temporary queue in the case of QCRE NN</td></tr>
</table>

DIAGAREA after a call of the INPUT exit

After the INPUT exit is called, open UTM enters the most important parameters that the exit receives or sets (KCPAC) in the KDCS record:

Bytes	Field name and meaning
16-19	KCOP: "CONT" is entered here
20-25	empty
26-33	KCIFCH: first 8 characters of input
34-35	KCICVST: service status: "ES"/"ET"/"RS"/"EC
36-37	KCIFKEY: value of F key: 1,...,20
38-39	blanks
40-41	KCICFINF: information about control fields: "UN"/"NO"/"ON"/"MO"
42-49	KCINTAC/KCINCMD: next TAC to be started or next user command
50-51	KCICCD: code for effect of input: "ER"/"CC"/"SC"/"ST"/"CD"
52	KCICUT: truncate TAC: "Y"/"N"
53	---
54-57	KCIERRCD: error info for data display terminal
58-...	the following is not relevant for diagnostics

Structure of an UTM-DIAGAREA entry after a call of the INPUT exit

Error texts in DIAGAREA after SYSTEM PEND ER

To allow more rapid diagnosis in the event of a SYSTEM-PEND-ER, the corresponding line of the UTM DIAGAREA contains a printable error text in bytes 22-57.

the following table contains a list of all error texts, plus the relevant cause of the error, the module where the PEND ER occurred and the type of error involved (system error or user error).

Error text	Cause, module	Error type
APPL. PROGRAM DOES NOT EXIST	No indicator for a program unit existed when program unit started (70Z with KR01).	System error
APPL. PROGRAM WITHOUT PEND	Application program was not terminated with PEND. (84Z)	User error
ASYNC. PROGRAM NOT FOUND	Asynchronous program no longer available (program may have been exchanged). PEND ER dump follows with 70Z and KR02.	User error

Error texts in DIAGAREA after SYSTEM PEND ER

Error text	Cause, module	Error type
ERROR IN "START-TP" OF LGCON	The language connection module supplied an incorrect return code when program unit was started.	System error
KB END LABEL OVERWRITTEN	The KB in the application program is greater than specified in the generation (70Z with KR04). Action: change KB= operand in the MAX statement	User error
SPAB END LABEL OVERWRITTEN	The SPAB in the application program is greater than specified in generation (70Z with KR05). Action: change SPAB= operand in MAX statement	User error
KCRCCC >= 70Z AFTER UTM SVC	UTM SVC set KCRCCC in KB header to ≥ 70Z	User/system error
ROOTDATA CODE INVALID	UTM SVC set ROOTDATA CODE to an invalid value	System error
ERROR ROUTINE XT... ENTERED	A signal with the number ...(in KDCRTDI) occurred.	User/ system error
ERROR ROUTINE EXIT ENTERED	An illegal exit() call (COBOL: STOP RUN statement) was recognized while a program unit was running; see also under return code KCRCDC= EXIT.	User error
DB ERRORCODE = TA_CHAIN_RSET	The database connection module supplied return code ' TA_CHAIN_RSET' . If this return code was issued by one of the messages K210, K211 or K216 with XAER_DUPID, the cause could be that the previous application run was terminated abnormally, and left a transaction in the database and that a new KDCFILE was generated in this state. Action in this event: shut down and restart the database.	System error
KDCS CALL IN VORGANG EXIT	An illegal KDCS call was made in the VORGANG exit program.	User error
VORGANGEXIT PROGRAM NOT LOADED	The VORGANG exit program is not loaded.	User error

Error texts in DIAGAREA after SYSTEM PEND ER

Error text	Cause, module	Error type
DATABASE DOWN AT USER DB CALL	The database was no longer connected at the time of the CALL to the database from the application program unit.	User/ system error
ILLEGAL RTCODE FROM DBCON	The database connection module supplied an illegal error code.	System error
NO DB CALL ALLOWED IN SIGN-ON	The sign-on service has issued an illegal DB-USER-CALL	User error
PROGRAM INDEX = 0 INVALID	UTM SVC set an invalid program index.	System error

Error texts in DIAGAREA after SYSTEM PEND ER

Service identification entry (type VGID)

Byte: 32 bit (64 bit)	Meaning
16 (16)	Service ID
17 (17)	Session counter
18-19 (18-19)	Transaction counter in the service
20-23 (24-31)	Transaction counter
24-27 (32-39)	Sum of USED and ERROR for the current TAC
28-31 (40-47)	Length of the global transaction identifier (GTRID) of the XID
32-35 (48-55)	Length of the branch qualifier (BQUAL) of the XID
36-67 (56-87)	The first 32 bytes of the global transaction identifier (GTRID)
68-115 (88-137)	The first 48 bytes of the branch qualifier (BQUAL)
116-117 (138-139)	Index to the program table
118-119 (140-141)	Index of the service exit in the program table
120-127 (142-149)	Name of the transaction code that started the service
128-135 (150-157)	Name of the current transaction code

Structure of the service identification entry

Example

In the first two examples, the hexadecimal part of the DUMP entry was omitted. The fourth example shows how the hexadecimal part is used in evaluating C names and length fields (KCLM, KCLA, etc.).

1. Incorrect operation code in a KDCS call:

```
KDCROOT : UTM Diagarea
     .
     .
     0012 08227F2C 0000      ...          ..INIT..........................
          08227F4C 0020      ...          ............  ..    000P0000
     0013 08227F90 0000      ...          ..MPUTNT....        *f0114  ....
          08227FB0 0020      ...          ............  ..    000P0000
     0014 08227FF4 0000      ...          ..SIGNOB........................
          08228014 0020      ...          ............  ..    000P0000
     0015 08228058 0000      ...          ..XXXXFI....        ............
          08228078 0020      ...          ............  ..    79ZP0000
     0016 082280BC 0000      ...          ..PENDER:   KCRCCC >= 70Z AFTER
          082280DC 0020      ...          UTM SVC  !!
```

2. A PEND ER via signal 11 is triggered by an address error:

```
KDCROOT : UTM Diagarea

   0004 082279B4 0000      ...          ..INIT..t.*.....................
        082279D4 0020      ...          ............  ..    000P0000
        082279F4 0040      ...                     .....J".............
   0005 08227A18 0000      ...          ..PENDER: ERROR ROUTINE XT11 ENT
        08227A38 0020      ...          ERED     !!
        08227A58 0040      ...                        ........POOLT001....
```

3. Evaluation of length fields in the dump

   For the entries in length fields you must take account of the machine-dependent type of representation (little endian or big endian)). With little endian machines, the octet with the lowest address is also the least significant octet; with big endian machines, the octet with the lowest address is the most significant octet.

   The different structures are illustrated in the following table.

Decimal	Hexadecimal INTEL processor (little endian)	Hexadecimal RISC processor (big endian)
228	E400	00E4

   In decimal fields (underscored in this example), the bytes are shown in reverse sequence.

INTEL processor (e.g. Windows-Systems)

```
 0008   B66EC3CC 0000   0700 4B44 43533D3D FF86AC45 53A20D00   ..KDCS==...ES...
           B66EC3DC 0010   494E4954 00000000 00000000 00000000   INIT............
           B66EC3EC 0020   00000000 00000000 00000000 00000000   ................
           B66EC3FC 0030   00000000 00000000 00002020 00002020   ..........  ..
           B66EC40C 0040   20203030 30503030 30302020 20202020     000P0000
           B66EC41C 0050   20202020 20202020 20203D3D 00000000             ==....
           B66EC42C 0060   0A3C49B7 02000000 4C545030 30303031   .<I.....LTP00001
 0009    B66EC454 0000   0800 4B44 43533D3D FF86AC45 67A20D00   ..KDCS==...Eg...
           B66EC464 0010   4D474554 0000 6D01 00000000 00000000   MGET..m.........  [1]
           B66EC474 0020   00002020 20202020 20200000 00000000   ..        ......
           B66EC484 0030   00000000 00000000 00000000 08004F43   ..............OC
           B66EC494 0040   204D3030 30503030 30302020 20202020    M000P0000
           B66EC4A4 0050   20202020 20202020 20203D3D 00000000             ==....
           B66EC4B4 0060   710756B7 02000000 4C545030 30303031   q.V.....LTP00001
```

[1] When the MGET call was issued, KCLA = 365 (hexadecimal: 016D) was entered.

RISC processor (e.g. Solaris, HP-UX, AIX)

```
0008   0004B2EC 0000   0007 4B44 43533D3D 45AC8284 000B2204   ..KDCS==E.....".
          0004B2FC 0010   494E4954 00000000 00000000 00000000   INIT............
          0004B30C 0020   00000000 00000000 00000000 00000000   ................
          0004B31C 0030   00000000 00000000 00002020 00002020   ..........  ..
          0004B32C 0040   20203030 30503030 30302020 20202020     000P0000
          0004B33C 0050   20202020 20202020 20203D3D 00000000             ==....
          0004B34C 0060   FFF25214 00000002 4C545030 30303031   ..R.....LTP00001
0009   0004B374 0000   0008 4B44 43533D3D 45AC8284 000B2230   ..KDCS==E....."0
          0004B384 0010   4D474554 0000 016D 00000000 00000000   MGET...m........  [1]
          0004B394 0020   00002020 20202020 20200000 00000000   ..        ......
          0004B3A4 0030   00000000 00000000 00000000 00084F43   ..............OC
          0004B3B4 0040   204D3030 30503030 30302020 20202020    M000P0000
          0004B3C4 0050   20202020 20202020 20203D3D 00000000             ==....
          0004B3D4 0060   FF1E1315 00000002 4C545030 30303031   ........LTP00001
```

[1] Beim MGET-Aufruf wurde in KCLA = 365 (sedezimal: 016D) angegeben.

3.3.5.4 DB-DIAGAREA

Each time a user call is issued to the DB system, a DB record is written to the DB-DIAGAREA. In the same way as UTM-DIAGAREA, this area is cyclically written with trace records.

Two cycles are separated by a dividing line comprising '=' characters. The record above the line is the most recent and the one below the line the oldest.

Byte: 32-Bit (64-Bit)	**Meaning**
0-1	Counter for the current entries in the DIAGAREAs (UTM and DB)
2-5	"DBCL"= ID for DB record
6-7	Unused at present (preset to '= =')
8-15	Timestamp
16-19 (16-19)	Status of DB transaction[1] prior to DB call, see table on page 111
20-23 (20-23)	Status of DB transaction[1] following DB call, see table on page 111
24 (24)	DB operation code, see table "DB operation codes" on page 112
25 (25)	DB secondary cpcode
26 (26)	DB error code, see table "DB error codes" on page 113
27 (27)	Identifier for DB system (hexadecimal): 02=XA interface
28-31 (28-31)	DB trace information of last XA call: 28: RM number, 29-30: XA function code, 31: XA return code
32-63 (32-63)	DB trace information for XA calls of this DB operation
64-67 (64-67)	Combined status info TAM (further diagnostic information, relevant only for multi-instance mode)
68-71 (68-71)	not used
72-73 (72-73)	Transaction counter within the service
74 (74)	Number of the UTM application run (starts at 1 after KDCDEF generation)

DB record in the DB-DIAGAREA

Byte: 32-Bit (64-Bit)	Meaning
75 (75)	Code for table index, =' T '
76-77 (76-77)	UTM table index that caused the database call
78-79 (78-79)	Action index within this table
80-83 (80-87)	Service counter (unique within the UTM application run)
84-87 (88-95)	Address used internally
88-91 (96-103)	Return address in the program unit. The address points to a location after the CALL-DB-SYSTEM call.
92 (104)	* to draw attention to the end of the diagnostics record

DB record in the DB-DIAGAREA

[1] For operation codes which refer to DB conversations the status of the VGM is shown here.

If the database system is connected to openUTM via the XA interface and a "dynamic XA switch" is incorporated (for Oracle, for example, the module "oraswd"), the following diagnostic record is written when a common transaction signs on or off (32-Bit and 64-Bit coincide):

Byte	Meaning
16-23	Printable "ax_reg" when the transaction signs on Printable "ax_unreg" when the transaction signs off
24-27	Number of the RM instance
28-31	not used
32-47	Printable return code for the call
48	Internal status of the transaction
49	Internal status of the sign-on/sign-off process
50 -51	not used
52-54	Is the call within an openUTM transaction? (printable, "yes" or "no")
55	not used
56	* to draw attention to the end of the diagnostics record

Diagnostic record

Status of the database transaction (contents of bytes 0 through 3 or 4 through 7 of the database record):

Contents (hex)	Meaning of the DB status
04	DB transaction was set to the preliminary end of transaction (PTC)
08	Updates were made within a DB transaction
10	DB transaction has been reset
20	DB transaction was closed
40	The program unit run issued a call CLOSE DB transaction; DB transaction is terminated from viewpoint of program unit run
80	DB transaction is open

Status of the DB transaction

These values can also occur in combination, e.g. "88".

DB operation codes (byte 8 of the DB record)

Contents (hex)	DB op. codes	Meaning
00	STPA	DB connection module is to check DB-specific start parameters. The call takes place at start of UTM application program
04	CONC	Connection setup between UTM task and connection module of DB system. Takes place at start of UTM application program
08	DCON	Connection cleardown between UTM task and DB system. Call takes place in end handling of openUTM application program
10	USRC	Execute DB call of application program (user call).
14	FITA	Terminate DB transaction. openUTM issues this call at the end of a joint DB/DC transaction
18	CATA	Terminate DB transaction abnormally, i.e. the DB transaction is reset to the last synchronization point
1C	BKTA	The call is used with PEND KP or PEND PA/PR with TASK change (due to TAC classes); the UTM tasks breaks off from DB/DC transaction processing.
20	COTA	The call takes place whenever a multi-step transaction is continued, i.e. following a PEND KP or PEND PA/PR with TASK change (due to TAC classes)
24	STAT	Display status of a DB transaction or delete all DB states. The DB system maintains status information on DB transactions as a means for coordinated restarts in openUTM and the DB system.
28	PETA	Request preliminary end of DB transaction
2C	EDVG	End of UTM conversation; DB conversation still open
30	BKVG	Interrupt of an open DB conversation
34	COVG	Continuation of an interrupted DB conversation following BKVG
38	RSVG	Display of reset of a DB conversation to last synchronization point
3C	CNFPTC	DB connection module confirms the status of the precommitted transactions to the DB system via the XA interface and deletes these from the internal list.
40	STRT	DB connection module notifies the DB system of the start of a DB transaction via the XA interface.
44	PEND	DB connection module notifies the DB system of the preliminary end of the transaction via the XA interface and initiates the first commitment phase.

DB operation codes

DB error codes

Contents (hex)	Meaning of the DB error codes
00	Job performed
04	DB transaction had to be reset. openUTM then also resets the UTM transaction. With a DB call (user call), control is restored to the application program unit
08	Causes conversation to abort with PEND ER when openUTM is used
0C	DB system is not available, not activated
10	DB system not available for another reason
14	Job not carried out, retry later!
18	A (possibly) recoverable system error occurred. openUTM creates a task dump and tries to terminate the open DB transaction
1C	An unrecoverable error occurred. Any further work with the DB system is pointless in this DB session
20	DB system detected a user error, e.g. during checking of the DB start parameters
24	DB system detected an interface error

DB error codes

3.3.5.5 ADMI-DIAGAREA

The ADMI-DIAGAREA is a task-specific trace area in KDCROOT. Trace records are written to this area cyclically in the same way as to UTM-DIAGAREA. A record is written to this area each time the administration program interface is called.

The area is large enough for 71 (64-Bit: 62) records, each record being 112 (64-Bit: 128) bytes in length. The area is written cyclically. A boundary consisting of "=" characters separates two cycles. The entry above the boundary is the most recent and the one below it the oldest. One record is written for every administration call.

Records have the following structure:

Byte: 32-Bit (64-Bit)	**Meaning**
0-1 (0-1)	Counter for the current entry in the ADMI-DIAGAREA
2-3 (2-3)	Counter for the corresponding entry (opcode "ADMI") in the UTM-DIAGAREA
4-5 (4-5)	Printable abbreviation for opcode
6-7 (6-7)	Printable abbreviation for object type or subopcode1, depending on the opcode specified
8-11 (8-15)	Address of parameter area
12-15 (16-23)	Address of identification area
16-19 (24-31)	Address of selection area
20-23 (32-39)	Address of data area
24-79 (40-95)	Parameter area
80-111 (96-127)	Name of the administered object in the length specified for the object type (2-32) from the identification area or from the data area

Structure of ADMI-DIAGAREA

The entry counter and the addresses of the areas are also logged before the UTM system code is called. The remaining data is logged after control has returned from the UTM system code and before branching back to the program unit. For this reason, the *parameter area* also contains return values - including the return code. The contents of the *identification*

area are only logged if the area was used during the administration call. The name from the *data area* is only logged for KC_CREATE_OBJECT.
If the administration call is terminated with PEND ER because the address of the *parameter area* cannot be accessed or is not aligned on a word boundary, this log shows the address specified.

The following abbreviations are used:

Abb. opcode	**Opcode**	**Abb. subopcode1/ object type**	**Subopcode1 or object type**
CA	KC_CHANGE_APPLICATION	N S O	KC_NEW KC_SAME KC_OLD
CD	KC_CREATE_DUMP		
CS	KC_CREATE_STATEMENTS		
EN	KC_ENCRYPT	V C D A N	KC_ACTIVATE_KEY KC_CREATE_KEY KC_DELETE_KEY KC_READ_ACTIV_PUBLIC_KEY KC_READ_NEW_PUBLIC_KEY
OI	KC_ONLINE_IMPORT	A	KC_ALL
PE	KC_PTC_TA	R	KC_ROLLBACK
LO	KC_LOCK_MGMT	UF US UA AB AA AP	KC_UNLOCK_USF KC_SIGNOFF_SINGLE KC_SIGNOFF_ALL KC_ABORT_BOUND_SERVICE KC_ABORT_ALL_BOUND_SERVICE KC_ABORT_PTC_SERVICE
UP	KC_UPDATE_IPADDR	A P	KC_ALL KC_PARTNER

Abbreviations for opcode and subopcode1 or object type

Abb. opcode	Opcode	Abb. subopcode1/ object type	Subopcode1 or object type
CR	KC_CREATE_OBJECT		
DL	KC_DELETE_OBJECT		
GT	KC_GET_OBJECT		
MD	KC_MODIFY_OBJECT		
		AB	KC_ABSTRACT_SYNTAX
		AC	KC_ACCESS_POINT
		AP	KC_APPLICATION_CONTEXT
		BC	KC_BCAMAPPL
		CL	KC_TACCLASS
		CN	KC_CLUSTER_NODE
		CC	KC_CLUSTER_PAR
		CO	KC_CON
		CP	KC_CURR_PAR
		DA	KC_DIAG_AND_ACCOUNT
		DB	KC_DB_INFO
		DP	KC_DYN_PAR
		ED	KC_EDIT
		GB	KC_GSSB
		KS	KC_KSET
		LC	KC_LTAC
		LM	KC_LOAD_MODULE
		LP	KC_LPAP
		LS	KC_LSES
		LT	KC_LTERM
		MC	KC_MSG_DST_PAR
		MM	KC_MESSAGE_MODULE
		MP	KC_MAX_PAR
		MX	KC_MUX
		OA	KC_OSI_ASSOCIATION
		OC	KC_OSI_CON
		OL	KC_OSI_LPAP
		PE	KC_PTC
		PO	KC_TPOOL

Abbreviations for opcode and subopcode1 or object type

Abb. opcode	Opcode	Abb. subopcode1/ object type	Subopcode1 or object type
CR DL GT MD (cont.)	KC_CREATE_OBJECT KC_DELETE_OBJECT KC_GET_OBJECT KC_MODIFY_OBJECT (cont.)	PR PT QP QU SI SF SP TA TC TI TR UP US UF U1 U2	KC_PROGRAM KC_PTERM KC_QUEUE_PAR KC_QUEUE KC_SIGNON KC_SFUNC KC_SYSTEM_PAR KC_TASKS_PAR KC_TAC KC_TIMER_PAR KC_TRANSFER-SYNTAX KC_UTMD_PAR KC_USER KC_USER_FIX KC_USER_DYN1 KC_USER_DYN2
SH	KC_SHUTDOWN	K N W G	KC_KILL KC_NORMAL KC_WARN KC_GREATEFUL
SL	KC_SYSLOG	I CS SC SW WB	KC_INFO KC_CHANGE_SIZE KC_SWITCH_AND_CHANGE KC_SWITCH KC_WRITE_BUFFER
SM	KC_SEND_MESSAGE		
SP	KC_SPOOLOUT		
UL	KC_USLOG	SW	KC_SWITCH

Abbreviations for opcode and subopcode1 or object type

3.3.5.6 ADMI-USERAREA

The ADMI-USERAREA is a task-specific trace area in KDCROOT.
This area is used to log the data passed from the program unit to openUTM at the program interface.

Since a large volume of data can be passed, only the data for one call is stored in the area. The area thus comprises a single record and covers 4140 bytes. The area is only written for those calls which pass data to openUTM. The contents of the *data area* or the *selection area* are logged, depending on the area used for the call. If this area is needed for diagnostic purposes, you must make sure that the call is the last call for which data is logged, otherwise the area will be overwritten by a subsequent call.

If a password for a user is passed in the *data area,* it is not logged, but is overwritten with binary zeros.

A record has the following structure:

Byte 32-Bit (64-Bit)	**Meaning**
0-1 (0-1)	Counter for the corresponding entry in ADMI-DIAGAREA
2-3 (2-3)	Irrelevant
4-59 (4-59)	Parameter area
60-4139 (60-4139)	Contents of the data area or the selection area in the length passed

Structure of ADMI-USERAREA

The contents of the *parameter area* and the data passed are logged in the UTM system code. The *parameter area* is logged as it is passed by the program unit, i.e. without return values. The subreturn code is set to zero.

The entry counter is logged after control has returned from the UTM system code and before branching back to the program unit.

3.3.5.7 The communication area KB

The communication area (KB) consists of the KB header, the KB return area and the KB program area of the generated length.

Byte 32-Bit (64-Bit)	**Field names (COBOL and C++) and contents**		
0-7 (0-7)	KCBENID kcuserid	UTM login name	
8-15 (8-15)	KCTACVG kccv_tac	TAC used to start this conversation.	
16-17 (16-17)	KCTAGVG: kccv_day	day	Date of the start of conversation
17-19 (17-19)	KCMONVG kccv_mon	month	
20-21 (20-21)	KCJHRVG kccv_year	year	
22-24 (22-24)	KCTJHVG kccv_doy	working day	
25-26 (25-26)	KCSTDVG kccv_hour	hour	Time of the start of conversation
27-28 (27-28)	KCMINVG kccv_minute	minute	
29-30 (29-30)	KCSEKVG kccv_second	second	
31 (31)	KCKNZVG kccv_status	conversation ID (see appendix or possible entries)	
32-39 (32-39)	KCTACAL kcpr_tac	TAC used to address the program	
40-41 (40-41)	KCTACAL kcpr_hour	hour	Time of the program start
42-43 (42-43)	KCMINAL kcpr_minute	minute	
44-45 (44-45)	KCSEKAL kcpr_second	second	
46 (46)	KCAUSWEIS kccard	card ID: A (ID card in reader) or blanks.	
47 (47)	KCTAIND kctaind	transaction ID: F (first) or N (follow-up transaction)	

KDCS communication area

Byte 32-Bit (64-Bit)	Field names (COBOL and C++) and contents	
48-55 (48-55)	KCLOGTER kclogter	name of the LTERM partner (sender)
56-57 (56-57)	KCTERMN kctermn	type of terminal or printer, see table for PTERM statement
58-59 (58-59)	KCLKBPB kclpa	maximum length of the KB program area as generated
60-61 (60-61)	KCHSTA: kchsta	number of stacked services from the point of view of the current service.
62 (62)	KCDSTA kcdsta	change in the number of stacked services
63 (63)		empty
64 (64)	KCPRIND kcprind	Program ID. "A" = program unit run in an asynchronous service "D" = program unit run in a synchronous service
65 (65)	KCOF1 kcof1	Permitted OSI-TP functions "B" = basic functions "H" = basic functions + handshake functions "C" = basic and commit functions with chained transactions "O" = other combination blank if service was not started via OSI-TP
66 (66)	KCCP kccp	UTM client protocol "0" = asynchronous service "1" = LU6.1 "2" = OSI TP "3" = UPIC "4" = DTP "5" = APPLI "6" = SOCKET
67 (67)	KCTARB kctarb	TA is marked for rollback.
68-71 (68-71)	KCYEARVG kccv_year	4-digit year specification for the start of the service
73-83 (73-83)		empty (FILLER)
84-115 (84-115)	KCRFELD	KB return area

KDCS communication area

Byte 32-Bit (64-Bit)	**Field names (COBOL and C++) and contents**			
84-85 (84-85)	KCRDF kcrdf KCRWVG kcrwvg	return screen function number of waiting services in the case of DGET		
86-87 (86-87)	KCRLM kcrlm	return actual length of the message		
	INFO CK call		MGET call	SIGN ON call
88 (88)	KCRINFCC/ kcrinfcc: KDCS return code of checked KDCS call		KCVGST/ kcpcv_state: service status	KCRSIGN1/kcrsign1: primary code
89 (89)			KCTAST/ kcpta_state: transaction status	KCRSIGN2: secondary code
90-91 (90-91)			empty	
92-94 (92-94)	KCRCCC kcrccc	KCDS return code		
95 (95)	KCRCKZ kcrcid	identifiers: P (production application), T (test application)		
96-99 (96-99)	KCRCDC kcrcdc	internal return code		
100-107 (100-107)	KCRMF kcrfn	return of the abstract syntax		
108-115 (108-115)	KCRPI kcrpi KCRUS kcrus KCRQN kcrqn	service ID return user ID in the case of SIGN ST or creator of message with DGET name assigned by openUTM to the temporary queue in the case of QCRE NN		
ab 116 (ab 116)	KB program area			

KDCS communication area

3.3.6 Memory areas in UTM cluster applications

The memory areas that are global to the cluster consist of the following tables:

UF-HDR	Header of the cluster user file
UF-ENT	Entries in the cluster user file
JF-1-HDR	Header of the first file in the cluster administration journal
JF-1-ENT	Entries in the first file in the cluster administration journal
JF-2-HDR	Header in the second file in the cluster administration journal
JF-2-ENT	Entries in the second file in the cluster administration journal
BUF-SGMT	Segments of buffer management at local node level
GF-HDR	Header of the cluster GSSB file
GF-ENT	Entries in the cluster GSSB file
LF-HDR	Header of the cluster lock file
LF-DLK	Area for deadlock detection in the cluster lock file
LF-ENT	Entries in the cluster lock file
CF-HDR	Header of the cluster configuration file
CF-ENT	Entries in the cluster configuration file
UL-HDR	Header of the cluster ULS file
UL-ENT	Entries in the cluster ULS file

3.3.7 Summary

The dump is concluded with the summary information, a table of contents and a message section that contains the messages that are output while preparing the UTM dump.

The summary information is an extract of all dump information. It contains the data frequently required for diagnosis. This saves you time looking through the dump, especially in the preliminary diagnosis stage and when looking for duplicates.
The summary information contains significant generation parameters of the application, version number of the operating system and openUTM, start parameters of the application, the most important current table entries and the last records of DIAGAREA and TRACE area.
If the dump was produced by KDCUPD, then only the first page of the summary information is written.

Some UTM dumps will not contain all tables, e.g. the PEND ER dump. Some data will therefore be missing in the summary of such dumps during analysis.

The INFO operand serves to control the output of summary information and of the dump.

4 UTM message concept

When a UTM application executes, openUTM generates messages that provide you with information on particular events.

A **UTM message** consists of a **message number**, a fixed **message text** and variable parameters referred to as inserts. These **inserts** are dynamically assigned current values when the respective message is output. Examples of inserts are the name of the application or of the communication partner, counters, error codes, etc.

Each message is associated with an identifier - a **message number**. UTM message numbers always consists of the letter K, P or U followed by a three-digit number, e.g. K008; this is then followed by the actual message text.

UTM messages serve different purposes and can be sent to different recipients (**message destinations**). Within certain limits you can define message destinations for K and P messages yourself on an application-specific basis.
U-messages, on the other hand, are always sent to *stderr* and *stdout*. It is not possible to specify other message destinations in this case.

Some explanations and examples for UTM messages are listed below:

- A user can be informed of a certain event at the terminal by means of a message and can be requested to make an input if necessary.

  Example

  During the sign-on check the password was found to be invalid. openUTM requests that the user repeat the KDCSIGN input at the terminal.

- An event in the UTM application is logged by means of a message in the UTM log file SYSLOG. In this way, data is collected for monitoring the application run and for diagnostic purposes.

  Example

  A message is issued stating that the size of the page pool in KDCFILE has exceeded a certain level.

- If certain UTM messages are generated and you have defined the message destination MSGTAC for them, openUTM calls an event-driven program unit in the application (see section on "MSGTAC event service" in the openUTM manual „Programming Applications with KDCS”). This event service can, for example, issue asynchronous calls to the

administrator using FPUT. This makes it possible to use programmed administration to react to events for which openUTM generates messages.

Example

If the event "service abort" occurs, message K017 or K055 is generated. The MSGTAC service can act on this message by locking the TAC, for example, and sending a message to the administrator.

In order to output a message, openUTM utilizes a message module specific to openUTM. This contains the properties and texts of all UTM messages. When using NLS, the message texts are taken from special message catalogs, see section “Message catalog source file for NLS” on page 129.

You can, to a certain extent, structure the output of UTM messages for specific applications. You can, for example, change the message destinations (recipients) of K and P messages and translate the message texts into a different language (not possible on Windows systems), or change the destination (recipient) for K and P messages. You will find more information in section “Modifying message output” on page 136.

You can also use NLS message catalogs to structure the message output. The message catalogs contain the message texts for a particular language and a character set (code set). This brings the following benefits:

- If message catalogs are used, UTM messages appear for the user in the selected language (assuming appropriate NLS message catalogs exist), i.e. an application can be operated in more than one language according to requirements.
- The language of a user need not be generated since the message texts are automatically selected in the correct language at run time. The %LANG% environment variable of the user is evaluated by the dialog terminal process of this user.

Message catalogs in German and English are supplied with openUTM as standard.

You can include message inserts in the message text or remove inserts from it.

4.1 Message module, message definition file

The message definition file *utm-path*/msgdescription and the following two standard message modules are supplied with openUTM.

kcsmsgs.o (K an P messages)
kcxmsgs.o (U messages)

These modules are contained in the libraries *utmpath*/sys/lib/libwork.*a* and *utmpath*/sys/lib/libwork.*so* respectively under Unix systems.

Under Windows systems, the modules are contained in the *utmpath*\sys\libwork.lib library.

The standard message modules contain the English message texts and the default settings for the message destinations (e.g. terminal or SYSLOG file). The message definition file is used as an input file if the user wishes to change the UTM messages. It contains message texts in both English and German and the framework definition of the messages (structure of the messages).

In the following description, the common term “standard message module” will be used for both standard message modules.

The message definition file can be expanded with message texts in other languages (not under Windows systems). Users can translate the message texts and enter them in the message definition file with the tool KDCMTXT (see page 137ff). From the message definition file users can use the tool KDCMMOD (see page 144ff) to create your own message module.

CAUTION!
You may only use the openUTM tools KDCMTXT and KDCMMOD to process the message definition file msgdescription! Any other write access, such as with an editor will **destroy** this file!

When a message is output, openUTM accesses the UTM message module. Among other things this contains the following for each message:

– the message number Knnn, Pnnn or Unnn
– the message text
– the message destinations
– the inserts

A particular type of message output is specified in the standard message module. Users who wish to change these must generate their own message module.
If there are no NLS message catalogs for openUTM and no user-specific message module, openUTM generates the messages from the standard message module.
The standard message module must be linked into **each** UTM application program, i.e. even if you are using NLS message catalogs or your own message module.

4.2 NLS message catalogs

With NLS message catalogs you can output the message texts in different languages irrespective of the linked message module. The message texts are then taken from the appropriate message catalog by the program at run time on a language-specific basis. The language is defined in Unix systems and Windows systems by evaluating the user's %LANG% environment variable.

NLS message catalogs contain only the message texts and no information on message destinations or message attributes. For evaluating the inserts, message destinations and message attributes and in response to an error, openUTM uses the standard message module or a user-defined message module, if one exists.

A UTM application can also be run without NLS message catalogs. For these reasons the UTM message module (supplied in the library *utmpath*`/sys/libwork.*` or, under Windows systems, in the library *utmpath*\`sys\libwork.lib`) must be linked into the application. In addition, a user-defined, modified message module can be linked in, as before.

The NLS message catalogs can be ignored when the application is generated and when the application program is linked.

NLS standard message catalogs in German and English are supplied with openUTM. Under Unix systems, they are stored in the *utm-path*`/nls/msg/`*lang* directories, where *lang* is the language identifier. Under Windows systems, they are stored in the *utmpath*\nls\msg*lang* directories.

The message catalogs under `.../lang` have the following names:

Unix systems	Windows systems	
`utmsys.cat`	`utmsys.dll`	K and P messages of the system
`utmutil.cat`	`utmutil.dll`	K messages of the utility programs
`utmxprog.cat`	`utmxprog.dll`	U messages

Application-specific message catalogs, i.e. message catalogs under *filebase,* are currently evaluated only for messages of the SYS function unit (message catalog name `utmsys.cat or utmsys.dll`).

4.2.1 Message catalog source file for NLS

Application-specific message catalogs can be generated with the aid of the KDCMTXT and KDCMMOD tools (see page 137ff and page 144ff). KDCMTXT stores the message texts for each language and function unit in a source file *FU_LN*`.msg`, where *FU* is the selected function unit and *LN* is the language identifier used in the FU statement.

The specific message catalogs (i.e. those modified by the operator of an application) must be stored for the SYS function unit under

utmpath`/nls/msg/`*lang*

with the name `utmsys.cat` where *lang* is a language identifier, as in the case of the standard message catalogs.

For the position of the inserts, the message texts of the NLS source files contain "pseudo printf control strings" `%`*nn*`$`, such as `%05$`, where the two-digit number *nn* is the number of the insert of the associated message in the message definition (general definition), in this case the fifth insert of the message.

4.3 Message destinations

openUTM generates messages while an application is active. K messages and P messages are sent to one or more of the following destinations:

STDOUT Output to *stdout*.

STDERR Output to *stderr*.

STATION Clients connected by means of a PTERM or a TPOOL with PTYPE=TTY.

SYSLINE System line of the data display terminal; user screen formats are maintained. If a message is to be output to the system line, both SYSLINE and STATION must be specified as destinations.

CONSOLE Console of the system operator; the application name is also output.

PARTNER Clients connected by means of a PTERM or a TPOOL with PTYPE=APPLI or SOCKET.

SYSLOG System log file SYSLOG (see page 155).

MSGTAC MSGTAC service (see control statement TAC in the openUTM manual “Generating Applications” and in the section "MSGTAC event service" of the openUTM manual „Programming Applications with KDCS”).

USER-DEST-1 to USER-DEST-4
User-specific message destinations to which you can assign a USER queue, a TAC queue, an asynchronous TAC or an LTERM partner as concrete message destinations at generation time.

Either cyclically or when requested by the user, WinAdmin and WebAdmin can retrieve messages from these UTM queues, output these in lists and save them in the configuration database if required.

The assignment between USER-DEST-1..4 and the concrete destination is specified by means of the KDCDEF statement MSG-DEST.

U-messages, are always sent to *stderr* and *stdout*. Message destinations can therefore not be changed.

4.3.1 Output format of the messages

Depending on the message destination, openUTM outputs the messages in different formats:

- to SYSLOG and MSGTAC:
  the message header and the current values of the parameters, as described in section “Structure of UTM system messages” on page 158.
- to CONSOLE:
  the application name, the message number and the message text with the text parameters.
- to USER-DEST-1...USER-DEST-4:
  – the message header and the current value of the inserts as described in the section “Structure of UTM system messages” on page 158, if USER-DEST-*n* is generated with MSG-FORMAT=FILE
  – the date/time followed by the message number, message text and inserts if USER-DEST-*n* is generated with MSG-FORMAT=PRINT
- to all other destinations:
  the message number and the message text with any text parameters contained in the message text.

In certain error situations during an application run (abnormal termination of an application, program or process), openUTM may only output a message to STDERR, STDOUT or CONSOLE, even though the message was meant for other destinations (e.g. SYSLOG). This is done to prevent any further errors that might result.

For the same reason, in certain error situations openUTM does not take the message text from the message module that can be modified by the user, but from the standard message module of the UTM system code.

Header for messages to STDERR / STDOUT

When outputting messages to the message destinations SSTDERR and STDOUT, by default openUTM precedes the messages with a header containing the date and time.

This header also contains the PID of the process that generated the message. One benefit of this information is that it allows errors to be diagnosed more easily.

The header has the following format:

pid yyyy-mm-dd hh:mm:ss

Messages K038 and K044 are always output without time and date. You can use the environment variable UTM_MSG_DATE=NO to suppress the output of date and time for all other messages as well. Output of the PID can be suppressed with the environment variable UTM_MSG_PID=NO.

4.3.2 UTM messages to the console

openUTM enters an application name in addition to the message text in messages that are output on the console (CONSOLE message destination).

The system administrator must ensure under Unix systems that the corresponding user ID (USER) is actually allowed to write to the console. When necessary, the system administrator must change the access rights for the console (`chmod` command on `/dev/console`).

Under Windows systems, a `console.txt` file is created in the *filebase* directory to which messages are written that have CONSOLE as their destination.

4.3.3 UTM messages to a TS application

If UTM messages are intended for a transport system application of the type PTYPE=APPLI or SOCKET (message destination = PARTNER), you must ensure that the application recognizes the messages and responds accordingly. If it does not, it can happen, for example, when two UTM applications are linked as TS applications, that the applications keep sending each other messages such as
`K009 Transaction code K009 is invalid`.

USP header in the case of UTM messages to a socket application

For UTM messages to a socket application (PTYPE=SOCKET, message destination = PARTNER), you can specify at generation whether openUTM is to precede the message with a USP header (open**U**TM **S**ocket **P**rotocol header). One of the purposes of the USP header is to output the length of the received messages to the socket partner.

To do this, specify the following at KDCDEF generation in the USP-HDR= operand of the PTERM or TPOOL statement:

```
USP-HDR = MSG or USP-HDR = ALL
```

If you generate `USP-HDR = NO` (default setting), a USP header is not created.

You will find a description of the USP header in the openUTM manual „Programming Applications with KDCS”.

4.3.4 UTM messages to user-specific message destinations

When a message occurs for which USER-DEST-1 ...USER-DEST-4 has been defined as the message destination, UTM creates an asynchronous job to this message destination internally. This asynchronous job is assigned the user KDCMSGUS and the LTERM partner KDCMSGLT as the originator. If the asynchronous job is rejected because, for example, the message destination is locked (STATUS=OFF), the message is lost to the message destination. If a message is created for this message destination again, UTM tries to create an asynchronous job again.

If an asynchronous TAC is generated as the message destination, UTM starts the program assigned to the TAC every time the relevant message is created (only one message can be read with FGET in a program run).

4.3.5 UTM messages to MSGTAC

MSGTAC is a special asynchronous program unit, that users can program themselves, (see the openUTM manual „Programming Applications with KDCS”).

If there is an MSGTAC program, and a message occurs for which MSGTAC was defined as the message destination, the asynchronous service MSGTAC is started. MSGTAC reads the message and can respond to it appropriately. The MSGTAC program can read a number of messages in a single program unit run.

The MSGTAC program unit runs under the internal UTM user ID KDCMSGUS with KSET=MASTER and PERMIT=ADMIN. It must be defined in a TAC statement with `TAC KDCMSGTC,PROGRAM=... .`

4.4 Message editing by openUTM

Initialization

To ensure that U, K and P messages are output in the same language, openUTM or the dialog terminal process first initializes the standard message catalogs. Application-specific message catalogs for a particular language are only used if the standard message catalogs are also available for the relevant language and if read access is permitted:

– `utmsys.cat` (Unix system) or `utmsys.dll` (Windows system)
– `utmxprog.cat` (Unix system) or `utmxprog.dll` (Windows system)

If, for example, only the standard message catalog for the K messages (`utmsys.cat` or `utmsys.dll`) exists for a language, then NLS message processing is not initialized.

The linked message catalogs are used in the following situations:

– if you do not use your own NLS catalogs
– for selecting the message destinations
– as a fallback in the event of an error

If there is no message catalog for the $LANG variable of a user, the message module linked into the application is used, as under the previous system.

Evaluation of message texts under Unix systems

Where message catalogs are in use, a UTM application always searches for message texts in the following order once it has started:

1. First an application searches in the *filebase*`/nls/msg/`*lang* directory for an NLS message catalog (which may have been modified by the application operator) with the name `utmsys.cat`, where *lang* corresponds to the contents of the $LANG variable of the relevant user. If there is no *lang* directory under *filebase*`/nls/msg`, then *lang* is shortened to the language component of the $LANG variable and the resultant name is used as the current catalog name. For example, `En_US.ASCII` is shortened to `En` if no directory called `../En_US.ASCII` is found.
2. If there is no message catalog in directory *filebase*`/nls/msg/`*lang*, the application searches in the UTM directory *utmpath* under `nls/msg/`*lang* for a message catalog. In the case of an error the value of *lang* is reduced as above to the language component. This directory contains NLS message catalogs which can be used on a shared basis by all the applications on the computer, i.e. if a message catalog under this directory is modified, then this change will apply to all the applications on the computer.
3. If the application cannot find a message catalog under *utm-path*`/nls/msg/`*lang*, the UTM application reverts to the linked standard message module or to an additionally linked application-specific message module.

Evaluation of message texts under Windows systems

Only message destinations can be changed in Windows systems (using the tool KDCMMOD). Changes cannot be made to the message texts. In other words, only the message catalogs containing German and English messages shipped with the product can be used. These message catalogs are supplied as DLLs in the `%UTMPATH%\nls\msg\`*lang* directories, where *lang* is the value of the %LANG% environment variable.

You can set values such as “De” for German and “En” for English for %LANG%. Other values are treated as if they were “En”.

openUTM loads either the German or English message catalog at startup, depending on the value set for %LANG%.

Editing the messages

If there are no NLS catalogs or if there is no agreement between the value of the %LANG% variable and the existing NLS catalogs, the linked standard message module is used. Since the dialog terminal process edits the messages of the transaction monitor itself, each user has his or her "own message system". If, for example, a user has entered LANG=**Fr**ench and there are only English and German NLS catalogs in the system and no French NLS catalogs, the standard messages of the linked message module are output for this user. For all other users the messages are edited in accordance with the entries in the NLS catalogs.

Message output exceptions

When UTM messages are output during operation, openUTM takes the message text and the current message destinations from the standard message module. However, the following exceptions to this exist:

- For some messages output by ROOT modules (e.g. for K078), English message texts are programmed in, and the output for these messages thus cannot be changed.
- In the start and end handling of the tasks, for technical reasons openUTM cannot access the application’s message module. In these cases, openUTM takes the message text and the message destinations from the German message module.

4.5 Modifying message output

The message definition file defines the way in which openUTM issues standard messages, i.e. which text is used, which destination it is sent to, etc., together with constraints as to the extent to which they can be modified. If you wish to change these default characteristics, you must generate your own message module and link it to the UTM application. You can modify the K and P messages of the work processes and the U messages. The following options are available:

- Adding or omitting message destinations (with KDCMMOD) as well as, for example, logging additional messages to STDERR or entry of the destination MSGTAC as a prerequisite for the use of an MSGTAC service.
- Translating message texts into a different language and incorporating them in the message definition file with the KDCMTXT tool (page 137).
- Changing message texts with the KDCMMOD tool (page 137), such as adding or removing (permitted) inserts to/from a message text, converting texts into lowercase/uppercase letters, etc.

  Note
  Only the message destinations can be changed in Windows systems. No changes can be made to the message texts (page 135).
- Defining text constants. These can be used in many message texts. Control characters are also included here.

Individual modification of message output applies only to the corresponding UTM application. It has no effect on other UTM applications on the same computer and requires no changes to be made in the operating system.

You may only make modifications with the aid of the KDCMTXT and KDCMMOD tools.

The KDCMTXT tool should be used when translating all message texts into another language. If only a few texts are modified for a language contained in the message definition file, the KDCMMOD tool can also be used.

Generation of a user-specific message module

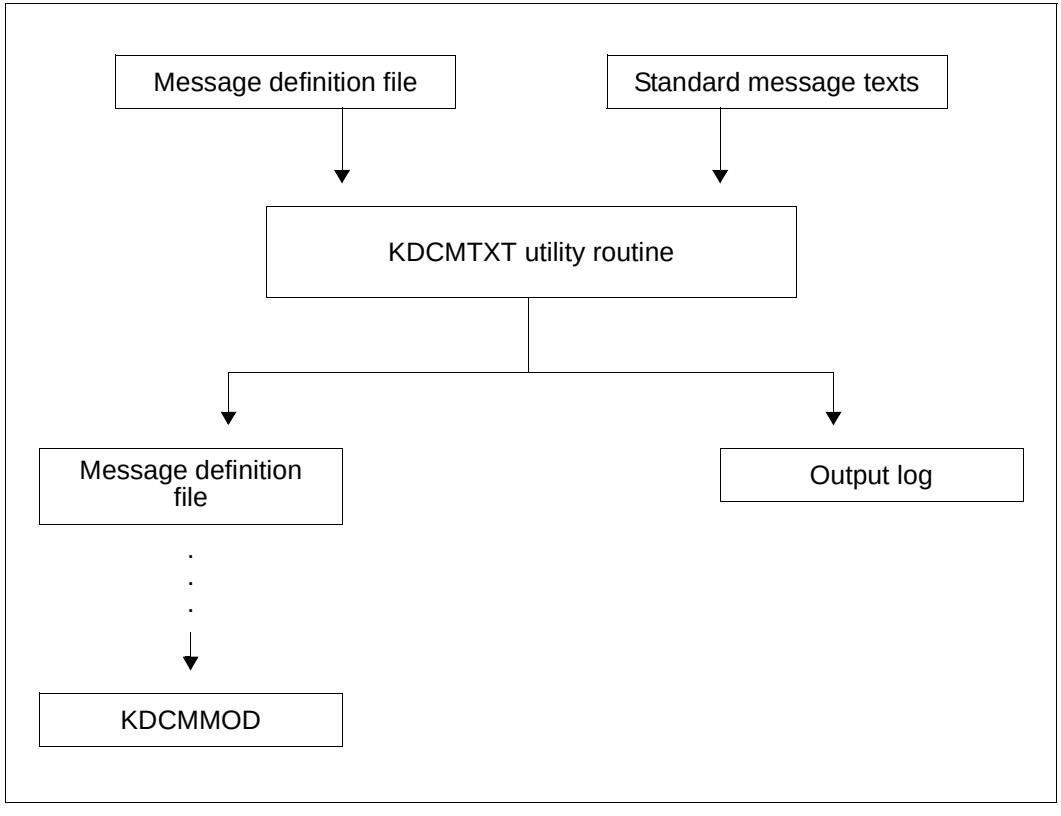
A user-own message module must be generated with the KDCMMOD tool, see page 144. From the message definition file and a user-generated modification definition, KDCMMOD creates a C source program and an NLS source file. Compilation of this C program results in the application-specific message module which is linked to the program units of an application instead of the standard message module.

4.5.1 Messages in other languages - the KDCMTXT tool

Message texts are output in English as standard (German message texts are generated using KDCMMOD). The KDCMTXT utility is a tool for adding message texts in other languages to the message definition file *utm-path*`/msgdescription`.

You need KDCMTXT if the standard message texts are to be output in a language other than German or English. You use KDCMTXT to insert your translations in the message definition file and create an NLS source file, from which you can generate an NLS message catalog by using `gencat`. From the extended message definition file, you can generate a C source program for a message module with KDCMMOD.

The following diagram illustrates the inputs and outputs for KDCMTXT.



Creating messages in other languages with KDCMTXT

4.5.1.1 Calling KDCMTXT

The KDCMTXT tool is called with:

utm-path`/ex/kdcmtxt`

4.5.1.2 Control statements for KDCMTXT

KDCMTXT recognizes the following control statements:

OPTION	Specify name of message definition file
FU	Define function unit and national language
MSGBASE	Select message group for FU SYS: UTM (K messages) or XAPTP (P messages)
STDTXT	Define start of standard message text
ENDTXT	Define end of standard message text
END	Define end of input of control statements

The following rules apply to the input of statements:

– Comment lines are marked by means of an asterisk (*) in column 1.

– If a line ends with a comma, KDCMTXT then interprets the following line as a continuation line of the statement.

It is expedient to write the control statements for KDCMTXT to a file which you then assign as the standard input when invoking the tool. A file named *utmpath*`/mtxtin` is supplied with openUTM. This file contains the English and German standard message texts in the syntax of the KDCMTXT control statements.

The file `mtxtin` can be used as a template for creating a customized input file.

Note

When defining message texts, an insert allowed for the corresponding message may only appear once.

OPTION statement

If specified at all, OPTION must be the first control statement. It is used to specify the name of the message definition file that is to be modified with KDCMTXT.

Operation	Operands
OPTION	MSGFILE=filename

filename is the name of the message definition file. In fully qualified form, the name may be up to 54 characters long. In partially qualified form, it may be up to 14 characters long.

If this statement is omitted, KDCMTXT uses the standard message definition file *utm-path*/msgdescription supplied with openUTM.

FU statement

This statement can be used to define the function unit and the natural language.

Operation	Operands
FU	[function-unit][,][LANG=language-id]

function-unit Function unit for which the new natural language is to be entered. The following values are permitted:

SYS: K messages (K001-K399) or P messages (P001 - P049)

XPROG: U messages (U101-U550)

Default: SYS

LANG=language-id

Identifier for the natural language, up to 3 characters in length. The identifier can be specified as you like, however it must be unique within the message definition file.

As language-id you can use for example the motor vehicle country symbols or the language identifiers specified in ISO IS/R639:

Country symbol	ISO IS/R639	Language
DK	Da	Danish
D	De	German
GB	En	English
E	Es	Spanish
FI	Fi	Finnish
F	Fr	French
GR	Gr	Greek
I	It	Italian
NL	Nl	Dutch
N	No	Norwegian
P	Pt	Portuguese
S	Sv	Swedish

Default: GB

It is only necessary to specify the comma if both *function-unit* and LANG=*language-id* are specified., e.g. `FU XPROG,LANG=DK.`

For any given functional unit, all the message texts for one language must be specified so that the message texts for this language and functional unit are incorporated in the message definition file.

After the FU statement has been processed, the MSGBASE UTM statement is executed implicitly.

CAUTION!
The message definition file supplied already contains the German and English message texts with language identifiers D and GB. If one of these identifiers is specified, KDCMTXT overwrites the corresponding standard message texts in the message definition file with the new message texts.

MSGBASE statement (only for FU SYS)

openUTM makes a distinction between two message groups within FU SYS: Messages of the UTM group, which start with the letter K, and messages of the XAPTP group, which start with the letter P.

The MSGBASE statement allows you to select the message group to which subsequent STDTXT statements are to apply until the next MSGBASE or FU statement is issued.

Operation	Operands
MSGBASE	{ UTM | XAPTP }

UTM Select the UTM message group (K messages).

XAPTP Select the XAPTP message group (P messages).

STDTXT and ENDTXT statements

The control statement STDTXT introduces the definition of the standard text of a message. The subsequent input lines describe the standard text.

ENDTXT terminates the definition of the message text.

Operation	Operands
STDTXT	msg-no text
ENDTXT	

msg-no — Message number for which the standard text is intended.

msg-no specifies the K or P message that is to be modified. The message number must lie within the message number range of the specified functional unit. The message number must be specified as three digits **without** the prefixed **K** or **P** .

Mandatory operand.

text — The message text must be specified here; the syntax is the same as with the KDCMMOD utility (see page 151). The same length restrictions similarly apply.

The control statements STDTXT and ENDTXT must each begin in a new line.
The STDTXT / ENDTXT statements for a single functional unit and a single language must all be located after the FU statement for this functional unit and must precede the next FU statement.

KDCMTXT will only incorporate the message texts for an additional language in the message definition file if the input data contain all the message texts for this language and if all message texts are error-free.

If the message texts for a language are already present in the message definition file, KDCMTXT will then also accept individual messages. In this case, the NLS source file contains only the modified messages.

Note

Use the KDCMMOD tool to change message texts.

END statement

This statement terminates the input of control statements.

4.5.1.3 KDCMTXT log

After the KDCMTXT tool has processed all the modifications and created a new message definition file, it outputs an edited list of all messages to *stdout* and *stderr* as a log (output log). The list contains the message texts edited in the form in which they would be output to a data display terminal. Here, the inserts are filled to the appropriate length as follows, depending on type:

Type	Fill character
CHAR	@
INT	#
HEXA	X

4.5.2 Generating a user-specific message module with KDCMMOD

If you choose not to use the supplied standard message module containing English message texts, openUTM allows you to create your own customized message module. The information in the standard message module can thus be modified within certain limits simply by describing the changes with respect to the standard module.

The K messages K001 - K149 and the P and U messages can be modified.

Using your own message module for K and P messages you can

– change the destinations of individual messages,
– output messages at the console,
– use a MSGTAC routine,
– output user-written message texts, and
– change the message texts, e.g. by translating them into another language.

Note
Only the message destinations can be changed in Windows systems. No changes can be made to the message texts (page 135).

For the U messages you can

– output user-written message texts, and
– change the message texts, e.g. by translating them into another language.

With the KDCMMOD tool you can create your own message texts, and modify inserts and message destinations. Each insert allowed may only appear once in the message, however. From the message definition file and your entries, KDCMMOD creates a C source file and an NLS source file with the modified texts for the appropriate NLS catalog. By compiling the C source file, you create a new user-specific message module. This can then be linked and used with the application program in addition to the standard message module.

The KDCMMOD tool stores the generated C source program in the file *module-name*`.c` in the current directory, where *module-name* denotes the name of the message module specified in the GEN statement.

Since message editing is performed by openUTM primarily via the NLS catalogs, a text modification in the application-specific message module has no effect (except in the event of an error) on the execution of a UTM application. In this case, i.e. if message texts are to be modified on an application-specific basis, an NLS catalog `utmsys.cat` must be made available for the K messages under the directory *filebase*`/nls/msg/`*lang*. This NLS catalog need only contain the modified messages.

If only message texts but not the message destinations are to be modified for an application, the C source file created by KDCMMOD can be deleted.

Note

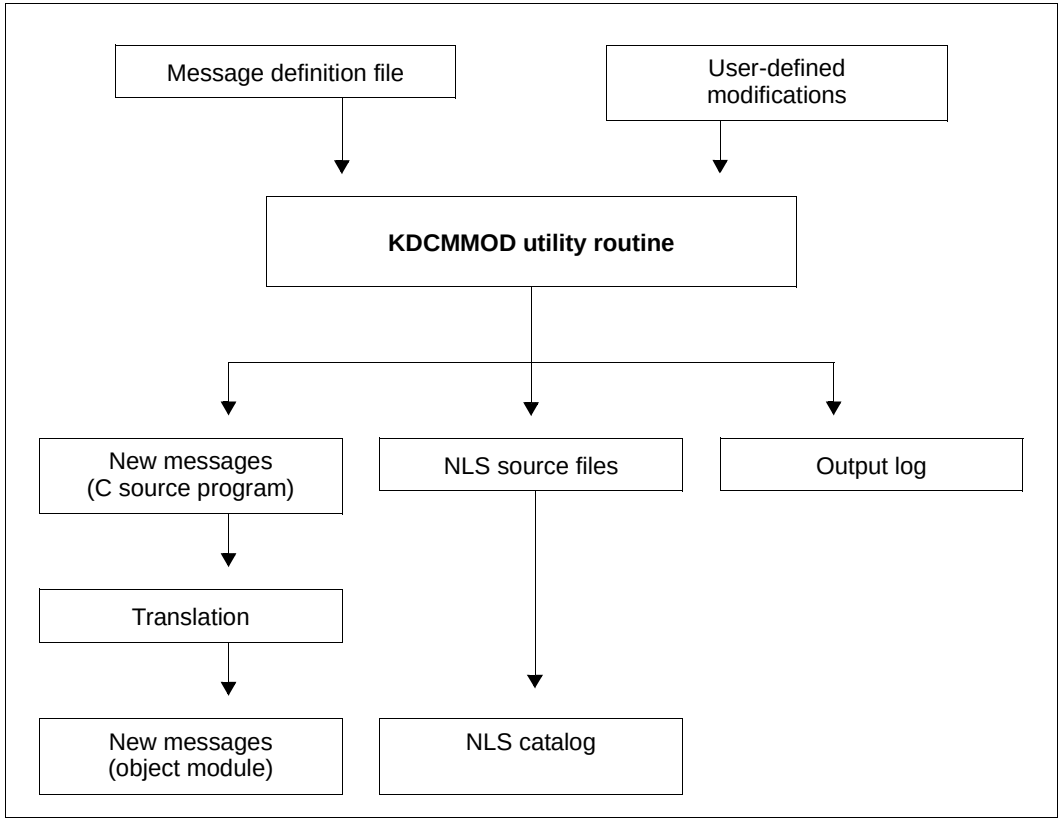
To avoid unnecessary catalog accesses, an application-specific NLS catalog should contain all the messages. You should therefore copy the standard NLS catalog of the language you require to *filebase*/nls/msg/*lang*, use the KDCMMOD tool to create an NLS source file with the modified message texts, and merge this file into the application-specific NLS catalog.

There is considerable scope for modifying the message texts but the length restrictions must be observed; see page 151.

The available inserts are listed on page 379ff. It is possible to remove inserts from the message text that were specified in the standard text or include inserts in a message that were not specified in the standard text. In addition, it is possible to alter the sequence of the inserts within the text. The message numbers can also be moved to another place in the message text or even removed altogether. However, this should only be done in certain exceptional cases since it can make diagnoses on the basis of the messages more difficult.

The message destinations can be modified within certain limits (see section “Destinations of UTM messages” on page 393ff). If the prescribed or prohibited destinations are not observed, KDCMMOD error messages result. You should also note that different destinations may be required or prohibited for each message.

KDCMMOD inputs and outputs



Sequence for creating a user-specific message module

Once the KDCMMOD tool has processed all the changes and set up a new message file, it writes an edited log of all messages to *stdout*. If you have modified messages, an NLS source file is also created. In addition, the program creates an overview of the message definitions (insert list, destinations, compression) derived from the framework definitions and modifications.
In the output list, the defined standard texts are edited in the form in which they were output to a terminal, with the inserts being filled as follows to the appropriate length depending on the type:

CHAR	with	@ characters
INT	with	# characters
HEXA	with	X characters

The messages for KDCMMOD will be found on page 347ff.

4.5.2.1 Calling KDCMMOD

The KDCMMOD tool is called with:

utmpath`/ex/kdcmmod`

4.5.2.2 Control statements for KDCMMOD

KDCMMOD recognizes the following control statements:

CONSTANT	Define constants
END	Terminate input of control statements
ENDMSG	Terminate message definition
GEN	Generate messages for a function unit
MODMSG	Modify messages
MSGBASE	Select message group: UTM (K messages) or XAPTP (P messages)
OPTION	Specify name of the message definition file

The following should be observed with regard to the sequence of KDCMMOD control statements:

1. The KDCMMOD tool reads the statements of the modification description from stdin. Individual lines are read, whereby
   - comment lines are marked by an asterisk ("*") in the first column, and,
   - for lines that have a comma as the last character, the next line is interpreted as a continuation line.

   All other lines are analyzed individually by the program.
2. If used, the OPTION statement must be the first control statement.
3. The CONSTANT statement must be in the input file before a text definition can refer to it.
4. The GEN statement must appear in the input file before any associated MODMSG/ENDMSG control statements.
5. The END statement must be at the end of the input file.

CONSTANT statement

The CONSTANT statement defines a constant which then can be used in the in the text definitions of messages.

Operation	Operands
CONSTANT	constant-name,constant-value

constant-name
: Denotes the name of a constant and can be up to 8 characters long. If a constant with the same name already exists, the CONSTANT statement is rejected with an error message.

  Mandatory operand.

constant-value
: Assigns a value to a constant. The value must be specified either in the hexadecimal format (X'.....') or in the printable format '...'. The maximum length is 50 characters.

  Mandatory operand.

The constants for new line (NL) and new page (NP) are already contained in the message definition file.

END statement

The statements for KDCMMOD are terminated with the END statement. END must be entered as the last statement.

Operation	Operands
END	

Without operands.

ENDMSG statement

A message text definition must be terminated with the ENDMSG statement.

Operation	Operands
ENDMSG	

Without operands.

The ENDMSG statement must always be on a separate line.

GEN statement

The GEN statement specifies the functional unit for which the messages are to be generated as well as the language in which they are to be generated. The GEN statement may be given only **once** per program run and must appear before any MODMSG statements.

Operation	Operands
GEN	[function-unit] [,LANG=language-id] ,MODULE=object-module-name

function-unit Name of the function unit for which the messages are to be generated. Possible function units are:

SYS for K and P messages

XPROG for U messages

Standard: SYS

LANG=language-id

notes the natural language for which the message texts are to be generated. A language identifier must be specified for which message texts are contained in the message file.

Default: GB

MODULE=object-module-name

Denotes the name of the message module. It corresponds to the MODULE name in the MESSAGE statement of the KDCDEF tool (see openUTM manual “Generating Applications”).

Mandatory operand.

MODMSG statement

You use the MODMSG statement to modify a message in the functional unit that was specified in the GEN statement. Changes to a message are initiated with the MODMSG statement and concluded with the ENDMSG statement..

Operation	Operands
MODMSG	msg-no [,BEL={**Y**ES | **N**O}] [1] [,COMPRESSION={**Y**ES | **N**O}] [,CONSOLE={**Y**ES | **N**O}] [,MSGTAC= {**Y**ES | **N**O}] [,PARTNER={**Y**ES | **N**O}] [,STATION={**Y**ES | **N**O}] [,SYSLINE={**Y**ES | **N**O}] [,SYSLOG= {**Y**ES | **N**O}] [,STDOUT= {**Y**ES | **N**O}] [2] [,STDERR= {**Y**ES | **N**O}] [2] [,USER-DEST-1= {**Y**ES | **N**O}] [,USER-DEST-2= {**Y**ES | **N**O}] [,USER-DEST-3= {**Y**ES | **N**O}] [,USER-DEST-4= {**Y**ES | **N**O}] [text] [3]

[1] The operands of the MODMSG statement must be separated by commas. If there are continuation lines in a MODMSG statement, the comma must always appear as the last character in the preceding line (as the continuation character).

[2] The synonym SYSLST is also permitted for STDOUT, as is the synonym SYSOUT for STDERR.

[3] The line prior to [text] must not be terminated by a comma; see example on page 154.

msg-no — Indicates which K message is to be modified. The message number must be in the message number range of the function unit specified in the GEN command. The message number must be given **without** the **K** or **P** prefix.

Mandatory operand.

BEL= — Indicates whether an audible signal is given when the message is output to the destination STATION or SYSLINE.

- YES — an acoustic signal is triggered.
- NO — no acoustic signal is triggered.

Default: NO

COMPRESSION=

YES Superfluous blanks are removed from the message.

NO Superfluous blanks remain in the message.

The default value varies for the different messages. To find out the value set as the default for the individual messages, refer to section “Destinations of UTM messages” on page 393.

Default: framework definition

Message destinations CONSOLE etc.

denotes the message destination to which the message is to be sent (YES) or not sent (NO). Only destinations that are ’permitted’ (+) or declared ’standard’ (S) in the framework definitions may be specified. The basic definitions apply to all message destinations that are not specified in the MODMSG command.

To assign the user-specific message destinations USER-DEST-1,..., USER-DEST-4 to the concrete message destination, you have to use the KDCDEF statement MSG-DEST.

You will find detailed descriptions of the various message destinations on page 127.

text In the MODMSG statement, a new message text can be defined. If no new text is defined, then the text as described in the chapter “UTM messages” on page 159 applies.

A new message text is defined in one or more lines. It consists of a series of text elements separated either by commas or end-of-line characters.

The message text should not contain any ASCII’00’ characters.

Text = text-element,text-element,.....
text-element,.....

The first text element must always occur in a new line.

Length restrictions

The message text, including the message number and any inserts which may be present, must not exceed 512 characters in length. For the lengths of the inserts, refer to the table on page 379. If the text is longer than 512 characters, KDCMMOD rejects it with message K686. The message definition is not then updated.

Message texts having the message destination SYSLINE (=systemline) must not exceed 40 characters in length. If the text is longer than 40 characters, KDCMMOD gives a warning with message K687. On output, longer messages are limited to 40 characters and the excess is truncated.

The following text elements can be used:

Literal
A literal is a character string enclosed in single quotes. Double quotes in a literal are used to designate a single quote in the literal.

Insert
The text element insert is the name of a field (insert) in the message text that is preceded by the "&" character. Only field names (inserts) that are specified in the framework definition of the corresponding message may be used in the text.

Insert fields are areas in the message texts into which current values are inserted before the message is output.

Constant
The text element constant is the name of a constant which is preceded by the "#" character.

Built-in function
The text element built-in function is a function that is executed at the point in question. The following built-in function is available:

MSGID
To simplify diagnosis, all message texts should start with MSGID.

This function returns the character string for the message identifier which corresponds to the message number.

The message text should not contain ASCII’00’ characters.

Each MODMSG command must be terminated with an ENDMSG statement.

MSGBASE statement (only for FU SYS)

openUTM makes a distinction between two message groups within FU SYS: messages of the UTM group, which start with the letter K, and messages of the XAPTP group, which start with the letter P.

The MSGBASE statement allows you to select the message group to which subsequent MODMSG statements are to apply within the SYS functional unit.

Operation	Operands
MSGBASE	{ UTM | XAPTP }

UTM Default; select the UTM message group (K messages).

XAPTP Select the XAPTP message group (P messages).

OPTION statement

The OPTION statement is used to define the name of the message definition file which the KDCMMOD tool is to process.

Operation	Operands
OPTION	MSGFILE=filename

filename Name of the message definition file to be processed. In fully qualified form, the name may be up to 54 characters long. In partially qualified form, it may be up to 14 characters long.

If the OPTION statement is omitted, the file with the name `msgdescription` in the UTM directory in used.

Example

The user wishes to assign the destination MSGTAC for message K006 or change the message text for K008 and otherwise retain the unchanged messages with standard English texts.

First, a file `input` is created which contains the following KDCMMOD control statements:

```
GEN  SYS,LANG=GB,MODULE=kcsmsgsd
MODMSG 006, MSGTAC=YES
ENDMSG
MODMSG 008
'Hello ',&USER,', What do you want?'
ENDMSG
END
```

The tool KDCMMOD is then called:

utmpath`/ex/kdcmmod < input`

As a result, the C source program `mymsgsd.c` is generated which must be compiled to form an object module using the C compiler. When utmwork is linked (see the openUTM manual “Generating Applications”), this object module must also be incorporated in the standard message module.

When you transfer modifications to the message system, please note:

If you are working with NLS catalogs and have used KDCMMOD to modify message texts only (not message destinations). In this case, you use the Unix command `gencat` and the NLS source file to transfer the modifications to the corresponding NLS message catalog. In this way you can create a modified NLS catalog which is used for all the UTM applications in the system, or an application-specific NLS catalog which you must store in directory *filebase*`/nls/msg/`*lang* (*filebase* = directory of the application).

You can delete the C source file for the message module.

4.5.3 NLS message catalogs under Windows systems

In Windows systems, the message catalogs are supplied as DLLs in the directories `%UTMPATH%\nls\msg\de` (German) and `%UTMPATH%\nls\msg\en` (English).

You can specify the message catalog to be used by means of the %LANG% environment variable. You can set the values “De” for German and “En” for English for %LANG%. Other values are treated as if they were “En”.

4.6 UTM log file SYSLOG

openUTM maintains a log file for every UTM application. This file is called the SYSLOG file. openUTM records all messages intended for the SYSLOG message destination in this file. You can - within certain limits - specify which messages these are to be (see section “Destinations of UTM messages” on page 393).

The SYSLOG file for the application is always located in the directory *filebase*, where *filebase* is the directory in which the application is installed (base name of the KDCFILE; defined in MAX KDCFILE).

You can create the system log file SYSLOG as:

- a simple file: a file with the name *filebase*/SYSLOG. If this file does not exist when the application is started and no FGG has been set up, openUTM creates the file.
- a file generation group (FGG): this directory must be set up by the user by means of the utility routine `kdcslog` before starting the application. It is sufficient to create the file generation directory; the individual files of the FGG are created for you by openUTM. If you have generated automatic size monitoring for the SYSLOG file, then the SYSLOG must be created as an FGG.

Events occurring during the execution of the application which could be useful for monitoring purposes or for making checks later are logged in the SYSLOG file in the form of UTM messages (e.g. K033, K070). In particular, the SYSLOG file contains important information for diagnostics.

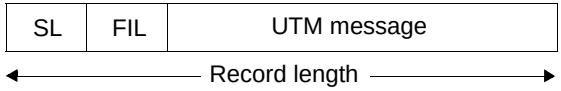
Each time the application is started subsequently, the SYSLOG file is overwritten by openUTM. The logging information from the previous run is lost. Therefore, you should backup the contents of the file if necessary when the application terminates.

4.6.1 Evaluating the SYSLOG file

openUTM creates the SYSLOG file in a format that cannot be evaluated directly. To print it out in a readable form or edit it using a program, you must process the file. Two editing tools are provided for this purpose: KDCCSYSL and KDCPSYSL.

4.6.1.1 The KDCCSYSL tool - editing the SYSLOG file

The editing tool KDCCSYSL converts a SYSLOG file into another file which can be processed with the user's evaluation tool program or the KDCPSYSL tool. Each UTM message in this new file comprises a record of variable length and is stored in the following format:



SL 2 bytes; content: binary record length (length of message + 4)

FIL 2 bytes; content: blanks

The structure of the individual UTM messages is described by the data structures *kcmsg.h* (C) or KCMSGC (COBOL). The messages do not contain a message text.
You can find the data structures for the UTM messages in the following locations:

Unix systems: *utmpath*`/include/kcmsg.h` or
utmpath`/copy-cobol85/KCMSGC` (Micro Focus COBOL compiler) or
utmpath`/netcobol/KCMSGC` (NetCOBOL compiler)

Windows systems: *utmpath*`\include\kcmsg.h` or
utmpath`\copy-cobol85\KCMSGC` (Micro Focus COBOL compiler) or
utmpath`\netcobol\KCMSGC` (NetCOBOL compiler)

Calling KDCCSYSL

Unix systems:

utmpath`/ex/kdccsysl` *file_in file_out*

Windows systems from a DOS window:

utmpath`\ex\kdccsysl` *file_in file_out*

Meaning of the parameters:

file_in is the SYSLOG file

file_out is the SYSLOG file edited by KDCCSYSL

The KDCCSYSL messages are output to *stderr*.

4.6.1.2 The KDCPSYSL tool -inserting message texts

KDCPSYSL reads from the file created with the editing tool KDCCSYSL and creates another file. For each UTM message, this new file contains the message number and message text with the current text parameters, date and time, as well as additional current parameters not contained in the message text. Some inserts are output in printable form.

The file created with KDCPSYSL is a normal text file which contains one or more lines for each UTM message. This file, which can be used to obtain a quick overview for example, can be viewed and, if necessary, processed or printed using an editor. The records in the file have a length of up to 136 characters (columns). This must be taken into account when printing.

Calling KDCPSYSL

Unix systems:

utmpath`/ex/kdcpsysl`

Windows systems from a DOS window:

utmpath`\ex\kdcpsysl`

KDCPSYSL works with the fixed file names `slogin` for the input file created with KDCCSYSL and `slogout` for the edited output file. For this reason, it may be necessary to use the mv command to change the name of the *file_out* file in the KDCCSYSL call to `slogin` before calling KDCPSYSL.

KDCPSYSL edits the data in `slogin` using NLS in the language set by the environment variable LANG, providing a message catalog `utmsys.cat` exists under *utmpath*`/nls/msg/`*set-language* (Unix systems)
or *utmpath*`\nls\msg\`*set-language* (Windows systems).

Otherwise, the incorporated message catalog is used.

4.6.1.3 KDCCSYSL and KDCPSYSL messages

The editing tools KDCCSYSL and KDCPSYSL write their messages to *stderr*, see section “Messages of the UTM tool KDCPSYSL” on page 346.

KDCCSYSL and KDCPSYSL edit their messages using NLS in the language set by the environment variable LANG, provided an NLS message catalog `utmsys.cat` exists under the following path:

utmpath`/nls/msg/`*set-language* (Unix systems) or
utmpath`\nls\msg\`*set-language* (Windows systems)

Otherwise, the incorporated message catalog is used.

4.7 Structure of UTM system messages

Structure of messages sent to SYSLOG or MSGTAC:

Message header	Message data, layout depending on message number

1 28 29

Structure of the messages:

Byte	Format	Meaning
1	X‘40‘	Blank
2-5	C‘Kxxx‘ or C‘Pxxx‘	Message number
6	X‘40‘	Blank
7-17	C‘mm/dd/yyjjj‘	Date, where mm=month, dd=day of month, yy=year, jjj=day of year
18	X‘40‘	Blank
19-24	C‘hhmmss‘	Time of day, where hh=hour, mm=minute, ss=second
25-28	C‘yyyy‘	4-digit year specification
29 & up		Message data

The complete data structures corresponding to the structure of the message header can be found the following locations:

Unix systems: *utmpath*`/include/kcmsg.h` or
utmpath`/copy-cobol85/KCMSGC` (Micro Focus COBOL compiler) or
utmpath`/netcobol/KCMSGC` (NetCOBOL compiler)

Windows systems: *utmpath*`\include\kcmsg.h` or
utmpath`\copy-cobol85\KCMSGC` (Micro Focus COBOL compiler) or
utmpath`\netcobol\KCMSGC` (NetCOBOL compiler)

5 UTM messages

The following lists contain all the messages which can be issued by openUTM. The messages of the UTM utilities are included in these lists. "Additional information" has been added to the descriptions to explain the responses to the messages.

A "&" character precedes the name of an insert. UTM messages may contain more inserts than the standard messages. The meanings of the inserts are given in the tables in section “Message inserts” on page 379ff and section “Destinations of UTM messages” on page 393ff.

5.1 Messages of the transaction monitor

K001 Connected to application &APPL - input please

K002 Connected to application &APPL - please sign on

K003 &CMD is not permitted now

K004 User identification &USER is invalid - please sign on

K005 User identification &USER is locked - please sign on

K006 Invalid password - please sign on

K007 User &USER already signed on - please sign on

K008 Sign-on accepted - input please

K009 Transaction code &TAC is invalid (&RCDC) - input please

The &RCDC insert contains the internal return code KCRCDC, see section “Internal return code KCRCDC” on page 416.

K010 Transaction code &TAC is locked - input please

K011 Transaction &ATAC1 accepted - input please

K012 &NUMMSGS asynchronous messages present

K013 Error in &CMD - input please

K016 Application shutdown pending - please sign off

K017 Service &TCVG terminated by UTM (&RCCC/&RCDC) - input please

This message informs a user of the abnormal termination of the service started by this user.

The K017 message contains the following inserts:

PTRM	Name of the PTERM from which the terminated service was started.
PRNM	Name of the processor to which the terminal is connected.
BCAP	Name of the BCAM application to which the user has signed on.
LTRM	Name of the LTERM from which the service was started.
USER	Name of the user who started the service.
TCVG	Name of the service TAC of the service.
RCCC	Value of the compatible KCRCCC return code.
RCDC	Value of the incompatible KCRCDC return code.
TAC	Only for RCDC=KMxx return codes: Name of the TAC that caused the service to terminate. In all other cases this insert contains a blank.

K018 Sign-off for application &APPL accepted - please sign on

K019 Sign-off accepted

K020 No message(s) present

K021 No input within the specified period

K022 The following message from &BCAP may have already been sent.

K023 &OMSG2

The message is called by the administration command KDCSHUT WARN or the corresponding call to the administrator interface.

K024 Input message lost - please repeat

K025 Message from application &BCAP to LTERM &LTRM was truncated

A message to an LTERM partner could not be transmitted completely, and was output in truncated form.

The following error recovery action is possible: Increase the length specification in the TRMSGLTH operand in the KDCDEF generation.

K026 Broadcast to &LTRM accepted - input please

K027 Terminal &LTRM is locked - contact administrator or sign off

K028 Please enter password

K029 Please insert card

K030 Card reader required - please sign on

K031 Wrong card - please sign on

K032 UTM-D connection message: &CON/&PRNM/&BCAP/&LPAP/&USER; reason1: &RCF1B; reason2: &RCF2B

In the following description of the return codes &RCF1B and &RCF2B, PLU (Primary Logical Unit) is the application in which SESCHA PLU=NO was generated, and SLU (Secondary Logical Unit) is the application in which PLU=YES was specified.

The &USER insert contains the UTM session name of the connection (LSES name).

&RCF1B	**Meaning**	
C01	BIND_REJECTED	The BIND request or the BIND response has been rejected. No session can be set up.
C02	BIND_CHANGED	The BIND parameters have been modified. The BIND request may be modified by the SLU; while PLU can reject but not modify the BIND response. If the BIND response cannot be accepted by PLU, no session is set up.
C03	CONNECTION / SESSION FAILED	A connection or session cannot be set up.
C04	SESSION_FAILED	Session warm start could not be carried out.
C05	STSN_CHANGED	SLU cannot accept the proposed restart point.

&RCF2B	**Meaning and possible action**
CR00	Session and connection do no belong to the same LPAP. Response: check the KDCDEF generation.
CR01	The sessions were not able to agree on a valid restart point at session warm start. The applications were probably started with different KDCDEF states. Response: check whether either of the applications has been regenerated.
CR02	SLU has suggested a new restart point. If the session cannot be set up, the applications were probably started with different KDCDEF states. Response: check whether either of the applications has been regenerated.
CR03	Quiet command entered.
CR04	No suitable session available; reasons: – more connections generated than sessions – KDCLSES command entered with ACT = QUIET – session not yet cleared down.

&RCF2B	Meaning and possible action
CR05	Connection setup request from TRANSIT-CD or partner application generated for this connection (NEA logs)
CR06	Connection setup in progress.
CR07	Inconsistency with respect to PLU specification in the connection message from the partner and in the generation. Both partners were possibly generated as PLU. Response: check the generations.
CR08	Inconsistency with respect to PLU specification in the connection message from the partner and in the generation. Both partners were possibly generated as SLU. Response: check the generations.
CR09 CR10 CR11 CR12	Incorrect structure of a connection message.
CR13	Resource bottleneck: no slot available for dynamic tables.
CR14	The name of the PLU session in the connection message is incorrect. Response: check the generation
CR15 CR16 CR19 CR21	Session is still set up.
CR17 CR18 CR20	No suitable session available; reasons: – more connections generated than sessions – KDCLSES command entered with ACT = QUIET – session not yet cleared down.
SC00	No reason given.
SC01[1]	The partner is responding with an invalid FM profile. UTM supports FM profile 18 only.
SC02[1]	The partner is responding with an invalid TS profile. UTM supports TS profile 4 only.
SC03[1]	UTM can process a message in small units (request units). For this reason, PLU must support "multiple RU chains".
SC04[1]	A request cannot be sent off until a previously requested response, if any, has arrived. PLU therefore has to operate in "immediate request mode".
SC05[1]	Depending on the message type, openUTM either requests all types of acknowledgment or only a negative acknowledgment. PLU must therefore accept both definite and exception response.
SC07[1]	Data compression is not supported for message transmission.
SC08[1]	PLU must always be able to close a bracket.

&RCF2B	Meaning and possible action
SC09[1]	Corresponds to reason SC03, but for SLU.
SC10[1]	Corresponds to reason SC04, but for SLU.
SC11[1]	Corresponds to reason SC05, but for SLU.
SC12[1]	openUTM does not support 2-phase commit.
SC13[1]	Corresponds to reason SC07, but for SLU.
SC14[1]	Corresponds to reason SC08, but for SLU.
SC15[1]	FM headers must be permitted.
SC16	The home session is insisting on a session warm start although the partner considers a session cold start to be sufficient.
SC17	The partner has indicated that the session has not terminated ("in bracket") without requesting a warm start. The session is assumed to have terminated (BETB).
SC18[1]	A service is not terminated until indicated as such by both partners. This means that openUTM makes use of the "Bracket Termination Rule 1".
SC19	The user data in BIND is not correct. The partner is probably sending an faulty BIND. The UTM BCAM trace is needed for diagnostics.
SC23[1]	In distributed processing, "half-duplex flip-flop" send or receive mode is used. This means that only one partner is authorized to send at any given time.
SC24[1]	Both partners must be responsible for a session restart, if applicable.
SC25	BIND and KDCDEF generation are inconsistent with respect to the contention winner. Response: check the specification of the contention winner in both generations (KDCDEF control statement SESCHA CONTWIN=Y/N).
SC26	PLU is requesting a send authorization at session warm start. If the session is not set up, the applications were probably started with different KDCDEF states. Response: check whether one of the applications has been regenerated.
SC27	SLU is requesting a send authorization at session warm start. If the session is not set up, the applications were probably started with different KDCDEF states. Response: check whether one of the applications has been regenerated.
SC28[1]	UTM supports "two-stage pacing".
SC29	Inconsistent pacing counts, i.e.: SLU receive count not equal to PLU send count, or PLU receive count not equal to SLU send count. Response: check the pacing count (= window size) in both generations. (KDCDEF control statement SESCHA PACCNT=)
SC30[1]	SLU is suggesting an illegal value for "maximum length of message segments from PLU" in the BIND response.

&RCF2B	Meaning and possible action
SC31[1]	SLU is suggesting an illegal value for "maximum length of message segments from SLU" in the BIND response.
SC32	PLU is suggesting an illegal value for "maximum length of message segments from SLU" in the BIND request.
SC33	PLU is suggesting a value for "maximum length of message segments from PLU" in the BIND request which is modified by SLU.
SC34[1]	Presentation Service (PS) usage field format must be defined as "basic format".
SC35[1]	LU session type 6 is the only permissible session type.
SC36[1]	Message coding is not supported.
SC37	Incorrect name of PLU session in BIND response. Response: check association of LSES and RSES in both generations (KDCDEF control statement LSES RSES=name)
SC38	Incorrect name of PLU session in BIND request. Response: see reason SC37
SC39	Incorrect name of SLU session in BIND response. Response: see reason SC37
SC40	Incorrect name of SLU session in BIND request. Response: see reason SC37
SC 41[1]	User Request Correlation (URC) is not supported UTM-D.
SY01	Session termination pending. Session cleardown has not yet been completed.
SY02	PET no send request The session is in the PTC status and there is no message for this partner.
SY03	PEND RS pending. PEND RS handling has not yet been completed.
SY04	Job-submitting conversation active. The job-submitting service is active.
SY05	Job submitter not available. The job submitted is not signed on or the job-submitting service is queued.
SY06	RESTART_VIA_SEND_RQ. The session is in the IN_BRACKET status and there is no message for this partner.
SY07	Session active The session is already active.

[1] Only in the case of heterogeneous links

K033 USER / LSES active : &PTRM/&PRNM/&BCAP/&LTRM/&USER &REST, &GLOBALSG

The K033 message is output:

– following successful connection setup when working without USERs
– following successful KDCSIGN when working with USERs
– following successful linkage of a session in a DTP connection.

&REST	Meaning
Y	Service restart
N	No service restart
U	Undefined, with UTM-D only

If a user signs on via an OSI-TP connection, the inserts have the following contents:

Insert	Contents
&PTRM	OSI-CON name
&PRNM	8 blanks
&BCAP	ACCESS-POINT name
&LTRM	OSI-LPAP name

A value is only entered for the insert &GLOBALSG 'Cluster Global Sign' in UTM cluster applications. The insert can have the following values:

&GLOBALSG	Meaning
Y	Global sign-on at the cluster
N	Local sign on at the node
A	The user was already signed on globally to the cluster at this node.
' '	Sign-on for an LU6.1 session user or a connection user.

K034 Transaction has been reset

K035 Service restart in progress

K036 Connection setup: &PTRM, &PRNM, &BCAP, &LTRM, &RSLT, &REA1

&RSLT	Meaning
Y	Connection set up
N	Connection was not set up; the cause is given in &REA1.

&REA1	Meaning
X'00'	Connection already set up

&REA1	Meaning
X'01'	PTERM/CON name unknown
X'02'	Processor name unknown
X'03'	PTYPE not accepted
X'04'	No LTERM assigned
X'05'	Incorrect connection password
X'06'	STATUS=OFF for this PTERM/CON
X'07'	STATUS=OFF for this LTERM/LPAP
X'08'	Resource bottleneck
X'0A'	Application shutdown
X'0B'	Partner already connected
X'0C'	Connection cleardown being executed
X'0D'	Negative transport system return code; see corresponding K065 message
X'0E'	Partner generated at another application
X'0F'	UTM-D connection request rejected, e.g. due to generation error or QUIET command
X'10'	PTERM name unknown and no LTERM pool available for this processor name
X'11'	No LTERM pool available for PTYPE and PTERM name unknown
X'12'	No further free entry available in terminal pool
X'13'	Partner already connected to LTERM pool
X'14'	Connection rejected due to insufficient characteristics of the transport connection (GROS)
X'16'	User is already connected
X‘17‘	The connection request has been rejected due to contention. The partner himself has initiated a connection setup.
X‘1B‘	On application start, it was not possible to determine the IP address of the partner computer.
X‘1F‘	LPAP is set to quiet
X‘20‘	There is no suitable session available
X‘21‘	The session is already active
X‘22‘	The session is currently being terminated
X‘23‘	The local page is in PTC and does not have an output message for this session; it is necessary to wait for the restart of the local service to establish the session.
X‘24‘	PEND RS is being processed
X‘25‘	The job-submitting service is active

&REA1	Meaning
X‘26‘	The job-submitting user is not available
X‘27‘	It is necessary to wait for the restart of the local service to establish the session
X'29'	Terminal pool is generated at another BCAM application
X'2E'	The connection has not yet been completely cleared down, or the MUX session is still in "RELEASE-PENDING" status.
X'45'	A connection cancel request occurred while waiting for confirmation of the establishment of a connection
X'46'	PTERM/CON was deleted by the administration
X'48'	A PTERM with the name of the multipool LTERM has already been generated.
X'55'	Rejected socket connection setup.
X'58'	Rejected confirmation of connection setup since the connection was interrupted in the interim.

This message, which is normally output to SYSLOG, helps to detect problems arising at connection setup.

For message output, the &REA1 insert is edited in printable form while at the program interface with MSGTAC / SYSLOG it appears in hexadecimal form.

K038 Syntax error &SYN

K039 Startparameters CLUSTER-FILEBASE and &STRTPAR are mutually exclusive!!

K040 Warning level &WLEV for &PGPOOL no longer exceeded

K041 Warning level &WLEV for &PGPOOL exceeded

K042 Error reading start parameters

K043 DMS error &DMSE for file &FNAM

The DMS error code is output in insert DMSE. The possible DMS error codes are described on page 377.

In the case of UTM cluster files, the insert &DMSE can contain the following error codes in addition to the error codes reported by the operating system:

ILCK An error occurred while initializing a file lock.

LOCK An error occurred while requesting a file lock.

ULCK An error occurred while releasing a file lock.

DLCK An error occurred while destroying a file lock.

K044 Reading the start parameters

K045 --- No message text ---

The only destination permitted for this message is MSGTAC; therefore no message text is defined for this message.

The message is output when a message output in acknowledgment mode is terminated with a positive print acknowledgment (see page 169).

No default destination is defined for this message.

K046 Print error: &PALTRM / &CID / &DPID / &ERPRT

The message is output only for printers that are assigned to a printer control terminal. It is primarily intended for programmed error handling with the help of the MSGTAC program unit.

Further inserts of this message are: &PTRM, &PRNM, &BCAP, &LTERM, &IMSG2

The message is output in the following cases:

– termination of a printout with negative print acknowledgment

– repetition of a printout (in addition to the K022 message)

– connection setup to a printer is rejected

No default destination is defined for this message.

The &ERPRT insert provides information on the error. Independent of this, the &IMSG2 insert gives further information.

&ERPRT	**Meaning and contents of &IMSG2**
A	Negative print acknowledgment. If the printer sends a logical print acknowledgment (return message), &IMSG2 contains the first 32 bytes of this acknowledgment (possibly padded with spaces). The exact structure of this print acknowledgment is described in the appropriate equipment user guides.
R	Repetition of a printout. &IMSG2 has no meaning. Connection setup is rejected.
C	&IMSG2 contains the DIAGNOSTIC-WORD.

K047 Invalid SVC 134 call (SVC for UTM): opcode = &OPCD1, RC = &RTCD

The SVC call is a UTM-internal interface. The return code (RC) in message K047 can have any of the following values:

nn	Meaning
04	Illegal ROOTDATA address (UTM error[1]).
08	The compiled ROOT source program and the library with the UTM system modules are incompatible. There are two solutions to this problem: – compile the ROOT source with the correct header files from utm-path/include – link the application with the correct library utm-path/sys/libwork.a (UTM system modules)
0C	Invalid parameter list address (UTM error).
10	Invalid KDCS parameter area address UTM error. If the include files made available by openUTM for the KDCS parameter areas are used, this is a UTM error. If user-defined areas are used in the program unit for the KDCS parameter area, this is a user error. Action: check the alignment and address of the parameter area
14	Illegal KTA address in ROOTDATA, possibly due to: – UTM error – KDCS call in START-EXIT program unit – KDCS call in SHUT-EXIT program unit – KDCS call after UTM Term Application
18	Illegal contents in KTA header. Possible cause: UTM error
1C	Application names in ROOTDATA and KAA do not match, or illegal KAA header (UTM error).
20	Application names in ROOTDATA and KTA do not match (UTM error).
24	Parameter list address not aligned with word boundary (UTM error).
28	KDCS parameter area not aligned. If the include files made available by openUTM for the KDCS parameter areas are used, this is a UTM error. If user-defined areas are used in the program unit for the KDCS parameter area, this is a user error (alignment with halfword boundary). Action: check the alignment and address of the parameter area.

[1] In the case of return codes of the UTM error type: Siemens Service should write problem report.

K049 Error &RCCC2 during application startup

UTM issues message K049 whenever the start of a UTM task is aborted due to an error, and the error code &RCCC2 shows the cause of the error.
The table below lists the possible error codes together with error causes and possible error recovery actions.

Start error codes

Code	Error cause	Response
2	There is not enough space available for the ROOTDATA area.	Check generation and system configuration
3	The ROOTDATA area is not allocated or the string ‚ROOTDATA' is not available.	Reconcile UTM versions
4	KDCROOT and the UTM system modules are of different versions.	Reconcile UTM versions
5	Application cannot be signed on to Name Manager.	System error; please write PR and notify Service
6	The start of a continuation task is aborted due to abortion of the start of the first task.	See error code of 1st task
7	The user's message module is not consistent with the message module of the UTM system code (e.g. incorrect number of inserts).	Correct message module and link application again
10	The request of a UTM task for 4K of memory has been rejected.	Check system generation and memory requirements of the application
11	Error at first OPEN call for the KDCA file of KDCFILE, possibly due to incorrect FILEBASE name specification in start parameters. Possible cause in UTM cluster applications: An online import is running for the node that is to be started.	See DMS error code, page 377.
12	DMS error when reading first 2KB area of the KAA (KDC Application Area) from the KDCA file.	See DMS error code; page 377.
13	DMS error when reading first or last check page of KDCA file; the file cannot be used.	Make copy or regenerate with KDCDEF
14	The UTM version in the KAA (KDC Application Area) which is read from the KDCFILE does not match the version of the UTM system code. The error can occur if you attempt to run two UTM applications having the same name in different UTM versions in parallel.	Change name of an application (MAX APPLINAME=...)
15	KDCDEF and UTM code inconsistent with respect to KAA structure.	Regenerate with correct KDCDEF

Code	Error cause	Response
16	KDCDEF and UTM system modules are of different versions.	Reconcile versions
17	Before the start KDCFILE was processed by KDCUPD, which terminated abnormally	Make KDCFILE consistent by means of either KDCUPD or KDCDEF
18	The KDCFILE in a follow-up task does not have the same generation time as the first task.	Start follow-up task with same KDCFILE file as first task
20	Shared memory cannot be allocated by the first task of the UTM application due to insufficient address space.	Check Unix generation and UTM generation
21	A follow-up task of a UTM application is unable to link up with the KAA shared memory due to insufficient address space or due to the application being terminated.	As for 20
22	DMS error when reading an NSR page in KAA.	See DMS error code, page 377.
23	A KAA NSR page has been destroyed, the KDCA file can no longer be used.	Regenerate with KDCDEF
24	DMS error when reading an SR page in KAA.	See DMS error code, page 377**.**
25	A KAA SR page has been destroyed, the KDCA file can no longer be used.	Regenerate with KDCDEF
26	At least one program for a generated event function is missing or an event-driven service is missing; i.e. the program is neither linked nor can it be loaded	Transfer program to the library specified at generation and relink if necessary
27	KCSBKAA returned a bad return code. Error during initialization of the SYSLOG data in the KAA	Write problem report
28	During a (follow-up) start of the UTM application an inconsistency in the database generation between KDCFILE and ROOT was detected. The error occurs when KDCFILE and ROOT source are incompatible. UTM rejects this in order to avoid consequent errors.	Generate KDCFILE and ROOT with the same start parameters
29	– The administration program unit (KDCADM) is missing or – TAC KDCSHUT is not generated.	– Add the KDCADM program to the generation and add to the UTM application – Generate TAC KDCSHUT
30	The request from a UTM task for a contiguous area for the task-specific UTM tables (= KTA) in class 5 memory has been rejected	Check system generation and memory requirements, reduce KTA by changing KDCDEF operands

Code	Error cause	Response
31	An attempt has been made to start more tasks than have been generated.	User error
32	The task lock bourse could not be created. This error occurs when there are too few semaphore entries available for the UTM processes. This can occur when the termination of a process and its restart overlap.	Regenerate with a larger number of semaphores.
33	UTM refuses to start more tasks for the application because the latter has already terminated (normal or abnormal termination).	Usual action
34	The KDCFILE in a follow-up task does not have the same KAA size as the first task.	Start follow-up task with same KDCFILE file as first task
35	While dynamically loading the application program, UTM detects that the application is being aborted.	Usual action
36	Error in Name Manager call for a continuation task of an application.	System error; please notify Siemens Service
37	Number of tasks to be started insufficient for PGWT processing or for a UTM cluster application	Start at least 2 tasks
39	When dynamically loading the application program, UTM detected that the generation has been illegally changed.	Check start procedure and generation
40	Number of entries in the program, load module or message module tables in ROOT and in the KDCFILE do not match	Generate KDCFILE and ROOT with the same start parameters
41	A program or load module attribute in ROOT and in the KDCFILE do not match (see K073 message).	Generate KDCFILE and ROOT with the same start parameters
44	The generations of KDCFILE and ROOT do not match with regard to MAX TRMSGLTH	Generate KDCFILE and ROOT with the same start parameters
45	Error in KCSLKLC when setting a TAC class lock	Increase parameter MAX RESWAIT=(...,time2)
46	UTM cluster application: A follow-up task was started with different cluster filebase specifications from the first task.	Start first task and follow-up task with same start parameters
47	The generations of KDCFILE and ROOT do not match with respect to the use of shared objects / dlls.	Generate KDCFILE and ROOT with the same start parameters

Code	Error cause	Response
48	Error while initializing buffer management in start case 1 or 2.	Increase the virtual address space. If this does not help, write problem report.
50	The first UTM task is trying to reserve the cache shared memory, or a continuation task is trying to link up to it. This is not possible due to insufficient address space.	Check system generation and memory requirements of the UTM application
51	The file is not a KDCDEF file; for file name see K082 message	Supply or generate correct file
52	The file has been inadvertently replaced by another KDCFILE file, e.g. pool file instead of KDCB file; for file name see K082 message.	As for 51
53	The file does not match the KDCA file (e.g. different KDCDEF time); for file name see K082 message; for details on the error cause see the log in *stdout*.	As for 51
54	The consistency check shows that the file has been destroyed. Possibly a version mix. For file name see K083 message	As for 51 Check versions
55	DMS error with KDCA/KDCB file	See DMS error code, page 377.
56	DMS error with page pool file	
57	DMS error with restart file	
58	Error in conjunction with the SYSLOG file to be made available by the user. Possible causes: – The <filebase>/SYSLOG directory is empty or it does not contain an info file for an FGG. – If there is no SYSLOG directory, then a “SYSLOG“ file is automatically created.	Delete or regenerate the prepared SYSLOG file.
59	Error when opening SYSLOG file	See DMS error code, page 377.
60	Error during sign-on of application or follow-up task to BCAM or socket. For exact error cases see DIAGNOSTIC-WORD in message K065 (default destination SYSLOG) or message K154 (default destination SYSLOG) for socket. User error or system error.	Evaluate K065 or K154 in the SYSLOG file
62	As for 70.	As for 70.
64	A request from the first task of the application for a work area of 8K of memory made during warm start is rejected	Check system generation and UTM generation
65	As for 64 (length = task no. * 2K)	As for 64

Code	Error cause	Response
69	KDCFILE and ROOT do not match regarding the OSI-TP generation. Possible causes: a new KDCFILE was created (new: with OSI-TP, old: without OSI-TP), the UTM application could not be relinked or the ROOT source could not be recompiled.	Regenerate, compile ROOT and link.
70	The first task of the application has detected inconsistent data in KDCFILE during warm start.	System error; please notify Service, regenerate KDCFILE with KDCDEF
71	As for 70.	As for 70.
72	As 70.	As for 70.
74	Database problem with DB status check during restart; error cause KU. is output, see KCRCDC error codes	Correct database, restart application
75	As for 70.	As for 70.
76	As for 70.	As for 70.
77	A user ID for an open asynchronous service could not be found	Write PR.
79	A UTM task requests, but is not allocated memory.	Check system generation and memory requirements of the application.
80	The first task of the application has detected inconsistent data in KDCFILE when setting up the page pool map.	System error; please notify Service, regenerate KDCFILE with KDCDEF
81	As for 80.	As for 80.
82	Database problem when erasing DB status information; error cause KUxx is output, see KCRCDC codes.	Correct database, restart application
83	Database problem when resetting a DB transaction; error cause KUxx is output, see KCRCDC codes.	Correct database, restart application
84	The task attempts to create a slot for an OSI service table. This attempt is unsuccessful due to a lack of memory space.	Check system generation and memory requirements of the application.
85	The task has failed to create a slot for an open service due to insufficient address space.	Check system generation and memory requirements of the application.
86	The 1st task of the application has detected unallocated page pool pages in the cache (during warm start)	System error; please write PR

Code	Error cause	Response
90	Error when starting an open distributed asynchronous service	System error; please write PR
91	Error when starting XAP-TP. A full description of the error can be found under message K124.	System error; please write PR and notify Service and check the available memory (see K124 on page 264)
92	No free NODE table was found during a restart for OSI-TP with XAP-TP.	System error; please write PR and notify Service
93	A user ID for a transaction which has not yet terminated was found during a restart for OSI-TP with XAP-TP.	System error; please write PR and notify Service
94	An inconsistent NODE table was found during a restart for OSI-TP with XAP-TP.	System error; please write PR and notify Service
95	DMS error while writing a UTM cluster file (start case 1)	See DMS error code
96	A UTM cluster file does not correspond to the cluster configuration file, e.g. different generation time. File name, see K082 message. For details on the cause of the error see the log message in *stdout*.	Use the correct UTM cluster file or regenerate the UTM cluster application.
97	DMS error while writing a cluster file (operation code CL_FIRST_NODE)	See DMS error code
98	The KDCFILE does not correspond to the cluster configuration file, e.g. different generation time. File name, see K082 message. For details on the cause of the error see the log message in *stdout*.	Use the correct KDCFILE or regenerate the KDCFILE.
99	Database problem during warm start via the XA interface.	System error; please write PR and notify Service
100	An inconsistency was detected between KDCFILE and ROOT while starting the UTM application in the KB length.	Create KDCFILE and ROOT with the same generation parameters
101	An inconsistency was detected between KDCFILE and ROOT while starting a follow-up task in the KB length.	Create KDCFILE and ROOT with the same generation parameters
103	An inconsistency was detected between KDCFILE and ROOT while starting the UTM application in the NB length.	Create KDCFILE and ROOT with the same generation parameters.
104	An inconsistency was detected between KDCFILE and ROOT while starting a follow-up task in the NB length.	Create KDCFILE and ROOT with the same generation parameters.

Code	Error cause	Response
109	An inconsistency was detected between KDCFILE and ROOT while starting the UTM application in the generation of the database.	Create KDCFILE and ROOT with the same generation parameters.
110	An inconsistency was detected between KDCFILE and ROOT while starting a follow-up task in the generation of the database.	Create KDCFILE and ROOT with the same generation parameters.
112	An inconsistency was detected while starting the UTM application in the first type of database.	Create KDCFILE and ROOT with the same generation parameters.
115	An inconsistency was detected between KDCFILE and ROOT while starting the UTM application in the second type of database.	Create KDCFILE and ROOT with the same generation parameters.
116	An inconsistency was detected between KDCFILE and ROOT while starting a follow-up task in the second type of database.	Create KDCFILE and ROOT with the same generation parameters.
118	UTM cluster application: Error reading filebase name of KDCFILE from the cluster configuration file Possible causes: – memory bottleneck – error accessing cluster file – invalid cluster file – own node not found For more details, see message K043 and K190	See message K043 or K190; correct start parameters or generation if necessary
119	UTM cluster application: The KDCFILE was generated as a UTM cluster application but no CLUSTER-FILEBASE start parameter was specified.	Correct start parameters
120	UTM cluster application: The KDCFILE was not generated as a UTM cluster application but a CLUSTER-FILEBASE start parameter was specified.	Correct start parameters
122	UTM cluster application: Error editing the cluster configuration file for initialization of the KAA	See message K043 and/or K190
123	UTM cluster application: Error registering node	See message K043 and/or K190
125	UTM cluster application: The sequence of node names in the KDCFILE is different from that in the cluster configuration file.	Correct the generation; Regenerate the KDCFILE and, if necessary, the UTM cluster files
127	UTM cluster application: Error opening cluster user file on start of first process in application (start case 1)	See message K043 and/or K190

Code	Error cause	Response
128	UTM cluster application: Error opening cluster user file on start of a follow-up process (start case 2)	See message K043 and/or K190
130	UTM cluster application: Error requesting start lock for serialization by KCSCONS (start case 1)	See message K043 and K190
131	UTM cluster application: Error opening administration journal files on start of a follow-up process (start case 2)	See message K043 and K190
132	UTM cluster application: Error opening cluster administration journal files when reloading the application program after a program replacement (start case 3)	See message K043/K190
133	UTM cluster application: Error creating administration journal files on start of first process in application (start case 1)	See message K043/K190
134	UTM cluster application: Error writing online copy on start of first process in application (start case 1)	See message K043 and/or K190
135	UTM cluster application: Error incorporating online copy on start of first process in application (start case 1)	See message K043/K190 and/or K174
136	UTM cluster application: Error editing cluster user file on start of first process in application (start case 1)	See message K043 and/or K190
137	UTM cluster application: Calling KCCCTRL with operation code REGISTER_NODE returns an unknown return code.	System error, please write problem report and inform Service.
138	UTM cluster application: Calling KCCJCTL with operation code WRITE_JOURNAL_PI returns an incorrect return code.	System error, please write problem report and inform Service.
139	UTM cluster application: No lock was requested for start serialization.	System error, please write problem report and inform Service.
140	UTM cluster application: Error concluding registration of node at cluster	See message K043 or K190
141	UTM cluster application: Calling KCCJCTL with operation code JFCT_SET_KAA_INFO returned an incorrect return code.	System error, please write problem report and inform Service.
142	UTM cluster application: Calling KCCJCTL with operation code JFCT_SET_KAA_INFO returned an incorrect return code.	System error, please write problem report and inform Service.

Code	Error cause	Response
143	UTM cluster application: Error opening administration journal files on start of first process in application (start case -1)	See message K043 and K190
144	UTM cluster application: Error incorporating administration journal	System error, please write problem report and inform Service.
145	UTM cluster application: Calling KCCJCTL with operation code JFCT_SET_KAA_INFO returned an incorrect return code.	System error, please write problem report and inform Service.
146	UTM cluster application: The runtime configuration of the node application that is to be started does not match the runtime configuration of the running node applications.	See message K174
147	UTM cluster application: Global administration actions should be incorporated in the warm start. However, the administration journal files cannot be opened.	Check whether the administration journal files exist, see message K043/K190
148	UTM cluster application: Global administration actions should be incorporated in the warm start. Internal error when reading journal files.	System error, please write problem report and inform Service.
149	UTM cluster application: Internal error when cleaning up node-specific information in the journal files.	System error, please write problem report and inform Service.
150	UTM cluster application: Error opening administration journal files on start of first process in application (start case 1)	See message K043 and K190
151	UTM cluster application: Error requesting lock for start serialization of nodes (during warm start).	See message K043 or K190
152	UTM cluster application: Error requesting lock for start serialization of nodes	See message K043 or K190
153	UTM cluster application: Cluster page pool control file defective.	See message K190
154	UTM cluster application: Error opening the LOCK file on the start of a follow-up process (start case 2)	See message K043
155	UTM cluster application: Error opening the LOCK file on the start of the first process in the application (start case 1)	See message K043
156	UTM cluster application: Error opening the LOCK file on the start of a follow-up process (start case 3)	See message K043

Code	Error cause	Response
157	UTM cluster application: Error while checking the cluster GSSB file	See message K043 or K190
158	UTM cluster application: Error while checking the cluster ULS file	See message K043 or K190
161	UTM cluster application: Error while releasing the ULS locks in the cluster ULS file	See message K190
162	UTM cluster application: Error while releasing the GSSB locks in the cluster GSSB file	See message K190
164	UTM cluster application: Error while opening the LOCK file on warm start	See message K043
165	UTM cluster application: On the warm start of a node application, it was not possible to lock the cluster lock file in the generated time.	Repeat the application start operation
166	UTM cluster application: Internal error when resetting a transaction in PTC state during node recovery.	System error, please write problem report and inform Service.
167	UTM cluster application: Internal error when outputting existing transactions in PTC state during node recovery.	System error, please write problem report and inform Service.
168	As for 167.	As for 167.
169	UTM cluster application: An attempt was made to perform a node recovery for a node application that terminated normally.	Check whether the incorrect node was specified in the start parameter.
170	UTM cluster application: Cluster page pool and KDCFILE generated with different BLKSIZE.	Regenerate all the UTM cluster files.
171	UTM cluster application: A node recovery was started in a dialog.	Start node recovery in batch operation.

K050 Successful warm start for application &APPL under UTM &VERS / &OST1 / &BMD1

K051 Successful cold start for application &APPL under UTM &VERS / &OST1 / &BMD1

The messages K050 and K051 are output after the successful startup of the application. The insert &VERS contains the UTM version, &OST1 the type of operating system and &BMD1 the bit mode (32/64).

Both messages have the additional inserts AMOD, &TERM, &ATYP and &FNOD which are not contained in the default message text; these have the following meanings:

&AMOD contains the application mode of the application:

"S" in the case of UTM-S
"F" in the case of UTM-F

&TERM contains the termination type entered in the KDCFILE at application startup; the following values are possible:

"C": The KDCFILE was created anew with KDCDEF.
"U": The KDCFILE was updated with KDCUPD.
"N": The last application run was terminated normally.
"A": The last application run was terminated abnormally.

&ATYP contains:

"C" in the case of a UTM cluster application
"S" in the case of a standalone application

&FNOD contains:

" " (blank) in the case of a standalone application
"Y" on the startup of the first node application in a UTM cluster application
"N" on the startup of each subsequent node application in a UTM cluster application

K052 Startup completed - task &PID activated for application &APPL, version &PRGVERS, System-Task: &STSK

The insert &PRGVERS is only used when exchanging a program with kdcprog. It specifies the version number of the FGG from which the application program was loaded. &PRGVERS contains the value null for normal application starts.

&STSK specifies whether (Y) or not (N) the process is a system process.

K053 New user log file created; old user log file contains &CNTR records.

K054 Copies of the user log files are not identical.

K055 Asynchronous service &ATAC1 terminated by UTM; KCRCCC=&RCCC; KCRCDC=&RCDC; USER=&USER; LTERM=&LTRM

K056 Task &PID terminated and will be restarted: &RSLT

The &RSLT insert indicates whether the process is restarted (Y) or not (N) after PEND ER.

K057 Application run terminated

K058 Abnormal termination of task &PID

K059 Abnormal termination of application run

K060 Application run aborted; reason = &TRMA

UTM dump error codes

UTM creates a memory dump whenever a UTM application is aborted or a dump requested. Such a dump is produced for each work process of the application (see chapter “The UTM dump” on page 57ff). The error code is indicated in a UTM dump with REASON= in the second line on each page of the edited UTM dump.

If a UTM application is terminated abnormally, openUTM also performs the following actions before restarting insofar as this is possible:

– immediately terminate all transactions currently being processed by the individual processes of the UTM application
– clear all connections to the communication partners of the UTM application
– clear connections to external resource managers (e.g. to database systems)
– close all files
– terminate all processes

For security reasons, when a UTM application terminates abnormally, no attempt is made to restore the KDCFILE, which contains all the data required for running the UTM application, to a consistent status. This is only done on restarting.

The **Group** column in the tables below describes to which reason group the dump code error code belongs. The following groups exist:

A The cause is a user error, e.g. an error in

  – generating and administering UTM applications
  – generating the system (e.g. division of the address space)

U The cause is an error in the UTM code.

S The cause is an error in another system component (software or hardware).

F The dump is a continuation dump, another task has caused the application to terminate abnormally.

D The UTM dump was created for diagnostic purposes. The UTM work process continues running.

M The cause is a memory bottleneck.

X The cause is an error in the XAP-TP code.

Multiple classifications are possible, e.g. ADS.

Errors of the reason group M in the XAP-TP component can occur when too small a value for the OSI-SCRATCH-AREA parameter in the MAX statement was selected when generating the application with KDCDEF.

You must write a problem report for error diagnosis for all errors of groups U and S and all error codes **not** listed in the table below. A number of different documents are required for diagnosis. For detailed information and a list of the required documents, refer to section “Producing error documentation” on page 42.

You should try to reproduce the errors using static libraries.

Code	Group	Reason
AHQA00	MX	XAP-TP component. KCOCOTA module, QueueAnno() function. The mGetBufferSpace() macro issued the return code LB_NOMEM.
AHSA02	SUX	XAP-TP component. Module KCOCOTA, function SendAnno(). The system interfaces call KCOBRSE supplied a bad return code.
AINF01	S	The return code ist not equal X'00' after the AINFcall.
ALAxxx	ASU	Error number xxx when calling shmat() in KCSALME
ALGxxx	ASU	Error number xxx when calling shmget() in KCSALME. For possible cause, see start error 33, or KDCREM was not called before appli-cation start or shared memory bottleneck due to over-large CACHESIZE or similar. Note: ALG022 can occur when the shared memory that is to be allocated is larger than the machine-specific maximum (tuning parameter SHMMAX). Response: modify the KDCDEF generation.
APFI00	A	FOBIB shell variable could not be reserved, e.g. in the event of a storage bottleneck. Response: check memory requirements.
ASA006	SUX	XAP-TP component. Module KCOASAM was called with the operation code ASAM_ATTACH. Bad return code from bBuildPAddr.
ASA007	SUX	XAP-TP component. Module KCOASAM was called with the operation code ASAM_ATTACH. First task of the application, but OSS issued return code NOTFIRST.
ASA009	SX	XAP-TP component. Module KCOASAM was called with the operation code ASAM_ATTACH. aufgerufen. OSS returned the return code "INVEREF" for an „attach“ call.
ASA010	SX	XAP-TP component. Module KCOASAM was called with the operation code ASAM_ATTACH. OSS returned a bad return code for an „attach“ call.
ASA011	FSX	XAP-TP module The module "KCOASAM" was called with the opcode "ASAM_DETACH". On the "detach" call, OSS returned the return code "ERROR". The task is already in the termination phase after another task has abnormally terminated the application. OSS returned an incorrect error code because openUTM still has to fetch an event from OSS.

Code	Group	Reason
ASA012	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_DETACH". On the "detach" call, OSS returned the return code "INVAREF".
ASA013	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_DETACH".On the "detach" call, OSS returned a bad return code
ASA033	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". the returned application context name is too long.
ASA034	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". OSS returns the return code "ERROR" on the "assin" call.
ASA035	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". OSS returns the return code "INVREF" on the "assin" call.
ASA036	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". OSS returns a bad return code on the "assin" call.
ASA043	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". OSS returns the return code "ERROR" on a positive "assrs" call.
ASA044	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". Bad return code from "PutElement".
ASA045	SX	XAP-TP component The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". OSS returns a bad return code on a positive "assrs" call.
ASA046	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". OSS returned the return code "ERROR" on the negative "assrs" call.
ASA048	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". OSS returned a bad return code on the negative "assrs" call.
ASA049	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_IND". Bad return code from "PutElement".
ASA051	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". The retruned Application Context Name is too long.

Code	Group	Reason
ASA052	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". OSS returned the return code "ERROR" on the "asscf" call.
ASA053	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". OSS returned the return code "INVREF" on the "asscf" call.
ASA054	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". OSS returned a bad return code for an "asscf" call.
ASA060	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". OSS returned the return code "ERROR" on the "aborq" call.
ASA062	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". OSS returned a bad return code for an "aborq" call.
ASA064	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". OSS returned an invalid diagnostic value on a negative association confirmation.
ASA065	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". OSS returned an invalid "result source" value on a negative association confirmation.
ASA066	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF_TIMEOUT". OSS returned the return code "ERROR" on the "aborq" call.
ASA068	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF_TIMEOUT". OSS returned a bad return code on the "aborq" call.
ASA071	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ABORT_IND". OSS returned the return code "ERROR" on the "aboin" call.
ASA072	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ABORT_IND". OSS returned the return code "INVREF" on the "aboin" call.
ASA073	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ABORT_IND". OSS returned a bad return code on the "aboin" call.

Code	Group	Reason
ASA080	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_P_ABORT_IND". OSS returned the return code "ERROR" on the "paboin" call.
ASA081	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_P_ABORT_IND". OSS returned the return code "INVREF" on the "paboin" call.
ASA082	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_P_ABORT_IND". OSS returned a bad return code on the "paboin" call.
ASA083	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". Bad return code from "PutElement".
ASA084	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF". Bad return code from "PutElement".
ASA085	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ASS_CNF_TIMEOUT". Bad return code fromn "PutElement".
ASA088	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned the return code "ERROR" on the "relin" call.
ASA089	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned the return code "INVREF" on the "relin" call.
ASA090	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned a bad return code on the "relin" call.
ASA091	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned an invalid value for "release reason".
ASA092	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned the return code "ERROR" on the "aborq" call.

Code	Group	Reason
ASA094	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned a bad return code on the "aborq" call.
ASA095	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned the return code "ERROR" on the "relrs" call.
ASA097	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_RELEASE_IND". OSS returned a bad return code on the "relrs" call.
ASA099	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ABORT_REQ". OSS returned an invalid value for "abort diagnostic".
ASA100	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ABORT_REQ". OSS returned the return code "ERROR" on the "aborq" call.
ASA101	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ABORT_IND" T Bad return code from "PutElement".
ASA102	SX	XAP-TP component. The module "KCOASAM" was called with the opcode "ASAM_ABORT_REQ". OSS returned a bad return code on the "aborq" call.
ASA104	AX	XAP-TP component. The function "bBuildPAddr" of the module KCOASAM was called. The presentation selector of a local access point or of a remote partner is too long.
ASA105	AX	XAP-TP component. The function "bBuildPAddr" of the module KCOASAM was called. The session selector of a local access point or of a remote partner is too long.
ASA116	SX	XAP-TP component. The function "BuildRemoteAet" of the module KCOASAM was called. The APT has "form2" format, but the AEQ does‘t have "form2" format.
ASA120	SX	XAP-TP component. The function "BuildRemoteAet" of the module KCOASAM was called. the APT has the "form1" format, but the AEQ does‘t have "form1" format.
ASA122	SX	XAP-TP component. The function "BuildRemoteAet" of the module KCOASAM was called. The APT has neither the "form1" format nor the "form2" format.
ASA128	M	XAP-TP component. The „CopyDefinedContext“ function of the module KCOASAM was called. Bad return code of „PutElement“.

Code	Group	Reason
ASA137	M	XAP-TP component The module “KCOASAM” was called with the opcode “ASAM_GO_IND”. The association is not locked.
ASA139	SX	XAP-TP component. KCOASAM was called with the operation code ASAM_P_ABORT_IND. Bad return code from "PutElement".
ASA151	SX	XAP-TP component. KCOASAM was called with the operation code ASAM_RELEASE_IND. Bad return code from "PutElement".
ASA152	SX	XAP-TP component. KCOASAM was called with the operation code ASAM_ABORT_REQ. Bad return code from "PutElement".
ASA153	SX	XAP-TP component. KCOASAM was called with the operation code ASAM_ABORT_REQ. Bad return code from “SacfSeparator”.
ASA155	SX	XAP-TP component. KCOASAM was called with the operation code ASAM_ABORT_IND. Bad return code from “SacfSeparator”.
ASA156	SX	XAP-TP component. KCOASAM was called with the operation code ASAM_P_ABORT_IND. Bad return code from “SacfSeparator”.
ASA157	SX	XAP-TP component. KCOASAM was called with the operation code ASAM_RELEASE_IND. Bad return code from “SacfSeparator”.
ASIO01	AU	File is not identified as open during KCSASIO call.
ASIO02	AU	Invalid (i.e. too large) position for writing to file detected when calling KCSASIO. Additional information: The file name the the current position are are output in an addtitional message "K078 KCSASIO ...".
ASIO03	AU	Invalid (i.e. too large) write request for writing to file detected when calling KCSASIO. Additional information: See ASIO02
ASIO04	AU	Invalid (i.e. too large) position for reading from file detected when calling KCSASIO. Additional information: See ASIO02
ASIO05	AU	Invalid (i.e. too large) read request for reading from file detected when calling KCSASIO. Additional information: See ASIO02
ASIO06	AU	Generic Error on positioning in file when calling KCSASIO.

Code	Group	Reason
ASIS99	D	Normal execution of command "KDCSHUT KILL" or the call to the program interface for administration with the opcode KC_SHUTDOWN and the sub-opcode1 KC-KILL.
ATAxxx	AU	Error on connecting a shared-memory to the work process. If xxx is the value for EINVAL, there is a conflict between a shared-memory of the user and a shared-memory of the UTM application. If a shared object is generated and used with LOADMODE=ONCALL, this can also cause an error with the value EINVAL. Response: in these cases, the error can be avoided by deactivating the shared-memory monitoring facility on starting the application.
BFMM05	M	UTM cluster application It was not possible to request another buffer segment. The virtual address space may be too small.
BFMM21	ASU	UTM cluster application A timeout occurred while releasing a file lock. Action see CCFG19.
BFMM22	SU	UTM cluster application A bad return code was returned when releasing a file lock.
BRSREM	F	UTM application was terminated with the KDCREM utility.
CACHT1	F	Another task has terminated the application abnormally (=continuation dump)
CACHT2	F	(See CAHCT1) Another task has terminated the application abnormally (=continuation dump)
CACHT3	F	(See CAHCT1) Another task has terminated the application abnormally (=continuation dump)
CACHT4	F	(See CAHCT1) Another task has terminated the application abnormally (=continuation dump)
CACHT5	F	(See CAHCT1) Another task has terminated the application abnormally (=continuation dump)
CC-...	D	Diagnostic dump generated on the basis of a primary KDCS return code. The prefix CC- is followed by the primary KDCS return code (e.g. CC-84Z). Activation and deactivation via the message dump function.
CCFG07	SU	UTM cluster application Module KCCCFG, entry KCCCGFB Bad return code from KCSGLHN.
CCFG19	ASU	UTM cluster application Module KCCCFG, opcode CCFG_READ_CLUSTER_FILE Timeout when requesting shared lock. Action see section „Actions when locking UTM cluster files“ below this table.

Code	Group	Reason
CCFG20	SU	UTM cluster application Module KCCCFG, opcode CCFG_READ_CLUSTER_FILE Bad return code when requesting shared lock.
CCFG21	ASU	UTM cluster application Module KCCCFG, Timeout when requesting exclusive lock. Action see CCFG19.
CCFG22	SU	UTM cluster application Module KCCCFG Bad return code when requesting the exclusive lock.
CCFG29	ASU	UTM cluster application Module KCCCFG, timeout when releasing file lock. Action see CCFG19.
CCFG30	SU	UTM cluster application Module KCCCFG, opcode CCFG_READ_CLUSTER_FILE Bad return code when releasing the file lock.
CCFG32	ASU	UTM cluster application Module KCCCFG, opcode CCFG_REGISTER_COMPLETE Invalid version of cluster configuration file.
CCFG33	ASU	UTM cluster application Module KCCCFG, opcode CCFG_REGISTER_COMPLETE Corrupt cluster configuration file.
CCFG34	ASU	UTM cluster application Module KCCCFG, opcode CCFG_REGISTER_COMPLETE Invalid application name in cluster configuration file.
CCFG35	ASU	UTM cluster application Module KCCCFG, opcode CCFG_REGISTER_COMPLETE The cluster configuration file was regenerated during operation.
CCFG41	ASU	UTM cluster application Module KCCCFG, opcode CCFG_OPEN_CLUSTER_FILE Timeout when initializing global lock. Action see CCFG19.
CCFG42	SU	UTM cluster application Module KCCCFG, opcode CCFG_OPEN_CLUSTER_FILE Bad return code when initializing the global lock.
CCFG43	ASU	UTM cluster application Module KCCCFG, opcode CCFG_CLOSE_CLUSTER_FILE Timeout when destroying global lock. Action see CCFG19.

Code	Group	Reason
CCFG44	SU	UTM cluster application Module KCCCFG, opcode CCFG_CLOSE_CLUSTER_FILE Bad return code when destroying the global lock.
CCFG45	SU	UTM cluster application Module KCCCFG, opcode CCFG_INIT_KAA Bad return code from KCSGLHN
CCFG49	S	UTM cluster application Module KCCCFG, opcode READ_CLUSTER_FILE Bad return code when reading the cluster file.
CCFG50	A	UTM cluster application Module KCCCFG, opcode READ_CLUSTER_FILE Corrupt cluster file.
CCFG51	A	UTM cluster application Module KCCCFG, opcode READ_CLUSTER_FILE Invalid version of cluster configuration file.
CCFG52	A	UTM cluster application Module KCCCFG, opcode READ_CLUSTER_FILE Invalid application name in cluster configuration file.
CCFG53	A	UTM cluster application Module KCCCFG, opcode READ_CLUSTER_FILE The cluster configuration file was regenerated during operation.
CCFG55	A	UTM cluster application Module KCCCFG, opcode NODE_FAILURE Corrupt cluster configuration file.
CCFG56	A	UTM cluster application Module KCCCFG, opcode NODE_FAILURE Invalid version of cluster configuration file.
CCFG57	A	UTM cluster application Module KCCCFG, opcode NODE_FAILURE Invalid application name in cluster configuration file.
CCFG58	A	UTM cluster application Module KCCCFG, opcode NODE_FAILURE The cluster configuration file was regenerated during operation.
CCFG62	ASU	UTM cluster application Module KCCCFG, opcode CCFG_RESET_START_SERIALIZATION Timeout when releasing lock. Action see CCFG19.
CCFG63	SU	UTM cluster application Module KCCCFG, opcode CCFG_RESET_START_SERIALIZATION Bad return code when releasing lock.

Code	Group	Reason
CCFG64	ASU	UTM cluster application Module KCCCFG, opcode CCFG_RESET_START_SERIALIZATION Timeout when destroying lock. Action see CCFG19.
CCFG65	SU	UTM cluster application Module KCCCFG, opcode CCFG_RESET_START_SERIALIZATION Bad return code when destroying lock.
CCFG71	AU	UTM cluster application KCCCFG module, opcode READ_CLUSTER_FILE or CHECK_CLUSTER_FILE The local node is marked as failed in the cluster configuration file.
CCFG72	AU	UTM cluster application KCCCFG module, opcode CCFG_NODE_FAILURE The local node is marked as failed in the cluster configuration file.
CCKF02	SU	UTM cluster application Module KCCCKF Unexpected return code from KCCFILA when opening the KDCFILE.
CCKF03	SU	UTM cluster application Module KCCCKF Unexpected return code from KCCGFLK on unlock.
CCKF04	SU	UTM cluster application Module KCCCKF Unexpected return code from KCCFILA when closing the KDCFILE.
CDTN02	M	XAP-TP component. Module KCOCOHF, function CheckDtnidTtnid(). The macro mGetBufferSpace() issued the return codeLB_NOREM.
CFMM05	D	On send, the length in DCF does not match that specified in the letter header.
CLREST	D	UTM cluster application The dump is only generated during a node application warm start with test mode enabled. It is used for diagnostic purposes following possible errors during the warm start of the cluster.
COBOL6	A	The internal NETCOBOL function CBL_SETJMP was called within libwork.so although NETCOBOL is not suported on AIX.
COBOL7	A	The internal NETCOBOL function CBL_LONGJMP was called within libwork.so although NETCOBOL is not suported on AIX.
COBOL8	A	The internal function coblongjump was called within libwork.so although no COBOL program is generated.
COBOL9	A	The internal function cobsavenv2 was called within libwork.so although no COBOL program is generated.

Code	Group	Reason
CONS03	A	The KDCFILE was overwritten during operation. Possible cause: KDCDEF run in current application.
CSND04	MX	XAP-TP component. Invalid return code after calling PutElement() to request a dynamic buffer for concatenator send data.
CSND05	SX	XAP-TP component. Invalid return code after calling an OSS presentation function.
CTPF04	M	XAP-TP component: The return code of PutElement was not equal to DM_OK.
DC....	D	Diagnostic dump generated on the basis of a secondary KDCS return code. The prefix DC- is followed by the secondary KDCS return code (e.g. DCKS17). Activation and deactivation via the message dump function.
DIAGCL	D	UTM cluster application A node application has been terminated abnormally. Before termination, all the other node applications were informed of the abnormal termination if this was still possible. The informed node applications write a diagnostic dump.
DIAGDP	D	A diagnostic dump has been generated by means of the administration command "KDCDIAG DUMP=YES" or by calling the program interface for administration with the opcode KC_CREATE_DUMP.
DMCA00	M	XAP-TP component. Module KCOCODM, function ConnectDynMemArea(). The function ConnectSharedMem() issued the return code MEM_NOMEM.
EHHP00	M	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The return code from mGetBufferSpace() was not equal to LB_OK.
EHHP01	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_datain() returned the return code P_ERROR.function
EHHP02	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_datain() returned the return code P_INVREF.
EHHP03	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_datain() returned an unknown return code
EHHP04	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_typein() returned the return code P_ERROR.
EHHP05	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_typein() returned the return code P_INVREF.

Code	Group	Reason
EHHP06	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_typein() returned an unknown return code .
EHHP07	SX	XAP-TP function. Module KCOXFEH, function HandlePresEvent(). The function CalcUserDataLth() returns 0 for length of user data after calling the OSS function p_typein().
EHHP08	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_synin() returned the return code P_ERROR.
EHHP09	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_synin() returned the return code P_INVREF.
EHHP10	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_synin() returned an unknown return code.
EHHP12	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The function CalcUserDataLth() returns zero for length of user data after calling the OSS function p_synin().
EHHP13	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_syncf() returned the return code P_ERROR.
EHHP14	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_syncf() returned the return code P_INVREF.
EHHP15	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_syncf() returned an unknown return code.
EHHP17	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The function CalcUserDataLth() returns zero for length of user data after calling the OSS function p_syncf().
EHHP18	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkgin() returned the return code P_ERROR.
EHHP19	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkgin() returned the return code P_INVREF.

Code	Group	Reason
EHHP20	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkgin() returned an unknown return code.
EHHP21	S	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkgin() did not return the token S_T_MINOR.
EHHP22	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkpin() returned the return code P_ERROR.
EHHP23	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkpin() returned the return code P_INVREF.
EHHP24	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkpin() returned an unknown return code.
EHHP25	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_tkpin() did not return the token S_T_MINOR.
EHHP26	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The function CalcUserDataLth() returns zero for length of user data after calling the OSS function p_tkpin().
EHHP27	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_minin() returned the return code P_ERROR.
EHHP28	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_minin() returned the return code P_INVREF.
EHHP29	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_minin() returned an unknown return code.
EHHP30	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The function CalcUserDataLth() returns zero for length of user data after calling the OSS function p_minin().
EHHP31	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_mincf() returned the return code P_ERROR.
EHHP32	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_mincf() returned the return code P_INVREF.

Code	Group	Reason
EHHP33	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function p_mincf() returned an unknown return code.
EHHP34	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The function CalcUserDataLth() returns zero for length of user data after calling the OSS function p_mincf().
EHHP35	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The function o_event() returned the unexpected event P_MAJIN.
EHHP36	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The function o_event() returned the unexpected event P_MAJCF.
EHHP37	SX	XAP-TP component. Module KCOXFEH, function HandlePresEvent(). The OSS function o_event() returned an unexpected event type.
EHRP01	MX	XAP-TP component. Module KCOXFEH, function ReloadPresEvent(). The macro mGetBufferSpace() issued a return code not equal to LB_OK.
EHSP00	SX	XAP-TP component. Module KCOXFEH, function StorePresEvent(). StorePresEvent() is to store a presentation event in DynMem. However, an event is already stored for the corresponding association (mValLth(g, &pAss->PendingEvnt.h) > 0).
EHSP01	M	XAP-TP component. Module KCOXFEH, function StorePresEvent(). The function PutElement() issued the return code DM_NOMEM. Response: Increase the value for the size of the OSI scratch area in the KDCDEF generation (parameter MAX OSI-SCRATCH-AREA).
EKAA11	S	UTM cluster application Timeout when releasing the file lock for the JKAA file. Action see CCFG19.
EKAA12	S	Incorrect return code from KCCGFLK when releasing the file lock for the JKAA file.
EKAA13	S	UTM cluster application Timeout when destroying the file lock for the JKAA file. Action see CCFG19.
EKAA14	S	Incorrect return code from KCCGFLK when destroying the file lock for the JKAA file.

Code	Group	Reason
ENCERR	DSU	UTM expected an encrypted message, but received an unencrypted message. There is no application abort pending. The dump is only for diagnostic purposes.
ENDE01	SU	The KAA pointer in KTA doesn‘t point to a KAA.
ENDE02	AU	Interrupt with locked KAA.
ENDE03	ASU	Abnormal process termination between writing a confirmatory record and the (temporary) end of the UTM transaction during database call.
ENDE04	SU	The cache cannot be released.
ENDE05	SU	The KAA cannot be released.
ENDE06	ASU	Error when opening the SYSLOG file.
ENDE11	DSU	The application was terminated normally, during sign-off of the OSS access point, OSS delivered a return code other than OK. If test mode is activated, a UTM diagnostic dump is initiated.
ENDE12	ASU	A task terminated (either normally or abnormally). Not enough tasks remain to continue processing the application (e.g. because there are tasks in the PGWT).
ENDE14	F	A task terminated (either normally or abnormally) and the task has still not called KCSTRMA. The status of the application is, however, TERM_APPL. KCSTRMA was called with ENDE14 to ensure that all tasks generate a UTM dump if the application terminates abnormally.
ENDE19	ASU	Only system tasks are running for the application. All normal tasks were terminalet. wurden alle beendet. The reason may be an error in the start procedure or a start error when restarting the tasks after PEND ER or program exchange.
ENDPET	A	An UTM-D application cannot be terminated normally, because services still exist with the transaction status PTC (prepare to commit) or because no acknowledgments have been received for asynchronous messages sent to other applications. In this case no UTM dump is generated. openUTM carries a warm start at the next application start.
EVGE00	M	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The return code of the macro mGetBufferSpace() was not equal to LB_OK.
EVGE01	S	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The return parameter <o_echain> contained an invalid value after calling the OSS function o_event().
EVGE03	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_ACS and the user data were not yet received completely on the call.

Code	Group	Reason
EVGE05	ASX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_ERROR.
EVGE06	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_INVEREF.
EVGE07	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_TIMEINT.
EVGE08	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_WAKEINT.
EVGE09	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_SYSTEM.
EVGE10	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_LOOK.
EVGE11	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_TRANSPORT.
EVGE12	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned the return code O_SESSION.
EVGE13	SX	XAP-TP component. Module KCOXFEV, function GetOssEvent(). The OSS function o_event() returned an unknown return code.
EVNT03	ASX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_ERROR.
EVNT04	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_INVEREF.
EVNT05	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_TIMEINT.

Code	Group	Reason
EVNT06	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_WAKEINT.
EVNT07	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_SYSTEM.
EVNT08	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_TRANSPORT.
EVNT09	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_SESSION.
EVNT10	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_PRESENTATION.
EVNT11	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned the return code O_ACSE.
EVNT12	SX	XAP-TP component. Module KCOXFEV, function aputm_event(). The OSS function o_event(O_EVALLOOK) returned an unknown return code.
EXIT00	AU	Illegal exit() call recognized during execution of UTM system coding.
EXPI32	ASU	UTM cluster application Bad return code from KCCGFLK when locking the cluster GSSB file. Action see CCFG19.
EXPI34	ASU	UTM cluster application Bad return code from KCCGFLK when locking the cluster ULS file. Action see CCFG19.
EXPI35	ASU	UTM cluster application Bad return code from KCCGFLK when locking the cluster ULS file. Action see CCFG19.
EXPI36	ASU	UTM cluster application Bad return code from KCCGFLK when locking the cluster GSSB file. Action see CCFG19.

Code	Group	Reason
EXPI51	A	A periodic write was called by a task because the conf. area of the task has become full. The task wishes to write another conf. page but no pages have become free through periodic write. The cause may be a conf. area which was generated too small, with the result that restart information of a single transaction is longer than the conf. area (e.g. transaction with access to "numerous" page pool pages). Action: Increase the number parameter in the generation statement "MAX RECBUF=(number,...).
EXPI95	A	See EXPI51, but occurs in RESET_TA_RTN.
FHCV03	S	Formatting system is not supported.
FMMM10	A	An input message cannot be stored because the page pool is full. Action: Increase page pool by increasing MAX PGPOOL=(number,...) in KDCDEF generation.
FMSM05	DSU	KCDFMSM was called with opcode PRSP_BID even though the session status is not WAIT_OF_BID_RSP.
FMSM12	S	Illegal LSES_WORK_STATE when receiving RTR.
FREE01	AX	XAP-TP component. Module KCOXFFO, function ap_free(). More than APFREE_MAX_TO_REL storage areas are to be released.
GETR00	AU	Record length less than 0 when calling KCSGETR. Possible cause: invalid input file.
GETR01	AU	On reading the record length field of a record, fewer than 4 bytes were read in KCSGETR. Possible cause: invalid input file
GETR02	AU	Record read has a record length < 5 after reading of the record length field in KCSGETR. Possible cause: invalid input file.
GETR03	AU	On reading a record, fewer bytes were read in than expected after reading of the record length field in KCSGETR. Possible cause: invalid input file.
GETR04	AU	On reading a record with KCSGETR an invalid record was found. Possible cause: invalid input file.
GFLKT1	F	UTM cluster application Lock hierarchy infringed after Term Application (continuation dump).
GFLKT2	F	UTM cluster application As for GFLKT1 (continuation dump).
GLHN03	A	The local computer name is longer than eight characters.
GMDT11	SU	UTM cluster application Module KCCGMDT, opcode UPDATE_AND_UNLOCK_GSSB. unexpected return code from KCCGFLK (lock GSSB file). Action see CCFG19.

Code	Group	Reason
GSYS00	S	XAP-TP component. Module KCOCOHF, function GetSystemInfo(). The function uname() issued a negative return code.
INPERR	AD	Error in INPUT exit was detected.
IOyxxx	ASU	ASIO return code: An unrecoverable error has occurred during file processing, yxxx = DMS error code. Response: See section "Error codes during file processing"
IPC000	ASU	IPC shared memory segment has been overwritten. A check takes place with every KDCS call.
IPC035	A	Error on locking the IPC shared memory segment, this may be caused by kdcrem being called while the application is still running.
IPC037	FU	IUTMIPC indicates that the application is terminated abnormally (= continuation dump). See also U306 with insert UERRNO=37.
IPCEND	F	A work process terminated outside of the control of openUTM. The application is therefore terminated. Possible cause: see note for message U231
IPCREM	F	UTM application was terminated with the KDCREM utility.
ISLP11	SU	Protocol error.
ISLP12	AU	Task-specific buffer for restart information too small. Action: Increase MAX RECBUF=(...,length) specification in KDCDEF generation
ISLPT1	F	Another task has terminated the application abnormally (=continuation dump)
ISLPT4	F	See ISLPT1
JFC011	ASU	UTM cluster application Timeout on KCCGFLK call in KCCJFCT with opcode CLOSE_FILES. Action see CCFG19.
JFC012	ASU	UTM cluster application Error on KCCGFLK call in KCCJFCT with opcode CLOSE_FILES. Action see CCFG19.
JFC016	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT with opcode CHECK_UNPROC_ENTRIES.
JFC018	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode CHECK_UNPROC_ENTRIES.
JFC055	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT with opcode SET_NODE_PROCESSING_STATE.

Code	Group	Reason
JFC058	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode SET_NODE_PROCESSING_STATE.
JFC067	SU	UTM cluster application Error when reading the administration page in KCCJFCT with opcode SET_NODE_CR_SEQNR.
JFC071	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT with opcode GET_NODE_CR_SEQNR.
JFC072	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode GET_NODE_CR_SEQNR.
JFC076	SU	UTM cluster application Error on the readControlPage call in KCCJFCT with opcode RESET_NODE_CR_SEQNR.
JFC082	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode RESET_DYNADM_LOCK.
JFC083	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT with opcode RESET_DYNADM_LOCK.
JFC400	ASU	UTM cluster application Timeout on KCCGFLK call (lock) in KCCJFCT, internal function setFileLock. Action see CCFG19.
JFC401	ASU	UTM cluster application Error on KCCGFLK call (lock) in KCCJFCT, internal function setFileLock. Action see CCFG19.
JFC402	SU	UTM cluster application Error on KCCGFLK call (lock) in KCCJFCT, internal function setFileLock.
JFC404	SU	UTM cluster application Timeout on KCCGFLK call (unlock) in KCCJFCT, internal function release-FileLock.
JFC405	ASU	UTM cluster application Error on KCCGFLK call (unlock) in KCCJFCT, internal function release-FileLock. Action see CCFG19.
JFC420	SU	UTM cluster application Error on KCCFILA call (write), first administration page in the journal file in KCCJFCT, internal function initControlPages.
JFC421	SU	UTM cluster application Error on KCCFILA call (write) in KCCJFCT, internal function initControlPages.

Code	Group	Reason
JFC430	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT, internal function switchFiles.
JFC431	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT, internal function switchFiles.
JFC460	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT, internal function check-ControlPages.
JFC501	SU	UTM cluster application Error on KCCFILA call in KCCJFCT, internal function readPages.
JFC507	SU	UTM cluster application Error on KCCFILA call in KCCJFCT, internal function writePages.
JFC512	SU	UTM cluster application Error on KCCFILA call in KCCJFCT, internal function writeControlPage.
JFC550	SU	UTM cluster application Error when reading an administration page in KCCJFCT with opcode GET_JOURNAL_FILE_INFO.
JFC561	SU	UTM cluster application Error on readControlPage call in KCCJFCT with opcode SET_COPY_STATE.
JFC570	SU	UTM cluster application Error on readControlPage call in KCCJFCT with opcode SET_KAA_INFO.
JFC572	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode SET_KAA_INFO.
JFC580	SU	UTM cluster application Error on readControlPage call in KCCJFCT with opcode GET_KAA_INFO.
JFC581	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode GET_KAA_INFO.
JFC583	SU	UTM cluster application Error on the 3rd readControlPage call in KCCJFCT with opcode GET_KAA_INFO.
JFC584	SU	UTM cluster application Error on the 4th readControlPage call in KCCJFCT with opcode GET_KAA_INFO.
JFC590	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT with opcode SET_GLOBAL_ADM_LOCK.

Code	Group	Reason
JFC591	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode SET_GLOBAL_ADM_LOCK.
JFC594	SU	UTM cluster application Error on the 3rd readControlPage call in KCCJFCT with opcode GET_GLOBAL_ADM_LOCK.
JFC595	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT with opcode RESET_GLOBAL_ADM_LOCK.
JFC596	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode RESET_GLOBAL_ADM_LOCK.
JFC598	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT with opcode GET_GLOBAL_ADM_LOCK.
JFC599	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT with opcode GET_GLOBAL_ADM_LOCK.
JFC611	SU	UTM cluster application Error on the 2nd readControlPage call in KCCJFCT, routine checkAllProcessed.
JFC613	SU	UTM cluster application Error on the first readControlPage call in KCCJFCT, routine checkAllProcessed.
JFC626	SU	UTM cluster application Error on readControlPage call in KCCJFCT, routine checkAndRepair.
LATC01	FSU	The lock mechanism is not functioning at the time when KCSLATC is called. See LATC02
LATC02	ASU	Error when calling KCXLOCK of the ATC lock bourse. Because the locks are implemented with semaphores in openUTM for Unix and Windows systems, an error code was returned for a semaphore operation. The cause may be that the user deleted a semaphore of the application (either by means of a Unix/Windows system function or KDCREM).
LATC03	SU	Max. number of locks per task exceeded.
LATC04	SU	KCSLATC called under IPC lock.
LATCT1	F	During a request for a lock, the status of an application is set to TERM APPLICATION. The task is also terminated (= continuation dump).
LCAC01	FSU	The lock mechanism is not functioning at the time when KCSLKAA / KCSLCAC / KCSPCMM is called. See LCAC02.

Code	Group	Reason
LCAC02	ASU	Error on calling KCXLOCK of the KAA/CACH/PCMM lock bourse. Since in openUTM for Unix and Windows systems the locks are implemented by means of semaphores, as error code was reported with a semaphore operation. The reason may be that the user has deleted a semaphore for the application (either through a Unix/windows system function or through KDCREM).
LCAC03	SU	Max. number of locks per task exceeded.
LCAC04	SU	KCSLCAC called under IPC lock.
LCACT1	F	During a request for a lock, the status of an application is set to TERM APPLICATION. The task is also terminated (= continuation dump).
LKAA01	FSU	The lock mechanism is not functioning at the time when KCSLKAA is called. See LKAA02.
LKAA02	ASU	Error on calling KCXLOCK of the KAA/CACH/PCMM lock bourse. Since in openUTM for Unix and Windows systems the locks are implemented by means of semaphores, as error code was reported with a semaphore operation. The reason may be that the user has deleted a semaphore for the application (either through a Unix/windows system function or through KDCREM).
LKAA03	SU	Max. number of locks per task exceeded.
LKAA04	SU	KCSLKAA called under IPC lock.
LKAAT1	F	During a request for a lock, the status of an application is set to TERM APPLICATION. The task is also terminated (= continuation dump).
LKLC26	FU	During locking of an entry, a situation is detected where the service is already entered in a queue chaining facility. System error or continuation dump if the application is currently being abnormally terminated (through error or KDCSHUT KILL). The case of a continuation dump may be recognized from the fact that the reason for the abort in message K060 is not LKLC26, and that a further dump has already been generated.
LKLC42	AU	Bourse wait time elapsed. Response: Change KDCDEF generation, increase the RESWAIT= (... , time2) parameter in the MAX statement. One cause for the elapse of the time limit may be the creation of diagnostic information (gcore) locked in another work process.
LKLC51	AU	See LKLC42
LKLC64	AU	See LKLC42
LKLCT1	F	Another task has terminated the application abnormally (=continuation dump)
LKLCT2	F	See LKLCT1
LKLCT3	F	See LKLCT1
LKLCT4	F	See LKLCT1

Code	Group	Reason
LKMTC0	F	Another task has abnormally terminated the application (= continuation dump before KCSBRSE call)
LKMTC1	F	Another task has abnormally terminated the application (=continuation dump after KCSBRSE call)
LKMTKP	D	This diagnostic dump is generated if TESTMODE=ON provided that the PEND KP flag is not reset in the lock field when the unlock is performed.
LPCM01	FSU	The lock mechanism is not functioning at the time when KCSPCMM is called. See LPCM02.
LPCM02	ASU	Error when calling KCXLOCK of the ATC lock bourse. Because the locks are implemented with semaphores in openUTM for for Unix and Windows systems, an error code was returned for a semaphore operation. The cause may be that the user deleted a semaphore of the application (either by means of a Unix/Windows system function or KDCREM).
LPCM03	SU	Max. number of tasks exceeded.
LPCM04	SU	KCSLPCM called under IPC lock.
LPCMT1	F	During a request for a lock, the status of an application is set to TERM APPLICATION. The task is also terminated (= continuation dump).
LWRT02	SU	Error when opening the user log file <filebase>/USLA/yyyy (copy A).
LWRT03	SU	Error when opening the user log file <filebase>/USLB/yyyy (copy B).
LWRT04	ASU	Error on positioning in user log file <filebase>/USLA/yyyy. Write manipulations may have been performed to the current file generation or the base may have been modified. Use the KDCLOG command to switch while the application is running. Action if the next UTM application start is also aborted with LWRT04: Delete the user log file (i.e.the entire USLA directory) and regenerate it.
LWRT05	ASU	Error on positioning in user log file <filebase>/USLB/yyyy The reason may be as described for LWRT04.
LWRT06	SU	Error on positioning in user log file <filebase>/USLB/yyyy during positioning back to beginning of file following an unsuccessfully concluded write job (e.g. due to disk storage bottleneck).
LWRT07	SU	Error on positioning in user log file <filebase>/USLA/yyyy during positioning back to beginning of file following an unsuccessfully concluded write job (e.g. due to disk storage bottleneck).
LWRT08	SU	Error with PUT call for writing an LPUT record to the user log file <filebase>/USLA/yyyy.
LWRT09	SU	Error with PUT call for writing an LPUT record to the user log file <filebase>/USLB/yyyy.
LWRT10	SU	Error on closing call for the user log file <filebase>/USLB/yyyy.
LWRT11	SU	Error on closing call for the user log file <filebase>/USLA/yyyy.

Code	Group	Reason
LWRT17	A	KCSLWRT requested a buffer via KCSALME for writing to the user log file, and the buffer cannot be made available. Action: Check memory requirements and operating system generation.
LWRT19	SU	Error on positioning to end of file for copy A of user log file.
LWRT20	SU	Same as LWRT19, but for copy B.
MACF02	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MACF03	M	XAP-TP component. The return code of SetTimer() was not equal to TI_OK.
MACF04	M	XAP-TP component. The return code of GetLogRecord was not equal to MACF_OK.
ME....	D	Diagnostic dump generated on the basis of a specific UTM message. The prefix ME is followed by the message number of the UTM message (e.g. MEK135). Activation and deactivation via the message dump function.
MFCR04	M	XAP-TP component. The return code of GetLogRecord was MACF_NO_MEM.
MFCR07	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR08	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR09	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR10	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR11	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR16	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR17	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR18	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR19	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR20	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFCR21	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.

Code	Group	Reason
MFCR24	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFDM03	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFDM04	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFDM05	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFDM06	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM05	AX	XAP-TP component. On TP_RECOVER_REQ, no free dialog table entry for a transaction branch is available. Possible cause: the number of associations in the previous application run was greater than the number of associations in the current run.
MFRM06	AX	XAP-TP component. See MFRM05
MFRM07	AX	XAP-TP component. See MFRM05
MFRM08	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFRM09	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM10	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM11	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM12	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM13	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM14	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM15	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM16	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM17	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.

Code	Group	Reason
MFRM18	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM19	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM21	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM24	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFRM25	AX	XAP-TP component. No free table entry available for a log damage record. Response: delete log damage records with TP_UPDATE_LOG_DAMAGE_REQ or increase the value of nMaxLogDamRec.
MFT102	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFT103	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT104	M	XAP-TP component. The return code of ChangeDescriptor was not equal to DM_OK.
MFT105	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT106	M	XAP-TP component. The return code of CopyElement was not equal to DM_OK.
MFT107	M	XAP-TP component. The return code of CopyElement was not equal to DM_OK.
MFT108	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFT109	M	XAP-TP component. The return code of CopyElement was not equal to DM_OK.
MFT110	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT111	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT113	M	XAP-TP component. The return code of ChangeDescriptor was not equal to DM_OK.
MFT114	M	XAP-TP component. The return code of CopyElement was not equal to DM_OK.
MFT115	M	XAP-TP component. The return code of ChangeDescriptor was not equal to DM_OK.

Code	Group	Reason
MFT119	M	XAP-TP component. The return code of GetLogRecord() was not equal to MACF_OK.
MFT120	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT121	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT122	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT123	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT124	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT126	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT127	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT128	M	XAP-TP component. The return code of CopyElement was not equal to DM_OK.
MFT129	M	XAP-TP component. The return code of CopyElement was not equal to DM_OK.
MFT130	M	XAP-TP component The return code of PutElement was not equal to DM_OK.
MFT131	M	XAP-TP component The return code of PutElement was not equal to DM_OK.
MFT132	M	XAP-TP component The return code of PutElement was not equal to DM_OK.
MFT133	M	XAP-TP component. The return code of CopyElement was not equal to DM_OK.
MFT134	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT135	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT138	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT139	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFT141	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.

Code	Group	Reason
MFT142	M	XAP-TP component. The function CopyElement issued a return code other than DM_OK.
MFT147	M	XAP-TP component. The function PutElement issued a return code other than DM_OK.
MFT151	M	XAP-TP component. The function PutElement issued a return code other than DM_OK.
MFTP03	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFTP04	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP05	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFTP06	M	XAP-TP component. The return code of PutElement was not equal to DM_OK.
MFTP07	M	XAP-TP component. The return code of SetTimer was not equal to TI_OK.
MFTP10	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP11	M	XAP-TP component. The return code of RequestBuffer() was not equal to LB_OK.
MFTP12	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP15	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP16	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP17	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP18	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP19	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP20	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MFTP24	M	XAP-TP component. The return code of the macro mGetBufferSpace() was not equal to LB_OK.
MOVE03	AU	Invalid overlapping of destination and source area in KCSMOVE or KCSMOVEP.

Code	Group	Reason
MSG000	DU	KCSCRMS was called with an invalid message ID. A diagnostic dump is written and the message "K000 MESSAGE NOT DEFINED" is output.
NDRCVY	D	The dump is only generated at the end of a node recovery with test mode enabled. It is used for diagnostic purposes following possible errors after a node recovery.
NET001	M	Error when requesting memory area for the Bcamappl table in the net process.
NET002	M	Error when requesting memory area for the connection table in the net process.
NET003	M	Error when requesting memory area for the application table in the net process.
NET006	ASU	In the net process, at least one Bcamappl could not attached to the transport system.
NET007	M	Error when requesting memory area for the child process table in utmnetm.
NET011	SU	Error when registering the CMX callback routine in utmnet.
NET014	SU	Error when calling t_conin() in the net prcoess.
NET019	SU	In the net process, an invalid DATAGO event received from the transport system.
NET020	SU	Error when calling t_event() in the net prcoess.
NET022	SU	Error when calling t_info() in the net prcoess.
NET023	SU	Error when calling t_datarq() in the net prcoess.
NET031	SU	Error when initializing the thread attributes in the net prcoess.
NET032	SU	Error when setting the detach attributes for the thread in the net prcoess.
NET033	SU	Error when creating the threads for the shared waiting point in the net prcoess.
NET034	SU	Error when creating the receive socket for the shared waiting point in the net prcoess.
NET035	SU	Error when setting the socket attribute REUSEADDR for the receive socket in the net prcoess.
NET036	SU	Error on bind() call for the receive socket in the net prcoess.
NET037	SU	Error when detecting the listener port for the receive socket in the net prcoess.
NET038	SU	Error when initializing the socket environment in the net prcoess.
NET039	A	On the computer there is not installed the socket library version that is necessary for the net process.
NET051	SU	An invalid event was reveived from transport system in utmnetm.
NMTE00	M	XAP-TP component. Module KCOCOHF, function NewMemTabEntry(). The function RequestBuffer() issued the return code LB_NOMEM.

Code	Group	Reason
NMTE02	M	XAP-TP component. Module KCOCOHF, function NewMemTabEntry(). The macro mGetBufferSpace() issued the return code LB_NOMEM.
NRDBER	A	The node recovery must be terminated abnormally due to errors during database recovery. No UTM dump is generated.
ODIA00	SX	XAP-TP component. Module KCOCOHF, function OssDiagInfo(). The OSS function o_error() returned the return code O_ERROR.
ODIA01	SX	XAP-TP component. Module KCOCOHF, function OssDiagInfo(). The OSS function o_error() returned the return code O_INVEREF.
ODIA02	SX	XAP-TP component. Module KCOCOHF, function OssDiagInfo(). The OSS function o_error() returned an invalid return code.
OREA00	SX	XAP-TP component. The OSS function o_reason() returned the return code O_ERROR.
OREA01	SX	XAP-TP component. The OSS function o_reason() returned the return code O_INVEREF.
OREA02	SX	XAP-TP component. The OSS function o_reason() returned an unknown return code.
OSAFT2	F	Return code APEXT_TERMAPPL from XAP-TP.
OSGO01	SX	XAP-TP component. Module KCOCOHF, function OssGo(). The OSS function o_go() returned the return code O_INVCREF.
OSGO02	SX	XAP-TP component. Module KCOCOHF, function OssGo(). The OSS function o_go() returned the return code O_ERROR.
OSGO03	SX	XAP-TP component. Module KCOCOHF, function OssGo(). The OSS function o_go() returned an unexpected return code.
OSTM01	AS	Serious error during interaction with database system.
OSTM05	AU	Task-specific buffer for restart information is too small. Response: Increase MAX RECBUF=(...,length) in the KDCDEF generation.
OSTM06	AS	Serious error during interaction with database system.
OSTM07	A	A log record cannot be backed up, since the page pool is full. Response: Increase the size of the page pool. Do this by increasing MAX PGPOOL=(number,...) in the KDCDEF generation.
OSTP01	SX	XAP-TP component. Module KCOCOHF, function OssStop(). The OSS function o_stop() returned the return code O_INVCREF.

Code	Group	Reason
OSTP02	SX	XAP-TP component. Module KCOCOHF, function OssStop(). The OSS function o_stop() returned the return code O_ERROR.
OSTP03	SX	XAP-TP component. Module KCOCOHF, function OssStop(). The OSS function o_stop() returned an unexpected return code.
PCMM05	AU	For KCSPCMM with the opcode Get Pagechain, the NR_PAGES parameter is invalid or larger than the number of all page pool pages. Response: increase MAX PGPOOL=number in KDCDEF generation.
PCTR00	M	XAP-TP component. Module KCOCOHF, function PrepareCtrlReq(). The macro mGetBufferSpace() issued the return code LB_NOMEM.
PEND02	A	No further TACB can be written for "System PEND ER" because the page pool on KDCFILE is full. Action: Increase page pool by increasing MAX PGPOOL=(number,...) in KDCDEF generation.
PEND03	AS	The DB system has reported a serious error, see DB-DIAGAREA in UTM dump. This error can occur if, in the RMXA statement of the Oracle database system, the SesTm parameter was set to a value smaller than the KDCDEF parameter for PEND KP monitoring.
PEND04	S	The DB system supplied an impermissible return code, see DB-DIAGAREA in UTM dump.
PEND05	AS	See PEND03
PEND07	ASU	Signal between writing of a confirmatory record and the (temporary) end of the UTM transaction (KCSEXPI call END_TA). Possible cause: error in the DB connection module. A timer signal (SIGALRM) during this period is ignored.
PEND11	S	Inconsistent XID
PEND26	A	With "System PEND ER" for a socket client, it is not possible to write a K017 message for a KDCDISP that may be required later, since the page pool on KDCFILE is full. Measure: As with PEND02
PEND97	A	In the case of a "System PEND ER" for a socket client, it is no longer possible to write a K017 message for any KDCDISP that may subsequently be necessary because the cluster page pool is full. Action: Increase the size of the cluster page pool by setting a larger value for CLUSTER PGPOOL=(number,...) in the KDCDEF generation.
PEND98	A	In the case of "System PEND ER", it is no longer possible to write a TACB because the cluster page pool is full. Action: Increase the size of the cluster page pool by setting a larger value for CLUSTER PGPOOL=(number,...) in the KDCDEF generation.

Code	Group	Reason
PENDER	ADU	Not caused by the application being aborted, but by a user-specified PEND ER, or a UTM-internal PEND ER following KCRCCC ≥ 70Z or a signal .
PENDT1	F	Another task has terminated the application abnormally (=continuation dump before KCSBRSE call).
PENDT2	F	Another task has terminated the application abnormally (=continuation dump after KCSBRSE call).
PENDT3	F	Another task has terminated the application abnormally (= continuation dump after call by KDCROOT with PEND ER).
PIPE39	AU	Error setting up an instance of a named pipe to the Windows system. Possible cause: There are two UTM applications of the same name on the system..
PLCA00	SX	XAP-TP component. Module KCOXFPL, function ActivateCmxCallback(). The CMX function t_callback() returned the return code T_ERROR.
PLCC00	SX	XAP-TP component. Module KCOXFPL, function CmxCallback(). The system function select() returned an unexpected return code.
PLCC01	S	XAP-TP component. Module KCOXFPL, function CmxCallback(). The system function select() returned an unkown return code.
PLCD00	SX	XAP-TP component. Module KCOXFPL, functionDeactivateCmxCallback(). The CMX function t_callback() returned the return code T_ERROR.
PMIO20	ASU	Integrity IDs of page inconsistent both in original and also in duplicate file. Possible cause: KDCFILE files were overwritten during live operation by copying in the start procedure. Action in this event: Correct start procedure. If this is not the cause, write PR. The following documentation is required for diagnosis: UTM dump, KDCFILE files, start procedure and task/process log. In the case of cluster applications, the cluster page pool files are also required.
PMIO22	ASU	The page type specified in the cache control table and the page type in the header of the scanned page do not match. Possible cause: see PMIO22 Action in this case: as for PMIO22
PMIO23	ASU	The page pool page to be read or written is not allocated. Cause and response: as for PMIO22
PMIO32	SU	UTM cluster application Timeout on locking a file of the cluster page pool. Response: See section „Actions when locking UTM cluster files“ below this table.

Code	Group	Reason
POLL03	MX	XAP-TP component. Module KCOXFPL, function ap_poll(). The return code of the macro mGetBufferSpace() was not equal to LB_OK.
POLL05	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() reports an ACSE event and the user data were not yet received completely on the o_event() call.
POLL07	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_ERROR.
POLL08	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_INVEREF.
POLL09	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_TIMEINT.
POLL10	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_WAKEINT.
POLL11	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_SYSTEM.
POLL12	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function ap_poll() returned the return code O_LOOK.
POLL13	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_TRANSPORT.
POLL14	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_SESSION.
POLL15	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned the return code O_SHUTDOWN.
POLL16	SX	XAP-TP component. Module KCOXFPL, function ap_poll(). The OSS function o_event() returned an unexpected return code.
PPMM10	ASU	UTM cluster application Timeout in the lockFile routine during attempt to lock a cluster page pool file. Action see CCFG19.

Code	Group	Reason
PPMM12	SU	UTM cluster application Timeout in the unlockFile routine during attempt to unlock a cluster page pool file. Action see CCFG19.
PPMM14	SU	UTM cluster application Bad return code from KCSBFMM in routine readControlPage.
PPMM15	ASU	UTM cluster application Control page of the cluster page pool has an invalid page header in the readControlPage routine.
PPMM41	SU	UTM cluster application Error when reading or writing the first CONS page of a cluster page pool file.
PPMM42	SU	UTM cluster application Bad return code from KCSBFMM when requesting a buffer.
PUTR01	AU	Not enough bytes could be written in KCSPUTR. Possible cause: disk storage bottleneck.
PWRT03	AMU	KCSPWRT has requested memory (via KCSALME), but did not get it. Action: Check memory requirements operating system generation.
PWRT06	F	With KCSPWRT, term application was called during the Periodic Write by another UTM task of the application (= continuation dump).
RALC01	AU	The KDCFILE was overwritten by KDCDEF during the application run (user error). Index is outside the permitted range when KCSRALC is called.
RCV009	MX	XAP-TP component. Module KCOXFRV, function ap_rcv(). The return code of the function CopyElement() was not equal to DM_OK.
RCV012	MX	XAP-TP component. Module KCOXFRV, function ap_rcv(). Inconsistency in the Boolean variables <bSwitchToNextTtnid> and <bClearTtnid>.
REME01	SU	Illegal system return code on KCSREME call.
RESTRT	D	The dump is only created during a warm start and in the debugging mode. It is used to diagnose any eventual errors in the warm start.
RQOB00	M	XAP-TP component. Module KCOCOHF, function ReqOssInBuff(). The function RequestBuffer() issued the return code LB_NOMEM.
RVCS03	M	XAP-TP component. Module KCOXFRV, function CheckSaRetc(). The return code issued by the function SetAttribute() was SA_NOMEM.
SACT14	MX	XAP-TP component. Invalid return code after calling PutElement() to request a dynamic memory area for the SACF action COPY.

Code	Group	Reason
SACT27	M	XAP-TP component. Invalid return code after calling PutElement() to perform save in SACF action FLUSHALL or FLUSHPAR.
SACT28	M	XAP-TP component. Invalid return code after call of ReAllocElement() or PutElement() in SACF action QUEUE. Action: Increase the value of the KDCDEF parameter MAX OSI-SCRATCH-AREA.
SC0005	SU	The half session is generated as PLU. An RU request arrived but the request code is not SDT, STSN or CLEAR.
SC0006	SU	The half session is generated as PLU. An RU response arrived but the request code is not RQR.
SC0007	SU	#FSM.SESS is not in "ACTIVE" status.
SC0008	SU	The half session is generated as SLU. An RU request arrived but the request code is not RQR.
SC0009	SU	The half session is generated as SLU. An RU response but the request code is not SDT,STSN or CLEAR.
SC0010	SU	Protocol error for at least one of the FSMs #FSM_DT, #FSM_STSN or #FSM_RQR.
SC0011	SU	Sense code is set by at least one of the FSMs #FSM_DT or #FSM_CNTL_IMMED_EXP.
SC0014	SU	FSM_SESS_LU_LU returned an unallowed return code.
SC0019	ADU	Protocol error reported by FSM. Action: Check whether both partners are generated as DTP partners in the KDCDEF generation.
SC0020	ADU	The status of the PLU-FSM is not permitted. Action: Check whether both partners are generated as DTP partners in the KDCDEF generation.
SC0021	ADU	The status of the SLU-FSM is not permitted. Action: Check whether both partners are generated as DTP partners in the KDCDEF generation.
SC0023	DSU	When checking UNBIND-RSP for FSM_SESS_LU_LU an error has been found
SC0024	DSU	When switching UNBIND-RSP for FSM_SESS_LU_LU an error has occurred
SC0025	DSU	When checking UNBIND Request for FSM_SESS_LU_LU an error has been found
SC0026	DSU	When switching UNBIND Request for FSM_SESS_LU_LU an error has occurred (diagnostic dump)
SC0027	DSU	FSM_STSN_SEND returns S,RQ,STSN

Code	Group	Reason
SC0028	DSU	Unauthorized STSN-REQ for PLU_TO_SLU
SC0029	DSU	Unauthorized STSN-REQ for SLU_TO_PLU
SC0030	DSU	ACT_SC_RECEIVE could not set any return code
SC0032	A	The session cannot be activated since there are no connections available. Action: Change KDCDEF generation, include more CON statements.
SC0034	DSU	The partner has made a protocol error on setting STSN response
SDCS02	M	XAP-TP component. Module KCOXFSD, function CheckSaRetc(). The SetAttribute() return code was SA_NOMEM.
SDUI01	M	XAP-TP component. Module KCOXFSD, function UserDataIn(). The macro mGetBufferSpace() issued the return code LB_NOMEM.
SEND05	SU	Severe BCAM error when processing a BCAM SENDLET calls. Error when canceling a timer for output terminals.
SG-nnn	D	Diagnostic dump generated on the basis of a KCRSIGN1/2 return code after a SIGN ON call. The prefix SG- is followed by the return code (e.g. SG-U17). Activation and deactivation via the message dump function.
SHCxxx	SU	Error number xxx when calling shmctl() in KCSSHMF.
SHM002	A	An attempt was made to generate a shared memory area that already exists. Response: check UTM generation.
SIGND1	DU	The input message is not yet deleted at the sign-off time of a user signed-on via a socket connection.
SIGxxx	AU	Signal with signal number xxx has occurred. Note: If the application terminates with SIG010/SIG011 (SIGBUS/SIGSEGV), the reason may be incorrect or missing parameters with KDCS call. Whether this is case can be ascertained by way of the Stack Traces in the gcore dump or Automatic Storages in the UTM dump. In this situation the function KCSAVAL may be found on the stack.
SLOG05	SU	KCSLKLC returned a bad return code when attempting to lock the message buffers (possibly TIMEOUT).
SLOG06	SU	KCSLKLC returned a bad return code when attempting to unlock the message buffers.
SLOG07	SU	KCSLKLC returned a bad return code when attempting to lock the SYSLOG file (possibly TIMEOUT).
SLOG08	SU	KCSLKLC returned a bad return code when attempting to unlock the SYSLOG file.
SLOG09	SU	Attempt to write message buffer to current SYSLOG file failed (the DMS error code in the preceding K043 message may provide information about the cause of the error).

Code	Group	Reason
SLOG10	SU	The attempt by a follow-up task to switch to the current SYSLOG file generation failed (please refer to any preceding K043 message).
SLOG21	ASU	In the administration call KDCSLOG INFO or in the call to the program interface for administration with the opcode KC_SYSLOG and the sub-opcode1 KC_INFO, KCSSLOG called the component KCSFST in order to determine the attributes of the file generation group of the SYSLOG (only when the SYSLOG is created as a FGG). KCSFST gave a DMS error (also take note of preceding K043 message).
SLOT07	AM	The address of a slot must be calculated in KCSSLOT. The task does not yet know the shared memory that contains the slot. The attempt to link up with the shared memory failed due to insufficient address space.
SMSG00	AU	SMSG00 is a diagnostic dump and does not lead to the end of the application run; a SMSG00 dump is only taken when test mode is activated. An SMSG dump is created if KCSSMSG wants to send a message to a message destination of the type MSG-DEST and receives a bad return code from KCSFPUT. The return code can be taken from the NKB table of the UTM dump; the CC and DC codes of the DPUT call are at distance X'5C' in this table. One possible reason is that the queue/TAC is locked.
SMSG03	ASU	The component KCSSLOG supplied a bad return code on writing a message to the SYSLOG. Cause: See DMS return code in K043 message
SMSG09	D	SMSG09 is a diagnostic dump and does not cause the application run to terminate; an SMSG09 dump is only created if test mode is enabled.
SND005	M	XAP-TP component. Module KCOXFSD, function ap_snd(). The UserDataIn() function returned the UDIN_NOMEM return code.
SND007	MX	XAP-TP component. Module KCOXFSD, function ap_snd(). The return code of the function GetVarLthAttr() was not equal to GA_OK.
STnnnn	ADSU	Error when processing the start of a UTM task, where nnnn is the number indicating the error cause in the UTM message "K049 Error <nnnn> during application startup".
STPF10	SX	XAP-TP component. Invalid indicator for HeurRep parameter in TP-HEUR-REP-RI.
STPF11	SX	XAP-TP component. Invalid indicator for FuUnits parameter in TP-BEG-DIAL(dialogue)-RC.
STPF12	SX	XAP-TP component. Invalid indicator for Result parameter in TP-BEG-DIAL(dialogue)-RC.
STPF13	SX	XAP-TP component. Invalid indicator for Diagnostic parameter in TP-BEG-DIAL(dialogue)-RC.

Code	Group	Reason
STPF14	SX	XAP-TP component. Invalid indicator for user data parameter in TP-BEG-DIAL(dialogue)-RC.
STPF15	SX	XAP-TP component. Invalid indicator for Result parameter in TP-BEG-DIAL(channel)-RC.
STPF16	SX	XAP-TP component. Invalid indicator for Diagnostic parameter in TP-BEG-DIAL(channel)-RC.
STPF17	SX	XAP-TP component. Invalid indicator for HeurRep parameter in TP-(ABORT-AND-)HEUR-REP-RI.
STPT02	SX	XAP-TP component. Ungültiger Wert in p_udt.p_udtnxt.
STPT10	SX	XAP-TP component. Invalid indicator for InitTPSUT parameter in TP-BEG-DIAL(dialogue)-RI.
STPT11	SX	XAP-TP component. Invalid indicator for RecTPSUT parameter in TP-BEG-DIAL(dialogue)-RI.
STPT12	SX	XAP-TP component. Invalid indicator for FuUnits parameter in TP-BEG-DIAL(dialogue)-RI.
STPT13	SX	XAP-TP component. Invalid indicator for BegTrans parameter in TP-BEG-DIAL(dialogue)-RI.
STPT14	SX	XAP-TP component. Invalid indicator for Conf parameter in TP-BEG-DIAL(dialogue)-RI.
STPT15	SX	XAP-TP component. Invalid indicator for LastPartId parameter in TP-BEG-DIAL(dialogue)-RI.
STPT16	SX	XAP-TP component. Invalid indicator for user data parameter in TP-BEG-DIAL(dialogue)-RI.
STPT17	SX	XAP-TP component. Invalid indicator for FuUnits parameter in TP-BEG-DIAL(channel)-RI.
STPT18	SX	XAP-TP component. Ungültiger ChanUtil parameter in TP-BEG-DIAL(channel)-RI.
STPT19	SX	XAP-TP component. Invalid indicator for LastPartId parameter in TP-BEG-DIAL(channel)-RI.
STPT20	SX	XAP-TP component. Invalid indicator for CcrTokReq parameter in TP-BID-RI.
STPT21	SX	XAP-TP component. Invalid indicator for LastPartId parameter in TP-BID-RI.
STPT22	SX	XAP-TP component. Invalid indicator for Result parameter in TP-BID-RC.
STPT23	SX	XAP-TP component. Invalid indicator for Conf parameter in TP-END-DIAL-RI.

Code	Group	Reason
STPT24	SX	XAP-TP component. Invalid indicator for ConfUrg parameter in TP-HSK-RI.
STPT25	SX	XAP-TP component. Invalid indicator for ConfUrg parameter in TP-HSK-AND-GRT-CTRL-RI.
STPT26	SX	XAP-TP component. Invalid indicator for Type parameter in TP-DEFER-RI.
STPT27	SX	XAP-TP component. Invalid indicator for Reason parameter in TP-TOKEN-GIVE-RI.
STPT28	SX	XAP-TP component. Invalid indicator for Correlator parameter in TP-TOKEN-GIVE-RI.
STRT01	DU	When changing the application program the number of processes which still are to be changed is less than zero, i.e. the counter is inconsistent. The application is terminated abnomallly if debug mode is activated, otherwise a diagnostic dump is written.
SYPM01	AU	Task-specific buffer for restart information is too small. Response: increase MAX RECBUF=(...,length) in KDCDEF generation.
SYPM03	AS	Serious error in communication with DB system (FITA).
SYPM11	AU	Task-specific buffer for restart information is too small. Response: increase MAX RECBUF=(...,length) in KDCDEF generation.
TADR03	SU	The so-called Application Reference in the announcement is invalid.
TC0040	DSU	The input message length supplied by the transport system does not match the length contained in the message.
TC0041	DSU	The input message length supplied by the transport system is shorter than the value in the Data Count Field (DCF) in the Transmission Header (TH).
TCPI13	SU	Invalid length of the IP address (TCP_GET_HOST_BY_ADDR).
TIMR02	SU	Error on locking IUTMIPC in KCSTIMR.
TIMR03	SU	Error on releasing IUTMIPC in KCSTIMR.
TIMR05	AU	utmtimer process not present when sending to IUTMIPC in KCSTIMR
UATC02	SU	(See UKAA02) Unlock failed.
UATC03	SU	(See UKAA03) Maximum number of locks per task exceeded.
UCAC02	SU	(See UKAA02) Unlock failed.
UCAC03	SU	(See UKAA03) Maximum number of locks per task exceeded.
UKAA02	SU	Unlock failed.
UKAA03	SU	Maximum number of locks per task exceeded.

Code	Group	Reason
UMDT12	SU	KCCUMDT opcode UMDT_LOCK_ULS Bad return code from KCSBFMM when reading a ULS administration file.
UMDT13	SU	KCCUMDT opcode UMDT_LOCK_ULS Bad return code from KCCLKMT.
UMDT15	ASU	UTM cluster application KCCUMDT opcode UMDT_UNLOCK_ULS: Error on locking the ULS file. Action see Action see section „Actions when locking UTM cluster files“ below this table.
UMDT32	SU	UTM cluster application KCCUMDT function readHdrPage Bad return code from KCSBFMM when requesting a buffer.
UMDT33	ASU	UTM cluster application KCCUMDT releaseFileLock function: Timeout while unlocking the cluster ULS file. Action see UMDT15.
UMDT34	SU	UTM cluster application KCCUMDT releaseFileLock function: Incorrect return code from KCCGFLK when unlocking the cluster ULS file. Action see UMDT15.
UMDT42	SU	UTM cluster application KCCUMDT function addrUlsEntry Bad return code from KCSBFMM when requesting a buffer.
UMDT44	SU	UTM cluster application KCCUMDT setFileLock function: Incorrect return code from KCCGFLK when locking the cluster ULS file. Action see UMDT15.
UMDT52	ASU	UTM cluster application KCCUMDT allocateUlsEntry function: Incorrect return code from KCCUSF when locking the cluster user file. Action see UMDT15.
UMDT63	ASU	UTM cluster application CUMDT entry KCCUGLI: The cluster ULS file cannot be locked. Action see UMDT15.
UMDT73	SU	UTM cluster application KCCUMDT function checkFile Bad return code from KCSBFMM when requesting a buffer.
UMDT74	SU	UTM cluster application KCCUMDT function checkFile Bad return code from KCSBFMM when requesting a buffer.

Code	Group	Reason
UMDT77	SU	UTM cluster application KCCUMDT function checkFile Bad return code from KCSBFMM when requesting a buffer.
UPCM02	SU	(See UKAA02) Unlock failed.
UPCM03	SU	(See UKAA03) Maximum number of locks per task exceeded.
USF002	ASU	UTM cluster application. Module KCCUSF, opcode OPEN_FILE. Timeout when initializing global file lock. Action: Increase the value of the FILE-LOCK-TIMER-SEC or FILE-LOCK-RETRY parameter in the CLUSTER statement of the KDCDEF generation.
USF003	SU	UTM cluster application Module KCCUSF, opcode OPEN_FILE. Invaild return code when initializing the global file lock.
USF009	ASU	UTM cluster application. Module KCCUSF, opcode CUSF_RESET_PTC_FLAG. Five timeouts on locking the cluster user file. Action see USF002.
USF013	ASU	UTM cluster application. Module KCCUSF, opcode CUSF_SIGNOFF_USER_WITH_CONTEXT. Five timeouts on locking the cluster user file. Action see USF002.
USF018	ASU	UTM cluster application. Module KCCUSF, opcode CLOSE_FILE. Timeout when destroying file lock. Action see USF002.
USF019	SU	UTM cluster application Module KCCUSF, opcode CLOSE_FILE. Invaild return code from KCCGFLK when destoying the file lock.
USF021	SU	UTM cluster application Module KCCUSF, function setFileLock Invaild return code from KCCGFLK when requesting the exclusive lock.
USF023	ASU	UTM cluster application. Module KCCUSF, function releaseFileLock. Timeout when releasing file lock. Action see USF002.
USF024	SU	UTM cluster application Module KCCUSF, function releaseFileLock Invaild return code from KCCGFLK when releasing the file lock.

Code	Group	Reason
USF025	ASU	UTM cluster application. Module KCCUSF, function readPage. Timeout when requesting shareable file lock. Action see USF002.
USF026	SU	UTM cluster application Module KCCUSF, function readPage Invaild return code from KCCGFLK when requesting the shareable file lock.
USF027	ASU	UTM cluster application. Module KCCUSF, function readPage. Timeout when releasing file lock. Action see USF002.
USF028	SU	UTM cluster application Module KCCUSF, function readPage. Invaild return code from KCCGFLK when releasing the file lock.
USF051	SU	UTM cluster application Module KCCUSF, function writePage Error when writing the cluster user file.
USF052	SU	UTM cluster application Module KCCUSF, function readPage. Error when reading the cluster user file.
WAI102	DU	Diagnostic dump.
WAI112	DU	When switching the system protocol file the number of processes which still are to be switched is less than zero, i.e. the counter is inconsistent. The application is terminated abnomallly if debug mode is activated, otherwise a diagnostic dump is written.
WAI113	DU	When changing the application program the number of processes which still are to be changed is less than zero, i.e. the counter is inconsistent. The application is terminated abnomallly if debug mode is activated, otherwise a diagnostic dump is written.
WAI114	DU	See WAI113
WAI118	DU	Aktion PROC_ACCESS_FROM_USER_INDX: UTM_INTERNAL_MSG_ANNO received but no UTM_INTERNAL_MSG_ANNO, UTM_START_PROG_ANNO, UTM_ROLLBACK_PET_ANNO and no UTM_COMNMIT_PET_ANNO. The application is terminated abnomallly if debug mode is activated, otherwise a diagnostic dump is written.

Code	Group	Reason
WAI119	DU	Aktion PROC_ACCESS_FROM_USER_INDX: UTM_INTERNAL_MSG_ANNO received but no UTM_START_PROG_ANNO, UTM_ROLLBACK_PET_ANNO and no UTM_COMNMIT_PET_ANNO. The application is terminated abnomallly if debug mode is activated, otherwise a diagnostic dump is written.
WAI120	DU	When administering the diagnostic trace settings the number of processes which still have to update the settings is less than zero, i.e. the counter is inconsistent. The application is terminated abnomallly if debug mode is activated, otherwise a diagnostic dump is written.
WAIT01	SU	Bad return code when calling KCSBRSE for the work bourse in action block ENQUEUE_WORKBOURSE von KCSWAIT.
WAIT02	DS	Invalid timer announcement type (action block ENQUEUE_WORKBOURSE).
WAIT03	S	Invalid type in announcement and no timer announcement (action block ENQUEUE_WORKBOURSE).
WAIT28	SU	Bad return code from Module KCSSLOG (before ENQUEUE_WORKBOURSE).
WAIT38	S	KCSWAIT starts a TU action for a service which is in PGWT status.
WAIT55	SU	Invalid code for a internal timer announcement (action block ANALYSE_INTERNAL_TIMER).
WAIT78	ASU	User error or fatal error when calling the internal BCAM interface for socket. Possible causes: 1. An attempt was made to actively establish a connection for a BCAMAPPL that could not be signed on by the network process because, for example, the port number for this local end point is already used or it is a reserved port number. Response: ensure that the port number is not being used for something else and that no reserved port number is specified. 2. The network process responsible for the connection has terminated abnormally. 3. One of the network processes of the application has terminated abnormally.
WAITT1	F	Another task has terminated the application abnormally (=continuation dump before calling KCSBRSE).
WAITT2	F	Another task has terminated the application abnormally (=continuation dump after calling KCSBRSE).
XATT02	F	XAP-TP component. Module KCOXFEX, function apext_att(). The function bCheckAndSetState() issued a bad return code and the application status was WAITING_DUMP_APPL. A different task had already caused the application to abort.

Code	Group	Reason
XATT04	MX	XAP-TP component. Module KCOXFEX, function apext_att(). The return code of the function EstablishBuffer() was not equal to LB_OK.
XATT12	MX	XAP-TP component. Module KCOXFEX, function apext_att(). The return code of the function EstablishBuffer() was not equal to LB_OK.
XATT13	MX	XAP-TP component. Module KCOXFEX, function apext_att(). The return code of the function RequestBuffer() was not equal to LB_OK.
XFAG09	M	XAP-TP component Module KCOXFHF, function GetAttribute(). The function RequestBuffer() issued the return code LB_NOMEM.
XFGA07	SUX	XAP-TP component. Module KCOXFHF, function GetAttribute(). The function AllocUserMem() returned an unexpected return code when reading the attribute AP_DTNID in the single task mode.
XFGA11	M	XAP-TP component. Module KCOXFHF, function GetAttribute(). The macro mGetBufferSpace() issued the return code LB_NOMEM.
XFGE01	F	XAP-TP component. Module KCOXFHF, function bCheckAndGetCallEnv(). The function bCheckAndSetState() issued a bad return code and the application status was WAITING_DUMP_APPL. A different task had already caused the application to abort.
XFSA07	MX	XAP-TP component. Module KCOXFHF, function SetAttribute(). The function PutElement() issued a return code other than DM_OK when setting the AP_DTNID attribute.
XFTM01	SX	XAP-TP component. Module KCOXFHF, function TraceMgmt(). The OSS function o_tron() returned the return code O_ERROR.
XFTM02	SX	XAP-TP component. Module KCOXFHF, function TraceMgmt(). The OSS function o_tron() returned the return code O_INVEREF.
XFTM03	SX	XAP-TP component. Module KCOXFHF, function TraceMgmt(). The OSS function o_tron() returned an unknown return code.
XFTM04	SX	XAP-TP component. Module KCOXFHF, function TraceMgmt(). The OSS function o_troff() returned the return code O_ERROR.

Code	Group	Reason
XFTM05	SX	XAP-TP component. Module KCOXFHF, function TraceMgmt(). The OSS function o_troff() returned an unknown return code.
XINI06	ASX	XAP-TP component. Module KCOXFEX, function apext_init(). The OSS function o_create() returned the return code O_ERROR. Possible cause: If the insert XP1INFO in the corresponding UTM message P001 has the value 1, then the error may be that the OSS shared memory has not been allocated.
XINI07	SX	XAP-TP component. Module KCOXFEX, function apext_init(). The OSS function o_create() returned an unknown return code.

Actions when locking UTM cluster files

The following workaround is recommended for all errors that occur when requesting or releasing locks for UTM cluster files.

- In the CLUSTER statement of the KDCDEF generation, increase the value of the FILE-LOCK-TIMER-SEC parameter or FILE-LOCK-RETRY parameter.

For a more detailed diagnosis, please also see the most recent K190 messages.

K061 Dump file &FNAM created

K062 Dump file could not be created.

K064 Message discarded : &PTRM/&PRNM/&BCAP/&LTRM &DEVC &FIL1A &FIL2A &FIL3 &VTRC &CBRC &IMSG2 &REA1

Meaning:

1. An invalid or unexpected announcement was received at the work bourse. In this event, insert &REA1 has a value of X‘05‘ and insert &IMSG2 contains the announcement received.
2. A message from an LTERM partner has been received which is logically inconsistent for UTM, e.g. because it does not comply with the strict dialog.
3. A message which does not contain a valid transmission header was received from an LU6.1 partner.

For diagnostic purposes, the first 32 characters of the message are output. In response to this message, the connection is cleared down or an automatic KDCDISP performed.

The standard record output to SYSLOG has the following structure:

Inserts	Meaning
&DEVC	PTYPE, according to KDCDEF generation
&FIL1A	Status of the application; possible values X'02' = start phase X'03' = normal run X'04' = shut warn X‘05‘ = shut grace X'07' = fast shut X‘08‘ = term application
&FIL2A	Status of the client or printer: X'00' = Connection clearing down X'01' = Connection not established X'02' = UTM waiting for a connection request to be confirmed X'03' = Connection established, user not yet signed on X'04' = UTM waiting for ID card to be inserted X'05' = UTM waiting for a password to be entered X'07' = 2nd part of the sign-on service X'08' = LTERM with USAGE=O: Printer connected, otherwise user logged on
&FIL3	Status of the physical terminal or printer, serves as a diagnostic aid in the event of errors
&VTRC	Internal UTM return code used for diagnostics in the event of errors
&CBRC	Internal UTM return code used for diagnostics in the event of errors
&IMSG2	The first 32 characters of the message
&REA1	Reason for K064 message: X'02' = invalid function key X'03' = no positive print acknowledgment X'04' = bad return code from ISLP X'05' = inconsistent input message X'06' = LU6.1: The announcement length is shorter than the DCF in the transmission header X'08' = inconsistent message from an LU6.1 partner X'09' = UPIC input message with invalid protocol X'0A' = inconsistent physical or logical acknowledgment X'0B' = input message from a socket client generated as an output partner X'0C' = unexpected GO signal received for a UPIC or socket connection

K065 Net message: &PTRM/&PRNM/&BCAP/&LTRM &FIL1B &FIL2B

The inserts &FIL1B and &FIL2B have the following meaning:

Inserts	Meaning
&FIL1B	BCAM call or (UTM) announcement
&FIL2B	Diagnostic word or UTM announcement code

&FIL1B is output in printable form. A value between X'FO' and X'FF' identifies a UTM announcement. The values X'E0', X'E2'and X'F0' to X'FF' indicate a UTM (timer) announcement. The values are explained in the following table.

&FIL1B	Meaning	Meaning of &FIL2B
X'01' - X'4C'	BCAM call or BCAM announcement [1]	BCAM, infoword (printable)
X'E0'	Idle timeout for an LU6.1 session (see KDCDEF statement SESCHA, parameter IDLETIME)	no meaning
X'E2'	Timeout on reception of message fragments at a socket connection since the message was not fully received within the permitted period of 10 minutes (internal value)	no meaning
X'F0'	UTM anno	no meaning
X'F1' - X'FA'	UTM anno	no meaning
X'FB'	UTM anno Timeout waiting for an acknowl-edgment from a printer or TS application after sending a queued message (see KDCDEF statement MAX, parameter LOGACKWAIT)	no meaning
X'FC' - X'FE'	UTM-Anno	no meaning

&FIL1B	Meaning	Meaning of &FIL2B	
X'FF'	UTM anno: timeout	The first byte in &FIL2B specifies the cause of the timeout:	
		X'21'	Expiry of the timer which monitors the establishment of an LU6.1 session. See KDCDEF statement UTMD, parameter CONCTIME=(**time1**,....).
		X'22'	Expiry of the timer which monitors the reception of the acknowl-edgment for a queued message sent via an LU6.1 session. See KDCDEF statement UTMD, parameter CONCTIME=(....,**time2**).
		X'30'	Expiry of the PEND-KP timer. See KDCDEF statement MAX, parameter TERMWAIT or PGWTTIME.
		X'40' to X'43'	Expiry of the PTERM idle timer. See KDCDEF statement PTERM or TPOOL, parameter IDLETIME or KDCDEF statement MAX, parameter PGWTTIME.

[1] The value and meaning of the BCAM call and announcement can be obtained from the BCAM diagnostic documentation or System Support

The values of the diagnostic word are specified in hexadecimal and are explained in the following table:

Bytes 1-4	Meaning	Response
00000008	Invalid parameter	PR
00000014	Connection letter too long	PR
00000028	Preceding send action not yet terminated (sending of data)	Normal behavior
0000002C	Message could not be received completely	Check UTM generation
00000030	Internal error	See dump error code WAIT 41
00000038	Preceding send action not yet terminated (conn. setup/cleardown)	Normal behavior
0400001C	Resource bottleneck	Check UTM generation
04000020	Application not signed on	Check UTM generation
04000024	Connection already cleared down	Normal behavior
08000024	Connection already set up	Normal behavior
10000024	UTM network conn. not yet active	Normal behavior
1C080000	End of application by kdcshut utility with time entry	Normal behavior
1C0C0000	End of application by kdcshut utility without time entry	Normal behavior
24000004	Connection already cleared down	Normal behavior
24000018	Negative response to connection setup request	Normal behavior
24000038	Connection cleardown request	Normal behavior
30000020	Error signing on	PR

K066 Mandatory parameter FILEBASE not specified

K067 Error in interoperation with language connection module &MOD: error code = &ERCD1 &ERCD2; opcode = &OPCD2

The inserts of the message have the following meaning:

&MOD	**Meaning**
KDCCC	Connection module for C
KDCCCOB2	Connection module for Micro Focus COBOL
KDCCCOBN	Connection module for NetCOBOL

&OPCD2	**Meaning**
INITIALISE	Initialize the language environment
DEACTIVATE	Release the language environment
S; XXXXXXXX	Start the program unit ' XXXXXXXX'
E; XXXXXXXX	Execute the end handling routine fortz program unit ' XXXXXXXX'

The ERRORCODE is made up of 2 x 4 characters.

&ERCD1	**Meaning and possible response**
0000	Job has already been executed
0004	Job was not executed correctly Compile error documentation and send to Siemens Service
0008	Error in application program
00FF	The language connection module for a generated language is not linked when the application is started

The inserts in &ERCD2 supply additional information on errors that have occurred and are used for diagnostic purposes.

K068 Database connection module &DBCON version &DBV1 cannot interoperate with KDCDB macro version &DBV2

K069 Disconnection : &PTRM/&PRNM/&BCAP/&LTRM/&REA4/&REA6/&COTM

The insert &REA4 indicates who initiated the connection cleardown. The insert &REA6 contains the cause of the connection cleardown.

&REA4	**Meaning**
B	Transport system reports connection cleardown
L	Connection lost for socket connection
other	Connection was cleared down by UTM, for reason see &REA6

&REA6	**Meaning**
X'00'	Reason not specified
X'08'	Resource bottleneck
X'09'	Connection cleared down
X'0A'	Application SHUTDOWN
X'0D'	Negative return code from transport system, see associated K065 message.
X'31'	Connection cleardown after KDCOFF
X'32'	Connection cleardown after SIGN OF
X'33'	Connection cleardown by the print administration
X'34'	Connection cleardown by the administration
X'35'	No messages for printer with PLEV > 0
X'36'	Page pool warning level 2 exceeded on receipt of an asynchronous DTP message.
X'39'	Invalid function key
X'3A'	Connection cleardown by DTP with session not yet established
X'3B'	Connection cleardown by UPIC
X'3C'	Connection cleardown after unsuccessful sign-on attempt
X'3D'	Connection cleardown after unsuccessful sign-on service
X'3E'	Connection cleardown after timeout
X'3F'	Message fragment received
X'40'	Negative transport or print acknowledgment
X'41'	Unexpected protocol element
X'42'	Inconsistent input message
X'43'	Connection cleardown through new connection setup request
X'47'	Connection cleared down because IDLE timer timed out.
X'49'	Socket - Connection cleared down because of invalid length of incoming message (negative or >32000)

&REA6	Meaning
X'4A'	Socket - Connection cleared down because of invalid version in the protocol header
X'4B'	Socket - Connection cleared down because of invalid type in the protocol header
X’4C'	Socket - Connection cleared down after an error occurred while writing the message in the message queue
X'4D'	Socket - Connection cleared down after timeout
X'4E'	Socket - Connection cleared down because of a length problem: The message generated is larger than the generated maximum length (see the MAX TRMSGLTH operand in the KDCDEF statement).
X’4F'	Socket - Connection cleared down because of unknown identifier in the protocol header
X'50'	Socket - Connection cleared down because of invalid version minor in the protocol header
X'51'	Socket - Connection cleared down because of invalid flag in the protocol header
X'52'	Socket - Connection cleared down because the maximum possible number of fragments was exceeded
X'53'	Socket - Connection cleared down because of an invalid message type FRAGMENT at the beginning of a message
X'54'	Socket - Connection cleared down because the last part of a fragment was missing.
X'57'	Cleardown of the connection to the client because insufficient processes are active to continue the service normally after timeout at the PGWT bourse.
X'59'	Connection cleared down by DTP while session established.
X'5A'	Internal cluster communication: Connection to client terminated, the signal in the received message could not be validated.
X'5B'	Internal cluster communication: Connection to client terminated, acknowledgment of the received message could not be sent.
X'5C'	Internal cluster communication: Connection to client terminated, response to "check alive" (ping signal) was sent successfully.
X'5D'	Internal cluster communication: Connection to client terminated, the signal to read the configuration file was recognized.
X'5E'	Internal cluster communication: Connection to client terminated, the signal to read the administration journal was recognized.

&REA6	Meaning
X’5F’	Internal cluster communication: Connection to client terminated, the received message contained a message for Administration.
X’60’	Cleardown of a socket connection due to no longer supported protocol version 1.0 (minor version X'00') in the protocol header.

K070 USER / LSES inactive : &USER, &GLOBALSG

A value is only entered for the insert &GLOBALSG 'Cluster Global Sign' in UTM cluster applications. The insert can have the following values:

&GLOBALSG	Meaning
Y	Global sign-off at the cluster
N	Local sign-off at the node
T	The global sign-off at the cluster was not successful because the cluster user file could not be locked in the generated time.
F	The global sign-off at the cluster was not successful because the user was not found in the cluster user file.
S	The global sign-off at the cluster was not successful because the user was not signed on globally at the cluster.
O	The global sign-off at the cluster was not successful because the user was not signed on globally at the cluster at this node.
' '	Sign-off for an LU6.1 session user or a connection user.

The message also possesses the following inserts:

Insert	Meaning
&COTM	Number of seconds since &USER signed on at this connection
&CPTM	Elapsed CPU time in msec in the user ID &USER if only one user was currently signed on under the user ID &USER. If multiple users have previously signed on under &USER at different connections then the value also contains their elapsed CPU time. The value is then reset to 0. If further users are still signed on under the user ID &USER then the value is 0.

K071 Internal Error in database connection: operation=&OPCD1, error=&ERCD3

The inserts have the following meanings:

DBCON	= XA (database connection via XA)
OPCD1	DB operation code, see section “DB-DIAGAREA” on page 109
ERCD	KCRCDC error code, see section “DB-DIAGAREA” on page 109
DBTRAC	DB trace information. The possible values and their meanings are described in the discussion of the relevant database system. If the DB system is connected via the XA interface then the 4 DB trace information bytes have the following meanings:

Byte 1	Resource Manager index (cf sequence in the start parameters)
Bytes 2 and 3	Operation code of XA call, output hexadecimal and platform-dependent (EBCDIC or ASCII),
Byte 4	Return code of XA call, output hexadecimal and platform-dependent

K072 There is a mismatch in the number of &STMT entries in ROOT and KDCFILE.

This message is output if when the application starts it is found that the generations of ROOT and KDCFILE do not match with regard to the number of programs. The start of the application or process is aborted with message K049, code 40.

Response: regenerate ROOT and/or KDCFILE

K073 The attribute &ATTR of &STMT &PROG in ROOT and KDCFILE does not match.

This message is output if when a task is started it is found that the entries of the PROGRAM table in ROOT and in the KDCFILE do not match. The first insert indicates the attribute in which the entries differ.

The start of the application or the process is aborted with message K049, code 41.

Response: regenerate ROOT and/or KDCFILE

K074 Program exchange completed; &CTYP &PROG &PVER

(see K075)

K075 Program exchange aborted by process &PID; &CTYP &PROG &PVER

K074 or K075 are issued after a positive or negative termination of a program exchange. The inserts give information as to which part of the application program were exchanged or which part could not be exchanged:

&PID:	Process ID of the process for which program exchange was aborted.
&CTYP:	APPL, Exchange of the entire application program LMOD, Exchange of a shared object
&PROG:	Name of the shared object (only when CTYP=LMOD)
&PVER:	Version number of the shared object (only when CTYP=LMOD)

K076 Error during asynchronous administration with transaction code &ADTC; KCRCCC=&RCCC, KCRCDC=&RCDC

K077 Internal cluster communication with &PTRM/&PRNM/&BCAP/&LTRM successful: &CLSIGT

The message K077 is only output if test mode is active (see page 39).

During internal cluster communication, a &CLSIGT message was sent to another node application. &CLSIGT can have the following values:

&CLSIGT	**Meaning**
1	Check the availability of the monitored node application (check alive, ping).
2	Check for changes in the cluster configuration
4	Administration job
8	Job to wake up one of the processes waiting at a GSSB or ULS
16	Response to an availability query from the monitoring node application (check alive response, pong)

No action is required.

K078 &ERRNAME &ERRCODE: in &REA3

The message can be output by openUTM when the application is started, when handling signals, or when a work process is terminated.

Message K078 is output by UTM in several variants that depend on the event that has occurred.

K078 yyyyyyyyy: IN module-name zzzzzzzz

K078 yyyyyyyyy: zzzzzzzz

K078 xxxxxxxx yyyyyyyyy: IN module-name text

xxxxxxxx
: contains a short code for the error that has occurred (see table).

yyyyyyyyy
: error code of the function called (for more information see preceding K078).

zzzzzzzz
: specific, context-related error message.

module-name
: contains the name of the module in which this error has occurred (for internal diagnostics)

text : contains additional information on specific errors

xxxxxxxx	**Error cause**	**Response**
ALME	Not possible to provide sufficient memory on memory request	Insufficient memory, tune system if necessary
APPLI 01	DMS error on accessing the applifile The applifile is located under utm-directory/applifile and contains the name and status of the UTM application	User error
APPLI 02	Application in applifile not known or utmwork was started for testing before utmmain prepared start	User error
APPLI 03	Application in applifile not started	User error
ARG 01	Invalid number of arguments at start of utmwork	User error: Start utmwork with correct arguments or utmwork has the wrong version
ATEXIT 00	The atexit() function was called on process termination.	Information message
ATEXIT 01	System program unit requests process termination with exit().	Information message
ATEXIT 02	Application program unit requests process termination with exit().	Information message, followed by PEND ERROR.
ATEXIT 03	The atexit() function was called recursively on process termination.	Information message

xxxxxxxx	Error cause	Response
ATEXIT 04	System program unit illegally requests process termination with exit().	Information message, followed by TERM APPLICATION.
BIND	Error loading a load module: The insert 'LMOD: name / version ': provides further information	See preceding K078
COB 01	cobtidy() suppressed after exit() in Micro Focus COBOL application program	Information message
COB 02	Start of COBOL application program suppressed after exit() in the Micro Focus COBOL application program.	Information message
COB 03	JMPCINT3() und JMPCINT4() werden unterdrückt nach exit() im NetCOBOL-Anwendungsprogramm.	Informationsmeldung
COB 04	Start des COBOL-Anwendungsprogramms wird unterdrückt nach exit() im NetCOBOL-Anwendungsprogramm.	Informationsmeldung
CPUT 00	The system was not able to supply the CPU time for the process.	Information message
CPUT 01	The system was not able to supply the CPU time for the child processes.	Information message
CPUT 02	The CPU time for the process was set to the maximum value LONG_MAX.	Information message
CRE 01	The environment variable UTM_MAIN_KILL_TIME is set	Information message
DIAG 01	The environment variable UTM_BREAK_BEFORE_KCSTRMA is set	Information message (only under Windows systems)
DIAG 02	The environment variable UTM_ABORT_WITH_EXCEPTION is set	Information message
DIAG 03	The environment variable UTM_CORE_DUMP is set to NO	Information message
DIAG 04	The environment variable UTM_EXIT_CORE is set to YES	Information message
DIAG 05	The environment variable UTM_ABORT_WITH_EXCEPTION is set	Information message
DIAG 06	exit() call in application program results in termination of the utmwork process	Information message
DIAG 07	exit() call in system section results in termination of the utmwork process	Information message

xxxxxxxx	Error cause	Response
DIAG 08	The environment variable UTM_MAIN_KILL_TIME is set	Information message
DLCLOSE 01 SHL_UNLOAD 01 FreeLibrary 01	Error closing a shared object.	Check UTM generation and return code.
DLOPEN 01 SHL_LOAD 01 LoadLibrary 01	Error loading a shared object.	Check UTM generation and return code. Check LD_LIBRARY_PATH environment variable or SHLIB_PATH environment variable on HP-UX
DLSYM 01 SHL_FINDSYM 01 GetProc-Adress 01	Error fetching the address for a symbol in a shared object.	
dlclose shl_unload FreeLibrary	Diagnostic message on closing a shared object.	Information message
dlopen shl_load LoadLibrary	Diagnostic message on dynamically loading programs. These messages only occur when test mode is on.	Information message
dlsym shl_findsym GetProcAddress	Diagnostic messages on fetching the address for a symbol in a shared object.	Information message
ENV 00	Specification of the type of UTM application: standalone / cluster / node_recovery / beanconnect	Information message
ENV 01	Specifications on the operating system	Information message
ENV 02	Specification of the bit mode of the UTM application: standalone/cluster/beanconnect	Information message
ENV 03	The compare and swap algorithm was switched off.	Information message
ENV 04	The environment variable UTM_NO_GCORE_DUMP is set to YES. I.e. no gcore is also generated when a UTM-DUMP is generated. By default, a gcore is also generated when a UTM dump is generated.	Information message (only on Unix systems)
EXIT xx	Signal number xx at start or shut exit. "STARTEXIT" or "SHUTEXIT" is entered in the additional field.	User error

xxxxxxxx	Error cause	Response
HOST 01	The environment variable UTM_NET_HOSTNAME is set. The host name file <file> is being used.	Information message
HOST 02	The environment variable UTM_NET_HOSTNAME is set. However, the host name file <file> could not be used because of DMS error <error>.	User error
IPC 01	Invalid return code when initializing the IPC.	Check UTM generation (MAX ICPTRACE) or call KDCREM
IPC 02	Memory bottleneck when creating the IPC shared memory	Check UTM generation Unix systems: tune SHMMAX kernel parameter
IPC 03	The environment variable UTM_IPC_LETTER is set	Information message
IPC 04	The environment variable UTM_IPC_EXTP_LETTER is set	Information message
IPC 05	Number of IPC letters changed	Information message
IPC 06	Number of IPC elements changed	Information message
IPC 07	The environment variable UTM_IPC_EXTP_LETTER is set	Information message
KCSASIO	Complete text: K078 KCSASIO: error KCSTRMA ("&REAS>") file: "<&FNAME>" position:<&POS>" Cause: Processing aborted with reason &REAS due to incorrect (= too large) file position &POS for file &FNAME.	On KDCDEF run: Reduce size of files, e.g. generate more page pool files (MAX PGPOOLFS=)
	Complete text: K078 "KCSASIO notice: write to "&FNAME" took "&SEC" secs" Cause: Writing to file &FNAME took &SEC seconds.	Information message

xxxxxxxx	**Error cause**	**Response**
	Complete text: K078 "KCSASIO notice: read from "&FNAME" took "&SEC" secs" Cause: Reading from file &FNAME took &SEC seconds.	Information message
KCSCHKD	Complete text: K078 "KCSCHKD notice: back to "lost" directory "&DIR" ok" Cause: A process has set its directory &DIR due to an I/O error	Information message
KCSROPN	Complete text: K078 "KCSROPN notice: reopening for file "&FNAME"" Cause: The file &FNAME has been opened again due to an I/O error.	Information message
	Complete text: K078 "KCSROPN notice: reopening return_code for file "&FNAME" is "&DMSERR"" Cause: A return code was returned when the file was reopened. &DMSERR = " " (i.e. blanks) means "OK"	Information message
KCXOFLK	A lock call was rejected with errno. Complete text K078 KCXOFLK LOCK-WAIT/ LOCK-IMM error "&ERRNO" for file " &FILENAME"	Diagnostic message
KDCS 00	"KDCS" call in START exit, SHUT exit or INPUT exit	Correct program unit
MEM 01	Memory bottleneck when starting the application. Full text: K078 MEM 01 in utmwork nn Bytes not available.	Check storage requirements; tune operating system
MSG 01	The environment variable UTM_MSG_DATE is set to NO	Information message
MSG 02	The environment variable UTM_MSG_PID is set to NO	Information message

xxxxxxxx	Error cause	Response
NET 01	The environment variable UTM_APPLI_CONLET is set	Information message
OSS 02	Error while allocating/signing on to the OSS shared memories	Check UTM generation or call kdcrem
OSS 03	Error while loading the OSS shared memories into the address space.	Check UTM generation
PIPE 01	The environment variable UTM_PIPE_TIME is set	Information message
READ 01	Inconsistent KDCFILE when starting the application.	User error.
SEM 01	Error when creating the semaphore. This can occur in openUTM under Windows systems if, for example, a dtp is still running when the application is started	In UTM generation reduce the value of *number1* in MAX SEMARRAY.
SEM 02	Error signing on to a semaphore	System error
SIGNAL 01	Signal received in utmwork process. Normal signal handling	Information message followed by either PEND ERROR or TERM APPLICATION
SIGNAL 02	SIGTRAP signal received in utmwork process.	Information message, signal is either ignored or application is terminated with TRMA IPCREM.
SIGNAL 03	Unexpected signal SIGALRM received in utmwork process	Information message, signal is ignored.
SIGNAL 04	Invalid signal received in utmwork process	Information message, signal is ignored.
SIGNAL 05	Application-specific signal routine for SIGALRM is not permitted.	Information message
SIGNAL 06	It is not permitted to set an application-specific signal routine for SIGALRM with sigaction().	Information message
STARTEXIT	Startexit terminated the work process with exit().	Information message
STXIT OFF	Message: "STXIT OFF in utmwork: termination of utmwork process creates gcore dump"	Information message
STXIT OFF	Message: "STXIT OFF in utmwork: exit() call produces termination of utm- work process with abort().	Information message

xxxxxxxx	Error cause	Response
SYSPROT 01	The environment variable UTM_REDIRECT_FILES is set	Information message
SYSPROT 02	The environment variable UTM_MAIN_NODIRECT is set	Information message
TIMR 01	The environment variable UTM_TIMER_RETRY is set	Information message
UNBIND	Error unloading a load module. The insert ' LMOD: name / version ' provides further information.	See preceding K078
VERS 01 VERS 02	Version or bit mode of the program and the KDCA file incompatible.	User error Response: set UTMPATH correctly
VSVI1	Error finding program addresses. The insert ' for program: name' provides further information.	See preceding K078

K078 WARNING : in KDCRTBF! No XA-Connection generated, but startparameters given!

A start parameter was specified for the Resource Manager but none was generated. Modify generation or start parameter!

K079 Accounting problem - reason: &REA2

The error cause is given in the insert &REA2:

&REA2	Meaning
28	Error in BS2000 accounting write task. UTM deactivates accounting. The application continues to run

Response: check to see if the utmlog process still exists.
Once the error has been rectified, the UTM administrator can reactivate the costing and/or accounting phase with the KDCAPPL command.

K080 KDCMON is not active.

K081 Statistics: &IMSG1/&OMSG1/&CONU/&ATAC2/&LWRT/&HITR/&WTBF

Inserts	Meaning	
&IMSG1	Number of terminal input messages	1)
&OMSG1	Number of terminal output messages	1)
&CONU	Number of connected users	
&ATAC2	Number of unprocessed ATACs	
&LWRT	Number of log writes	1)
&HITR	Cache hit rate	
&WTBF	Cache waits for buffer	1)

This message is written every hour on the hour and in the case of normal termination of the application. The values marked with 1) are subsequently reset to 0.

K082 Wrong file &FNAM

K083 File &FNAM has been destroyed.

K084 &OBJ1 &VER1 &OST1 &BMD1 and &OBJ2 &VER2 &OST2 &BMD2 are not compatible.

K085 &FNKT functions are not available.

K086 UTM-D error information: &PTRM, &PRNM, &BCAP, &LTRM, &USER, &SYSD, &USSD, &FMH7, &AGUS

Inserts	Meaning
&PTRM	Name of physical terminal or printer
&PRNM	Processor name
&BCAP	Application name
&LTRM	Name of LTERM partner
&USER	Login name
&SYSD	SNA sense code (system)
&USSD	SNA sense code (user)
&FMH7	Error message from the remote application
&AGUS	ID of job-submitting user

The following values are possible for the inserts &SYSD and &USSD with homogeneous linkage:

&SYSD&USSD	Meaning
08120000	Resource bottleneck (e.g. page pool full.
08130000 081B0000	Contention: A request to occupy a session was rejected by the contention loser application because the contention winner application had already occupied the session for a job.
08641003	Invalid or locked TAC
0864C5E2	PEND ER from UTM
0864C5D9	PEND ER from the application program unit
0864D9E2	PEND RS from UTM
0864D9E4	PEND RS from the application program unit
0866D9E2	PEND RS from UTM
0866D9E4	PEND RS from the application program unit
10030000	Invalid or locked TAC

K088 UTM-D session start : &LSES/&RSES/&LPAP
SR-STATE: &SRFG PET &PSQN
SAVED : &ESQS &EBSS
ACT. : &ESQR &ESRR &EBSR

UTM-D session start. The message contains diagnostic information

Inserts	Meaning
&LSES	LSES name
&RSES	RSES name
&LPAP	LPAP name
&SRFG	Status bits from the LSES table entry (SR)
&PSQN	PET sequence numbers Two 2 byte numbers: – sequence number sent saved – sequence number sent current
&ESQS	Secured sequence numbers Two 2-byte numbers: – SLU TO PLU – PLU TO SLU

Inserts	Meaning
&EBSS	Secured bracket state – "BETB" = BETween Bracket – "INBR" = IN Bracket Receive (local side without send authorization) – "INBS" = IN Bracket Send (local side with send authorization)
&ESQR	Current sequence numbers (from STSN request) Total of 5 bytes: – SEC_TO_PRI_SQN (2 bytes) – PRI_TO_SEC_SQN (2 bytes) – ACTION_CODE_SEC_TO_PRI (2 bits) 00: ignore 01: set 10: test 11: set and test – ACTION_CODE_PRI_TO_SEC (2 bits) 00: ignore 01: set 10: test 11: set and test – RESERVED (4 bits)
&ESRR	Current sequence numbers RSP(STSN) Total 5 bytes: – SEC_TO_PRI_SQN (2 bytes) – PRI_TO_SEC_SQN (2 bytes) – ACTION_CODE_SEC_TO_PRI (2 Bit) 00: ignore 01: set 10: test 11: set and test – ACTION_CODE_PRI_TO_SEC (2 bits) 00: ignore 01: set 10: test 11: set and test – RESERVED (4 bits)
&EBSR	Current bracket state – "BETB" = BETween Bracket – "INBR" = IN Bracket Receive (local side without send authorization) – "INBS" = IN Bracket Send (local side with send authorization)

K089 Request to delete asynchronous message (&GNDATE/&GNTIME) for destination &DEST accepted.
&GNUSER / &USER / (&DLDATE/&DLTIME) / &CHAIN

&CHAIN	Meaning
NO	No negative follow-up message of the deleted message was present.
DEL	The negative follow-up messages should also be deleted.
ACT	The negative follow-up message should be activated, i.e. inserted in the message chain of its receiver.

Any positive follow-up message is always deleted when the job is executed.

K090 Request to delete all asynchronous messages for destination &DEST accepted.
&USER / (&DLDATE/&DLTIME)

K091 Due to a resource bottleneck no sign-on can be processed by application &BCAP at this time.

K092 Please enter password and optional new password

K093 Service cannot be stacked.

K094 Sequence of unsuccessful sign-on attempts

With this message openUTM initiates a silent alarm to indicate a sequence of unsuccessful sign-on attempts by a user or from a client. A response to this situation can be made in the application via the MSGTAC program unit, for example. The SIGNON SILENT-ALARM=*nnn* parameter of the KDCDEF utility can be used to set the number of unsuccessful attempts after which openUTM is to generate message K094.

The message contains the following inserts:

Inserts	Meaning
&PTRM	Terminal name
&PRNM	Processor name
&BCAP	Application name
&LTRM	Name of LTERM partner
&USER	Login name
&RCF1B	Reason for rejection
&REA4	Originator of the rejection (L=LTERM, U=USER, B=BOTH)

Insert &USER contains the login name of the last unsuccessful sign-on attempt in the sequence. This may be a login name generated for the application or any character string passed to UTM as a login name.

Insert &RCF1B consists of three characters indicating the reason why the last sign-on attempt in the sequence was rejected. It is set by UTM as for return fields KCRSIGN1 and KCRSIGN2 of KDCS call SIGN ST.

The &REA4 insert specifies, if the user (U), the client LTERM (L) or both (B) have caused the message. The counter for the unsuccessful sign-on attempts is reset to zero.

K095 KDCOFF effective - input please

K096 End of stacking - input please

K097 Input for new password cannot be used - please sign on

K098 Input exit &RCF1C &RCF2B - input please

This message is generated if errors are detected by UTM or by the exit itself when the INPUT exit is called.

Insert &RCF1C indicates what openUTM discovered during or after the INPUT exit:

&RCF1C	Meaning
IN00	INPUT exit call ok from point of view of UTM
IN01	Error in calling up INPUT exit via IUTMHLL
IN02	KDCS call in INPUT exit
IN03	Output parameters do not match KCICCD
IN04	Illegal value in KCICCD
IN05	DB-USER-CALL in INPUT exit
IN06	STXIT in INPUT exit

Insert &RCF2B indicates what the INPUT exit has entered in the KCIERRCD output parameter in the case of KCICCD = 'ER'.

K099 &MSG

K101 Bottleneck - please repeat last input

This message is output:

- If a dialog input message should be buffered in the page pool due to a TAC class waiting situation, and there is not enough room in the page pool
- If an input message destined for an asynchronous program, a TAC queue or a temporary file is to be saved to the page pool and either warn level 2 has been exceeded or there is not enough room in the page pool
- If an input message is intended for an asynchronous program, a TAC queue or a temporary queue, and the number of messages saved for this asynchronous TAC, this TAC queue or this temporary queue has already reached or exceeded the generated value of QLEV, and the temporary queues were generated with QMODE=STD

K104 UTM-D &UTMDEVT (&RCVDANNO): &LSES , &LPAP , &AGUS
; old state: (&OCVST, &OTAST); action: &ACTION; new state: (&NCVST, &NTAST).

In the case of communication via LU6.1, the message is output on the following events:

- One of the following timers times out:
  - Session/association occupation timer (*time1* in the WAITTIME parameter in the LTAC statement). In this case, &LSES contains blanks.
    Exception: Timeout after occupation of a contention loser session.
    Special case: If the session occupation timer of a contention winner session equals 0 and there is no session free at the end of the program unit, this is treated like the timeout of a timer.
  - Response timer (*time2* in the WAITTIME parameter in the LTAC statement). In this case, &LSES and &AGUS contain different names.
  - Prepare-to-commit timer (value of PTCTIME in the UTMD statement). &LSES and &AGUS are identical.
- A connection loss if the session is occupied
  - by a conversation with an open transaction or
  - by a conversation that has initiated transaction termination.
- In the case of a session restart where the session is occupied by a service that has initiated transaction termination

In the case of communication via OSI TP, the message is output on the expiry of one of the following timers. &LSES always contains blanks.

- Association occupancy timer (for dialog jobs see value *time1* in KCDEF statement LTAC, parameter WAITTIME, for asynchronous jobs internal value of 60 seconds).
  In the case of an asynchronous job, &ACTION contains the value ASYNCH.
- Response timer (for dialog jobs, see value *time2* in KCDEF statement LTAC, parameter WAITTIME, for asynchronous jobs, see value *time2* in KCDEF statement UTMD, parameter CONCTIME).
  In the case of a dialog job, processing waits for the response and in the case of asynchronous jobs, processing waits for the acknowledgment from the job receiver. In the case of an asynchronous job, &ACTION contains the value ASYNCH.
- PEND KP and PGWT KP timers (see KDCDEF statement MAX, parameter TERMWAIT or PGWTTIME).
- Ready timer (see KDCDEF statement UTMD with parameter PTCTIME or KDCDEF statement MAX with parameter PGWTTIME)

Inserts	**Meaning**
&UTMDEVT	Event in response to which the message is issued RESTART Session restart DISCON Connection loss TIMEOUT Time out
&RCVDANNO	Last announcement received at the UTM exchange. In the case of &UTMDEVT = TIMEOUT, the last two bytes in &RCVDANNO have the following meaning: X'F330' X'F331' Timer expiry in OSI-TP job-receiving service after sending a message to a job submitter in a transaction (see KDCDEF statement MAX, parameter TERMWAIT or PGWTTIME). X'F332' Timer expiry in OSI-TP job submitter service after all job receivers have been requested to initiate the end of the transaction. The timer corresponds to the greater of the generated values *time* in the KDCDEF statements MAX PGWTTIME= and *time2* in the KDCDEF statement LTAC WAITTIME= of the participating LTACs. X'F333' Expiry of the ready timer in the OSI-TP job-receiving service (KDCDEF statement UTMD, parameter PTCTIME or KDCDEF statement, parameter MAX PGWTTIME). X'F400' Expiry of the OSI-TP association occupancy timer for a dialog job (*time1* in KDCDEF statement LTAC, parameter WAITTIME). X'F520' Expiry of the OSI-TP association occupancy timer for an asynchronous job (internal 60-second timer).

Inserts	Meaning	
&RCVDANNO (cont.)	X'F522'	Expiry of the timer which monitors the reception of an acknowledgment for a queued message sent via an OSI-TP association (*time2* in KDCDEF statement UTMD, parameter CONCTIME).
	X'F534'	Expiry of the OSI-TP response timer in the job-submitter service for a dialog job. The timer corresponds to the greater of the generated values *time* in the KDCDEF statements MAX PGWTTIME= and *time2* in the KDCDEF statement LTAC WAITTIME= of the participating LTACs
	X'F800'	Expiry of the LU6.1 session occupancy timer for a dialog job (*time1* in KDCDEF statement LTAC, parameter WAITTIME).
	X'F933'	Expiry of the Prepare-to-Commit timer in the LU6.1 job-receiving service (KDCDEF statement UTMD, parameter PTCTIME or KDCDEF statement MAX, parameter PGWTTIME).
	X'F934'	Expiry of the LU6.1 response timer in a dialog job (The timer corresponds to the greater of the generated values for *time* in the KDCDEF statement MAX PGWTTIME= and *time2* in the KDCDEF statement LTAC WAITTIME= of the participating LTACs).
	X'F935'	Expiry of the LU6.1 response timer in a dialog job after re-establishment of the LU6.1 session (The timer corresponds to the greater of the generated values for *time* in the KDCDEF statement MAX PGWTTIME= and *time2* in the KDCDEF statement LTAC WAITTIME= of the participating LTACs)
	X'F936'	Expiry of the LU6.1 response timer on service restart after the establishment of a session (*time2* in KDCDEF statement LTAC, parameter WAITTIME)
&AGUS	Name of the job submitter (user or session)	
&ACTION	Response by openUTM	
	COMMIT	Transaction terminated
	RESET	Transaction reset
	WAIT	Nothing
	STPROG	Continuation program start
	ASYNCH	Re-execution of the asynchronous job.

Inserts	Meaning
&OCVST &OTAST &NCVST &NTAST	Service and transaction status of the job submitter specified in &AGUS (before and after the action specified in &ACTION) Possible values: O O Service and transaction are open. O P The service is open and has initiated transaction termination. O C The service is open, the transaction terminated. C C The service and transaction were terminated. O R The transaction is reset, the service remains open. Z R The transaction is reset and the service terminated.

K105 UTM-D &SYST mismatch; &LSES , &LPAP , &AGUS

&SYST contains the system with which a mismatch occurred:
DB = database / LPAP = other application

K119 OSI-TP error information: &OSLPAP, &USER, &TAC, &DIA1, &DIA2, &DIA3

OSI-TP error information: `&OSLPAP , &USER , &TAC , &DIA1 , &DIA2 , &DIA3`

The explanations in the tables below refer to (primitive) log elements at the XAP-TP interface. These data elements have the following meaning:

Data element	Meaning
APM_ALLOCATE_CNF	Response (confirmation) to an association's request for an OSI-TP dialog. A negative response means that no association could be provided.
TP_BEGIN-DIALOGUE_REQ	Request to the partner to start an OSI-TP dialog.
TP_BEGIN_DIALOGUE_IND	Indication that the partner wants to start an OSI-TP dialog.
TP_BEGIN_DIALOGUE_CNF	Response to the request to start an OSI-TP dialog. A negative response means that the partner has rejected the dialog.
TP_END_DIALOGUE_REQ	Request to the partner to terminate the OSI-TP dialog.
TP_END_DIALOGUE_IND	Indication that the partner wants to terminate the OSI-TP dialog.
TP_U_ABORT_IND	Indication that the remote TP Service User[1] has aborted the OSI-TP dialog.
TP_P_ABORT_IND	Indication that the remote TP Service Provider[2] has aborted the OSI-TP dialog.
TP_DATA_IND	Indication that data has been transferred by the partner.
TP_U_ERROR_IND	Indication that the partner has reported an error.
TP_HANDSHAKE_IND	Indication of a synchronization (handshake) request by the partner.
APUTM_ABORT_REQ	Request to abort a connection.

[1] TP Service User: UTM application

[2] TP Service Provider: System section which provides the OSI-TP service. In the case of openUTM; this is the XAP-TP component.

The &DIA1 insert contains the reason for outputting message K119.

&DIA1	**Meaning**
01	A negative APM_ALLOCATE_CNF was received.
02	A negative TP_BEGIN_DIALOGUE_CNF was received.
03	A TP_BEGIN_DIALOGUE_IND was rejected by openUTM.
04	A TP_U_ABORT_IND or TP_P_ABORT_IND was received.
05	A TP_ERROR_IND was received.
06	UTM does not support reception of TP-HANDSHAKE-IND for a dialog message.
07	openUTM does not support reception of TP-END-DIALOGUE-IND from the client for a dialog message. A dialog RTAC may have been generated for an asynchronous LTAC.
08	With the unchained transactions functional unit, openUTM requires that the transaction starts with a dialog.
09	The association was rejected with APUTM_ABORT_REQ.
10	Resource bottleneck in UTM

The inserts &DIA2 and &DIA3 contain additional information depending on the value of &DIA1:

&DIA1	&DIA2 / Meaning	&DIA3 / Meaning
01	Cause of rejection:	Diagnostic information:
	00 ACSE Service User 01 ACSE Service Provider	-1 No reason given 02 Application Context Name is not supported 03 Job submitter's Application Process Title (APT) is unknown 04 Job submitter's Application Identifier (APID) is unknown 05 Job submitter's Application Entity Qualifier (AEQ) is unknown 06 Job submitter's Application Entity Identifier (AEID) is unknown 07 Job receiver's Application Process Title (APT) is unknown 08 Job receiver's Application Identifier (APID) is unknown 09 Job receiver's Application Entity Qualifier (AEQ) is unknown 10 Job receiver's Application Entity Identifier (AEID) is unknown
	02 Presentation Service Provider	– not relevant
	06 TP service provider	-1 No reason given. 00 CCR Version 2 not available. 01 Incompatible TP protocol version 02 No Contention Winner association available Possible cause: The number of associations in the partner application is smaller than in the local UTM application.
	07 Association Pool Manager	-1 No reason given. 04 Manager cannot establish a further association 06 Local or remote Application Entity Title (AET) is unknown 07 No appropriate association found for specified AET

&DIA1	&DIA2 / Meaning	&DIA3 / Meaning
02	Cause of rejection:	Diagnostic information:
	02 TP service provider 03 TP service user	00 No reason specified 01 Unknown TPSU title. 02 TPSU title permanently unavailable. 03 Possible reasons: – TPSU title temporarily unavailable. – No free OSI-TP instance table for the ACCESS-POINT is available in the application of the job-receiving service. – No free dialog table available. – There is another resource bottleneck in openUTM. Message K119 with &DIA1=10 at the job receiver contains more detailed information about the type of bottleneck. Action: Generate a larger number of associations for the ACCESS-POINT's OSI-LPAP partner at the job receiver. 04 No TPSU title specified. 05 Functional unit not supported. 06 Combination of functional units not supported. 07 Association reserved for partner. 08 Partner AEI unknown. 11 Functional Unit Shared Control is not supported by openUTM 12 TPSU title cannot be decoded. 13 Type of TPSU title not supported. 14 TPSU title too long. 15 Decoding error for user data 16 Decoding error for security PDV. 17 Abstract syntax (UTMSEC) unknown or not generated. 18 User ID unknown or not generated or partner application generated without users or user ID rejected, e.g. due to an incorrect password. 19 Partner unavailable, e.g. deactivated (STATUS=OFF) or due to QUIET command.

&DIA1	&DIA2 / Meaning	&DIA3 / Meaning
02 (cont.)	02 TP service provider 03 TP service user	20 User ID or password too long. 21 Warning level 2 for page pool exceeded. 22 The possible reasons are: – Invalid transaction code – Transaction code locked – No administration rights. 23 Transaction code not generated with CALL=FIRST. 24 Asynchronous service cannot be started. The queue level generated for the asynchronous TAC (maximum number of messages in the message queue) has already been reached. 25 Dialog service cannot be started because the transaction code has been generated as an asynchronous TAC (KDCDEF statement TAC TYPE=A). 26 Combination of functional unit and restart functionality (KDCDISP) is not supported.
03	Cause of rejection:	Diagnostic information:
	03 TP service user	04 TPSU title not specified 11 FU shared control is not supported by openUTM. 12 TPSU title cannot be decoded. 13 Type of TPSU title not supported. 14 TPSU title too long. 15 Decoding error for user data 16 Decoding error for security PDV. 17 Abstract syntax (UTMSEC) unknown or not generated. 18 User ID unknown or not generated or partner application generated without users or user ID rejected, e.g. due to an incorrect password. For more detailed information, see the previous message K147. 19 Partner unavailable, e.g. due to QUIET command. 20 User ID or of the password are too long 21 Warning level 2 for the page pool has been exceeded.

&DIA1	&DIA2 / Meaning	&DIA3 / Meaning
04	Cause of rejection:	Diagnostic information:
	02 TP service provider 03 TP service user	00 No reason specified. 01 A permanent error occurred. 02 TP_BEGIN_TRANSACTION_ REQ rejected. 03 A temporary error occurred. 04 A protocol error occurred. 05 Collision between two TP_END_DIALOGUE_REQ primitives. 06 Collision between TP_BEGIN_TRANSACTION_ REQ and TP_END_DIALOGUE_REQ .
09	Cause of rejection:	Diagnostic information:
	03 TP service User	21 Warning level 2 for the page pool reached. The queued message could not be stored. Action: Generate a larger page pool[1] 31 The KDCFILE is full: The message could not be stored Action: Generate a larger page pool[1] 32 Decoding error for UDT data of a TP-DATA-IND primitive. The association is interrupted in all cases.
10	Resources affected:	Diagnostic information:
	11 No free OSI-TP instance table for this ACCESS-POINT is available in the application of the job-submitting service. The cause is a message to an OSI-TP partner when all OSI-TP instance tables are occupied. The message is entered in a queue. Action: Generate a larger number of associations for the access point's OSI-LPAP partners[2].	Not relevant

&DIA1	&DIA2 / Meaning		&DIA3 / Meaning
10 (cont.)	12	No free OSI-TP instance table for this OSI-LPAP partner is available in the application of the job-submitting service. The cause is a message to an OSI-TP partner when all OSI-TP instance tables are occupied. The message is entered in a queue. Action: Generate a larger number of associations for the access point's OSI-LPAP partners[2]	Index OSI LPAP
	13	The instance is occupied in XAPTP in the application of the job-submitting service. The message is entered in a queue. Action: Generate a larger number of associations for the access point's OSI-LPAP partners[2].	Index UTM instance
	14	A error occurred in the application of the job-submitting service while reading the queued message to be sent. The message is entered in a queue	Not relevant
	15	No free OSI-TP node table available. This error can occur in both a application of a job-submitting service as in a application of a job-receiving service. In a application of a job-submitting service, the message is entered in a queue. Action: Generate a larger number of associations for the access point's OSI-LPAP partners[2].	not relevant

&DIA1	&DIA2 / Meaning	&DIA3 / Meaning
10 (cont.)	16 No free OSI-TP service table available in the application of the job-receiving service. Action: Generate a larger number of associations for the access point's OSI-LPAP partners[2].	not relevant
	17 There is no free OSI-TP user table for the OSI-LPAP partner in the application of the job-receiving service. Possible cause: A dialog program unit whose association has been disconnected in the interim is still active for the OSI-LPAP partner.	Index OSI-LPAP

[1] KDCDEF statement MAX PGPOOL=

[2] KDCDEF statement OSI-LPAP ASSOCIATIONS=

K120 Password expired

K121 Your password is valid for &NUMDAYS more day(s) only

K122 Your password is valid for &NUMDAYS more day(s) only

K123 LTERM does not have the rights to continue the service
- please sign on

K124 Error: &RCXAPTP at startup of XAP-TP occurred in phase: &PHAXAPTP

This message is output as a diagnostic aid for errors which occur when starting XAP-TP

Meaning of the inserts:

&PHAXAPTP Phase during start of XAP-TP. Possible values:
INIT (passing of the generation values to XAP-TP)
START/RECOVERY (OPEN and BIND of the instances and recovery of incomplete transactions).

&RCXAPTP Return code specifying the error in more detail. Depending on the phase (value of &PHAXAPTP), the return codes have different meanings. If the return code is not listed in the following table, the error is internal (in this event, you must write a problem report).

&PHAXAPTP	&RCXAPTP	Meaning
INIT	18	The OSS version is lower than 4.0A20.
	20	KDCFILE and ROOT do not match. Possible cause: A new KDCFILE was generated with KDCDEF (new: with OSI-TP, old: without OSI-TP); the UTM application was, however, not relinked or the ROOT source was not recompiled.
	16 60 85	Not enough memory space could be allocated to cover the OSI-TP-specific generation values.
	21	The KCSALME call to create the transfer buffer for the transfer syntaxes returned a bad return code.
	41	The "apext_init" sequence has already been executed.
	44	The KCSALME call to create the transfer buffer for the abstract syntaxes returned a bad return code.
	63	The KCSALME call to create the transfer buffer for the access points returned a bad return code.
	99	A function called by "apext_att" returned a bad return code. This return code can also mean that the address space that the OSS wants to allocate for its shared memory in a follow-up task is already being used as shared memory (e.g. by KAA, XAPTP global memory). Response: try to start additional tasks.
	102	Error on attach. Action: Check whether an application is already signed-on under this name at this computer.

&PHAXAPTP	&RCXAPTP	Meaning
INIT (cont.)	106	XAPTP "apext_att" call Return code: APEXT_ATTACH_INVEREF Meaning: The waiting point reference specified in the OSS o_attach call is invalid or is not (or no longer) recognized. Possible cause: Follow-on task cannot be started because the application has already been terminated.

K125 Password not complex enough - contact administrator or sign off

K126 SAT error information: &SATRC

K127 Internal error in UTM - ROOT: &ERCD6

Insert &ERCD6 contains the reason why message K127 was output.

&ERCD6	Meaning
RT04	Incorrect action index in KDCRTMM

K128 UTM-D job rejected: &CON/&PRNM/&BCAP/&LPAP &LSES &REA1 &RCDC &TAC

If the job is an OSI-TP UTM-D job, the inserts have the following meanings:

&CON: OSI-CON name
&PRNM: Eight blanks
&BCAP ACCESS-POINT name
&LPAP OSI-LPAP name

Insert &REA1 contains the reason why message K128 was output.

&REA1	Meaning
X'01'	Invalid transaction code LU6.1: DPN or PRN in FMH-5 or FMH-6 or the first 8 characters of the message OSI-TP: Recipient tpsu title in TP-BEGIN-DIALOGUE-RI &RCDC contains an error code KCRCDC
X'02'	Transaction code not generated with CALL=FIRST LU6.1: DPN or PRN in FMH-5 or FMH-6 or the first 8 characters of the message OSI-TP: Recipient tpsu title in TP-BEGIN-DIALOGUE-RI
X'03'	An asynchronous service is to be started, TAC is generated with TYPE=A LU6.1: Message with begin bracket and end bracket OSI-TP: Receipt of an TP-END-DIALOGUE-RI protocol element

&REA1	Meaning
X'04'	A dialog service is to be started, TAC is generated with TYPE=A LU6.1: Message with begin bracket and change direction OSI-TP: Receipt of TP-GRANT-CONTROL-RI or TP-HANDSHAKE-AND-GRANT-CONTROL-RI.
X'05'	An asynchronous service is to be started but the QLEV of the asynchronous TAC has already been reached. LU6.1: The connection is cleared down. OSI-TP: The connection is cleared down.
X'06'	Only in the case of heterogeneous connections via LU6.1: A asynchronous message was received with RQE. openUTM expects RQD2. Action: In the case of CICS, it is necessary to specify NOCHECK and PROTECT in the START command.
X'07'	Only for heterogeneous connection using LU6.1: A message was received with EC and RQD2 but neither CD nor EB is set. Response: modify CICS or IMS program.

K129 &CMD - start parameter not allowed at this moment! Statement ignored

K130 Task priority &TPRIO not allowed for task &TASK! Priority not changed.

K131 &OBJ1 &LOAD1 &OBJ2 &LOAD2 &REA5

K132 &OBJ1 and &OBJ2 have not been generated by the same KDCDEF run

K133 Program for &EXIT missing!

K134 Message to &DEST has been placed into dead letter queue. &NMSG message(s) in dead letter queue

This message informs the administrator that faulty messages have been placed in the dead letter queue.

The inserts have the following meaning:

Insert	Meaning
&DEST	Recipient of the faulty message that led to the dead letter queue threshold being reached.
&NMSG	Generated dead letter queue threshold in number of messages.

K135 UPIC message :
&PTRM/&PRNM/&BCAP/&LTRM/&UPCREAS/&UPCSTAT/&UPCPROT/&UPVENC1/&UPPENC2

A problem occurred in interoperation with a UPIC partner.
The value of insert &UPCREAS indicates the cause:

&UPCREAS	Meaning	Reason / Response
01	UPIC partner sends without send right	Error in product UPIC or UTM
02	UPIC partner sends protocol which is too short	Error in product UPIC
03	UPIC partner uses unsupported protocol version	UPIC version incompatible with UTM version
04	UPIC partner sends invalid protocol	UPIC error
05	UPIC partner has initiated abortion of the conversation	Effect of the UPIC function Deallocate() (CMDEAL())
06	UPIC partner has not sent a TAC	UPIC error
07	TAC sent by the UPIC partner – is not generated – LTERM/USER don't have a key – is admin.TAC and the USER are not admin – is existing and program does not exist.	User error (Side info or SETTP)
08	UPIC partner has sent a TAC generated with CALL=NEXT	User error
09	UPIC partner has sent a TAC which is not permitted	User error (Side info or SETTP)
0A	UPIC partner has sent a TAC which is too short	UPIC error
0B	UPIC partner has sent a TAC when a conversation is open	UPIC error
0C	UPIC partner has sent a TAC for a continuation section	UPIC error
0D	UTM page pool is too small for input message	Page pool for application generated too small (statement MAX PGPOOL)
0E	UPIC partner grants only send right when a message is open	UPIC error
0F	UPIC partner grants send right when a message is open without closure of current message section	UPIC error

&UPCREAS	Meaning	Reason / Response
10	UPIC partner has sent a TAC generated with TYPE=A (asynchronous TAC)	User error
11	UPIC partner is assigned to an LTERM with STATUS=OFF (locked) on connection setup	Normal behavior
12	UPIC partner is assigned to an LTERM with STATUS=OFF (locked) on message receipt	Normal behavior Administrator has locked LTERM
13	UPIC partner has sent USER with invalid length	UPIC error
14	UPIC partner has sent invalid USER or PASSWORD	User error
15	UPIC partner has sent invalid data for service restart, e.g. KDCDISP without USER or with additional user data.	User error
16	UPIC partner has sent invalid data for service restart, e.g. KDCDISP for USER with RESTART=NO	User error
17	Invalid length of the protocol token received.	UPIC error
19	Invalid protocol token has been received.	UPIC error
1A	Overall length of the received data is inconsistent.	UPIC error or transport system error
1B	The protocol token for data was not the last protocol token.	UPIC error
1C	Two protocol tokens for the service were received.	UPIC error
1D	Two protocol tokens for the user ID were received.	UPIC error
1E	Two protocol tokens for the password were received.	UPIC error
1F	Two protocol tokens for the form were received.	UPIC error
20	Two protocol tokens for the function key were received.	UPIC error
21	No protocol tokens for data were received.	UPIC error
22	Inconsistent sign-on data was received.	UPIC error
23	A protocol token was received for a user ID, but not for a service.	UPIC error

&UPCREAS	Meaning	Reason / Response
24	Invalid function key.	UPIC error
25	Protocol token for a form was received in a data fragment.	UPIC error
26	Protocol token for a function was received in a data fragment.	UPIC error
27	Invalid encryption level	UPIC or UTM error
28	Length of the received user message is invalid	UPIC error
29	An error occurred during the protocol discussion	UPIC or UTM error
2A	Two protocol tokens were received for encrypted data	UPIC error
2B	Two protocol tokens were received for one encrypted password	UPIC error
2C	Two protocol tokens were received for one RSA key	UPIC error
2D	Two protocol tokens were received for one DES key	UPIC error
2E	Two protocol tokens were received for one cursor	UPIC error
2F	Two protocol tokens were received for one protocol discussion	UPIC error
30	No protocol tokens were received for encrypted data	UPIC error
31	One protocol token for one password and one encrypted password were received	UPIC error
32	An error occurred while decrypting the password	UPIC or UTM error
33	An error occurred while encrypting the password. The encrypted password is too long.	UPIC or UTM error
34	An error occurred while reading the DES key	UPIC or UTM error
35	The RSA key cannot be sent	UPIC error
36	An error occurred while reading the RSA key	UPIC error
37	An error occurred while encrypting the data	UPIC error
38	An error occurred while decrypting the data	UPIC or UTM error

&UPCREAS	Meaning	Reason / Response
39	Inconsistent message and conversation characteristics have appeared	UPIC error
3A	The protocol discussion could not be completed	UPIC error
3B	An error occurred while exchanging keys	UPIC error
3C	The UPIC partner does not support encryption	Normal behavior, change generation
3D	Two protocol tokens were received for the transaction status	UPIC error
3E	Two protocol tokens were received for the client context	UPIC error
3F	The length of the user message in the connection letter is not compatible with the protocol version	UPIC error
40	openUTM does not support encryption	Normal behavior, change generation
41	Two protocol tokens received for new password	UPIC error
42	Two protocol tokens received for encrypted new password	UPIC error
43	Protocol tokens received for new password and encrypted new password	UPIC error
44	Impermissible operation code occurred with the call of the internal function PASSWD_ENCRYPT_PROC to decrypt a password.	UPIC error
45	Two protocol tokens received for the client type.	UPIC error
46	No RSA key could be found.	Activate one of the RSA keys

The insert &UPCPROT is the hexadecimal UPIC protocol,
&UPCSTAT is used for diagnostic purposes when an error occurs.

Inserts &UPVENC1 and &UPPENC2 are used for diagnosing data encryption.

The first byte of &UPVENC1 contains the ENCRYPTION_LEVEL of the message, the second byte of &UPVENC1 contains the ENCRYPTION_LEVEL of the conversation.

The first byte of &UPPENC2 contains the ENCRYPTION_LEVEL of the session, the second byte of &UPPENC2 contains the ENCRYPTION_LEVEL of the partner.

K136 (First) SYSLOG file is &FNAM

openUTM outputs this message in the start phase. &FNAM contains the name of the SYSLOG file. If the SYSLOG is created as a file generation group, &FNAM then contains the name of the first file generation which is written by openUTM.

K137 SYSLOG switched to file &FNAM

openUTM has successfully switched to a new SYSLOG file generation. The switchover was initiated by the administration or by the automatic size monitoring facility. &FNAM contains the name of the new SYSLOG file generation.

K138 SYSLOG file &FNAM closed

Two situations should be differentiated:
– the SYSLOG is maintained as a single file:
  The last UTM task of the application has closed the SYSLOG file. &FNAM contains the name of the SYSLOG file.
– the SYSLOG is maintained as a SYSLOG FGG:
  A SYSLOG file generation was closed by the last openUTM task (i.e. completely). This file generation is now freely available. It is no longer required by UTM. &FNAM contains the name of the closed file generation.

K139 Switching SYSLOG failed! Still using file &FNAM

The attempt to switch to a new SYSLOG file generation has failed. openUTM continues working with the file generation &FNAM. It may be possible to ascertain the reason for the error occurring on switchover from the DMS error code in the preceding message K043.

K140 There is no supported MUX protocol version in the range from &MXP1 to &MXP2

K141 The MUX protocol version &MXP1 is not supported

K142 Relaese pending timeout for session. PTERM: &PTRM MUX-PTERM: &MXPT

K143 UTM-D: STSN sequence numbers response differ from request. Request: &STS1, &STS2 Response: &STS3, &STS4

K145 Due to a transaction recovery no sign-on can be processed by user &USER at this time - please sign on

K146 Monitoring BCAM waiting time. OPCODE=&BCMOPCD, RTCODE= &BCMRTCD, standard header= &STDHEAD, TSN= &TASK, BCAM appliname= &BCAP

K147 Sign-on for &USRTYPE user &USER not successful.
&PTRM/&PRNM/&BCAP/&LTRM reason: U&REA7

`&PTRM/&PRNM/&BCAP/&LTRM` Reason: `&REA7`.

&USRTYP contains the following values:
– CONNECTION at sign-on of the connection user ID of a connection to a UPIC or TS client
– CLIENT at sign-on of a genuine user ID via TS application, UPIC client or OSI TP partner.

The values in &REA7 Have the following meanings:

&REA7	Meaning
U1	The specified USER does not exist.
U2	The specified USER is locked.
U3	Somebody has already signed on with this USER name.
U4	The “old” password specified is incorrect.
U5	The new password specified cannot be used.
U6	There is no card reader available.
U7	The card information is incorrect
U8	It is not possible to sign on at the moment: – because of a resource bottleneck – because the maximum number of users who can be signed on simultaneously has already been reached – because a password could not be changed, since an inverse KDCDEF is currently running.
U10	The current LTERM partner is not authorized to continue the service.
U11	The password is no longer valid. The password must be changed by the administrator.
U12	The new password does not fulfill the requirements.
U13	The new password is too short.
U14	The password passed by KDCUPD does not fulfil the generated complexity requirements or is too short.
U15	A transaction restart is necessary for the specified user.
U16	The open service cannot be continued from this partner type.
U17	The administrator entered SHUT WARN; normal users can no longer sign on to the application (an administrator can still sign on).
U18	The encryption mechanism required for continuing the open service is not available on the connection

&REA7	Meaning
U19	The validity period of the password has expired. Because Grace Sign-On is generated, sign-on can be repeated by transferring a new password.
U22	The specified USER does not exist in the cluster user file.
U23	Someone else has already signed on to another node with this USER.
U24	It is currently not possible to sign on because the cluster user file could not be locked in the generated time (CLUSTER statement, FILE-LOCK-TIMER-SEC parameter, FILE-LOCK-RETRY parameter).
U25	It is not possible to sign on at this node application because the user has a service that is bound to another node application and may not be terminated.
U26	Sign-on rejected because the open service belonging to the user has a transaction in the PTC state but no service restart has been requested.

K149 Internal diagnostic information &DIA5

An event has occurred which is logged with diagnostic information. &DIA5 is the internal UTM diagnostic information.

K151 Run of inverse KDCDEF terminated. Return code: &IDEFRC &DMSE &FNAM

The insert &IDEFRC contains a 16-digit return code for the inverse KDCDEF. The return code comprises two 8-digit printable numbers.

The first 8 digits have the following meanings:

First 8 digits	Meaning
00000000	The inverse KDCDEF was terminated normally.
00000001	A file handling error occurred.
00000002	A memory bottleneck occurred.
00000003	No KDCDEF statements were generated.

Digits 9 - 16 are only relevant if the first 8 digits contain 00000001. In this event, digits 9 - 16 contain a more detailed description of the error which occurred.
In all other cases, digits 9 - 16 contain 00000000.

Digits 9 - 16	Meaning
00000001	Invalid name for an output file.
00000002	Output file could not be created.
00000003	Output file could not be opened.
00000004	Output file could not be written.

The insert &DMSE contains the printable DMS error code if an error occurs during file handling and digits 9 - 16 of the insert &IDEFRC contain one of the values 00000002, 00000003 or 00000004.

The insert &FNAM contains the name of the file for which the error occurred.

K152 Heuristic report: &COND &MTYPE &OSLPAP &USER &LTAC &AAIS &AAID

The inserts have the following meaning

&COND	**Meaning**
MIX	The server reported data inconsistency. This is not possible with an asynchronous message to a UTM application.
HAZ	Possible data inconsistency. Connection to OSI TP job receiver (subordinate) was lost after a PREPARE was sent and the OSI TP job submitter (superior) had not carried out a backup. The transaction is rolled back on the job submitter and a asynchronous message is sent again. The transaction is also reset in a UTM job receiver, or an asynchronous message is rejected (immediately after a restart), so that the data is consistent.

&MTYPE	**Meaning**
DIAL	A dialog message was sent.
ASYN	An asynchronous message was sent.

&OSLPAP: Name of the partner application.

&USER: Name of the user who issued the job.

&LTAC: for an asynchronous message: LTAC name of the job for a dialog massage: TAC of the transaction for which inconsistency is possible

&AAIS: Size of the atomic action identifier in bytes.

&AAID: contains the encoded atomic action identifier (up to 64 bytes).

K154 Network message: &PTRM/&PRNM/&BCAP/&LTRM &TCPCL &TCPRC

The inserts of the message have the following meaning:

Insert	Meaning
&TCPCL	internal socket function or internal socket event
&TCPRC	Diagnostic Word

&TCPCL is output in printable form:

&TCPCL	Meaning
ATTACH	Create a socket
CONNECTION_REQ	Request the establishment of a connection
CONNECTION_RSP	Confirm the establishment of a connection
DISCONNECT_REQ	Clear connection
DATA_REQ	Send data
DATA_IND	Receive data
GET_HOST_BY_ADDR	Determine computer name of an IP address
GET_HOST_BY_NAME	Determine IP address of a computer

The diagnostic word is 4 bytes long and are specified in hexadecimal. The values have the following meaning:

&TCPRC	Meaning	Cause / Response
00000008	Invalid parameter	
00000014	Connection letter too long	
00000028	The previous send action was not yet completed when the data was sent.	Normal response
00000030	Internal error	See WAIT41 in message K060
00000038	The previous send action was not yet completed when the connection was established/cleared	Normal response
0400001C	Resource bottleneck	Check UTM generation
04000020	Application is not signed on	Check UTM generation
04000024	Connection already cleared	Normal response
08000024	Connection already established	Normal response
10000024	UTM network connection not yet active	Normal response
1C080000	End of application through KDCSHUT utility with time specification	Normal response

&TCPRC	Meaning	Cause / Response
1C0C0000	End of application through KDCSHUT utility without time specification	Normal response
24000004	Connection already cleared	Normal response
24000018	Negative response to request to establish connection	Normal response
24000038	Request to clear connection	Normal response
30000020	Error during sign-on	
FF013008	Only spaces were passed when determining a host's IP address	Check UTM generation
FF023008	It was not possible to determine the host's IP address	Check UTM generation

K155 Your password is expired! - please enter password and new password

K156 UTM-UPIC encryption function available: &RSLT

K158 Inconsistent value for CPU time used: &CPUTEXT

The calculation of the CPU time used by the user resulted in an inconsistent value. The message contains the following inserts:

Insert	Meaning
PTRM	Name of the current PTERM
PRNM	Name of the current processor
BCAP	Name of the current BCAM application
LTRM	Name of the current LTERM
USER	Name of the current user
CPUTEXT	NEGATIVE: A negative value was observed OVERFLOW: Too much time used by the user
CPUBEGIN	CPU time used at the beginning of the period monitored
CPUEND	CPU time used at the end of the period monitored
CPUUSED	CPU time used calculated for the period monitored
CPUCLNT	CPU time actually used by the current user
CPUREAS	Internal diagnostic value

K159 Password for User &USER changed

The password has been changed for the specified user. The message is sent every time the password is changed independently of how the password was changed (administrator, via SIGN CP, etc.)

The message possesses the following inserts:

Insert	Meaning
USER	Name of the user whose password was changed.
ENCPW	Modified password in encrypted form.

This message can be used to transfer modified user passwords to a standby application. To this end, the message destination MSGTAC or USER-DEST must be defined for this message in the private message module. The new user password can be set in encrypted form in the standby application by defining pw_encrypted='Y' via programmed administration KC_MODIFY_OBJECT, object type USER.

Notes:

- You should note that no UTM-D communication is permitted in the MSGTAC program unit. The modified password cannot therefore be sent directly from the MSGTAC program unit via a UTM-D connection to the standby application.
- If the application terminates during productive operation and password changes are made during this period then it is possible that the corresponding messages to the MSGTAC program will not be received but will instead be lost.
- UTM generates the new message for each type of password change and in particular if the password is modified in the standby application. You should make sure that the standby application does not transfer the password back to the productive application in order to avoid any ping-pong effect.
- You should note that UTM uses different encryption algorithms on different platforms. As a result, the function cannot be used on a cross-platform basis.
- The passwords should be generated with the same level of complexity in the involved applications since UTM does not check the complexity when entering the encrypted password.

K160 The &TACNTR. transaction of service &TCVG has been rolled back by &RBCAUSER (&RCCC/&RCDC); (pid: &PID)

The message has the following inserts:

Insert	Meaning
PTRM	Name of the current PTERM
PRNM	Name of the current processor
BCAP	Name of the current application
LTRM	Name of the current LTERM
USER	Name of the current user
TCVG	Name of the service TAC in which the transaction was reset
TAC	Name of the TAC in which the transaction was reset
TACNTR	Number of the reset transaction within the service
RBCAUSER	Initiator of the reset. There are the following values for the initiator, – User PEND RS: Reset was triggered by a PEND RS in the program unit. – System PEND RS: Reset was triggered by a PEND RS initiated by UTM. – User PGWT RB: Reset by triggered by a PGWT RB in the program unit. – System PGWT RB: Reset was triggered by a PGWT RB initiated by UTM. – User PEND ER: Reset was triggered by a PEND ER/FR in the program unit. – System PEND ER: Reset was triggered by a PEND ER initiated by UTM. – DBSTATUS ERROR: Error when querying the database status during node recovery. – DBSTATUS OPEN: The reset was triggered during the warm start due to an unsuccessful database status query. – DB CONFIRM PTC: Error on XA database call CONFIRM PREPARE TO COMMIT STATE during node recovery. – DB CATA VTV: Error on database call CANCEL TRANSACTION during node recovery. – JOURNAL: The reset was triggered during the warm start due to an incompletely written journal record.
RCCC	Value of the compatible KCRCCC return code
RCDC	Valid of the incompatible KCRCDC return code
PID	pid of the process

K161 Synchronous Periodic Write started for task &PID

K162 Long IO for task &PID: &IOPG pages, &IOMS milliseconds

K163 Long Periodic Write for task &PID: &IOPG pages, &IOMS milliseconds

K169 action: &ACTION; node: &NNM1/&HST1/&STATE; node monitoring this node: &NNM2/&HST2; node monitored by this node: &NNM3/&HST3

The message is output when a node signs on or off at a cluster or when a node fails. It provides information during the monitoring relationships in the UTM cluster application.

The message possesses the following inserts:

Insert	Meaning
&ACTION	Describes the situation in which the message is output. It contains one of the following values: – ADD – REMOVE – CHANGE – FAIL
&NNM1, &NNM2, &NNM3	Contain the node names.
&HST1, &HST2, &HST3	Contain the computer names.
&STATE	Contains the status of the node that has been added or removed. If the node has been added then the old status of this node is output; if it has been removed then its new status is output. The insert &STATE can have the following values: G: generated — Node application has not yet been started R: registered — Node application is active T: terminated — Node application was terminated normally A: abterm — Node application was terminated abnormally F: failure — Node application is marked as failed

K170 Cluster: new KDCFILE detected; generation time: &DTTM, state: &STATE

The message is output if the generation time reveals a newly generated KDCFILE which is defined as the basis for future starts of application instances.

The message possesses the following inserts:

Insert	Meaning
&DTTM	Generation time of the new KDCFILE
&STATE	Start type of the local application. The insert &STATE can have the following values: G: generated — Node application has not yet been started R: registered — Node application is active T: terminated — Node application was terminated normally A: abterm — Node application was terminated abnormally F: failure — Node application is marked as failed

K174 Configuration mismatch: &DIA1

The configuration of the starting node application does not match the configuration of the running UTM cluster application.

The insert &DIA1 describes the cause of the error:

– Values ≤ 10: Runtime configuration error
– Values > 10: KDCDEF configuration error

Values of the insert &DIA1 and their meanings:

&DIA1	Meaning
1	Mixed system log file SYSLOG as single file and file generation group (FGG)
2	The encryption capability of the starting node application is different from that of the running node applications.
3	The application program version of the starting node application is different from the application program version of the running node applications.
11	Mixture of UTM-F and UTM-S applications (MAX APPLIMODE)
12	Mixture of single and duplicate file maintenance (second parameter in MAX KDCFILE, MAX USLOG)
13	Mixture of applications with and without users
14	Mixture of applications with and without multiple sign-on permission (SIGNON MULTI-SIGNON)
16	The number of database systems has been modified. (DATABASE, RMXA)
17	The sequence of database statements (DATABASE) has been modified.
18	The password history has been modified (SIGNON PW-HISTORY)

&DIA1	Meaning
19	The number of LSSBs has been changed (MAX LSSB)
20	The number of GSSBs has been changed (MAX GSSB)
21	The number of ULS has been changed (ULS)
22	A ULS from the old KDCFILE no longer exists in the new KDCFILE.
23	The maximum number of services that a user is permitted to stack has been reduced (MAX NRCONV)
24	The maximum number of asynchronous services that may be open simultaneously has been reduced (2nd value in MAX ASYNTASKS)
25	The maximum length of the card information has been reduced (MAX CARDLTH)
26	The maximum length of the Kerberos principal has been reduced (MAX PRINCIPAL-LTH)
27	The size of the page pool has been reduced (1st parameter value in MAX PGPOOL)
28	The size of the process-specific buffer for the restart data has been reduced (2nd parameter value in MAX RECBUF)
29	The length of the communication area has been changed (MAX KB)
30	The length of the default primary work area has been reduced (MAX SPAB)
31	The length of the message area has been reduced (MAX KB)
32	The maximum length of physical output messages has been changed (MAX TRMSGLTH)
33	The maximum length of the user data in LPUT records has been reduced (MAX LPUTLTH)
34	The value of the parameter ABORT-BOUND-SERVICE in the CLUSTER statement has been changed.

K175 File &FNAM created

K176 Procedure &PRCN started. &MSG2 RC = &RCHX

The configured procedure was started on the recognition of a node application failure.

The insert &PRCN contains the started procedure including the passed parameters.

The insert &MSG2 contains additional return information.

Values of the insert &RCHX1 and their meanings:

&RCHX1	Meaning
0	The procedure could not be started successfully
-1	The procedure could not be started. &MSG2 contains additional information.

K178 Cluster journal files: action &ACTION; states (&STATE,&STA2); used pages (&PGS1,&PGS2); number switches &SWNR

Diagnostic information is output for the two files in the administration journal.

The message possesses the following inserts:

Insert	Meaning	
&ACTION	Action of the administration journal:	
	CHANGE	The administration journal file to which data is not currently written has been fully incorporated by all running node applications and an online copy of the administration data has been created; the entries in the file are no longer needed.
	CREATE	The administration journal files have been (re)generated. This is performed on the start of the first node application if the node application was previously regenerated.
	EXTEND	The administration journal file to which data is currently written has been extended because the other administration journal file has not yet been incorporated by all running node applications or the online copy of the administration data has not yet been completed.
	OPEN	The administration journal files have been opened. Output on the start of every UTM process.
	SWITCH	The administration journal file to which data is currently written has been switched.

Insert	Meaning
&STATE / &STA2	Status of an administration journal file: At any time, only one file may have the status C; the other file must have the status O or R . C (Current) This is the file to which data is currently written. O (Old) This file is still being incorporated by running node applications or the online copy of the administration data is not yet terminated. R (Reusable) The data in this file is no longer required and can be overwritten after the administration journal files have been switched.
&PGS1 / &PGS2	Valid pages of an administration journal file: Outputs the number of logically valid pages of an administration journal file: When the file to which data is currently written is switched, it contains only one valid page: the first page containing the control information. Other pages containing old, invalid journal records are not physically released but continue to be occupied.
&SWNR	Number of times the administration journal is switched (ACTION SWITCH)

K179 Data to import completely imported by Online Import

K180 Data to import partially imported by Online Import

K181 No data to import existing for Online Import in file &FNAM.

K182 Abnormal termination of Online Import from file &FNAM.

K183 Normal termination of Online Import from file &FNAM.

K184 There is already an Online Import or a run of KDCUPD active for the old KDCFILE.

K185 The Online Import states that the generation times of the KDCFILEs are not equal

K186 The source application is not part of the cluster application.

K187 Start of Online Import from file &FNAM.

K188 Journal creation time is different. In &OBJ1: &DTTM; in &OBJ2: &DTM2

In the case of UTM cluster applications, this warning is output for diagnostic purposes if different administration journal generation times have been found in files.

The message possesses the following inserts:

Insert	**Meaning**	
&OBJ1 / &OBJ2	File ID.	
	jkaa	JKAA file
	journal	JRN1/JRN2 file
	kdcfile	KDCFILE
&DTTM / &DTM	Journal generation time found in file 1 or file 2.	

K189 Signon rejected - service restart in node application on host &HST1 required

K190 Diagnostic information &DIA1 for cluster &SUFF-file: &INF1: &INF2

The insert &SUFF designates the file to which the message refers.

The insert &DIA1 describes the cause of the error; the inserts &INF1 and &INF2 provide additional information.

The **Grp.** (Group) column in the tables below describes the reason group to which the error code belongs. The following groups exist:

A The cause is a user error, e.g. an error in
- generating and administering UTM applications
- operating UTM applications
- generating the system (e.g. division of the address space)

U The cause is an error in the UTM code.

S The cause is an error in another system component (software or hardware).

M The cause is a memory bottleneck.

I The message is for information only.

Message K190 for the cluster configuration file

If &DIA1 is in the range 1000 to 1099, then &SUFF has the value ’CFG’ and the message refers to the cluster configuration file.

The table below provides an overview of the values of the insert &DIA1 for the cluster configuration file and their meanings, together with the values and meanings of the inserts &INF1 and &INF2 which may occur in combination with &DIA1:

&DIA1	Grp.	Meaning	&INF1	&INF2
1001	A	APPLNAME_MISMATCH Different application names in the cluster configuration file and KDCFILE.	Application name from the cluster configuration file	Application name from the KDCFILE
1002	A	HOSTNAME_NOT_FOUND The host computer name was not found in the cluster configuration file.	Local host name	Host name of the cluster node or blank
1003	A	INVALID_KDCDEF_TIME The generation time of the KDCFILE is too old; another node application with a more recent KDCFILE has already been started.	KDCDEF time from the cluster configuration file	KDCDEF time from the KDCFILE
1004	I	OLD_STATE_INVALID (warning) The old status of the node is "registered" and the start was not a warm start.	G generated: The node has never been started. R registered: The node is active T terminated: The node was terminated normally A abterm: The node was terminated abnormally F failure: The node is marked as failed	D: Start after KDCDEF run C: Cold start W: Warm start U: Start after KDCUPD run
1006	A	CLUSTER_FILE_CHANGED The cluster configuration file was regenerated during the application run.	Generation time from the cluster configuration file or KDCFILE filebase from the cluster configuration file	Generation time from the KDCFILE or KDCFILE filebase from the KDCFILE
1008	U	NODE_NOT_REGISTERED The local application is not entered as registered in the KAA.		

&DIA1	Grp.	Meaning	&INF1	&INF2
1009	A	PLATFORM_MISMATCH The operating system platform of the local application does not match the information from the cluster configuration file.	OS type or bit mode of the local computer	OS type or bit mode from the cluster configuration file
1010	AS	NODE_START_SERIALIZATION The lock for the serialization of the start of different node applications could not be initialized or could not be set.	INIT: Error when initializing lock LOCK: Error when setting lock	1: Timeout 2: Locked 3: Try later
1014	A	INVALID_FILE_VERSION The file version of the cluster configuration file does not match the UTM system code.	Expected version of the cluster configuration file	Actual version of the cluster file
1015	ASU	CLUSTER_FILE_CORRUPTED The start and end markers of the cluster configuration file could not be found; the file is probably not a cluster configuration file.		
1016	A	NODE_KDCFILE_MISSMATCH A node application has already been started on another host computer with this KDCFILE.	Name from the cluster configuration file of the host computer on which a startup has already been performed using this KDCFILE.	Name from the cluster configuration file of the host computer on which a startup is now to be performed using this KDCFILE.
1017	A	NODE_GT_KDCFILE_START The node application was not last started with this KDCFILE.	Time of the last start with this KDCFILE	Time of the last start of this node application.
1018	A	NODE_START_GT_KDCFILE_DEF The newly generated KDCFILE was generated before the last start of the node application.	Generation time of the KDCFILE	Time of the last start of this node application.
1019	A	NODE_NAME_OF_OWN_HOST Node recovery of the node application at the local computer is not permitted.		
1020	A	NODE_NAME_NOT_FOUND Node name for node recovery does not exist in the cluster configuration file.	Node name in the cluster configuration file	Node name in the KDCFILE
1021	A	NODE_NAME_INCONSISTENCY The sequence of node names in the cluster configuration file and the KDCFILE do not match.		

Message K190 for the cluster user file

If &DIA1 is in the range 1100 to 1199, then &SUFF has the value 'USER' and the message refers to the cluster user file. In this case, the message can also be output by the KDCDEF utility program.

The table below provides an overview of the values of the insert &DIA1 for the cluster user file and their meanings:

&DIA1	Grp.	Meaning
1101	AS	TIMEOUT_FOR_USERFILE A timeout occurred on the request for a file lock when opening or closing the cluster user file.
1102	A	NO_USER_FILE The first page in the file is not a CONS page of a cluster user file.
1103	A	INVALID_FILE_VERSION The version number in the cluster user file does not match the version number in the UTM system code.
1104	A	APPLINAME_MISMATCH Different application name in the cluster user file and the KDCFILE.
1105	A	GEN_TIME_MISMATCH The generation time of the cluster user file specified in the cluster user file is not the same as that specified in the KDCFILE.
1106	A	CLUSTER_FILEBASE_MISMATCH The cluster filebase specified in the cluster user file is not the same as that specified in the KDCFILE.
1107	ASU	LAST_PAGE_CORRUPTED The last page in the file is not a CONS page of a cluster user file.
1108	U	PAGE_COUNTERS_INCONSISTENT The counters for the free and used entries in the cluster user file are inconsistent.
1109	U	FREE_ANCHOR_INCONSISTENT The free chain administration fields in the cluster user file are inconsistent.
1110	U	ENTRY_NOT_FOUND An error occurred while addressing an entry in the cluster user file.
1111	U	FREE_ANCHOR_BROKEN The chaining of the first entry in the free chain is inconsistent.
1112	U	USER_STATE_INVALID The status of a user entry in the free chain is invalid.
1113	U	ENTRY_NOT_FREE An entry in the free chain is not identified as free.

&DIA1	Grp.	Meaning
1114	U	END_OF_CHAIN_NOT_FOUND The last element was not found when the free chain was searched.
1115	U	FREE_CHAIN_BROKEN The number of elements in the free chain does not correspond to the counter in the control page.
1116	AU	MAX_FILE_SIZE_REACHED The cluster user file cannot be extended any further since it already contains the maximum number of entries.
1118	A	VERSION_MISMATCH The cluster user file and KDCFILE come from different UTM versions.
1119	A	OS_TYPE_MISMATCH The cluster user file and KDCFILE have a different OS type and/or bit mode.

Message K190 for the cluster JKAA file

If &DIA1 is in the range 1200 to 1399, then &SUFF has the value 'JKAA' and the message refers to the cluster JKAA file.

The table below provides an overview of the values of the insert &DIA1 for the cluster JKAA file and their meanings:

&DIA1	Grp.	Meaning
1202	AU	FILE_IS_EMPTY The JKAA file is empty.
1203	AU	FILE_DOES_NOT_EXIST The JKAA file does not exist.
1204	AU	DMS_ERROR DMS error for the cluster JKAA file Insert &INF1 contains the DMS error code.
1205	M	MEMORY_INSUFFICIENT Error allocating memory for FCB and read buffer. Action: Increase the virtual memory.
1206	AU	FILE_OPEN_ERROR DMS error when opening the JKAA file. Insert &INF1 contains the DMS error code.
1207	ASU	LOCK_INIT_TIMEOUT Timeout when initializing file lock (KCCGFLI).
1208	SU	LOCK_INIT_ERROR Error when initializing file lock (KCCGFLI).
1209	ASU	FILE_LOCK_TIMEOUT Timeout when requesting file lock (KCCGFLK).
1210	SU	FILE_LOCK_ERROR Error when requesting file lock (KCCGFLK).
1211	AU	INVALID_FILE_TYPE The file is not a JKAA file.
1212	AU	FILE_CORRUPTED The second CONS page has an invalid format.
1213	ASU	FILE_UNLOCK_TIMEOUT Timeout when releasing file lock (KCCGFLK).
1214	SU	FILE_UNLOCK_ERROR Error when releasing file lock (KCCGFLK).
1215	ASU	DESTROY_LOCK_TIMEOUT Timeout when destroying file lock (KCCGFLK).
1216	SU	DESTROY_LOCK_ERROR Error when destroying file lock (KCCGFLK).

&DIA1	Grp.	Meaning
1217	SU	FILE_CLOSE_ERROR DMS error when closing the JKAA file.
1218	AU	READ_FILE_ERROR DMS error when reading the KAA pages of the JKAA file. Insert &INF1 contains the DMS error code.
1219	A	VERSION_MISMATCH The KDCFILE and JKAA file have different UTM versions. Insert &INF1 contains the UTM version of the KDCFILE. Insert &INF2 contains the UTM version of the JKAA file.
1220	A	OS_TYPE_MISMATCH The KKAA file and KDCFILE have a different OS type and/or bit mode. Insert &INF1 contains the OS version of the KDCFILE. Insert &INF2 contains the OS version of the JKAA file.
1221	A	APPLINAME_MISMATCH The KDCFILE and JKAA file have different application names. Insert &INF1 contains the application name of the KDCFILE. Insert &INF2 contains the application name of the JKAA file.
1223	A	DATETIME_NOT_EQUAL The KAA generation time of the JKAA file is not the same as the KAA generation time of the KDCFILE even though the node application that is starting has the same generation time as the running UTM cluster application. &INF1 contains the KAA generation time of the KDCFILE. Insert &INF2 contains the KAA generation time of the JKAA file.
1224	A	DATETIME_NOT_LATER The KAA generation time of the KDCFILE is not later than the KAA generation time of the JKAA file even though the starting node application has been newly generated. Insert &INF1 contains the KAA generation time of the KDCFILE. Insert &INF2 contains the KAA generation time of the JKAA file.
1225	A	KCSANT_ERROR The KCSANT component returned an incorrect return code on the inclusion of a KSET for generation in the object tree of the local application.
1226	A	OBJECT_TYPE_ERROR On the generation of a user (USER) or an LU6.1 session in the local application, it was found that an LU6.1 session or user already exists under the index in the local application.
1301	A	BLKSIZE_ERROR The block size of the KDCFILE is not equal to 4K. Action: Generate KDCFILE with MAX BLKSIZE=4K.

&DIA1	Grp.	Meaning
1302	A	BUFFER_RQ_ERROR Error when requesting write/read buffer. Action: Increase the virtual memory.
1303	A	FCB_RQ_ERROR Error requesting memory for FCB. Action: Increase the virtual memory.
1304	AU	FILE_CMD_ERROR DMS error when setting up the JKAA file. Insert &INF1 contains the DMS error code.
1305	AU	OPEN_ERROR DMS error when opening the JKAA file. Insert &INF1 contains the DMS error code.
1306	U	WRITE_NSR_1_ERROR DMS error when writing the first NSR page. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1307	U	WRITE_NSR_ERROR DMS error when writing the follow-up NSR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1308	U	WRITE_NSR_END_ERROR DMS error when writing the last block of NSR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1309	AU	PWRT_LOCK_ERROR Error when requesting periodic write lock.
1310	U	READ_DSR_1_ERROR DMS error when reading the first DSR page. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1311	U	WRITE_SR_ERROR DMS error when writing SR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1312	U	WRITE_SR_END_ERROR DMS error when writing the last block of SR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.

&DIA1	Grp.	Meaning
1313	U	READ_SR_1_ERROR DMS error when reading the first SR page. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1314	U	READ_SR_ERROR DMS error when reading the follow-up SR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1315	U	SR_PAGE_TYPE_ERROR Read SR page has incorrect page type. Insert &INF1 contains the type of the read page. Insert &INF1 contains the half page number.
1316	U	READ_DSR_ERROR DMS error when reading the follow-up DSR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1317	U	DSR_PAGE_TYPE_ERROR Read DSR page has incorrect page type. Insert &INF1 contains the type of the read page. Insert &INF1 contains the half page number.
1318	U	WRITE_DSR_ERROR DMS error when writing DSR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1319	U	WRITE_DSR_END_ERROR DMS error when writing the last block of DSR pages. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1320	U	READ_CONS_ERROR DMS error when reading the first CONS page. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1321	AU	WRITE_CONS_1_ERROR DMS error when writing the first CONS page. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.
1322	AU	WRITE_CONS_2_ERROR DMS error when writing the last CONS page. Insert &INF1 contains the DMS error code. Insert &INF1 contains the half page number.

&DIA1	Grp.	Meaning
1323	U	CLOSE_ERROR DMS error when closing the JKAA file. Insert &INF1 contains the DMS error code.
1324	SU	GFLI_ERROR Error when initializing file lock KCCGFL). Insert &INF1 contains the KCCGFLI return code.
1325	ASU	GFLI_TIMEOUT Timeout when initializing file lock KCCGFLI. Insert &INF1 contains the KCCGFLI return code.
1326	SU	GFLK_LOCK_ERROR Error when requesting file lock KCCGFLK. Insert &INF1 contains the KCCGFLK return code.
1327	AU	GFLK_LOCK_TIMEOUT Timeout when requesting file lock KCCGFLK. Insert &INF1 contains the KCCGFLK return code.
1328	SU	GFLK_UNLOCK_ERROR Error when releasing file lock KCCGFLK. Insert &INF1 contains the KCCGFLK return code.
1329	ASU	GFLK_UNLOCK_TIMEOUT Timeout when releasing file lock KCCGFLK. Insert &INF1 contains the KCCGFLK return code.
1330	SU	GFLK_DESTROY_ERROR Error when destroying file lock KCCGFLK. Insert &INF1 contains the KCCGFLK return code.
1331	ASU	GFLK_DESTROY_TIMEOUT Timeout when destroying file lock KCCGFLK. Insert &INF1 contains the KCCGFLK return code.
1332	U	JFCT_START_ERROR Error when writing the copy status (Started) to the journal file.
1333	U	JFCT_END_ERROR Error when writing the copy status (Completed) to the journal file.

Message K190 for the administration journal files

If &DIA1 is in the range 1400 to 1499, then &SUFF has the value 'JRN' and the message refers to the administration journal.

The table below provides an overview of the values of the insert &DIA1 for the administration journal files and their meanings:

&DIA1	Grp.	Meaning
1401	AU	One or both journal files do not exist. See previous K043 message(s) Action: Regenerate the application.
1402	AU	Error when opening a journal file. See previous K043 message.
1403	AU	Error when creating a journal file. See previous K043 message.
1404	SU	Error when setting up lock for the journal files.
1405	SU	Error when closing a journal file. See previous K043 message.
1406	ASU	Error when releasing reserved pages in a journal file. See previous K043 message.
1407	A	Errored journal file(s): Incorrect ID. &INF1/&INF2: ID in file with suffix JRN1/JRN2. Action: Delete files, regenerate the application.
1408	A	Errored journal file(s): Incorrect file ID. &INF1/&INF2: ID in file with suffix JRN1/JRN2.
1409	A	Errored journal file(s): Incorrect UTM application name. &INF1/&INF2: Application name in JRN1 file / KDCFILE.
1410	A	Errored journal file(s): Incorrect UTM application name. &INF1/&INF2: Application name in JRN2 file / KDCFILE.
1411	A	Errored journal file(s): Incorrect cluster filebase. &INF1/&INF2: Cluster filebase in JRN1 file / KDCFILE.
1412	A	Errored journal file(s): Incorrect cluster filebase. &INF1/&INF2: Cluster filebase in JRN2 file / KDCFILE.
1413	A	Errored journal file(s): Different creation time. &INF1/&INF2: Creation time of file with suffix JRN1/JRN2.
1414	AU	Errored journal file(s): Invalid file status. &INF1/&INF2: File status in file with suffix JRN1/JRN2.
1415	SU	Error when destroying lock for the journal files.
1416	AU	Errored journal file(s): Incorrect file sequence numbers. &INF1/&INF2: File sequence numbers in JRN1/JRN2 file

&DIA1	Grp.	Meaning
1417	A	Errored journal file(s): Incorrect file version. &INF1/&INF2: Version of JRN1 file / expected version.
1418	A	Errored journal file(s): Incorrect file version. &INF1/&INF2: Version of JRN2 file / expected version.

Message K190 for the cluster page pool control file

If &DIA1 is in the range 1500 to 1599, then &SUFF has the value 'CPMD' and the message refers to the cluster page pool control file.

The table below provides an overview of the values of the insert &DIA1 for the cluster page pool control file and their meanings:

&DIA1	Grp.	Meaning
1501	A	Control file has incorrect file version. &INF1/&INF2: Version of CPMD file / expected version. Action: Regenerate all the UTM cluster files.
1502	A	A cluster update with transfer to the CPMD file was terminated abnormally. Action: Regenerate the cluster configuration files.
1503	A	The size of the cluster page pool has been reduced. &INF1/&INF2: Size in CPMD file / size in KDCFILE. Action: Regenerate all the UTM cluster files.
1504	A	The number of files in the cluster page pool has been changed. &INF1/&INF2: Number in CPMD file / number in KDCFILE. Action: Regenerate all the UTM cluster files.
1505	A	Error when trying to increase the size of the files in the cluster page pool. &INF1: 4K page number that could not be written. &INF2: Generated size of the cluster page pool in 4K pages. Action: Make sufficient disk space available for the UTM cluster files.
1506	AU	The size of the free page quota in the cluster page pool is invalid. &INF1: Size of quota as number of UTM pages. Action: Regenerate all the UTM cluster files or write problem report.
1507	AU	The number of pages in the free page quota in the cluster page pool is invalid. &INF1: Number of pages for the full free page quota. Action: Regenerate all the UTM cluster files or write problem report.
1508	AU	Error in management of the free page quota in the cluster page pool. Action: Regenerate all the UTM cluster files or write problem report.
1509	AU	Invalid ID for file size increase in the cluster page pool. &INF1: ID for file size increase. Action: Regenerate all the UTM cluster files or write problem report.

&DIA1	Grp.	Meaning
1510	AU	Invalid ID for running a cluster update with transfer to the CPMD file. &INF1: ID for running cluster update. Action: Regenerate all the UTM cluster files or write problem report.
1511	A	MAX APPLIMODE of the node application has been changed. To change between Secure and Fast, it is necessary to recreate all the UTM cluster files. &INF1/&INF2: APPLIMODE in CPMD file / in KDCFILE
1512	A	MAX VGMSIZE of the node application has been changed. All UTM cluster files must be regenerated. &INF1/&INF2: VGMSIZE in CPMD file / in KDCFILE
1513	A	MAX KB of the node application has been changed. All UTM cluster files must be regenerated. &INF1/&INF2: KB in CPMD file / in KDCFILE
1514	A	MAX NB of the node application has been changed. All UTM cluster files must be regenerated. &INF1/&INF2: NB in CPMD file / in KDCFILE
1515	A	MAX LSSBS of the node application has been changed. All UTM cluster files must be regenerated. &INF1/&INF2: LSSBS in CPMD file / in KDCFILE
1516	A	MAX TRMSGLTH of the node application has been changed. All UTM cluster files must be regenerated. &INF1/&INF2: TRMSGLTH CPMD file / KDCFILE.
1517	A	The number of generated databases has been changed. All UTM cluster files must be regenerated. &INF1/&INF2: Number in CPMD file / KDCFILE.
1518	A	Invalid combination with and without user IDs. All UTM cluster files must be regenerated.
1520	A	Change of CLUSTER ABORT-BOUND-SERVICE. All UTM cluster files must be regenerated.
1521	A	Change of generated database systems. All UTM cluster files must be regenerated.

Message K190 for the cluster ULS file

If &DIA1 is in the range 1600 to 1699, then &SUFF has the value 'ULS' and the message refers to the cluster ULS file in which the administration data for ULS areas is stored in UTM cluster applications.

The table below provides an overview of the values of the insert &DIA1 for the cluster ULS file and their meanings:

&DIA1	Grp.	Meaning
1601	AU	The file is not a cluster ULS file or the file is corrupt.
1602	A	The version of the cluster ULS file is invalid.
1603	A	The application names are different in the cluster ULS file and KAA.
1604	AU	The last page of the cluster ULS file is destroyed.
1605	U	Generation error: The number of ULS blocks from the cluster ULS file is different from the number of ULS blocks in the KDCFILE.
1606	U	Generation error: A ULS from the cluster ULS file is missing in the KDCFILE.
1607	ASU	A timeout occurred while requesting a lock for the cluster ULS file.

Message K190 for the cluster lock file

If &DIA1 is in the range 1700 to 1799, then &SUFF has the value 'LOCK' and the message refers to the cluster lock file for locking global resources in UTM cluster applications (GSSB and ULS).

The table below provides an overview of the values of the insert &DIA1 for the cluster lock file and their meanings:

&DIA1	Grp.	Meaning
1700	AU	An error occurred while opening the existing cluster lock file. See previous K043 message.
1701	AU	An error occurred while creating the cluster lock file. See previous K043 message.
1702	SU	Error while writing the cluster lock file. See previous K043 message.
1703	SU	Error while closing the cluster lock file. See previous K043 message.
1704	SU	Error while opening the cluster lock file. See previous K043 message.
1705	SU	Error while replacing the cluster lock file. See previous K043 message.

&DIA1	Grp.	Meaning
1706	I	Information message - the cluster lock file has been extended. The number of new pages is output.
1707	SU	Fehler beim Schließen der Cluster-Lock-Datei. Siehe vorherige K043-Meldung.

Message K190 for the cluster GSSB file

If &DIA1 is in the range 1800 to 1899, then &SUFF has the value 'GSSB' and the message refers to the cluster GSSB file in which the administration data for GSSB areas is stored in UTM cluster applications.

The table below provides an overview of the values of the insert &DIA1 for the cluster GSSB file and their meanings, together with the values and meanings of the inserts &INF1 and &INF2 which may occur in combination with &DIA1:

&DIA1	Grp.	Meaning	&INF1	&INF2
1800	ASU	GSSBFILE_LOCK_ERROR An error or timeout occurred when requesting or releasing a lock for the cluster GSSB file.	LOCK: Error when requesting lock LOCK: Error when releasing lock	TIMEOUT: Timeout (KCSGFLK return code) TRY LATER: Try later (KCSGFLK return code) LOCKED: File is already locked by own task NOT LOCKED: File is not locked
1801	A	INVALID_FILE_VERSION The version of the cluster GSSB file is invalid.	File version in the GSSB file	Expected file version
1802	AU	HEADER_PAGE_CORRUPTED The control page of the cluster GSSB file is destroyed.	Incorrect field content in the GSSB file	Expected field content
1803	AU	FREE_PAGES_ANCHOR_CORRUPTED The free page chaining of the cluster GSSB file is corrupt.		
1804	AU	NO_GSSBFILE The file is not a cluster GSSB file or the file is corrupt.	File type on the CONS page of the GSSB file	Expected file type
1805	AU	PAGE_COUNTER_INCONSISTENT The page counters of the cluster GSSB file are corrupt.		

&DIA1	Grp.	Meaning	&INF1	&INF2
1806	AU	LAST_PAGE_INCONSISTENCY The file sizes on the CONS and control pages of the cluster GSSB file are different.		
1808	SU	CONS_PAGE_READ_ERROR Error when reading the first CONS page of the cluster GSSB file.		
1809	SU	CONS_PAGE_WRITE_ERROR Error when writing the first CONS page of the cluster GSSB file.		
1810	SU	READ_ERROR_GSSBFILE Error when reading a page of the cluster GSSB file.		
1811	SU	WRITE_ERROR_GSSBFILE Error when writing a page of the cluster GSSB file.		
1812	U	RELEASE_BUFFER_ERROR Error when releasing a page of the cluster GSSB file.		
1814	U	FREE_ENTRY_CHAIN_CORRUPTED The chaining of the free GSSB entries in the cluster GSSB file is corrupt.		
1815	U	HASH_CHAIN_CORRUPTED The hash chaining of the GSSB entries in the cluster GSSB file is corrupt.		
1816	U	FREE_PAGE_CHAIN_CORRUPTED The free page chaining of the cluster GSSB file is corrupt.		
1817	U	GSSB_CNTR_ERROR The counter for the occupied GSSB entries in the cluster GSSB file is corrupt.		
1818	AU	MAX_GSSB_ERROR The value of MAX GSSB in the KDCFILE is different from the value in the cluster GSSB file.	MAX_GSSB in the GSSB file	MAX_GSSB in the KDCFILE

Message K190 for the UTM cluster files

If &DIA1 is in the range 1900 to 1949 then the message refers to the request for a lock for a UTM cluster file and is used for diagnostic purposes only.

The table below provides an overview of the values of the insert &DIA1 and their meanings:

&DIA1	Grp.	Meaning
1900	I	The lock request could not be executed in the generated time (operand CLUSTER FILE-LOCK-TIMER-SEC). The lock request is repeated in accordance with the value generated by the FILE-LOCK-RETRY operand. &INF1: Internal error code of the lock request.
1901	I	This message is issued on the release of the file lock if the lock has been maintained by this process for at least half the time generated in the operand FILE-LOCK-TIMER-SEC in the CLUSTER statement. &INF1: Status of the requested lock. &INF2: Lock maintenance period in seconds.

&SUFF specifies the suffix of the UTM cluster file for which the lock was requested. In Unix and Linux systems, the following suffixes, which stand for real KDCFILE files, are also generated:

IKDC for the imported KDCFILE

OKDC for the local KDCFILE

EKDC for the KDCFILE of another node

Message K190 on the failure of a node application

If &DIA1 is in the range 2001 to 2099, then &SUFF has the value 'KDCA' and the message refers to the KDCFILE of a node application. This message is output if a node failure is detected.

The table below provides an overview of the values of the insert &DIA1 and their meanings:

&DIA1	Grp.	Meaning
2001	I	The application is not active.
2002	S	An error occurred while requesting memory.
2003	ASU	An error was reported by the file system; see also the message K043
2004	SU	An error occurred while initializing the lock.
2005	U	The node could not be found in the UTM cluster file

&INF1 contains the node number of the node that has been identified as failed.

&INF1 contains the file name of the KDCFILE of the node that has been identified as failed.

K191 Dump will be written without &SUFF-file because of request memory error (&RQM bytes requested).

K192 Node recovery for node &NNM1 on host &HST1, RESET-PTC=&RSPTC

This message is output at *stdout* and *stderr* at the start of a node recovery.

K193 PTC found : ID=&PTCID, USER=&USER, LPAP=&LPAP, LSES=&LSES, USER-type=&USTYPPTC

On a node recovery, this message is output once for every transaction with PTC status.

K194 After node recovery: Number of still locked GSSB: &GBLNBR, number of still locked ULS: &ULLNBR

This message is output at the end of node recovery.

If the failed node application for which a node recovery has been performed still has GSSB locks, then this can impair the running UTM cluster application.

K201 XA(&TSNPID) Resource Manager support using X/Open &XASPEC

&XASPEC identifies the version of the XA connection. The insert can have the values `XA-CAE-Spec` or `XA-P-Spec`.

K202 XA(&TSNPID) RM instance &INSTNUM, &TEXT32, &RMSTAT

The message provides information about the status of the XA connection.

The inserts have the following meanings:

&INSTNUM Serial number of the Resource Manager instance, starting with 1. The numbering corresponds to the sequence of the start parameters.

&TEXT32 Name of the Resource Manager.

&RMSTAT Status of the connection. The following values are possible:

&RMSTAT	Meaning
opened	The Resource Manager has been opened
reopened	The Resource Manager has been opened after a restart
closed	The Resource Manager has been closed

K203 XA(&TSNPID) RM &TEXT32,&INSTNUM - recovered transactions: &RTAANZ

The inserts have the following meanings:

&TEXT32 Name of the Resource Manager.

&INSTNUM Serial number of the Resource Manager instance, starting with 1. The numbering corresponds to the sequence of the RMXA start parameters.

&RTAANZ Number of recovered transactions.

K204 XA(&TSNPID) precommit requires global rollback - reason: &XATXT
TA=&INTTAID

Precommit failed, global transaction will be rolled back.

The inserts have the following meanings:

&XATXT Reason for precommit failure. The following values are possible:

&XATXT	Meaning
XA_RBROLLBACK	Rollback for unspecified reason
XA_RBCOMMFAIL	Rollback due to a internal communication error in the Resource Manager
XA_RBDEADLOCK	Rolback due to a deadlock
XA_RBINTEGRITY	Rollback due to a resource inconsistency
XA_RBOTHER	Rollback for unspecified reason

&XATXT	Meaning
XA_RBPROTO	Rollback due to an internal protocol error in the Resource Manager
XA_RBTIMEOUT	Rollback due to transaction period timeout
XA_RBTRANSIENT	Rollback due to a temporary error

&INTTAID Description of the UTM transaction which triggered the global rollback of the other transactions.

Byte 0 contains the instance number of the Resource Manager.

If the XID (= XA transaction identifier) was created in an OSI-TP job-receiver transaction then the AAID (Atomic Action IDentifier) is added here.
Otherwise the subsequent bytes have the following content:

Byte	Meaning
1 - 8	Host name
9 - 16	Application name
17	Type of BS2000 hardware platform (7 ≙ /390, 8 ≙ RISC, 9 ≙ SPARC)
18	Bit mode (1 ≙ 32-bit platform)
19	Byte format (2 ≙ big-endian)
20	Number of the openUTM session
21 - 22	Number of the transaction in the service
23 - 24	Reserved
25 - 28	Service number
29 - 32	Service index

K205 XA(&TSNPID) transaction committed - reason: &XATXT
TA=&INTTAID

The inserts have the following meanings:

&XATXT Reason for committing the transaction. Possible reasons: "Recovery" or "Int.Event".

&INTTAID Description of the UTM transaction that was committed (for contents, see K204).

K206 XA(&TSNPID) transaction rolled back - reason: &XATXT
TA=&INTTAID

The inserts have the following meanings:

&XATXT Reason for the rollback. Possible reasons: "Recovery" or "Int.Event".

&INTTAID Description of the UTM transaction that was committed (for contents, see K204)

K207 XA(&TSNPID) transaction is unknown to the RM&INSTNUM
TA=&INTTAID

The inserts have the following meanings:

&INSTNUM Instance number of the Resource Manager, see K203.

&INTTAID Description of the UTM transaction that is unknown to the RM (for contents, see K204)

K210 XA(&TSNPID) Error: &XATXT - open RM: &TEXT32,&INSTNUM

K211 XA(&TSNPID) Error: &XATXT - close RM: &TEXT32,&INSTNUM

The inserts in the messages K210, K211 and K216 have the following meanings:

&TEXT32 Name of the Resource Manager.

&INSTNUM Instance number of the Resource Manager, see K203.

&XATXT Keyword for the return code from the called XA function. Possible values:

&XATXT	**Meaning**
XA_OK	Normal execution
XA_NOMIGRATE	The transaction was suspended but the context cannot be migrated and the transaction can only be continued in the same process.
XA_HEURHAZ	The transaction may have been heuristically terminated due to a possible internal RM error.
XA_HEURRB	The transaction has been internally rolled back due to a heuristic Resource Manager decision.
XA_HEURMIX	The transaction has been internally partially committed and partially rolled back due to a heuristic Resource Manager decision.
XA_RETRY	The call to the routine must be repeated
XA_RDONLY	The transaction was "read_only" and was committed
XA_RBROLLBACK	Rollback for unspecified reason
XA_RBCOMMFAIL	Rollback due a communication error

&XATXT	**Meaning**
XA_RBDEADLOCK	Rollback due to a deadlock
XA_RBINTEGRITY	Rollback due to a resource inconsistency
XA_RBOTHER	Rollback for unspecified reason
XA_RBPROTO	Rollback due to an internal RM protocol error
XA_RBTIMEOUT	Rollback due to a transaction period timeout
XA_RBTRANSIENT	Rollback due to a temporary error
XAER_ASYNC	An asynchronous operation is still outstanding
XAER_RMERR	A non-recoverable error has occurred in the Resource Manager. Possible cause: The Resource Manager was not initialized or was incorrectly initialized
XAER_NOTA	The transaction identifier is invalid or is unknown to the Resource Manager
XAER_INVAL	Invalid function arguments were specified. It is possible that the Open or Close strings contain incorrect parameters.
XAER_PROTO	The routine was called internally in the RM in the wrong context.
XAER_RMFAIL	The Resource Manager is no longer available
XAER_DUPID	The transaction identifier already exists in the Resource Manager. openUTM terminates the service. To eliminate this residual XID left by XAER_DUTSNPID from the Resource Manager, the database administrator should remove this transaction. Possible action: shut down and restart the database.
XAER_OUTSIDE	The Resource Manager is operating outside of the transaction.

K212 XA(&TSNPID) xa_start(&XAFLAG) - return code: &XATXT
TA=&INTTAID

K213 XA(&TSNPID) xa_end(&XAFLAG) - return code: &XATXT
TA=&INTTAID

K214 XA(&TSNPID) xa_commit() - return code: &XATXT
TA=&INTTAID

K215 XA(&TSNPID) xa_rollback() - return code: &XATXT
TA=&INTTAID

The inserts in the messages K212 to K215 have the following meanings:

&XATXT	Keyword for the return code from the called XA function. Possible values, see K211.
&XAFLAG	Keyword that describes the flags on an xa_start() or xa_end() call.
&INTTAID	Description of the UTM transaction during which the return code occurred (for contents, see K204)

K216 XA(&TSNPID) return code: &XATXT - recover PTC list, RM: &TEXT32,&INSTNUM

For the meaning and content of the inserts, see K211.

K217 XA(&TSNPID) xa_prepare() - return code: &XATXT
TA=&INTTAID

For the meaning of the inserts, see K211 (`&XATXT`) and K215 (`&INTTAID`).

K218 XA(&TSNPID) xa_forget() - return code: &XATXT
TA=&INTTAID

For the meaning of the inserts, see K211 (`&XATXT`) and K215 (`&INTTAID`).

K220 XA(&TSNPID) Error: xa_switch definition not found for specified RM: &TEXT32

A Resource Manager (RM) for which the RMXA start parameter was specified was not defined in the KDCDEF generation.
&TEXT32 contains the name Resource Manager.

K221 XA(&TSNPID) Error: Start parameters not found for defined RM: &TEXT32

No start parameters are specified for a Resource Manager (RM) present in the KDCDEF generation.
&TEXT32 contains the name Resource Manager.

K222 XA(&TSNPID) Error: Linked RM is not &XASPEC compatible: &TEXT32

The Resource Manager (RM) does not use the XA interface that was specified during KDCDEF generation.
&TEXT32 contains the name of the Resource Managers and &TEXT32 the XA interface specified during generation.

K223 XA(&TSNPID) Syntax error in start parameters:

The following line contains the incorrect line from the start parameter file.

K224 XA(&TSNPID) &XACALL - return code &XATXT from RM instance &INSTNUM, &TEXT32 is not XA(CAE) compliant

The Resource Manager (RM) returns an unexpected return code.

The inserts have the following meanings:

&XACALL	Name of the call to the XA interface, e.g. xa_start().
&XATXT	Keyword for the return code from the called XA function (see K211).
&INSTNUM	Instance number of the Resource Manager, see K203.
&TEXT32	Name of the Resource Manager.

K225 XA(&TSNPID) recursive call of XA module: function: &XADBC1 - error/signal in DB/XA connection for &XADBC2

The meaning of the inserts is as follows:

&XADBC1	Name of the called function in the UTM/XA connection module.
&XADBC2	Name of the function in which an error was detected in the UTM/XA connection module.

K230 XA(&TSNPID) Int. error: &TEXT32

&TEXT32	Specifies an internal error. This may, for example, indicate a lack of memory space for malloc().

K231 XA(&TSNPID) Int. error: PETA not supported

Preliminary end of transaction not supported.

K232 XA(&TSNPID) Int. error: DBSTAT secondary opcode inconsistent

The secondary operation code of the primary operation code DBSTAT is inconsistent.

K233 Heur. decision in task / process &TSNPID, RM=&INSTNUM in &XACALL(&DBCALL), xa-rc=&XATXT, LTHGTRID=&LTHGTRID, GTRID=&GTRID

The meaning of the inserts is as follows:

&INSTNUM	Instance number of the Resource Manager (RM), see K203.
&XACALL	Name of the call to the XA interface, e.g. xa_start().
&DBCALL	Name of the called UTM-DB interface function
&XATXT	Keyword for the return code from the called XA function (see K211).
&GTRID	Global transaction ID (gtrid) of the UTM transaction in accordance with the XA/CAE specification (for contents, see K204).
&LTHGTRID	Length of the global transaction ID.

K235 Name resolution for &PRNM lasts &TCPMS milliseconds (socket call: &TCPCL, returncode: &TCPRC, IP address: &IPADDR).

K251 Version V&IMPVER of filebase &FBASUPD does not match current version V&DEFVER.

K252 &UPDERR error

If &UPDERR is set to the value PARAM, then this points to a syntax error. KDCUPD aborts and sets switch 3 to ON.
Response: enter the correct command!
Otherwise an error will occur when requesting memory, when reading in data, etc.
Response: write a problem report with documentation.

K255 DMS error &DMSE on file &FNAM

Error in file processing.
Response: according to DMS error code, see page 377.

K256 File type of &FNAM is not a valid UTM type

The file is not of type KDCFILE (KDC, pool, conf file)

K257 Application run was not terminated correctly

K258 File &FNAM has already been used

K260 Unknown version &DEFVER of filebase &FBASUPD

K261 Inconsistent file &FNAM

The file is not consistent in itself
Response: restore the file or repeat generation.

K262 Wrong file &FNAM

The file was overwritten by another KDCFILE file; see sdterr log for further details.
Response: make correct file available or repeat generation.

K263 File &FNAM has not yet been used

K269 &OBJ1 &OST1 &BMD1 and &OBJ3 &OST2 &BMD2 are not compatible

K273 Error &TRMA in module &UPDMODUL

Description of the causes of the error for message K273, plus action:

&TRMA	Cause	Response
BFMMER	Error initializing buffer management (cluster)	if necessary, increase virtual address space or PM
CFGERR	Error processing the cluster configuration file	previous diagnostic messages, or if there are none: inconsistent node name sequence
CONSER	Internal error	PR + documents
DMSERR	A DMS error has occurred	See message K255
DxxxRD	DMS error xxx on reading the first page	Depends on DMS error code, see also page 377
DxxxWR	DMS error xxx on writing the first page	
xxxx01	DMS error xxxx on reading the first KAA page	
GMDTER	Error checking the cluster GSSB file	See previous diagnostic messages
INCONS	Error during consistency checks	See messages K261/K262
LOOKLT	Error searching for LTERM	PR + documents
LOOKFL LOOKFU	Error searching for initiator (LTERM, USER)	PR + documents
LOOKLT	Error searching for LTERM	PR + documents
LOOKTC	Error searching for TAC	PR + documents
LOOKUS	Error searching for USER	PR + documents
OPWRON	Internal error	PR + documents
PPMMER	Cluster page pool management file incorrect.	See previous diagnostic messages
REQM01	'Request memory' error for 1st page	Regenerate system if necessary
REQM02	Insufficient memory when creating shared memory for UTM cache	Check operation system generation or reduce the CACHESIZE operand
REQKTA	Error in memory request for KTA	See REQM01
SHM002	Cause: UTM application is running at the moment. A KDCUPD run is therefore not possible.	Response: terminate the application.
UMDTER	Error checking the cluster ULS file	See previous diagnostic messages
UPDSTA	File has already been processed with KDCUPD	None
USFERR	Error opening the cluster user file	See previous diagnostic messages

&TRMA	Cause	Response
...GSB ....GB	Internal error	PR + documents
...TLS ....TL	Internal error	PR + documents
...ULS ....UL	Error during ULS handling	PR + documents
..DIAL	Error during transfer of chained services	PR + documents
...ASY ....AS	Error handling FPUTs or DPUTs to LTERM	PR + documents
...IMS ....IM	Error handling FPUTs or DPUTs to TAC	PR + documents
...UPI ....UP	Error while processing a dialog message for UPIC	Write problem report and create documentation
...SOC ....SO	Error while processing a dialog message for SOCKET	Write problem report and create documentation
...UMS ....UM	Error while processing a user queue message	Write problem report and create documentation
...QMS ....QM	Error while processing a queue message	Write problem report and create documentation
...LSB	Error in LSSB handling	PR + documents
xxxx99	Internal error in KDCUPD	PR + documentation

Further &TRMA error codes may be the term application codes of openUTM. In this case, you should always write a PR and compile documentation.

K274 Terminated with a dump

K277 At least one node application was not terminated correctly.

For more details, see stderr log.

K278 At least one node application has already been started.

K279 Error in cluster-handling, see (possible) diagnostic message before

K300 &UPDPRO percent of &PGPOOL used.

K303 &UKCOP data transferred. KCRN = &UKCRN, type = &UPDTYP, KCLM = &UKCLM.

K304 Service data transferred for user &USER. Service type = &TACTYPE, height of stack = &UKCHSTA

K305 Used pages of &PGPOOL for &UPDTYP: &PGS1 overall, &PGS2 for &UKCRN.

K306 Used pages of &PGPOOL for &UPDTYP: &PGS1.

K310 * &UPDTYP &UKCRN not found

K311 * &UPDTYP &UKCRN not found. No data transferred

K314 * Warning: LPUT data cannot be transferred

K317 * &UKCOP data not transferred. KCRN= &UKCRN, Type = &UPDTYP, KCLM = &UKCLM, KCRCCC = &RCCC, KCRCDC = &RCDC . Caused by LTERM = &LTRM , user = &USER

K318 * &UPDTYP &UKCRN : sender of asynchronous message not found. LTERM = &LTRM, user = &USER

K320 * Service data for user &USER not transferred. Service type = &TACTYPE, reason: &UERCODE, &UERINFO, &RCDC

5.2 Messages of the XAP-TP provider

The messages from the XAP-TP provider all start with the letter “P”. The values for the inserts are either described following the message or (if the insert occurs a number of times) in section “General inserts for the XAP-TP messages” on page 330.

P001 Error on OSS call (&XPFUNC): &ACPNT, &XPRET, &XPERR, &XP1INFO, &XP2INFO

This message is output if a call to an OSS function (&XPFUNC) returns an error. If the error has been reported by the transport system, message P012 is also output.

The inserts have the following meaning:

&XPFUNC	Name of the OSS function
&ACPNT	Name of the local ACCESS-POINT
&XPRET	See table on page 330
&XPERR	See table on page 330
&XP1INFO	Supplementary OSS information
&XP2INFO	Supplementary OSS information

P002 Error on association establishment (&XPFUNC): &ACPNT, &OSLPAP, &XPRET, &XPERR, &XP1INFO, &XP2INFO

This message is issued if the call to an OSS function (&XPFUNC) required to establish an association returns an error. If the error has been reported by the transport system, message P012 is also output. If the error has not been reported by the transport system, the application is terminated with “Termapplication”.

The inserts have the following meaning:

&XPFUNC	Name of the OSS function
&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application
&XPRET	See table on page 330
&XPERR	See table on page 330
&XP1INFO	Supplementary OSS information
&XP2INFO	Supplementary OSS information

P003 Association rejected (a_assin()):&ACPNT, reason: &XPRJCT, length: &XPLTH

This message is issued if a request to establish an association was rejected from outside.

The inserts have the following meaning:

&ACPNT	Name of the local ACCESS-POINT
&XPLTH	Incorrect length
&XPRJCT	See table on page 333

P004 Association rejected (a_assin()):&ACPNT, &OSLPAP, reason: &XPRJCT

This message is issued if a request to establish an association was rejected from outside.

The inserts have the following meaning:

&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application
&XPRJCT	See table on page 333

Possible causes in the case of &XPRJCT = 34 or &XPRJCT = 35 (NO_MORE_CONTENTION_LOSER / WINNER_ASSOCIATIONS, page 333):

– Associations have been disconnected without the local UTM application being informed of this. The local UTM application considers that these associations still exist. The OSI-TP partner application attempts to re-establish these associations. However, these cannot be re-established until either the affected associations have been disconnected due to the expiry of the idle timer or all associations to the relevant partner have been disconnected at the administrative level.

– More associations are generated in the OSI-TP partner application than in the local UTM application.

– Different numbers of contention losers/winners in the local and partner applications

P005 Association rejected (a_assin()):&ACPNT, reason: unknown partner
N-SEL: &XPNSEL, T-SEL: &XPTSEL
S-SEL: (&XPLSSEL,&XPCSSEL,&XPHSSEL)
P-SEL: (&XPLPSEL,&XPCPSEL,&XPHPSEL)

This message is issued if a request to establish an association was rejected from outside because the remote partner is not known to the local application.
The inserts have the following meaning:

&ACPNT Name of the local ACCESS-POINT

&XPNSEL Network selector of the remote partner

&XPTSEL Transport selector of the remote partner

&XPLSSEL Length of the session selector of the remote partner

&XPCSSEL Session selector (printable) of the remote partner

&XPHSSEL Session selector (hexadecimal) of the remote partner

&XPLPSEL Length of the presentation selector of the remote partner

&XPCPSEL Presentation selector (printable) of the remote partner

&XPXPSEL Presentation selector (hexadecimal) of the remote partner

P006 Association rejected (a_assin()): &ACPNT, &OSLPAP, reason: wrong application context name (&XP0OBID, &XP1OBID, &XP2OBID, &XP3OBID, &XP4OBID, &XP5OBID, &XP6OBID, &XP7OBID, &XP8OBID, &XP9OBID)

This message is issued if a request to establish an association was rejected from outside. The application context name for the remote partner does not match the application context name generated for this partner in the local application.

The inserts have the following meaning:

&ACPNT Name of the local ACCESS-POINT

&OSLPAP Name of the partner in the local application

&XP0OBID - &XP9OBID
These are (up to) ten elements of the object identifier which form the application context name of the remote partner.
-1 is output for elements which do not have a value assigned.

P007 Error on association establishment (a_assrs()): &ACPNT, &OSLPAP, &XPRET, &XPERR, &XP1INFO, &XP2INFO

This message is output when a call to the OSS function *a_assrs()* to respond to a request to establish an association from outside returns an error. If the error has been reported by the transport system, message P012 is also output.

The inserts have the following meaning:

&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application
&XPRET	See table on page 330
&XPERR	See table on page 330
&XP1INFO	Supplementary OSS information
&XP2INFO	Supplementary OSS information

P008 Association established: &ACPNT, &OSLPAP

This message is issued when an association has been established.

The inserts have the following meaning:

&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application

P009 Association rejected (a_asscf()): &ACPNT, &OSLPAP, reason: &XPRJCT, length: &XPLTH

This message is issued when active establishment of an association is rejected because the confirmation from the partner cannot be accepted.

The inserts have the following meaning:

&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application
&XPRJCT	See table on page 333
&XPLTH	Possible incorrect length

P010 Association rejected (a_asscf()): &ACPNT, &OSLPAP, reason: unknown partner
N-SEL: &XPNSEL, T-SEL: &XPTSEL
S-SEL: (&XPLSSEL,&XPCSSEL,&XPHSSEL)
P-SEL: (&XPLPSEL,&XPCPSEL,&XPHPSEL)

This message is issued when active establishment of an association is rejected, because the remote partner confirms establishment of an association with an address (&XPADDR) which is unknown to the local application.

The inserts have the following meaning:

&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application
&XPNSEL	Network selector of the remote partner
&XPTSEL	Transport selector of the remote partner
&XPLSSEL	Length of the session selector of the remote partner
&XPCSSEL	Session selector (printable) of the remote partner
&XPHSSEL	Session selector (hexadecimal) of the remote partner
&XPLPSEL	Length of the presentation selector of the remote partner
&XPCPSEL	Presentation selector (printable) of the remote partner
&XPXPSEL	Presentation selector (hexadecimal) of the remote partner

P011 Association rejected (a_asscf()): &ACPNT, &OSLPAP, reason: wrong application context name (&XP0OBID, &XP1OBID, &XP2OBID, &XP3OBID, &XP4OBID, &XP5OBID, &XP6OBID, &XP7OBID, &XP8OBID, &XP9OBID)

This message is issued when active establishment of an association is rejected, because the remote partner confirms establishment of an association with an application context name other than the one configured for this partner in the local application.

The inserts have the following meaning:

&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application
&XP0OBID - &XP9OBID	These are (up to) ten elements of the object identifier which form the application context name of the remote partner. *-1* is output for elements which do not have a value assigned.

P012 CMX diagnostic information: &XPCTYPE, &XPCCLS, &XPCVAL

This message is issued if a preceding message is issued as a result of an error reported by the transport system. The diagnostic code of the transport system is print-edited. The following table describes a number of values for &XPCTYPE, &XPCCLS and &XPCVAL. The CMX header file `cmx.h` contains a complete list.

XPCTYPE	Meaning (CMX error type)
0	T_CMXTYPE: CMX error detected by the CMX library
2	T_DSTEMPERR: Temporary TNS error
3	T_DSCALL_ERR: TNS call error
4	T_DSPERM_ERR: Permanent TNS error
5	T_DSWARNING: TNS warning
>15	CMX error on the basis of error codes from the transport system

XPCCLS	Meaning (CMX error class, valid for &XPCTYPE < 15)
0	T_CMXCLASS: CMX class
2	T_DSNOT_SPEC: TNS class not specified
3	T_DSPAR_ERR: TNS parameter error
4	T_DSILL_VERS: Invalid TNS version
5	T_DSSYS_ERR: TNS system error
6	T_DSINT_ERR: Internal TNS error
7	T_DSMESSAGE: TNS note

XPCVAL	Meaning (CMX error value)
0	T_NOERROR: No error
5	T_EIO: Temporary bottleneck or error in the transport system
14	T_EFAULT: IO_Area not allocated
100	T_UNSPECIFIED: Unspecified error, generally a system call error
101	T_WSEQUENCE: Invalid call sequence
103	T_WPARAMETER: Invalid parameter
104	T_WAPPLICATION: The task is not authorized to sign on to the application or the application has already been opened by this task.
105	T_WAPP_LIMIT: The limit for the number of simultaneously active connections has already been reached.

XPCVAL	Meaning (CMX error value)
106	T_WCONN_LIMIT: The limit for the number of simultaneously active applications has already been reached.
107	T_WTREF: Invalid transport reference or the transport connection has already been established.
111	T_NOCCP: The transport system does not support the requested application or connection.
114	T_CCP_END: The transport system has been terminated or the application was closed by the administrator.
255	T_WLIBVERSION: No connection to the CMX subsystem possible.
-100	T_INVREF: Invalid evid. CMX cannot assign the call to a wait point.

P013 Association rejected (a_asscf()): &ACPNT, &OSLPAP, reason: &XPCRES, &XPSRC, &XPNDIA
CCR V2 = &XP1BOOL, Version Incompatibility = &XP2BOOL
ContWin Assignment rejected = &XP3BOOL
Bid mandatory rejected = &XP4BOOL, No reason = &XP5BOOL

This message is issued when active establishment of an association is rejected by the remote partner.

The inserts have the following meaning:

&ACPNT Name of the local ACCESS-POINT

&OSLPAP Name of the partner in the local application

&XPCRES Specifies whether the rejection is temporary or permanent:
0= permanent reject
1= transient reject

&XPCSRC Specifies who has rejected establishment of the association:
0 = ACSE service user
1 = ACSE service provider
2 = Presentation service provider

&XPNDIA See table on page 337

&XP1BOOL - &XP5BOOL

These inserts can take the values TRUE or FALSE. Values of TRUE indicate the reasons the partner reported for rejecting the request to establish an association:

&XP1BOOL: CCR Version 2 is not available
&XP2BOOL: The TP protocol versions are not compatible
&XP3BOOL: The contention winner assignment has been rejected
&XP4BOOL: The specification “Bidding is mandatory” or “Bidding is not mandatory” has been rejected
&XP5BOOL: No reason is specified

Possible causes for the rejection of the contention winner assignment (&XP3BOOL=TRUE):

– Fewer associations are generated in the OSI-TP partner application than in the local UTM application.

– Different number of contention losers/winners in the local and the partner application.

– Associations have been disconnected without the OSI-TP partner application being informed of this. The OSI-TP partner application rejects the establishment of associations since it considers that they are still established.
In this situation, the message P004 with &XPRJCT = 34 or &XPRJCT = 35 is generated in the OSI-TP partner application (NO_MORE_CONTENTION_LOSER / WINNER_ASSOCIATIONS, page 333).

P014 Error on association disconnection (&XPFUNC): &ACPNT, &OSLPAP, &XPRET, &XPERR, &XP1INFO, &XP2INFO

This message is issued if the call to an OSS function (*&XPFUNC*) required to establish an association returns an error. If the error has been reported by the transport system, message P012 is also output. If the error has not been reported by the transport system, the application is terminated with “Termapplication”.

The inserts have the following meaning:

&XPFUNC	Name of the OSS function
&ACPNT	Name of the local ACCESS-POINT
&OSLPAP	Name of the partner in the local application
&XPRET	See table on page 330
&XPERR	See table on page 330
&XP1INFO	Supplementary OSS information
&XP2INFO	Supplementary OSS information, currently always set to zero.

P015 Association disconnected (&XPFUNC): &ACPNT, &OSLPAP, &XPLNK, &XPSRC, &XPNDIA, &XPINI, &XP1INFO, &XP2INFO

This message is issued when an association is cleared. The inserts have the following meaning:

&XPFUNC Name of the OSS function

&ACPNT Name of the local ACCESS-POINT

&OSLPAP Name of the partner in the local application

&XPLNK Represents the internal status of the association
0 = Association not linked
1 = Association linked to channel
2 = Association linked to instance

&XPCSRC Originator of clear-down
0 = ACSE service user
1 = ACSE service provider
2 = Presentation service provider

&XPNDIA See table on page 337

&XP1INFO Supplementary OSS information

&XP2INFO Supplementary OSS information

&XPINI See table below

XPINI	Meaning
0	Association was cleared internally.
401	O_LOC_TRAN The originator is the local transport system. &XP1INFO contains the CMX return code. This is output in detail in the subsequent message P012.

XPINI	Meaning
402	O_REM_TRAN The originator is the remote transport system. &XP1INFO contains the reason for the CMX event t_disin. The values are defined in `cmx.h`. Some of the possible values for &XP1INFO and their meaning are contained in the list below. The following abbreviations are used: – CCP (**C**ommunication **C**ontrol **P**rogram) for the application program that controls communication – TSAP (**T**ransport **S**ervice **A**ccess **P**oint) for the access point to the transport service, – NSAP (Network Service Access Point) for the access point to the network service. – PDU (**P**rotocol **D**ata **U**nit) for data elements in a protocol layer. &XP1INFO can, for example, take on the following values: 0 (T_USER) The communication partner cleared the association, possibly as a result of a user error on the partner side. 1 (T_RTIMEOUT) The connection was cleared locally by CMX because the connection had been inactive for too long according to the t_timeout parameter. 2 (T_RADMIN) The connection was cleared locally by CMX because the administrator closed down CCP. 3 (T_CCPEND) The connection was cleared locally by CMX because CCP failed. 256 (T_RUNKOWN) Either the partner or CCP cleared the connection. No reason was given. 257 (T_RSAP_CONGEST) The partner CCP cleared the connection because of a TSAP-specific bottleneck. 258 (T_RSAP_NOTATT) The partner CCP cleared the connection because the addressed TSAP was not registered there. 259 (T_RUNSAP) The partner CCP cleared the connection because the addressed TSAP was not known there. 261 (T_RPERMLOST) The connection was cleared by the network administrator or the partner CCP administrator. 262 (T_RSYSERR) Error in the network

XPINI	Meaning
402 (cont.)	385 (T_RCONGEST) The partner CCP cleared the connection as a result of a resource bottleneck. 386 (T_RCONNFAIL) No connection could be established. The partner CCP aborted the attempt due to failure. 387 (T_RDUPREF) The partner CCP cleared the connection because a second connection reference was assigned for an NSAP pair (system error). 388 (T_RMISREF) The partner CCP cleared the connection because a connection reference could not be assigned (system error). 389 (T_PROTERR) The partner CCP cleared the connection because of a protocol error (system error). 391 (T_PREFOFLOW) The partner CCP cleared the connection because of a connection reference overflow. 392 (T_RNOCONN) The partner CCP rejected the request to establish a network connection. 394 (T_RINLNG) The partner CCP cleared the connection because of an incorrect length header or parameter (system error). 448 (T_RLCONGEST) The local CCP cleared the connection because of a resource bottleneck. 449 (T_RLNOQOS) The local CCP cleared the connection because the “quality of service” could not be maintained. 451 (T_RILLPWD) Invalid connection password. 452 (RNETACC) Access to the network was refused. 464 (T_RLPROTERR) The local CCP cleared the connection because of a transport protocol error (system error). 465 (T_RLINTIDU) The local CCP cleared the connection because it received an interface data unit which was too long (system error).

XPINI	Meaning
402 (cont.)	466 (T_RLNORMFLOW) The local CCP cleared the connection because of an infringement of the flow control rules for normal data (system error). 467 (T_RLEXFLOW) The local CCP cleared the connection because of an infringement of the flow control rules for expedited data (system error). 468 (T_RLINSAPID) The local CCP cleared the connection because it received an invalid TSAP identification (system error). 469 (T_RLINCEPID) The local CCP cleared the connection because it received an invalid TCEP identification (system error). TCEP **= T**ransport **C**onnection **E**nd **P**oint. 470 (T_RLINPAR) The local CCP cleared the connection because of an invalid parameter value (e.g. user data too long or expedited data not permitted). 480 (T_RLNOPERM) The administrator of the local CCP prevented establishment of a connection. 481 (T_RLPERMLOST) The administrator of the local CCP cleared the connection. 482 (T_RLNOCONN) The local CCP could not establish the connection because no network connection is available. 483 (T_RLCONNLOST) The local CCP cleared the connection because the network connection was lost. Most common cause: generation error on the CCP and PDN side, e.g. incorrect link addresses. Other possible causes: partner is not available, modem is faulty or has been set incorrectly, data transfer connection not plugged in, data transfer card faulty. 484 (T_RLNORESP) The local CCP cannot establish the connection, because the partner did not respond to the CONRQ. 485 (T_RLIDLETRAF) The local CCP cleared the connection because the connection was lost (Idle Traffic Timeout). 486 (T_RLRESYNC) The local CCP cleared the connection because the resynchronization was not successful (more than three attempts were made). 487 (T_RLEXLOST) The local CCP cleared the connection because the expedited data channel is faulty (more than three attempts were made).

XPINI	Meaning
403	O_LOC_SESS The originator is the local session provider. &XP1INFO can take the following values: 4 (S_PROTERROR) Protocol error: Incorrect establishment of the session PDU or incorrect SPDU parameter 16 (S_PICSREST) Violation of implementation-specific restrictions.
404	O_REM_SESS The originator is the remote session provider. &XP1INFO can take the following values: 1 (S_TCDISCON) transport disconnect 4 (S_PROTERROR) protocol error 8 (S_UNDEFINED) undefined 16 (S_PICSREST) violation against restriction stated in PICS
405	O_LOC_PRES The originator is the local presentation provider. &XP1INFO can take the following values: 0 (P_ARRNO) reason not specified – A decoding buffer requested internally cannot be provided due to a lack of memory. – Overflow of the internal data buffer when reassembling fragmented messages – An unknown session event was reported. System bottleneck or system error. 1 (P_ARNRPDU) unrecognized PPDU – No session user data is available or the presentation part of the session user data cannot be decoded (system error). 4 (P_ARNRPAR) unrecognized PPDU parameter – Error on decoding the ACSE, presentation or user syntax. 5 (P_ARNEPAR) unexpected PPDU parameter – PPDU parameter not in normal mode.

XPINI	Meaning
405 (cont.)	6 (P_ARNIPAR) invalid PPDU parameter – Invalid context identifier on decoding. – Invalid PPDU parameter, e.g. incorrect length. This “abort” can be triggered by the UTM user by specifying invalid presentation or session selectors.
406	O_REM_PRES The originator is the remote presentation provider. &XP1INFO can take the following values: -1 (O_NOVALUE) Optional parameter is not present 0 (P_ARNNO) Reason not specified 1 (P_ARNRPDU) Unrecognized PPDU 2 (P_ARNEPDU) Unexpected PPDU 3 (P_ARNESSP) Unexpected session service primitive 4 (P_ARNRPAR) Unrecognized PPDU parameter 5 (P_ARNEPAR) Unexpected PPDU parameter 6 (P_ARNIPAR) Invalid PPDU parameter value
407	O_LOC_ACSE The originator is the local ACSE provider &XP1INFO always has the following value: 1 (A_ABSASP) ACSE service provider initiated the abort The instance is specified which initiated the abort (“abort source”) from the point of view of ACSE.

XPINI	Meaning
408	O_REM_ACSE The originator is the ACSE service provider. &XP1INFO can take the following values: 0 (A_ABSASU) ACSE service user initiated the abort 1 (A_ABSASP) ACSE service provider initiated the abort

P016 Association disconnected (a_relin()): &ACPNT, &OSLPAP, &XPLNK, &XPNDIA

This message is issued if an association is cleared because a “release indication” was received. The inserts have the following meaning:

&ACPNT Name of the local ACCESS-POINT

&OSLPAP Name of the partner in the local application

&XPLNK Represents the internal status of the association
0 = Association not linked
1 = Association linked to channel
2 = Association linked to instance

&XPNDIA See table on page 337

P017 OSS decoding error: &XPPDU, &XP1DIA, &XP2DIA, &XP3DIA

This message is issued if OSS detects an error on decoding a TP PDU, CCR PDU or user data PDU. The insert `&XPPDU` indicates the type of PDU in question. The inserts have the following meanings:

XPPDU	Meaning
0	PDU_UNKNOWN (unknown PDU type)
1	TP_BEGIN_DIALOGUE_RI
2	TP_BEGIN_DIALOGUE_RC
3	TP_BID_RI
4	TP_BID_RC
5	TP_END_DIALOGUE_RI
6	TP_END_DIALOGUE_RC
7	TP_U_ERROR_RI
8	TP_U_ERROR_RC
9	TP_ABORT_RI
10	TP_GRANT_CONTROL_RI

XPPDU	Meaning
11	TP_REQUEST_CONTROL_RI
12	TP_HANDSHAKE_RI
13	TP_HANDSHAKE_RC
14	TP_HSK_AND_GRT_CTRL_RI
15	TP_HSK_AND_GRT_CTRL_RC
16	TP_DEFER_RI
17	TP_PREPARE_RI
18	TP_HEURISTIC_REPORT_RI
19	TP_TOKEN_GIVE_RI
20	TP_TOKEN_PLEASE_RI
21	TP_RECOVER_RI
22	TP_INITIALIZE_RI
23	TP_INITIALIZE_RC
24	CCR_INITIALIZE_RI
25	CCR_INITIALIZE_RC
26	CCR_BEGIN_RI
27	CCR_BEGIN_RC
28	CCR_PREPARE_RI
29	CCR_READY_RI
30	CCR_COMMIT_RI
31	CCR_COMMIT_RC
32	CCR_ROLLBACK_RI
33	CCR_ROLLBACK_RC
34	CCR_RECOVER_RI
35	CCR_RECOVER_RC
50	PDU_ANY
51	PDU_UASE_RI

XP1DIA / XP2DIA	Meaning
1	not supported parameter was received and skipped
2	received data truncated
4	required transfer syntax name missing in user data or not specified in AVX list, error codes in &XP2DIA

XP1DIA / XP2DIA	Meaning
6	no transfer syntax name in user data though presentation negotiation was not completed
7	transfer syntax name encoded in user data not found in AVX list
10	invalid value in data structure
11	invalid object identifier in data structure
12	invalid length or count in data structure
13	invalid index in data structure (EXTERNAL, CHOICE)
14	invalid value of ax_typtag in corresponding syntax table

&XP3DIA Corresponding index in the syntax table

P018 FSM protocol error: &ACPNT, &OSLPAP, &XPPTYP, &XPFSMN

This message is issued when the finite state machine reports an error.

The inserts have the following meaning:

&ACPNT Name of the local ACCESS-POINT

&OSLPAP Name of the partner in the local application

&XPPTYP Type of the service protocol element

&XPFSMN Name of the finite state machine

P019 APDU contains invalid value: &ACPNT, &OSLPAP, &XPAPDU, &XP3INFO

This message is issued if an invalid APDU is received.

The inserts have the following meanings:

&ACPNT Name of the local ACCESS-POINT

&OSLPAP Name of the partner in the local application

&XPAPDU Type of the APDU

&XP3INFO Supplementary information on the error

P020 OTRACE implicitly switched off. Reason: &XPTRFAIL

This message is issued when an attempt to write a trace record fails. The OSS trace is deactivated implicitly as a result of the error. After the error has been corrected, the administrator can reactivate the OSS trace.

The inserts have the following meaning:

XPTRFAIL	Meaning
1	The OSS function o_wutr() issued the return code O_ERROR. The preceding P001 message provides further information on the error.
2	The OSS function o_wutr() issued the return code O_INVEREF.
3	The OSS function o_wutr() issued an unknown return code.

P021 Unexpected event &XPEVT occurred for association, event discarded: &ACPNT, &OSLPAP, &XPOSAS, &XPASST

This message is output if an event occurs which is incompatible with the current status of the association. XAPTP does not take account of this event.

The inserts in the message have the following meanings:

&XPEVT Type of event that has occurred.

&ACPNT Name of the local access point (KDCDEF statement ACCESS-POINT).

&OSLPAP Name of the OSI-LPAP partner in the local application.

&XPOSAS Index of the relevant association.

&XPASST Status of the relevant association.

5.2.1 General inserts for the XAP-TP messages

XPRET	Meaning
2	Not first process of application
-1	Function call not successful due to permanent error.
-2	Function call not successful due to transient error. Retry the call later
-3	Function call not successful, data flow stopped Continue after event GO
-4	Session call: Expedited function call stopped due to expedited data flow control shortage Continue after event S_XGO/S_GO Presentation call: Function call not successful, apref invalid Local function call: Invalid connection reference ACSE call: Function call not successful, apref resp. aref invalid
-5	Invalid waiting point reference
-6	Invalid application reference
-7	Waiting period to obtain a lock on a shared association expired

XPERR	Meaning
1	No memory available (temporary)
100	Call sequence error
101	Application not attached
102	Sending of data not allowed; wait for GO event
103	Internal error
104	Shared association is not locked
200	Missing ACSE/presentation reference
201	Invalid ACSE/presentation reference
202	Presentation call: missing AVX list (o_attach) ACSE call: missing application reference
203	Presentation call: invalid AVX list ACSE call: invalid application reference

XPERR	Meaning
204	Presentation call: invalid abstract syntax name in AVX ACSE call: missing ACSE parameters
205	Presentation call: invalid decoding mode in AVX ACSE call: missing presentation parameters
206	Presentation call: invalid user data length ACSE call: missing session parameters
207	Presentation call: invalid context id in p_udl ACSE call: missing application context name
208	Presentation call: invalid next parameter in p_udl ACSE call: invalid application context name
209	Presentation call: invalid pdv parameter in p_udl ACSE call: invalid calling AP Title
210	Presentation call: invalid chaining parameter ACSE call: invalid calling AE Qualifier
211	Presentation call: missing token parameter ACSE call: invalid called AP Title
212	Presentation call: invalid token parameter ACSE call: invalid called AE Qualifier
213	Presentation call: missing rtype parameter ACSE call: invalid responding AP Title
214	Presentation call: invalid rtype parameter ACSE call: invalid responding AE Qualifier
215	Presentation call: missing type parameter ACSE call: missing called p_address
216	Presentation call: invalid type parameter ACSE call: invalid called p_address
217	Presentation call: invalid syncp parameter ACSE call: missing calling p_address
218	Presentation call: missing syncp parameter ACSE call: missing responding p_address
219	Presentation call: invalid ctxlst parameter ACSE call: no mode parameter
220	Presentation call: invalid number of abstract syntaxes passed to OSS ACSE call: invalid mode parameter
221	Presentation call: invalid transfer syntax name ACSE call: missing result
222	Presentation call: invalid number of transfer syntaxes ACSE call: invalid result

XPERR	Meaning
223	Presentation call: invalid number of abstract syntaxes ACSE call: missing result source
224	Presentation call: same abstract syntax occurred already in transparent or non-transparent mode ACSE call: invalid result source
225	Presentation call: invalid data separation parameter ACSE call: invalid diagnostic
226	ACSE call: missing reason
227	ACSE call: invalid reason
228	ACSE call: missing provider reason
229	ACSE call: invalid provider reason
230	ACSE call: missing abort source
231	ACSE call: invalid p-requirements
232	ACSE call: invalid s-requirements
233	ACSE call: invalid syntax identifier
234	ACSE call: invalid p-context identifier
235	ACSE call: invalid p-context definition list
236	ACSE call: invalid p-context definition result list
237	ACSE call: invalid result in p-context definition result list
238	ACSE call: invalid default p-context result
239	ACSE call: invalid default p-context name
240	ACSE call: invalid user data length
241	ACSE call: invalid quality of service
242	ACSE call: invalid sync point serial number
243	ACSE call: invalid tokens
244	ACSE call: invalid SS-user reference
245	ACSE call: invalid SS-common reference
246	ACSE call: invalid SS-additional reference
250	Presentation call: ASN encoding error ACSE call: ASN encoding error
251	Presentation call: ASN decoding error ACSE call: ASN decoding error
252	Presentation call: ASN: invalid value in data struct ACSE call: ASN: invalid value in data struct

XPERR	Meaning
253	Presentation call: ASN: invalid object id in data struct ACSE call: ASN: invalid object id in data struct
254	Presentation call: ASN: invalid length in data struct ACSE call: ASN: invalid length in data struct
255	Presentation call: ASN: invalid index in data struct ACSE call: ASN: invalid index in data struct
256	Presentation call: ASN: invalid tag in syntax table ACSE call: ASN: invalid tag in syntax table
300	Presentation call: invalid protocol state ACSE call: invalid protocol state Local function call: error on system call
301	Presentation call: protocol error ACSE call: protocol error Local function call: error on transport system call
302	Local function call: error on local function call
305	Local function call: error on session call
306	Local function call: error on presentation call
307	Local function call: error on ACSE call

XPRJCT	Meaning
0	NO_REJECT
1	APPLICATION_CONTEXT_NAME_TOO_LONG The object identifier received from the partner, which forms the application context name, contains more elements than supported by openUTM.
2	CALLING_APT_TOO_LONG A length was specified for the application process title in the association indication which is not supported by openUTM.
3	CALLING_AEQ_TOO_LONG A length was specified for the application entity qualifier in the association indication which is not supported by openUTM.
4	CALLED_APT_TOO_LONG The application process title which was called is longer than that supported by openUTM.
5	CALLED_AEQ_TOO_LONG The application entity qualifier which was called is longer than that supported by openUTM.

XPRJCT	Meaning
6	CONTEXT_DEFINITION_LIST_TOO_LONG More abstract syntaxes were passed for the association indication than are supported by openUTM.
7	CONTEXT_RESULT_LIST_TOO_LONG The list of supported abstract syntaxes passed when establishing an association (association indication or confirmation) contains more elements than are supported by openUTM.
9	ADDRESS_NO_PSAP_INFO The address passed for association indication or confirmation does not contain any information on PSAP.
10	ADDRESS_NO_INFO_VERS_0_PSAP The address passed for association indication or confirmation contains an incorrect version for the PSAP information.
11	ADDRESS_INVALID_P_SEL_LENGTH The address passed for association indication or confirmation contains an invalid length for the presentation selector.
12	ADDRESS_NO_SSAPINFO The address passed for association indication or confirmation does not contain any information on SSAP.
13	ADDRESS_NO_INFOVERS_0_SSAP The address passed for association indication or confirmation contains an incorrect version for the SSAP information.
14	ADDRESS_INVALID_S_SEL_LENGTH The address passed for association indication or confirmation does not contain a valid part for the session selector.
15	ADDRESS_NO_PARTNER_MODE The address passed for association indication or confirmation does not contain a valid part for the network and transport selector.
16	ADDRESS_TNSX_ERROR The address passed for association indication or confirmation has been rejected by TNS.
17	UNKNOWN_PARTNER The address passed for association indication or confirmation is not known in the local application.
18	WRONG_APPLICATION_CONTEXT_NAME The application context passed for association indication or confirmation does not correspond to the application context name generated in the local application.
19	ABSTRACT_SYNTAX_MISSING The association indication or confirmation supports less abstract syntaxes than are generated in the local application.

XPRJCT	Meaning
20	OSITP_SYNTAX_MISSING The association indication or confirmation does not support the abstract syntax for OSI-TP.
21	NO_TP_INITIALIZE No TP-INITIALIZE-RI/RC PDU was passed with the association indication or confirmation.
22	OSITP_NO_VERSION_1 The partner does not support Version 1 of the OSI-TP protocol.
23	OSITP_RCH_WRONG_LENGTH The recovery context handle passed with the TP-INITIALIZE-indication or TP-INITIALIZE-confirmation is of a length not supported by openUTM.
24	NO_CCR_INITIALIZE The CCR-INITIALIZE-RI PDU is missing.
25	CCR_NOT_VERSION_2 The partner does not support Version 2 of the CCR protocol.
26	SESSION_NO_FDX Session functionality “full duplex” has not been set.
27	SESSION_NO_DATA_SEPARATION Session functionality “data separation” has not been set although CCR is in the context.
28	SESSION_NO_TYPED_DATA Session functionality “typed data” has not been set although CCR is in the context.
29	SESSION_NO_MINOR_SYNCHRONIZE Session functionality “minor synchronize” has not been set although CCR is in the context.
30	SESSION_NO_RESYNCHRONIZE Session functionality “resynchronize” has not been set although CCR is in the context.
31	TOKEN_CONTENTION_WINNER_AND_NO_TOKEN The local application is the contention winner, but does not possess the “token” (only if CCR is in the context).
32	TOKEN_CONTENTION_LOSER_AND_TOKEN The local application is the contention loser, but possesses the “token” (only if CCR is in the context).
33	INITIAL_SYNC_POINT_SERIAL_NUMBER_NOT_SET The initial syncpoint serial number is not set, although CCR is in the context.
34	NO_MORE_CONTENTION_LOSER_ASSOCIATIONS The request to establish an association from outside is rejected because all the contention loser associations have already been established in the local application.
35	NO_MORE_CONTENTION_WINNER_ASSOCIATIONS Request to establish an association from outside is rejected because all the contention winner associations have already been established in the local application.

XPRJCT	Meaning
36	CCR_BUT_NO_PARTNER_AET Partner did not specify an application entity title, although CCR is in the context.
37	CCR_BUT_NO_OWN_AET No application entity title is specified in the local application, although CCR is in the context.
38	RESPONDING_APT_TOO_LONG The application process title specified in the association confirmation is longer than that supported by openUTM.
39	RESPONDING_AEQ_TOO_LONG The application entity qualifier specified in the association confirmation is longer than that supported by openUTM.
40	ASS_ESTABLISHMENT_TIMEOUT The establishment of an association started by the local application cannot be completed in the specified time.
41	PARTNER_IS_IN_QUIET_STATE The request for establishment of an association will be rejected because the partner in the local application has been set to Quiet.
42	NO_SPACE_FOR_RCH The PutElement call for storing the recovery context handle returned a bad value.
43	REMOTE_AET_2_BIG The application entity title of the partner is longer than that supported by openUTM.
44	REMOTE_AET_CHANGED When establishing parallel associations to a partner the partner did not provide the same application entity title as for the first association established.
45	NO_SPACE_FOR_REMOTE_AET The PutElement call for storing the application entity title of the partner returned a bad value.
46	PARTNER_HAS_STATUS_OFF The establishment of the association is rejected because the partner is locked in the local UTM application (STATUS=OFF is set).

XPNDIA	Meaning
0	NO_REASON_GIVEN
1	NO_COMMON_ACSE_VERSION The partner rejected the request to establish an association because there is no common ACSE version.
2	APPL_CONTXT_NAM_NOT_SUPPORTD The partner rejected the request to establish an association because it does not support the application context name.
3	CALLING_AP_TITLE_NOT_RECON The partner rejects the request to establish a connection because the partner has not been generated correctly at the sender (e.g. incorrect N-SEL). or (only for heterogeneous links): The partner rejects the request to establish a connection because it does not know the Calling Application Process Title
4	CALLING_AE_QUALI_NOT_RECON The partner rejected the request to establish an association because it does not know the calling application entity qualifier.
5	CALLING_AP_INVOC_ID_NOT_RECON The partner rejected the request to establish an association because it does not know the calling application process invocation identifier.
6	CALLING_AE_INVOC_ID_NOT_RECON The partner rejected the request to establish an association because it does not know the calling application entity invocation identifier.
7	CALLED_AP_TITLE_NOT_RECON The partner rejected the request to establish an association because it does not know the called application process title.
8	CALLED_AE_QUALI_NOT_RECON The partner rejected the request to establish an association because it does not know the called application entity qualifier.
9	CALLED_AP_INVOC_ID_NOT_RECON The partner rejected the request to establish an association because it does not know the called application process invocation identifier.
10	CALLED_AE_INVOC_ID_NOT_RECON The partner rejected the request to establish an association because it does not know the called application entity invocation identifier.
11	PERMANENT_FAILURE The partner cleared the association because a permanent error occurred.
12	BEGIN_TRANSACTION_REJECT The partner cleared the association because it rejected the start of a transaction.

XPNDIA	Meaning
13	TRANSIENT_FAILURE The partner cleared the association because a temporary error occurred.
14	PROTOCOL_ERROR The partner cleared the association because a protocol error occurred.
15	UNRECOGNIZED_PDU The association was cleared from outside with P-ABORT because the presentation layer received an unknown presentation PDU.
16	UNEXPECTED_PDU The association was cleared from outside with P-ABORT because the presentation layer received an unexpected presentation PDU.
17	UNEXPECTED_SESSION_SERVICE_PRIMITIVE The association was cleared from outside with P-ABORT because the session layer received an unexpected session service primitive.
18	UNRECOGNIZED_PDU_PARAMETER The association was cleared from outside with P-ABORT because the presentation layer received an unknown PPDU parameter.
19	UNEXPECTED_PPDU_PARAMETER The association was cleared from outside with P-ABORT because the presentation layer received an unexpected PPDU parameter.
20	INVALID_PPDU_PARAMETER_VALUE The association was cleared from outside with P-ABORT because the presentation layer received an invalid PPDU parameter value.
21	RELEASE_NORMAL The association was cleared by the partner with release. The partner specified release normal as the reason.
22	RELEASE_URGENT The association was cleared by the partner with release. The partner specified release urgent as the reason.
23	RELEASE_USER_DEFINED The association was cleared by the partner with release. The partner specified user defined as the reason
24	IDLE_TIMEOUT_ABORT The association was cleared by the local application because the association was not used in the time generated with IDLETIME.
25	ABORT_BY_ASS_INDICATION The association was terminated by the local UTM application because a request to establish an association was received and no suitable, non-established association was available.

5.3 Messages of the KDCDEF generation tool

K400 KDCDEF &DEFTYP &VERS started

K401 Please enter control statements

K402 Syntax error: &SYNERR.

K403 Illegal value entered for parameter &PARAM15.

K404 DMS error &ERROR for file &FNAM.

The DMS error codes are described on page 377.

K405 &ERROR error.

K406 Illegal value entered for operand &PARAM15. &PARVAL8 assumed

K407 Unable to create entry for user "&PARVAL8" in cluster user-file

K408 &STMTNAME name &UTMNAME must not start with "KDC".

K409 LTERM parameter is mandatory for PTERM statements with PTYPE=APPLI, SOCKET, UPIC-L or UPIC-R.

K410 Warning: LTERM is assigned to a PTERM with PTYPE=APPLI, SOCKET, UPIC-L or UPIC-R and an illegal USER or USAGE=O was specified.

K411 Mandatory parameter &PARAM15 has not been declared in any &PARAM15 control statement.

K412 DESTADM &UTMNAME is not defined as asynchronous TAC or as LTERM. No DESTADM assumed

K413 Ambiguous definition of &STMTNAME &OPERAN32 &STRING64

K414 No LSES control statement assigned to LPAP &UTMNAME.

K415 Illegal parameter for MAX DPUTLIMIT1 or DPUTLIMIT2. Default assumed

K416 &PARAM30 defined more than once.

K417 &PARAM10 but type is not "&PARAM30".

K418 CID parameter is mandatory for a PTERM controlled by a printer control terminal.

K419 Printer pool defined but type is not "OUTPUT".

K420 LTERM with USAGE=D must not be assigned to a PTERM with USAGE=O.

K421 &PARAM30 and &PARAM50 are mutually exclusive.

K422 Mandatory parameter &PARAM10 has not been specified.

K423 Warning: size page pool (PGPOOL) is not greater than size LPUT buffer (LPUTBUF).

K424 Invalid key value: &KEYVALUE. Valid key values are: 0 <= key <= MAX KEYVALUE.

K425 &STMTNAME name &PARAM32 has not been defined.

K426 &STMTNAME control statement missing

K427 Event exit &EXITNAME has already been defined.

K428 Invalid CARD parameter: &CARDINFO or POSITION = 0.

K429 Too many database types

K430 Too many &STMTNAME control statements

K431 User "&PARVAL8" from cluster user-file is missing in KDCFILE.

K432 No &STMTNAME with &PARAM32 .

K433 Filebase name is too long. The configuration run will be aborted.

K434 Minimum length of buffer for recovery data, RECBUF parameter, is 1024. 1024 assumed

K435 The event exit &PARAM8 must be assigned to an ASSEMBLER program unit.

K436 MAX statement is missing. The configuration run will be aborted.

K437 Program error in KDCDEF; reason: &TERMREAS.

TERMREAS describes the cause that lead to the abnormal termination of the utility.

The description of UTM message K060 also helps to clear the error.

Action: Proceed as described for K060.

In the other cases, there is an error in KDCDEF. Store the documents and write a PR.

K438 More LSES statements than CON statements have been assigned to LPAP &UTMNAME.

K439 In parameter KDCFILE the file base name was specified with a catalogue ID.

K440 &PARAM8 is not allowed in &STMTNAME statements with &PARAM10.

K441 SHARE table &UTMNAME is empty.

K442 Inclusion of &PARAM8 is only meaningful if the &STMTNAME is assigned to a &PARVAL8 program unit.

K443 Inclusion of ULS control statements is only meaningful if USER statements were generated.

K444 OPTION control statement is only effective when reading from stdin.

K445 &PARAM30 no longer supported.

K446 Parameter ignored.

K447 Control statement ignored.

K448 KDCFILE generated with warnings; KAA size: &KAASIZE K

K449 There was at least one ERROR. Configuration run aborted.

K450 KDCFILE generated; KAA size: &KAASIZE K

K451 File &FNAM generated.

K452 Too many &PARAM10 names. The configuration run will be aborted.

K453 BCAMAPPL name different from APPLINAME is only allowed for a &STMTNAME with PTYPE=APPLI, SOCKET or UPIC-R.

K454 Length of LTERM parameter &UTMNAME incompatible with NUMBER parameter.

K455 TAC assigned to STACK parameter may not be defined with TYPE=A or CALL=NEXT.

K456 It is not possible to assign a group-LTERM to a PTERM

K457 Warning: &PARAM10 larger than &PARAM10A

K458 LTERM &PARAM8 has been assigned as CTERM for LTERM &PARAM8A. "USAGE = OUTPUT" is not meaningful.

K459 The LTERM specified at GROUP must not be a group-LTERM itself.

K460 File &FNAM processed.

K461 &STMTTP30 &STMTNM30 not supported by &UTMVAR

K462 Only one OSI-CON connection group is allowed to be active with local partner application &PARAM8.

K463 &PARAM30 set to &PARAM30.

K464 Length of session names + length of associations > 8.

K465 &STMTNAME control statement is mandatory if &PARAM30 is generated.

K466 Parameter &PARAM30 is mandatory if &PARAM30 is generated.

K467 Error: Multiple entries for user "&PARAM8" found in cluster user-file

K468 Password parameter does not fit security level.

K469 Parameter &PARAM32 only allowed if APPLIMODE = SECURE.

K470 No user is able to call KDCSHUT.

K471 &OPERAND&PARAM10 already set.

K472 &PARAM32 too long.

K473 Parameter &PARAM10 used more than once for &PARAM30

K474 Parameter &PARAM30 is illegal if &PARAM30 is generated.

K475 Parameter &PARAM30 not needed if &PARAM30 is generated.

K476 In &STMTNAME control statements &OPERAND is given more than once for PRONAM=&PARAM8 and BCAMAPPL=&PARAM8 .

K477 Parameter &PARAM30 is illegal if any &PARAM30 is given.

K478 Control statement &STMTNAME is illegal if &PARAM30 is given.

K479 &STMTNAME &PARAM50 not used.

K480 Too many &STMTNAME control statements for this &PARAM30.

K481 The primary LTERM of a group must not be slave of a bundle.

K482 EXIT and TAC must be in the same LOAD-MODULE if the exit program is referencing a LOAD-MODULE with LOAD-MODE=ONCALL.

K483 &PARAM32 must not reference a &STMTNAME with &PARAM32 .

K484 &PARAM32 with &PARAM32 not generated.

K485 Control statement too long.

K486 Different libraries given for same DATABASE TYPE.

K487 Cannot assign a master LTERM of bundle to PTERM.

K488 All slave LTERMs of bundle must be assigned to PTERMs with identical PTYPE (APPLI or SOCKET).

K489 Master LTERM of bundle must not be a slave itself.

K490 Master LTERM must be generated with QAMSG=YES and RESTART=YES. Parameters changed.

K491 &OPERAND&PARAM32&PARAM20 will not be supported in the next UTM version.

K492 Note: multiple ACCESS-POINT statements may cause problems.

K493 The creation of the TNS input file failed.

K494 Operand &PARAM30 is mandatory if &STMTNAME control statement is given more than once.

K495 Standard message module not found in any MESSAGE control statement or not defined by MAX control statement.

K496 File &FNAM corrupted or not a KDCFILE.

K497 KDCDEF &VER1 / &OST1 / &BMD1 cannot read KDCFILE &VER2 / &OST2 / &BMD2

K498 There are no &STMTNAME control statements to be created.

K499 &PARAM30 in &STMTNAME control statement and &PARAM30 do not match.

K500 No file(s) generated.

K501 Illegal character "&CHAR1" in string: &STRING64

K502 Number of free &PARAM10 entries:&PARAM11

K503 The referenced ACCESS-POINT &PARAM8 must include an APPLICATION-ENTITY-QUALIFIER because the referenced OSI-LPAP &PARAM8 includes an APPLICATION-CONTEXT with the ABSTRACT-SYNTAX CCR.

K504 OSI-CON &PARAM8 references same OSI-LPAP but different ACCESS-POINT.

K505 The operand &PARAM32 is mandatory if the referenced APPLICATION-CONTEXT includes the ABSTRACT-SYNTAX CCR.

K506 An APPLICATION-CONTEXT including the ABSTRACT-SYNTAX CCR must not be used because the UTMD control statement does not specify an APPLICATION-PROCESS-TITLE.

K507 Too many &PARAM32

K508 Error occurred during generation of RSA-Keys.

K509 OSI-CON &PARAM8 references same LOCAL-ACCESS-POINT and an OSI-LPAP with identical APPLICATION-ENTITY-QUALIFIER and APPLICATION-PROCESS-TITLE.

K510 Operand &PARAM32 requires at least &PARAM8 parameters.

K511 At most &PARAM11 concurrent stacked services will be possible.

K512 At most &PARAM11 concurrent sign on and/or stacked services will be possible.

K513 KDCDEF will now generate new RSA-Keys. Depending on key length and system configuration this process can take a considerable amount of time.
Please wait ...

K514 Generation of RSA-Keys completed

K515 Warning: For proper operation the application needs RSA-keys. However, no RSA-keys have been generated.

If objects are generated with encryption levels, the application requires RSA keys in order to run correctly. If no RSA keys are available in the application, the application can be operated but with certain restrictions. TACs with encryption levels cannot be called and no connection can be set up to PTERMs or TPOOLs with encryption levels. Three responses are possible.

– The RSA keys can be transferred from an old KDCFILE to the new KDCFILE using KDCUPD before the application is started, or
– the KDCDEF run can be repeated with the GEN-RSA-KEYS=YES operand in the OPTION statement, or
– the application can be started and the required RSA keys can be created and activated by administration (e.g. using WinAdmin/WebAdmin).

K516 Error &DIAG1 for cluster user-file &STRING64

The insert &DIAG1 describes the cause of the error; the insert &STRING64 contains the file name.

The insert &DIAG1 can have the following values:

Value	Meaning
1	OPEN_ERROR Error opening the file.
2	CLOSE_ERROR Error closing the file.
3	FILE_ERROR The file is inconsistent.
4	LOCK_ERROR Error requesting the file lock.
5	UNLOCK_ERROR Error releasing the file lock.
6	USER_NOT_FOUND KCCUFUE sends an incorrect return code on an update to a user entry; the second insert contains the name of the user
7	UNEXPECTED_RETURNCODE An unexpected return code was returned by a called function; the second insert contains the value of the return code
8	KDCDEF_RUNNING The cluster user file is already locked by a KDCDEF run
9	FILE_IS_EMPTY The cluster user file is cataloged but empty

Value	Meaning
10	Error on getLocalHostName call The second insert contains the error code.
11	NO_XCS_GROUP The computer does not belong to an XCS cluster; the "distributed lock" function is not therefore available.

K517 &PARAM30 not supported for &PARAM30

K518 File &FNAM already exists. The configuration run will be aborted.

K521 MAX PRIVILEGED-LTERM=&UTMNAME is not defined as Dialog-LTERM.

K522 Note: No privileged LTERM generated (see MAX PRIVILEGED-LTERM=)

5.4 Messages of the UTM tool KDCPSYSL

K600 Program &PRGNMSG &VERS started.

K602 Program KDCPSYSL terminated normally - SYSLOG file edited

K604 Program KDCPSYSL terminated abnormally - SYSLOG file not edited

K605 &ERROR error

K607 DMS error &DMSE on file &LINK

The DMS errors are described on page 377.

K611 Program error in &PRGNMSG; reason: &TRMA

K621 NLS catalogue &NLSCAT for >&NLSLANG< not available

K622 Setup of NLS for >&NLSLANG< failed

5.5 Messages of the UTM tools KDCMMOD / KDCMTXT

K650 Program &PRGNMSG terminated abnormally.

K651 Program error (&TRMA).

K652 DMS error &DMSE on file &FNAM.

The DMS errors are described on page 377.

K653 Line &LINENR: mandatory operand missing.

K654 Line &LINENR: syntax error.

K655 Line &LINENR: insert &INSMSG is not permitted for message &IDMSG.

K656 Line &LINENR: constant &CONMSG not defined.

K657 EOF reached on SYSDTA. END statement generated.

K658 * * * * * Statement ignored. * * * * *

K659 File &FNAM is not a valid message definition file.

K660 Program &PRGNMSG &VERS started.

K661 Program &PRGNMSG terminated normally.

K662 Line &LINENR: function unit &FUMSG not defined.

K663 Line &LINENR: language &LANGMSG not defined for function unit &FUMSG.

K664 Line &LINENR: message &IDMSG is not permitted for function unit &FUMSG.

K665 Line &LINENR: GEN statement already entered.

K666 Line &LINENR: message destination &DESTMSG is not permitted for message &IDMSG.

K667 Line &LINENR: message destination &DESTMSG is required for message &IDMSG.

K668 Line &LINENR: GEN statement missing.

K669 Line &LINENR: function unit &FUMSG may not be modified.

K670 Source file &FNAM for message module created.

K671 Source file for message module not created.

K672 Line &LINENR: constant &CONMSG already defined.

K673 Wrong version &VERS of message definition file &FNAM.

K681 NLS source file &FNAM created.

K682 NLS source file &FNAM not created.

K686 Line &LINENR: text for message &IDMSG longer than &MSGMAXL.

The text of the message including the insert is longer than 512 characters. The utilities KDCMMOD and KDCMTXT cannot process the text. KDCMMOD does not generate a source for a new message module. KDCMTXT does not amend the message definition file.

K687 Line &LINENR: warning - text for message &IDMSG (SYSLINE) longer than &MSGMAXL.

The text of the message (&IDMSG) with the destination SYSLINE including the insert is longer than 40 characters. The utilities KDCMMOD and KDCMTXT accept the message text. On subsequent output of the message in the system line, UTM outputs only the first 40 characters of the message text.

K688 Line &LINENR: message number > 999 not allowed.

A message number greater than 999 was specified. The utilities KDCMMOD and KDCMTXT reject this. KDCMMOD does not generate a source for a new message module. KDCMTXT does not amend the message definition file.

K690 Message text for message &IDMSG in language &LANGMSG and function unit &FUMSG not defined.

K691 Message definition file &FNAM generated.

K692 Message definition file not generated.

K693 Message definition file &FNAM modified.

K694 Message definition file not modified.

K695 Line &LINENR: insert &INSMSG not defined.

K696 Line &LINENR: FU/constant/insert &CONMSG already defined.

K697 Line &LINENR: illegal value for SOURCE length, default value assumed.

K698 Line &LINENR: range already used for another function unit.

5.6 Messages of the UTM tool KDCDUMP

K700 KDCDUMP &VERS started.

K701 Difference between the number of existing UTM tables and the number of UTM tables which are known by KDCDUMP.
UTM area(s): &UTMA1&UTMA2&UTMA3&UTMA4&UTMA5&UTMA6&UTMA7

K702 Abnormal end

K703 Term application reason &TRMA

K704 Table entry index not valid.

K705 UTM dump &FNAM or part of it read in memory.

K706 This type of preparation is not allowed for table &TABNAM.

K707 No further process is available.

K708 The editor given by the editor shell variable does not exist.

K709 Only one entry of a slot table can be prepared.

K710 Error on command

K711 No UTM dump in memory.

K712 Requested address X'&SADDR' not in present UTM dump or selected domain.

K713 Requested address X'&SADDR' is present in table &TABNAM, but not within one table entry.

K714 Error on link &LNAM: DMS return code &DMSE.

K716 Open error on file &FNAM. DMS return code : &DMSE.

K717 Request memory error

K718 File &FNAM does not contain a UTM dump.

K719 KDCDUMP &VER1 cannot prepare a UTM dump &VER3.

K720 Only the directory of the UTM dump exists.

K721 &UTMA1 is not present in UTM-dump or in selected domain.

K722 &NUM1 bourse cycle with the criterias CREF=&CREF and ANNO=&ANNO1 is found.

K723 Write error on file &FNAM. DMS return code: &DMSE.

K724 Command not allowed at present time.

K725 Table index is too low or too high.

K726 Address is too low or too high.

K727 FSTAT: Error on file &FNAM. DMS return code: &DMSE.

K728 Normal end

K729 Output file &FNAM is written.

K730 Read error on file &FNAM. DMS return code: &DMSE.

K731 The name &DEFTYPE is not a valid UTM type.

K732 UTM dump does not contain UTM type &DEFTYPE.

K733 Table name &TABNAM does not exist.

K734 The UTM dump &FNAM could not be written completely. Last written UTM area is &UTMA1.

This message indicates that the UTM dump creator was unable to write the UTM dump completely. There may have been insufficient space available under the ID. The UTM areas are dumped in the following order:

KAA, SLOT, CACHE, MPGP, XAPTP-GLOBAL, KTA, XAPTP-LOCAL, STACK, ROOT, User File, Journal File 1, Journal File 2, Buffer Segments, GSSB File, Lock File, CFG File, ULS File

In the case of a PENDER dump, only ROOT is dumped.

The individual tables for the UTM areas can be output by means of the KDCDUMP statement HELP TABLE-NAMES (page 72); they appear in the dump order.

The XAPTP sections form a block.

K735 Table &TABNAM does not exist in UTM dump or in selected domain.

K736 Index out of range - index is set to lowest or highest allowed table index.

K737 End-index lower than start-index - end-index is given the value of the start-index.

K740 Abbreviation of table name ambiguous with regard to &TABNAM &TABNAM1 &TABNAM2 &TABNAM3 &TABNAM4 &TABNAM5.

K741 Information not available. Table name: &TABNAM

This message indicates that a UTM area or a table could not be dumped by the UTM dump creator because the area or the table was not fully accessible.

K742 The KDCDUMP version on the hardware &HW1 with &OS1 as operating system cannot read a UTM dump written on the hardware &HW2 with the operating system &OS2.

K743 For the table &TABNAM no displacement is possible.

K744 Offset lower than 0 or higher than the no. of bytes for one table entry.

K745 No symbolic preparation for this table &TABNAM possible.

K746 Error on symbolic preparation.

K747 Command has no result.

K753 The name &FIRES is not a UTM resource of UTM type &DEFTYPE.

K754 No further entry with this UTM type in this dictionary table.

K755 Input error: No further command can be read.

K756 False value for DB operand. Only &DBMAX database(s) are generated.

K757 The version of &UTMA1 (&VER1) is not compatible with the version of KDCDUMP (&VER2).

K758 No summary was written.

K759 Input string is longer than the permitted length (256 char).

K760 No entry with the name &FIRES found.

K761 No further entry with the name &FIRES found.

K770 Error detect while checking compression of file &FNAM. Error code: &ERRC.

K771 Error detect while creating temporary file for &FNAM. Error code: &ERRC.

K772 Error detect while decompressing file &FNAM. Error code: &ERRC.

K773 Requested information can not be found in the selected domain.

K774 Requested domain for command SFIND not read from dump file.

Recommendation: Read domain with `FILE=<dumpfile>,DOMAIN=<domain>` and repeat SFIND command.

K780 No dump file was closed.

K781 Compressed dump file could not be uncompressed.

K782 No hit found.

K783 Value of HITS is neither ALL nor between 0 and 32767.

K784 Size of address greater than 32 bit is forbidden.

5.7 Messages of the UTM tool KDCUPD

The UTM tool KDCUPD outputs both transaction monitor messages and its own messages. With all KDCUPD messages where the text begins with an asterisk (*****), it was not possible to transfer the data.

K800 KDCUPD &VERS started

K801 Please enter parameters

K802 Control statement KDCFILE: parameter &UPDCMD is not specified

K803 Base names of old and new &PARAM17 must be different

K804 Transfer from UTM &UPDVERS to UTM &UPDVERS not supported

K805 Consistency check for filebase &FBASUPD okay

K806 All requested data transferred

K807 Requested data partially transferred

K808 No data to transfer from &FNAM!

K809 Control statement KDCFILE/CHECK: base name &FBASUPD is too long

K810 CHECK and KDCFILE commands cannot be combined

K812 KDCUPD abnormal end

K813 KDCUPD normal end

K814 CLUSTER-Transfer of different versions not supported

K831 KDXUPDS: Error reading kdcfile.

K832 KDXUPDX: Invalid number of arguments.

K833 &PRGUPDX: Shared memory couldn't be created, errno: &ERRNO.

K834 &PRGUPDX: Shared memory couldn't be attached, errno: &ERRNO.

K835 Process &UPDMODUL couldn't be created, errno: &ERRNO.

K836 Process &UPDMODUL couldn't be called, errno: &ERRNO.

K837 KDCUPD: Invalid child process died.

K838 KDCUPD: Child process &UPDMODUL died.

K839 KDCUPD: Parent process died.

K840 &PRGUPDX: Error &ERRNO in semaphore operation &SEMOP.

K841 KDCUPD: Transfer from 32-Bit to 64-Bit architecture.

In addition, information messages with the prefix "KDCUPD:" are also output. Details on the started help processes can be read in these messages.

Example

Information messages on a Unix system if openUTM is installed under the directory "/opt/lib/utm63a00":

```
KDCUPD: full Path of READ process: "/opt/lib/utm63a00/64/../32/ex/kdcrV63A"
KDCUPD: full Path of WRITE process: "/opt/lib/utm63a00/64/ex/kdcwV63A"
```

K851 &UPDTYP data transferred. KCRN = &UKCRN,&PARAM4 = &UKCLA.

K852 &PARAM3 data transferred. KCRN = &UKCRN, &PARAM4 = &UKCLT, KCLA = &UKCLA

K853 Database configuration. number: &DBCOUNT FILEOLD: &DBOLD FILENEW: &DBNEW

K854 Warning: Security level of user &USER increased. Password may be invalid.

Meaning: The complexity level of the password for the USER &USER is higher in the new KDCFILE than in the old KDCFILE, see USER statement in the KDCDEF generation. If the password transferred does not satisfy this condition, the USER can then no longer sign on.

Action: The administrator must then issue a new password.

K855 * &UPDTYP data not transferred. KCRN = &UKCRN, &PARAM4 = &UKCLA, KCRCCC = &RCCC, KCRCDC = &RCDC.

K856 * &PARAM3 data not transferred. KCRN = &UKCRN, &PARAM4 = &UKCLT, KCLA = &UKCLA, KCRCCC = &RCCC, KCRCDC = &RCDC

K857 * Load module &PROG not found. Current version &PVER not transferred.

K858 Current version &PVER of load module &PROG transferred.

5.8 U messages

5.8.1 Messages of the dialog terminal process

U101 Invalid arguments for utmdtp process

U102 Input for utmdtp process may not be redirected

U103 Please wait ...

U104 Please enter data ...

U106 Please enter application name:

U107 Login name:

U108 Password:

U109 Resource bottleneck in UTM system

U110 Terminal already connected to UTM application &APPL

U111 UTM application &APPL not started

U112 UTM application &APPL terminated

U113 UTM application &APPL startup in progress or terminated

U114 Connection rejected by UTM application &APPL

U115 Sign-on with login name rejected by UTM application &APPL

U116 Connection to UTM application &APPL cleared down

U117 No answer received from UTM application &APPL within timeout

U118 Input terminated with "END" key

U119 Input terminated with "DEL" key

U120 utmdtp process terminated with error number &UERRNO

The insert &UERRNO has the following meanings:

&UERRNO	Meaning	Cause / response
1	utmdtp internal error	PR
71	Negative length for line mode output	PR
72	Error in write() for line mode output	PR
80	Environment variables not set correctly: – $LANG $NLSPATH not correct. – $LINES < 25 or $COLUMN < 80	– Correct variables – Set to 25 or 80 – Use em97801 with TERM=97801 for graphical interfaces Other cause: PR
81	Error in n_cpi.	PR
82	Environment variable $LINES < 25 or $COLUMNS < 80	Set to 25 / 80
83	Error in n_chr.	PR
84	Error in n_wir.	PR
85	Error in n_wrt.	PR
86	Error in n_ger.	PR
88	Error in n_csm.	PR
89	Error in n_dpi.	PR
90	Error in n_cls.	PR
91	Error in n_rac.	PR
92	Error in n_mom.	PR
95	Error in f_open Possible causes: Same as for &UERRNO = 80	Same as for &UERRNO=80
96	Error in f_read	PR
99	Error in f_switch	PM

If this message occurs, a core dump is generated for diagnostic purposes with abort(). This dump must accompany any problem report on a data medium.

U121 USER with login name already connected

U123 New password:

U124 Repeat new password:

U125 utmdtp process terminated by kdcrem

5.8.2 Messages of the printer process

U151 utmprint: PTERM &PTRM terminated with error &UERRNO

The insert &UERRNO has the following meaning:

&UERRNO	Meaning	Response
1	Internal utmprint error	PR
50	Connection setup request from utmprint rejected	PR
60	Invalid TIAM protocol	PR
71	Negative length for line mode output	PR
72	Error in line mode output	PR
131	Shell script "utmlp" found neither under $PATH nor under *utmpath*/shsc or no execution authorization	UTM installation error Provide script

If this message occurs, a core dump is generated for diagnostic purposes with abort(). This dump must accompany any problem report on a data medium.

U154 utmprint: PTERM &PTRM not configured as printer group for lpr

U155 utmprint: Invalid arguments for utmprint process

U156 utmprint: PTERM &PTRM error in output errno: &ERRNO

U157 utmprint: PTERM &PTRM error in output, utmlp exit code: &EXITC

5.8.3 Messages of the utmlog process

U171 utmlog: Invalid number of arguments

U172 utmlog process gets terminated. Error occurred while accessing UTM logpipe. CMD = &CMD errno = &ERRNO

U173 utmlog process gets terminated. Error occurred while reading UTM logpipe (record header). errno = &ERRNO, return value = &RETVALUE

U174 utmlog process gets terminated. Error occurred while reading UTM logpipe (record). errno = &ERRNO, return value = &RETVALUE, nominal value = &NBRBYTES

U175 utmlog: wrong identification in UTM logging record (&LOGREC)

U176 utmlog process gets terminated. Error occurred while calling &CMD for file = &FNAM, errno = &ERRNO

U177 utmlog process terminates normally (&FNAM)

5.8.4 General U messages

U181 Program &OBJ1 &VERS started (pid: &PID, &STRTIME)

U182 &OBJ3 &VERS and &OBJ2 &VERS not compatible

U184 &OBJ1 DMS error &DMSE for file &FNAM

The possible error codes which are output in insert `&DMSE` are described on page 377.

U185 kdcdef not allowed during application run

U186 &OBJ1 KCSTRMA called - reason: &TRMA (&STRTIME)

U187 &OBJ1 used applifile: &OBJ3

U188 &OBJ1 UTMPATH not set

U189 &OBJ1 (&PTRM, &PRNM): IPC shortage of &IPCOBJ &IPCREAS

The inserts &IPCOBJ and &IPCREAS have the following meaning:

&IPCOBJ	&IPCREAS	Meaning	Response
NET	NOT ATT	utmnetm/utmnet/utmnets process not yet started	Normal response at start of application
NET	PROC DEAD	UTM net process terminated	See the message from the network process
TSAP	tsapname	UTM network process could not sign on to transport system with the BCAMAPPL or ACCESS-POINT (tsapname)	Normal response at start of application
EXTP EXTP EXTP	WORK USED NET USED EXTP USED	Entire process administration table in use	Check UTM generation regarding semaphores
SEMA	USED	All semaphores in use	Check UTM generation regarding semaphores
LETT	IPC FULL	Data area already too full to take any more data for this connection	Check UTM generation or UTM_IPC_LETTER environment variable. (See openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.)

&IPCOBJ	&IPCREAS	Meaning	Response
LETT	EXTP FULL	Data area for received messages already fully occupied	Check UTM generation or UTM_IPC_LETTER environment variable. (See openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.)
LETT	USED	Data area is occupied	Check UTM generation or UTM_IPC_LETTER environment variable. (See openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”.)
LETT LETT	MAX ILETT MAX OLETT	Maximum data area per connection in use	Check UTM generation of enlarge the data area of the connection in shared memory before the next start using the environment variable UTM_IPC_EXTP_LETTER (see openUTM manual “Using openUTM Applications under Unix Systems and Windows Systems”) (Default: 64 KB, allowed values: multiples of 4KB).
ANNO	USED	All IPC ANNOs in use	PR
ELEM	USED	All IPC ELEMENTs in use	PR

U190 &OBJ1 SHM error (key: &SHMKEY, lth: &SHMLTH): &UERRNO

The &UERRNO insert has the following meaning.

&UERRNO	Meaning	Cause/Response
1	IPC shared memory cannot be set up with the requested size.	User error Response: It may be necessary to tune the system. Refer to the Release Notice for appropriate information.
2	IPC shared memory cannot be set up because it already exists.	User error
3	Error creating IPC shared memory.	System error
4	IPC shared memory with the requested size cannot be loaded into the process address space.	User error
5	Error loading IPC shared memory into the process address space.	System error

U191 Setup of NLS for >&NLSLANG< failed

U192 &OBJ1Systemcall &SCALL failed; rc: &RETVALUE errno: &UERRNO

U193 &OBJ1Internal error '&UABNREAS' causes abnormal termination of application run

5.8.5 Messages of the timer process

U201 utmtimer: Invalid number of arguments

U202 utmtimer: Invalid application name &PARAM10

U203 utmtimer: A utmtimer process already exists for application &APPL

U204 utmtimer: Application &APPL does not exist in &OBJ3

U205 utmtimer: Error &UERRNO during utmtimer run

The insert &UERRNO has the following meaning:

&UERRNO	Meaning	Response
1	Parameter error at sign-on utmtimer linked inconsistently	PR
2	Buffer bottleneck at sign-on	PR
3	Timeout on sign-on to IPC Shared Memory Possible cause: period of start exit - 8 minutes	Check application start period
9	Unexpected return code in sign-on	PR
10	Memory bottleneck on enlarging the timer list	
11	Parameter error when receiving utmtimer linked inconsistently	PR
12	Work process does not exist when receiving - application ended	
19	Unexpected return code when receiving	PR
21	Parameter error when sending, utmtimer linked inconsistently	PR
22	Work process does not exist when sending - application ended	
29	Unexpected return code when sending	PR
31	Parameter error in sign-on utmtimer linked inconsistently	PR
32	Communication shared memory is locked when signing off. May occur in certain situations after error 12 or 22	
40	The signal handler routine for the signal SIGALRM could not be set.	Problem report + documentation

U206 utmtimer: Message with incorrect type received

U207 utmtimer: Reallocation of timer list from &UDIA1 to &UDIA2 elements

5.8.6 Messages of the utmmain process

U221 &OBJ1 UTM application &APPL &VERS terminated (&STRTIME)

U222 &OBJ1 Invalid number of arguments

U223 &OBJ1 UTM application &APPL still running according to internal status, applifile: &OBJ3

An application of this name is already running. This can result from the following:

– The application name selected is not unique throughout the system.
Response: ask the UTM administrator.

– The user-own application ran as a UTM test application. The initial status was not then produced by calling the kdcrem utility.
Response: call the kdcrem utility.

– The application was not terminated before the operating system shut down.
Response: call the kdcrem utility.

– The application was running as a productive application with the TEST option. After the application is terminated normally, the utmmain process is not terminated automatically.
Response: terminate the utmmain process with kill -9.

U224 &OBJ1 KDCROOT terminated

U225 utmmain: Error while creating pipe &FNAM,errno: &ERRNO

U227 &OBJ1 UTM application &APPL terminated by kdcrem

U228 utmmain: Read error on pipe, errno: &ERRNO

U229 utmmain: &OBJ1 process died, pid: &PID, &SIGEXIT(&STRTIME)

U230 utmmain: utmwork process died, pid: &PID utmwork is being restarted (&STRTIME)

U231 utmmain: utmwork process died, pid: &PID
Unexpected exit code: &EXTCODE &SIGEXIT(&STRTIME)

This message is generated when a utmwork process is not terminated under the control of openUTM, for instance if terminated with kill -9.
In the UTM application a work process is lost and is not restarted. The UTM application is terminated abnormally in this case in order to avoid inconsistencies. This is indicated with the message U221.

The list below shows some of the reasons why work processes can terminate without being under the control of openUTM:

– application was started with STXIT = OFF and a signal occurred

– utmwork processes receive signals from a source other than openUTM

– a program unit overwrites signal routines

– the runtime environment of COBOL or C++ causes the process to terminate

– the database terminates the process

Action:
Eliminate the problem which resulted in the termination of the process and restart the UTM application.

U232 &OBJ1 Error &UERRNO during utmmain run

The values of &UERRNO and their meanings are listed in the following table. For most errors, additional information on the associated actions is available at *stderr*.

&UERRNO	Meaning	Action
1	Standalone application: FILEBASE parameter not present in start parameter file.	User error
	UTM cluster application: CLUSTER-FILEBASE parameter not present in start parameter file.	
2	utmmain could not sign on at the shared memory areas of the application.	User error
11	First utmwork process terminated immediately after start.	See utmwork messages
12	utmwork process terminated in undefined state.	See utmwork messages
13	Invalid job received by utmwork.	Problem report + documentation
21	Memory bottleneck while requesting lock management for cluster configuration file (only for UTM cluster applications).	Check memory requirement
22	utmmain could not set a shared lock for the cluster configuration file (only for UTM cluster applications).	User error
23	Memory bottleneck while requesting area for reading in cluster configuration file (only for UTM cluster applications).	Check memory requirement
24	Cluster configuration file has inconsistent content (only for UTM cluster applications).	User error
25	Invalid version in cluster configuration file (only in UTM cluster applications).	User error
26	Invalid operating system type in cluster configuration file (only in UTM cluster applications).	User error

&UERRNO	Meaning	Action
27	Invalid bit mode in cluster configuration file (only in UTM cluster applications).	User error
28	Name of local host could not be evaluated (only for UTM cluster applications).	Problem report + documentation
29	Node entry for local host could not be found in cluster configuration file (only for UTM cluster applications).	User error
30	Memory bottleneck while requesting lock management for KDCFILE (only for UTM cluster applications).	Check memory requirement
31	utmmain could not set a shared lock for the KDCFILE (only for UTM cluster applications).	User error
32	File is not a cluster configuration file	User error
33	The node name specified during node recovery could not be found in the cluster configuration file.	User error

U233 &OBJ1 &UREAS

U234 &OBJ1 fork() error &ERRNO ; process type: &PRTYPE

U235 &OBJ1 Starting error &ERRNO for file &FNAM

U236 &OBJ1 Starting of &FNAM, pid: &PID

U237 &OBJ1 Error while deleting a semaphore, key: &SEMKEY, error: &ERRNO

U238 &OBJ1 Error while attaching to a shared memory, key: &SHMKEY, error: &ERRNO

U239 &OBJ1 Error while deleting a shared memory, key: &SHMKEY, error: &ERRNO

U240 utmmain started for test in dialog

U241 &OBJ1 <FILEBASE> more than 29 chars or max. no. of processes

U242 &OBJ1 &FBTYPE names inconsistent, reason: &UERRNO
&FILEBAS1
&FILEBAS2

The insert `&FBTYPE` has the following meaning:

&FBTYPE	Meaning
FILEBASE	FILEBASE specification in start parameter file
ARGUMENT	FILEBASE specification as argument of utmmain
CLUSTER-FILEBASE	CLUSTER-FILEBASE specification in start parameter file
CONFIG-FILE	FILEBASE specification in cluster configuration file

&FBTYPE	Meaning
NODE-RECOVERY-DIR	Directory for node recovery

The insert &UERRNO has the following meaning

&UERRNO	Meaning
1	File base name specification missing in the start parameter file.
2	It was not possible to switch to the directory specified in the start parameter file.
3	It was not possible to change to the directory specified in the argument in utmmain.
4	It was not possible to change to the original directory.
5	The specification of the filebase name as a utmmain argument does not correspond to the specification in the start parameter file (for standalone applications) or in the cluster configuration file (for UTM cluster applications)
6	The directory in which utmmain was started does not match the filebase directory of the node application for which a node recovery is to be performed.

The inserts `&FILEBAS1` and `&FILEBAS2` have the following meanings:

Insert	Meaning
&FILEBAS1	Specification in start parameter file
&FILEBAS2	Specified as argument

U244 utmmain: Please start &FNAM with arguments:
&VERS &APPL &FILEBASE &FNAM &PID &WID &STIND &WTYP

U245 &OBJ1 System file stdout will be switched from &FNAM to file &FNAM.

U246 &OBJ1 System file stderr will be switched from &FNAM to file &FNAM.

U247 &OBJ1 New system file &FNAM opened. Previous system file was &FNAM.

5.8.7 Messages of the kdcuslog and kdcslog utilities

U251 &PARAM10 Invalid number of arguments

U252 &PARAM10 FILEBASE name &FILEBASE invalid

U253 &PARAM10 Value &GENUSL not valid for number of generations

U254 &PARAM10 Value &ARG2 invalid method of file operation

U255 &PARAM10 Error &DMSE when creating directory &DIRECT

U256 &PARAM10 Directory &DIRECT created

U257 &PARAM10 Error &DMSE when creating FGG files for &FNAM

U258 &PARAM10 FGG files for &FNAM created

U259 &PARAM10 &OBJ1 not allowed during application run

5.8.8 Messages of the kdccsysl utility

U271 kdccsysl: Invalid number of arguments

U272 kdccsysl: File name &FNAM invalid

U273 kdccsysl: Input file = output file not permitted

U274 kdccsysl: Open error &ERRNO on file &FNAM

U275 kdccsysl: Error &ERRNO when creating file &FNAM

U276 kdccsysl: File &FNAM empty

U277 kdccsysl: Read error &ERRNO on file &FNAM

U278 kdccsysl: File &FNAM not a SYSLOG file

U279 kdccsysl: Write error &ERRNO on file &FNAM

U280 kdccsysl: File &FNAM cannot be written

U281 kdccsysl: Normal termination - SYSLOG file converted

5.8.9 Messages of the main network process

U301 &OBJ1 (pid: &PID): Invalid number of arguments

U302 &OBJ1 (pid: &PID, &TNSNAME): &TNSPROP : Error &TNSCODE &TNSCLASS &TNSVALUE

U303 &OBJ1 (pid: &PID, &NETPROC): &TNSPROP does not exist for &TNSNAME

U304 &OBJ1 (pid: &PID, &TNSNAME): &NETFCT call: Error &NETERR

The error causes in U304 relate to the UTM-internal mapping of the CMX transport interface to the socket interface in the utmnets process. The meaning of the NETFTC and NETERR inserts is given in the table on page 390.

U305 &OBJ1 (pid: &PID): CMX application &BCAP already attached

U306 &OBJ1 (pid: &PID, &TNSNAME): Error &UERRNO during process run

The insert &UERRNO has the following meaning:

&UERRNO	Meaning	Response
1	Parameter error during sign-on; network process inconsistently linked	PR
2	Network process started too often	User error
4	Buffer bottleneck at sign-on	PR
9	Unexpected return code during sign-on	PR
11	Parameter error when receiving; network process inconsistently linked	PR
12	No work process when receiving	Normal behavior when an application terminates
13	Invalid message received from work process	PR
14	Concurrent clearing of connections	Normal behavior
19	Unexpected return code when receiving	PR
21	Parameter error when sending; network process inconsistently linked	PR
22	No work process when sending - application terminated	Normal behavior when an application terminates
23	Buffer bottleneck when sending to work process	Normal behavior under high loads
24	UTM buffer locked for sending	PR
25	Connection was cleared down by openUTM	Normal behavior
26	Connection cleardown was identified on sending	Normal behavior

&UERRNO	Meaning	Response
27	Waiting for a resource to be released on sending. Note: This message is output once a second as of the 2nd second.	Normal behavior under high loads
29	Unexpected return code when sending	PR
32	Connection could not be redirected due to CPU bottleneck	Normal behavior under high loads
34	Memory bottleneck on receiving data in the network process	Normal behavior under high loads
35	Memory shortage in the network process while storing an output message because sends are not permitted	Normal behavior under high loads
36	Stored output message not found.	PR
37	Sign-on failed for a local communication end point	User error: generation incorrect or incomplete or port occupied by another application. In the case of the implicit BCAMAPPL which is formed from the application name, this is port 0. The implicit BCAMAPPL can also be explicitly defined with a port; see U320 / U323
38	Invalid send data request	PR
39	Length of the message received from ICMX(L) is greater than 64K	User error: Message sent by partner system too long
41	Invalid sender in connection setup request to openUTM	PR
42	Buffer bottleneck when processing connection setup request from openUTM	PR
43	Internal status incompatible with connection setup request from openUTM	PR
44	It was not possible to determine any IP address for the host name of the recipient of the connection setup request from openUTM.	User error: TNS entry for recipient missing, generation incorrect or incomplete
45	Connection cleardown contention: The partner application has already cleared down a connection over which openUTM wishes to send a further message.	Normal behavior

&UERRNO	Meaning	Response
46	Invalid port number on connection setup request from openUTM	User error: Generation incorrect or incomplete
47	Error getting address of recipient of connection setup request from openUTM with t_getaddr()	PR
48	Error constructing address of recipient of connection setup request from openUTM with t_setaddr()	PR
51	No connection redirection indication received	PR
52	Data received before connection redirection	PR
53	Request received to send data before connection redirection	PR
54	Cleardown of connection received before connection redirection	PR
55	Invalid indication of connection redirection received	PR
56	tref for CMX does not match tref for redin	PR
57	Invalid call of callback function	PR
58	Length conflict on receiving data (> 64K)	User error: Change data length
82	Sender of connection setup request from ICMX(L) could not be determined	User error: generation incorrect or incomplete
83	Recipient of connection setup request from ICMX(L) is not known	PR
84	Buffer bottleneck when determining connection ID during processing of a connection setup request from ICMX(L)	User error: Increase number of semaphores
85	Work process not available when determining the connection ID for a ICMX(L) connection	
86	Buffer bottleneck when processing connection setup request from ICMX(L)	Increase number of semaphores or or maximum number of connections
87	Internal status does not correspond to connection setup confirmation from ICMX(L)	PR
88	Invalid connection setup confirmation from ICMX(L)	PR
89	Invalid connection cleardown from ICMX(L)	PR

&UERRNO	Meaning	Response
100	Error on call of semaphore in IPC receive thread	PR or normal behavior on end of application
101	Error while sending socket message in IPC receive thread	PR
201	t_attach in socket network process. Parameter error: option pointer is NULL	PR
202	t_attach in socket network process. Parameter error: invalid option structure	PR
203	t_attach in socket network process. Initialization of socket environment failed	PR
204	t_attach in socket network process. Not enough freely available sockets	Increase number of socket network processes.
211	t_event in socket network process. Parameter error: option pointer is NULL	PR
212	t_event in socket network process. Parameter error: invalid option structure	PR
213	t_event in socket network process. Parameter error: invalid value for Cmode	PR
214	t_event in socket network process. Parameter error: invalid connection cleardown stored	PR
215	t_event in socket network process. Not enough freely available sockets	Increase number of socket network processes.
216	t_event in socket network process. An unexpected except socket occurred with the select call.	Client cannot send out-of-band data.
221	t_conrq in socket network process. Parameter error: invalid option structure	PR
222	t_conrq in socket network process. Not enough freely available sockets	Increase number of socket network processes.
223	t_conrq in socket network process. bind() call for sender address was unsuccessful.	Check generation and network
224	t_conrq in Socket net process. Invalid recipient address.	Check generation
226	t_conin in socket network process. Parameter error: invalid value for transport reference	PR

&UERRNO	Meaning	Response
227	t_conin in socket network process. Parameter error: a connection cleardown already exists for the specified transport reference	PR
228	t_conin in socket network process. Parameter error: the specified transport reference is no longer valid	PR
231	t_concf in socket network process. Parameter error: invalid value for transport reference	PR
232	t_concf in socket network process. Parameter error: a connection cleardown already exists for the specified transport reference	Normal behavior
235	t_conrs in socket network process. Parameter error: invalid value for transport reference	PR
236	t_conrs in socket network process. Parameter error: invalid option structure	PR
237	t_conrs in socket network process. Parameter error: a connection cleardown already exists for the specified transport reference	Normal behavior
238	t_conrs in socket network process. Parameter error: the specified transport reference is no longer valid	Normal behavior
241	t_datain in socket network process. Parameter error: invalid value for transport reference	PR
242	t_datain in socket network process. Parameter error: there is no data for the specified transport reference	PR
243	t_datain in socket network process. Parameter error: invalid value for data length	PR
244	t_datain in socket network process. Parameter error: a connection cleardown already exists for the specified transport reference	Normal behavior
251	t_datarq in socket network process. Parameter error: invalid value for transport reference	PR

&UERRNO	Meaning	Response
252	t_datarq in socket network process. Parameter error: invalid value for data length	PR
253	t_datarq in socket network process. Parameter error: a connection cleardown already exists for the specified transport reference	Normal behavior
261	t_disin in socket network process. Parameter error: invalid value for transport reference	PR
266	t_disrq in socket network process. Parameter error: invalid value for transport reference	PR
271	t_datastop in socket network process. Parameter error: invalid value for transport reference	PR
272	t_datastop in socket network process. Parameter error: the specified transport reference is no longer valid	PR
276	t_datago in socket network process. Parameter error: invalid value for transport reference	PR
277	t_datago in socket network process. Parameter error: the specified transport reference is no longer valid	PR
281	t_info in socket network process. Parameter error: invalid value for transport reference	PR
282	t_info in socket network process. Parameter error: option pointer is NULL	PR
283	t_info in socket network process. Parameter error: invalid option structure	PR

U307 &OBJ1 (pid: &PID, &TNSNAME): Invalid event &EVENT

U308 &OBJ1 (pid: &PID, &TNSNAME): UTM application &APPL terminated

U309 &OBJ1 (pid: &PID, &TNSNAME): KCSTRMA called - reason: &TRMA (&STRTIME)

U310 &OBJ1 (pid: &PID, &APPL): Invalid fork() call errno: &ERRNO

U311 &OBJ1 (pid: &PID, &APPL): Starting error errno: &ERRNO

U312 &OBJ1 (pid: &PID, &TNSNAME): Resource bottleneck on transmission

U313 &OBJ1 (pid: &PID, &NETPROC): Warning: &TNSNAME not found in TNS

U315 &OBJ1 (pid: &PID): Network message: &TNSNAME &NETREAS

The insert &NETREAS has the following meaning:

&NETREAS	Meaning
1	Connection setup confirmation for a connection already set up by the partner
2	Rejection of the connection setup request for a connection already cleared down by the partner
3	Connection cleardown for a connection already cleared down by the partner
6	Network main process detected that the application was terminated
7	Message received from openUTM for a connection which has already been cleared down
8	openUTM detected that the connection has been cleared down during sign on
11	openUTM rejected a request to establish a connection
12	openUTM received request to clear down connection
13	Partner received request to clear down connection
14	Partner detected that the connection has been cleared down when sending data
15	Partner detected that the connection has been cleared down when receiving data
20	Active connection setup
21	Passive connection setup

U316 &OBJ1 (pid: &PID, &NETPROC): TSEL-FORMAT(&NETNAME) change: &TFOLD to &TFNEW.

&TFOLD corresponds to the value of TSEL-FORMAT (format of the transport selectors), &TFNEW corresponds to the new value TSEL-FORMAT.

The inserts have the following meanings:

&TFOLD /&TFNEW	Meaning
?	Undefined TSEL-FORMAT (not for &TFNEW)
T	TRANSDATA format
E	EBCDIC character format
A	ASCII character format

The KDCDEF generation should be checked. Recommendation: Set the TSEL-FORMAT in the corresponding KDCDEF statement.

U317 &OBJ1 (pid: &PID, &NETPROC): &TNSNAME CMX error(&CMXERR) in &CMXFUNC.

The insert &CMXERR contains the CMX error codes (see the CMX manual).

U318 &OBJ1 (pid: &PID, &NETPROC): Listen-port(&TNSNAME) : &LPOLD not valid.

The insert &LPOLD corresponds to the current values of LISTENER-PORT (listener port number), where only the port numbers 102, 1025 through 32767 are allowed.

The KDCDEF generation should be checked.
Recommendation: set the LISTENER-PORT in the corresponding KDCDEF statement.

U319 &OBJ1 (pid: &PID, &NETPROC): IP address (0.0.0.0) for proname &PRNM not valid.

There is an invalid IP address stored for the partner `&NETPROC`.

The KDCDEF generation or the name service (e.g hosts file in Unix systems or Windows systems) should be checked.
Recommendation: after checking, the IP address should be changed using the dynamic administration or WinAdmin/WebAdmin using the function KC_UPDATE_IPADDR.

U320 &OBJ1 (pid: &PID, &TNSNAME): &SOCKFCT call: Error &ERRNO

The ERRNO insert corresponds to the UERRNO insert of message U306 (see the table for U306).

U321 &OBJ1 (pid: &PID, &TNSNAME): Resource bottleneck

U322 &OBJ1 (pid: &PID, &TNSNAME): No response from partner within wait time

U323 &OBJ1 (pid: &PID): &BCAP port &PORT permission denied

5.8.10 Messages of kdckaa

U341 kdckaa: Invalid number of arguments

5.8.11 Messages of the UTM tool kdcshut

U351 kdcshut: Invalid number of arguments

U352 kdcshut: Invalid application name &PARAM10

U353 kdcshut: A kdcshut process already exists for application &APPL

U354 kdcshut: Application &APPL does not exist in &OBJ3

U355 kdcshut: Error &UERRNO during kdcshut run

&UERRNO	Meaning
21	Parameter error while sending
22	Work process not available while sending because the application has ended
23	Lock error while sending
29	Unexpected return code while sending
31	Parameter error while signing off
32	Communication shared memory was locked while signing off

U356 kdcshut: Specified time &PARAM10 is not numeric

U357 kdcshut: Specified time &PARAM10 is too long

U358 kdcshut: Invalid value &ARG2 for shutdown type

5.8.12 Messages of the UTM tool kdcrem

U361 kdcrem: Invalid number of arguments

U362 kdcrem: Application &APPL not found in &OBJ3

U363 kdcrem: Normal termination

5.8.13 Messages of the UTM tool kdcprog

U370 usage:
kdcprog CREATE <filebase> <number of FGG entries>
INFO <filebase>
TRANSFER <filebase> [<fgg-number>]
SWITCH <filebase> <new base generation>

U371 kdcprog: Invalid command &CMD

U372 kdcprog: Filebase name &FILEBASE invalid.

U373 kdcprog: Value &GENUSL for number of generations invalid

U374 kdcprog: Filebase directory &FILEBASE not accessible

U375 kdcprog: Error &DMSE during creation of the FGG files for &FNAM

U376 kdcprog: FGG files for &FNAM created

U377 kdcprog: Cannot show info for &FNAM - return code is &DMSE

U378 INFO for FGG &FNAM
FGG maximum number of versions &GENUSL
FGG base &PRV1
FGG first generation &PRV2
FGG last generation &PRV3

U379 File PROG/&PRV1 is PROG(&PRV3) &ARROW

U380 The following program files are available:

U381 kdcprog cannot transfer into &FNAM. Illegal format of FGG number

U382 kdcprog cannot transfer - return code is &DMSE

U383 kdcprog: Transfer : &PRCMD

U384 kdcprog: Transfer failed - return code of cp is &ERRNO

U385 kdcprog: Transfer failed - return code of chmod is &ERRNO

U386 kdcprog: &CMD not allowed because application &APPL is active.

U387 kdcprog cannot SWITCH &FNAM - return code of KCSSWGG is &DMSE

U388 kdcprog: New base of program FGG &FNAM is &GENUSL

U389 kdcprog: TRANSFER successful

U390 kdcprog: SWITCH failed - illegal format of base generation

U391 kdcprog: TRANSFER for KDCAPPL PROG=NEW initiated

U392 kdcprog: File &FILEBASE not accessible

5.9 Error codes during file processing (DMS errors)

If errors occur during file processing and calls to other C runtime routines then error codes are output in the form yxxx in the messages. When they refer to file processing, these are also referred to as DMS errors.

The following inserts are affected:

- &DMSE
- &ERRNO
- &UERRNO

The DMS errors have the following meanings:

y The first character y denotes the function in which the error occurred.
y may have the following values:

A	Error in loading shared memories into the address space
C	Error in close call
D	Error in signing off from a shared memory area
E	Error in remove call
F	Error in fstat/stat call
G	Error in allocating shared memory
L	Error in lseek call
M	Error in mkdir call
O	Error in open call
R	Error in read call
S	Error in system call
W	Error in write call
X	Error in create call

xxx The three characters xxx represent, in printable form, the error number which is stored by the operating system in the external variable 'errno'. If necessary, the error number is padded out to the length of three digits with leading zeros. The meanings of the individual error numbers are described in the manuals for the systems and in the errno.h header file.

For example, the error code O002 means that on an attempt to open a file (O=open), the file was not present (2=errno ENOENT).

In addition, the following errors codes can occur:

CONS	The contents of the file are inconsistent.
GPOS	GPOS means that it was not possible to get the position in the start parameter file stream with *fgetpos()*.
LERR	lseek could not be positioned at the desired point.
OERR	An attempt was made to open a directory as a normal file.
REND	End-of-file reached on reading from a file.
RERR	Insufficient bytes could be read.
WERR	Insufficient bytes could be written.
LOCK	The file cannot be written because it is locked.
MARK	The expected file marks could not be found. The file has probably been destroyed.
USED	The file cannot be written because it is currently being used.
VERS	The expected version number could not be found. The file may have been destroyed.

5.10 Standard message definition file, inserts

5.10.1 Constants of the standard message definition file

Constant name	Constant value	Meaning
NL	0A	NEW LINE
NP	0C	NEW PAGE

NEWPAGE cannot be supported with certain PTYPES. A smudge character appears instead.

5.10.2 Message inserts

The "Length" column contains the output length of the individual inserts in bytes, i.e. on output of the message text the insert occupies as many characters as specified in the "Length" column. The length of the inserts is significant particularly for the creation of message texts using KDCMMOD and KDCMTXT.

The abbreviations in the ’Data type’ column have the following meanings:

Char printable characters

Int numeric field

Hexa hexadecimal information

5.10.2.1 Inserts in K and P messages

Insert name	Data type	Length	Meaning
AAID	Hexa	128	FIRST 64 BYTE OF ATOMIC ACTION IDENTIFIER
AAIS	Int	4	ATOMIC ACTION IDENTIFIER SIZE
ACPNT	Char	8	ACCESS-POINT-NAME
ACTION	Char	6	SYSTEM ACTION
ADTC	Char	8	ADMINISTRATION TAC
AGUS	Char	8	JOB-SUBMITTING USER
AMOD	Char	1	APPLICATION MODE
APPL	Char	8	APPLICATION NAME
ATAC1	Char	8	ASYNCHRONOUS TAC
ATAC2	Char	10	NUMBER OF UNPROCESSED ASYNCHRONOUS TACS
ATTR	Char	11	ATTRIBUT OF LOAD-MODULE/PROGRAM
ATYP	Char	1	APPLICATION TYPE (STANDALONE/CLUSTER)
BCAP	Char	8	APPLICATION NAME
BCMOPCD	Hexa	8	BCMM-OPCODE
BCMRTCD	Hexa	8	BCMM-RETURNCODE
BMD1	Char	8	BIT MODE OF SYSTEM
BMD2	Char	8	BIT MODE OF SYSTEM
CBRC	Hexa	8	RETURN CODE
CHAIN	Char	3	CHAINED MESSAGE INFORMATION
CID	Char	8	PRINTER CONTROL ID
CLSIGT	Int	2	CLUSTER COMMUNICATION SIGNAL TYPE
CMD	Char	8	COMMAND NAME
CNTR	Char	6	NUMBER OF LPUT RECORDS
CON	Char	8	CONNECTION NAME
COND	Char	3	CONDITION
CONU	Char	10	NUMBER OF CONNECTED USERS
COTM	Int	10	ELAPSED CONNECTION TIME IN SECONDS
CPTM	Int	10	CPU TIME SINCE SIGN-ON IN MILLISECONDS
CPUBEGIN	Hexa	8	CPU TIME AT TAC START IN MILLISECONDS
CPUCLNT	Hexa	8	CPU TIME USED OF THIS CLIENT
CPUEND	Hexa	8	CPU TIME AT TAC END IN MILLISECONDS
CPUREAS	Char	1	INTERNAL REASON

Insert name	Data type	Length	Meaning
CPUTEXT	Char	8	TEXT: OVERFLOW OR NEGATIV
CPUUSED	Hexa	8	CPU TIME USED OF THIS TAC IN MILLISECONDS
CTYP	Char	4	TYPE OF PROGRAM EXCHANGE
DBCALL	Char	12	FUNCTION-CALL OF IUTMDB INTERFACE
DBCON	Char	8	DATABASE CONNECTION MODULE
DBV1	Char	8	VERSION OF DB CONNECTION MODULE
DBV2	Char	8	VERSION OF KDCDB MACRO
DEFVER	Char	5	VERSION NUMBER IN KAA
DEST	Char	8	DESTINATION OF ASYNCHRONOUS MSG
DEVC	Hexa	2	DEVICE TYPE
DIA1	Int	11	DIAGNOSTIC INFORMATION
DIA2	Int	11	DIAGNOSTIC INFORMATION
DIA3	Int	11	DIAGNOSTIC INFORMATION
DIA5	Char	80	INTERNAL DIAGNOSTIC INFORMATION
DLDATE	Char	3	DAY OF KDCS CALL PADM DL/DA
DLTIME	Char	8	TIME OF KDCS CALL PADM DL /DA
DMSE	Char	4	DMS ERROR CODE
DPID	Char	8	ASYNCHRONOUS MESSAGE ID
DTM2	Char	18	TIME STAMP 2
DTTM	Char	18	TIME STAMP
EBSR	Char	4	ACTUAL BRACKET STATE
EBSS	Char	4	SAVED BRACKET STATE
ENCPW	Hexa	16	ENCRYPTED PASSWORD
ERCD1	Hexa	4	ERROR CODE (IUTMHLL)
ERCD2	Hexa	4	INFO RETURN CODE (IUTMHLL)
ERCD3	Char	4	ERROR CODE (IUTMDB)
ERCD4	Char	4	ERROR CODE (IUTMFORM)
ERCD5	Char	4	INFO RETURN CODE (IUTMFORM)
ERCD6	Char	4	ERROR CODE (ROOT)
ERPRT	Char	1	PRINT ERROR CODE
ERRCODE	Char	16	RETURN CODE OF FAULTY FUNCTION
ERRNAME	Char	8	NAME OF FAULTY FUNCTION
ESQR	Hexa	10	ACTUAL REQUEST SEQUENCE NUMBER

Insert name	Data type	Length	Meaning
ESQS	Hexa	8	SAVED SEQUENCE NUMBER
ESRR	Hexa	10	ACTUAL RESPONSE SEQUENCE NUMBER
EXIT	Char	10	CURRENT ACTIVE EXIT
FBASUPD	Char	42	FILE BASE NAME KDCUPD
FIL1A	Hexa	2	APPLICATION STATE
FIL1B	Hexa	2	BCAM REQUEST OR ANNO TYPE / UTM ANNO TYPE
FIL2A	Hexa	2	LTERM STATE
FIL2B	Hexa	8	DIAGNOSTIC WORD
FIL3	Hexa	4	PTERM STATE
FMH7	Char	80	ERROR RECOVERY PROCEDURE MESSAGE
FNAM	Char	54	FILE NAME
FNKT	Char	6	FUNCTION
FNOD	Char	1	FIRST NODE (Y/N) IN CLUSTER APPLICATION
FORM	Char	8	FORMAT NAME (FOR K015 ONLY)
GBLNBR	Int	11	NUMBER OF LOCKED GSSB
GLOBALSG	Char	1	CLUSTER GLOBAL SIGNON/SIGNOFF
GNDATE	Char	3	GENERATION DATE ASYNCHRONOUS MESSAGE
GNTIME	Char	8	GENERATION TIME ASYNCHRONOUS MESSAGE
GNUSER	Char	8	USER NAME OF ASYNCHRON. MESSAGE GENERATION
GTRID	Hexa	128	FIRST 64 BYTE OF GLOBAL TRANSACTION ID
HITR	Char	3	CACHE HIT RATE
HST1	Char	8	HOST NAME
HST2	Char	8	HOST NAME
HST3	Char	8	HOST NAME
HSTACK	Int	2	HEIGHT OF STACK
IDEFRC	Hexa	16	RETURNCODE OF INVERSE KDCDEF
IDX1	Char	4	CLUSTER NODE INDEX
IDX2	Char	4	CLUSTER NODE INDEX
IDX3	Char	4	CLUSTER NODE INDEX
IMPVER	Char	5	VERSION NUMBER IN KAA OF KDCFILE TO IMPORT
IMSG1	Char	10	NUMBER OF TERMINAL INPUT MESSAGES
IMSG2	Hexa	64	FIRST PART OF INPUT MESSAGE
INF1	Char	65	ADDITIONAL INFORMATION

Insert name	Data type	Length	Meaning
INF2	Char	65	ADDITIONAL INFORMATION
INSTNUM	Int	2	RM-INSTANCE NUMBER
INTTAID	Hexa	130	INTERNAL TRANSACTION ID
IOMS	Int	11	DURATION OF IO IN MILLISECONDS
IOPG	Int	11	NUMBER PAGES OF IO
IPADDR	Char	39	IPV4 (123.456.789.012) OR IPV6 (1234:5678:9ABC:DEF0:1234:5678:9ABC:DEF0) ADDRESS
LPAP	Char	8	LPAP NAME
LSES	Char	8	LSES NAME
LTAC	Char	8	TAC OR LTAC
LTACINDX	Hexa	8	LTAC INDEX
LTHGTRID	Int	2	LENGTH OF GLOBAL TRANSACTION ID
LTRM	Char	8	LTERM NAME
LWRT	Char	5	NUMBER OF USLOG FILE WRITES
MOD	Char	7	MODULE NAME
MSG	Char	80	MESSAGE TEXT
MSG2	Char	100	MESSAGE TEXT
MSTACK	Int	2	MAXIMUM STACK HEIGHT
MTYPE	Char	4	MESSAGE TYPE
MXLT	Char	8	MUX LTERM
MXP1	Char	4	MUX PROTOCOL VERSION (LOWER BOUNDARY)
MXP2	Char	4	MUX PROTOCOL VERSION (UPPER BOUNDARY)
MXPR	Char	8	MUX PROCESSOR
MXPT	Char	8	MUX PTERM
NCVST	Char	1	NEW CONVERSATION STATE
NMSG	Int	11	NUMBER OF MESSAGES
NNM1	Char	8	CLUSTER NODE NAME
NNM2	Char	8	CLUSTER NODE NAME
NNM3	Char	8	CLUSTER NODE NAME
NTAST	Char	1	NEW TRANSACTION STATE
NUMDAYS	Char	2	NUMBER DAYS PASSWORD VALID
NUMMSGS	Int	11	NUMBER OF WAITING OUTPUT MESSAGES
OBJ1	Char	10	OBJECT NAME

Insert name	Data type	Length	Meaning
OBJ2	Char	10	OBJECT NAME
OBJ3	Char	54	OBJECT OR FILENAME
OCVST	Char	1	OLD CONVERSATION STATE
OMSG1	Char	10	NUMBER OF TERMINAL OUTPUT MESSAGES
OMSG2	Char	74	BROADCAST MESSAGE
OPCD1	Char	4	OPCODE
OPCD2	Char	35	OPCODE (IUTMHLL)
OPCD3	Char	5	OPCODE (IUTMFORM)
OSLPAP	Char	8	OSI-LPAP NAME
OST1	Char	24	TYPE OF OPERATING SYSTEM
OST2	Char	24	TYPE OF OPERATING SYSTEM
OTAST	Char	1	OLD TRANSACTION STATE
PALTRM	Char	8	LTERM NAME PRINT ADMIN STATION
PAS1	Char	20	SPACE FOR PASSWORD
PAS2	Char	20	SPACE FOR PASSWORD
PAS3	Char	20	SPACE FOR PASSWORD
PGPOOL	Char	16	(NODE/CLUSTER) PAGEPOOL
PGS1	Int	11	NUMBER OF UTM PAGES
PGS2	Int	11	NUMBER OF UTM PAGES
PHAXAPTP	Char	14	INIT or START/RESTART of XAP-TP
PID	Int	11	UNIX/NT PROCESS ID
PRCN	Char	200	PROCEDURE/SCRIPT/COMMAND-FILE NAME
PRGVERS	Int	11	PROGRAM VERSION IN CASE OF PROGRAM EXCHANGE
PRNM	Char	8	PROCESSOR NAME
PROG	Char	32	PROGRAM OR LOAD MODULE NAME
PSQN	Hexa	8	SAVED PET SEQUENCE NUMBER
PTCID	Char	27	PTC IDENTIFICATION
PTRM	Char	8	PTERM NAME
PVER	Char	24	PROGRAM VERSION
RBCAUSER	Char	14	CAUSER OF ROLLBACK
RCCC	Char	3	KCRCCC
RCCC2	Char	4	STARTUP ERROR CODE
RCDC	Char	4	KCRCDC

Insert name	Data type	Length	Meaning
RCF1A	Char	4	KCRCDC
RCF1B	Char	3	RETURN CODE 1
RCF1C	Char	4	RETURN CODE 1
RCF2A	Char	4	INTERNAL RETURN CODE
RCF2B	Char	4	RETURN CODE 2
RCHX	Hexa	8	RETURNCODE IN HEX-FORM
RCVDANNO	Hexa	8	FIRST 4 BYTES OF RECEIVED ANNO
RCXAPTP	Int	3	RETURNCODE XAP-TP STARTFUNCTIONS
REA1	Hexa	2	REASON
REA2	Char	2	REASON
REA3	Char	136	ERROR MESSAGE
REA4	Char	1	DIAGNOSTIC INFORMATION
REA6	Hexa	2	DIAGNOSTIC INFORMATION (DISCONNECT USER REASON)
REA7	Int	2	DIAGNOSTIC INFORMATION (REJECT USER REASON)
REST	Char	1	RESTART INDICATOR OF LTERM
RMSTAT	Char	8	CONNECTION STATUS OF AN RM
RQM	Int	11	REQUESTED NUMBER OF BYTES
RSES	Char	8	RSES NAME
RSLT	Char	1	RESULT
RSPTC	Char	1	RESET-PTC (Y/N) FOR NODE RECOVERY
RTAANZ	Int	2	NUMBER OF RECOVERED TRANSACTIONS
RTCD	Hexa	8	RETURN CODE
SATRC	Hexa	8	SAT RETURNCODE
SESSCNTR	Int	3	SESSION COUNTER OF ACTUAL SERVICE
SRFG	Hexa	8	SAVED SESSION STATE
STA2	Char	1	STATE
STATE	Char	1	STATE
STDHEAD	Hexa	16	BS2000 STANDARDHEADER
STMT	Char	11	STATEMENT OF KDCDEF
STRTPAR	Char	8	STARTPAR LTH = 8
STS1	Hexa	4	STSN-REQ SEQUENCE NUMBER RCV-CNT
STS2	Hexa	4	STSN-REQ SEQUENCE NUMBER SEND-CNT

Insert name	Data type	Length	Meaning
STS3	Hexa	4	STSN-RSP SEQUENCE NUMBER SLU-PLU
STS4	Hexa	4	STSN-RSP SEQUENCE NUMBER PLU-SLU
STSK	Char	1	SYSTEM TASK (Y/N)
SUFF	Char	5	FILE SUFFIX
SWNR	Int	11	NUMBER OF JOURNAL SWITCHES
SYN	Char	50	SYNTAX ERROR
SYSD	Hexa	4	SYSTEM SENSE DATA
SYST	Char	4	SYSTEM
TAC	Char	8	TRANSACTION CODE
TACINDX	Hexa	8	TAC INDEX
TACNTR	Int	5	TA CNTR OF ACTUAL SERVICE
TACTYPE	Char	1	TAC TYPE
TASK	Char	4	PID (MAX. 4 DIGITS) OF UTM PROCESS
TCPCL	Char	18	SOCKET FUNCTION
TCPMS	Int	11	DURATION OF SOCKET FUNCTION IN MILLISECONDS
TCPRC	Hexa	8	SOCKET DIAGNOSTIC WORD
TCVG	Char	8	CONVERSATION TAC
TERM	Char	1	TERMINATION TYPE
TEXT32	Char	32	STANDARD-TEXTPUFFER
TPRIO	Int	3	EXTERNAL TASK-PRIORITY
TRMA	Char	6	TERM APPLICATION REASON
TSNPID	Char	10	TSN (BS2000) / FULL RECESSION PID (UNIX/WIN)
UERCODE	Char	8	ERROR CODE
UERINFO	Char	8	ERROR INFORMATION
UKCHSTA	Int	5	HEIGHT OF STACK
UKCLM	Int	10	LENGTH OF KCLM USED BY KDCUPD
UKCOP	Char	4	OPCODE OF KDCS CALL USED BY KDCUPD
UKCRN	Char	8	REFERENCE NAME USED BY KDCUPD
ULLNBR	Int	11	NUMBER OF LOCKED ULS
UPCPROT	Hexa	8	UPIC PROTOCOLL
UPCREAS	Hexa	2	UPIC ERROR REASON
UPCSTAT	Hexa	4	USRTNSR UPIC STATE
UPDERR	Char	5	UPD ERROR CODE

Insert name	Data type	Length	Meaning
UPDMODUL	Char	8	UPD MODULE READxxxx/WRITxxxx
UPDPRO	Int	3	PERCENT USED PAGES IN NEW FILE
UPDTYP	Char	6	TYPE OF KCRN FOR UPDATE (LTERM, TAC, LPAP)
UPPENC2	Hexa	4	UPIC ENCRYPTION PTRMDYN INFO
UPVENC1	Hexa	4	UPIC ENCRYPTION VGTDYN INFO
USER	Char	8	USER/LSES/OSI-ASS NAME
USRTYPE	Char	10	USER-TYP: CLIENT/CONNECTION
USSD	Hexa	4	USER SENSE DATA
USTYPPTC	Char	1	TYPE OF USER IN PTC
UTMDEVT	Char	7	UTM-D EVENT
VER1	Char	6	VERSION NUMBER
VER2	Char	6	VERSION NUMBER
VERS	Char	8	UTM VERSION
VGCNTR	Int	11	SERVICE COUNTER OF ACTUAL SERVICE
VTRC	Hexa	8	RETURN CODE
WLEV	Char	1	WARN LEVEL OF PAGE POOL
WTBF	Char	3	CACHE WAITS FOR BUFFER
XACALL	Char	12	FUNCTION-CALL OF XA-CAE INTERFACE
XADBC1	Char	8	TEXT FOR DB-XA-CALLS
XADBC2	Char	8	TEXT FOR DB-XA-CALLS
XAFLAG	Char	8	FLAGS FOR XA-CALLS
XASPEC	Char	12	VERSION OF XA-SPECIFICATION
XATXT	Char	16	READABLE XA-RETURNCODE
XP0OBID	Int	11	OSI-TP OBJECT IDENTIFIER 0
XP1BOOL	Char	5	OSI-TP CCR V2 NOT AVAILABLE
XP1DIA	Int	11	OSI-TP DIAGNOSTIC INFORMATION 1
XP1INFO	Int	11	OSI-TP ADDITIONAL INFORMATION 1
XP1OBID	Int	11	OSI-TP OBJECT IDENTIFIER 1
XP2BOOL	Char	5	OSI-TP PROTOCOL VERSION INCOMPATIBILITY
XP2DIA	Int	11	OSI-TP DIAGNOSTIC INFORMATION 2
XP2INFO	Int	11	OSI-TP ADDITIONAL INFORMATION 2
XP2OBID	Int	11	OSI-TP OBJECT IDENTIFIER 2
XP3BOOL	Char	5	OSI-TP CONTENTION WINNER ASSIGNMENT REJECTED

Insert name	Data type	Length	Meaning
XP3DIA	Int	11	OSI-TP DIAGNOSTIC INFORMATION 3
XP3INFO	Char	40	OSI-TP ADDITIONAL INFORMATION 3
XP3OBID	Int	11	OSI-TP OBJECT IDENTIFIER 3
XP4BOOL	Char	5	OSI-TP BID MANDATORY REJECTED
XP4OBID	Int	11	OSI-TP OBJECT IDENTIFIER 4
XP5BOOL	Char	5	OSI-TP NO REASON GIVEN
XP5OBID	Int	11	OSI-TP OBJECT IDENTIFIER 5
XP6OBID	Int	11	OSI-TP OBJECT IDENTIFIER 6
XP7OBID	Int	11	OSI-TP OBJECT IDENTIFIER 7
XP8OBID	Int	11	OSI-TP OBJECT IDENTIFIER 8
XP9OBID	Int	11	OSI-TP OBJECT IDENTIFIER 9
XPAPDU	Char	20	OSI-TP APDU TYPE
XPASST	Char	20	ASSOCIATION STATE
XPCCLS	Int	11	CMX ERROR CLASS
XPCORR	Int	11	MESSAGE CORRELATOR NUMBER
XPCPSEL	Char	16	OSI-TP P-SEL OF PARTNER (CHAR)
XPCRES	Int	4	OSI-TP NEGATIVE CONFIRMATION RESULT
XPCSSEL	Char	16	OSI-TP S-SEL OF PARTNER (CHAR)
XPCTYPE	Int	11	CMX ERROR TYPE
XPCVAL	Int	11	CMX ERROR VALUE
XPERR	Int	11	OSI-TP ERROR CODE
XPEVT	Char	10	XAPTP EVENT
XPFSMN	Char	10	OSI-TP FSM NAME
XPFUNC	Char	20	CALLED OSI-TP FUNCTION
XPHPSEL	Hexa	32	OSI-TP P-SEL OF PARTNER (HEX)
XPHSSEL	Hexa	32	OSI-TP S-SEL OF PARTNER (HEX)
XPINI	Int	11	OSI-TP INITIATOR
XPLNK	Int	11	OSI-TP LINK
XPLPSEL	Int	2	OSI-TP LENGTH P-SEL OF PARTNER
XPLSSEL	Int	2	OSI-TP LENGTH S-SEL OF PARTNER
XPLTH	Int	11	OSI-TP INVALID LENGTH
XPNDIA	Int	4	OSI-TP NEGATIVE DIAGNOSTICS
XPNSEL	Char	8	OSI-TP N-SEL OF PARTNER

Insert name	Data type	Length	Meaning
XPOSAS	Int	8	OSI-TP ASSOCIATION REFERENCE
XPPDU	Int	11	OSI-TP PDU TYPE
XPPTYP	Int	11	OSI-TP PRIMITIVE TYPE
XPRET	Int	11	OSI-TP RETURN CODE
XPRJCT	Int	4	OSI-TP ASSOCIATION REASON FOR REJECT
XPSRC	Int	4	OSI-TP RESULT SOURCE FROM PARTNER
XPTRFAIL	Int	11	OSI-TP WRITE TRACE FAILURE REASON
XPTSEL	Char	8	OSI-TP T-SEL OF PARTNER

5.10.2.2 Inserts in U messages

Insert name	Data type	Length	Meaning
APPL	Char	8	APPLICATION NAME
ARG2	Char	2	ARGUMENT
ARROW	Char	2	POINTS TO CURRENT ENTRY IN FGG
BCAP	Char	8	APPLICATION NAME
CMD	Char	8	COMMAND NAME
CMXERR	Int	4	CMX ERROR
CMXFUNC	Char	20	CMX FUNCTION
DIRECT	Char	29	DIRECTORY NAME
DMSE	Char	4	DMS ERROR CODE
ERRNO	Int	4	UNIX/NT ERROR NUMBER
EVENT	Int	4	CMX OR NEABX EVENT
EXITC	Int	4	EXIT CODE OF SCRIPT UTMLP
EXTCODE	Int	4	EXIT CODE OF TERMINATED PROCESS
FBTYPE	Char	20	FILEBASENAME
FERRNO	Int	4	FORMATSYSTEM ERROR NUMBER
FILEBAS1	Char	80	FILEBASENAME
FILEBAS2	Char	80	FILEBASENAME
FILEBASE	Char	40	NAME OF FILEBASE
FMTN	Char	8	FORMAT NAME
FNAM	Char	54	FILE NAME
GENUSL	Int	11	NUMBER OF USLOG GENERATIONS
IPCOBJ	Char	4	IPC OBJECT TYPE
IPCREAS	Char	10	IPC ERROR REASON
LOGREC	Hexa	12	UNIX/NT UTM LOGGING RECORD
LPOLD	Int	5	OLD LISTEN(ER)-PORT
NBRBYTES	Int	11	UNIX/NT NUMBER BYTES
NETERR	Hexa	8	ERROR VALUE RETURNED BY T_ERROR()
NETFCT	Char	10	CMX FUNCTION NAME
NETNAME	Char	17	SHORT NET CONNECTION NAME
NETPROC	Char	17	NET PROCESS NAME
NETREAS	Int	4	NET MESSAGE REASON
NLSLANG	Char	14	NLS LANGUAGE VARIABLE

Insert name	Data type	Length	Meaning
OBJ1	Char	10	OBJECT NAME
OBJ2	Char	10	OBJECT NAME
OBJ3	Char	54	OBJECT OR FILENAME
PARAM10	Char	10	PARAMETER LTH=10
PID	Int	11	UNIX/NT PROCESS ID
PNAME	Char	8	PARTNER NAME
PORT	Int	5	PORT NUMBER
PRCMD	Char	100	SYSTEM CMD FOR COPYING
PRNM	Char	8	PROCESSOR NAME
PRTYPE	Char	8	PHYSICAL PRINTER TYPE OR PROCESS TYPE
PRV1	Char	4	INFO ABOUT FGG
PRV2	Char	4	INFO ABOUT FGG
PRV3	Char	5	INFO ABOUT FGG
PTRM	Char	8	PTERM NAME
RETVALUE	Int	11	UNIX/NT RETURNED VALUE BY SYSTEM CALLS
RNAME	Char	8	REFERENCE NAME
SCALL	Char	40	SYSTEM CALL
SEMKEY	Int	11	SEMAPHOR KEY
SHMKEY	Int	11	SHARED MEMORY KEY
SHMLTH	Int	11	SIZE OF SHM
SIGEXIT	Char	10	EXIT CODE OF TERMINATED PROCESS
SOCKFCT	Char	10	SOCKET FUNCTION NAME
STIND	Char	6	UTMWORK START INDICATOR
STRTIME	Char	30	DATE AND TIME AS CHAR ARRAY
TFNEW	Char	1	TRANSPORT SELECTOR NEW
TFOLD	Char	1	TRANSPORT SELECTOR OLD
TNSCLASS	Int	4	ERRCLASS IN TNS STANDARD HEADER
TNSCODE	Int	5	RETCODE IN TNS STANDARD HEADER
TNSNAME	Char	83	NET CONNECTION NAME
TNSPROP	Char	8	TNS PROPERTY NAME
TNSVALUE	Int	4	ERRVALUE IN TNS STANDARD HEADER
TPROT	Char	8	TRANSPORT PROTOCOL TYPE
TPROT1	Char	8	TRANSPORT PROTOCOL TYPE

Insert name	Data type	Length	Meaning
TRMA	Char	6	TERM APPLICATION REASON
TSEL	Char	1	TRANSPORT SELECTOR FORMAT
UABNREAS	Char	20	ABNORMAL TERMINATION REASON
UDIA1	Int	11	DIAGNOSTIC INFORMATION
UDIA2	Int	11	DIAGNOSTIC INFORMATION
UERRNO	Int	4	UTM ERROR NUMBER
UREAS	Char	200	DETAILED ERROR REASON
VERS	Char	8	UTM VERSION
WID	Hexa	8	WORK PROCESS ID
WTYP	Char	10	WORK APPLICATION TYPE

5.11 Destinations of UTM messages

The table below shows which options are available for each message. It contains only those messages which the user can modify.

For the individual messages, the meanings of the entries in the columns of the table are as follows:

R (Required) The destination is assigned permanently to the message; it cannot be modified.

\+ The destination is permitted for this destination.

D (Default) The destination is specified for this message; it may be canceled.

\- The destination cannot be defined for this message.

in the COMPRESS column:

Y (Yes) Superfluous blanks are removed from the message.

N (No) Superfluous blanks are left in the message.

To make diagnosing errors that occur when a UTM application or follow-up process is started easier, all K messages from openUTM that appear in the start phase are output to *stderr* and *stdout*, regardless of which message destination was specified for these messages.

Under Windows systems, messages with the message destination CONSOLE are written to the CONSOLE.TXT file in the *filebase* directory.

The function unit can be modified for the user.

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K001[1]	PTRM , PRNM , BCAP , LTRM , APPL	R	-	+	+	+	+	+	+	-	Y
K002[1]	PTRM , PRNM , BCAP , LTRM , APPL	R	-	-	+	+	+	+	+	+	Y
K003	PTRM , PRNM , BCAP , LTRM , CMD	R	D	-	+	+	+	+	+	+	N
K004[1]	PTRM , PRNM , BCAP , LTRM , USER , REA7	R	-	-	+	+	+	+	+	+	N
K005[1]	PTRM , PRNM , BCAP , LTRM , USER	R	-	-	+	+	+	+	+	+	N
K006[1]	PTRM , PRNM , BCAP , LTRM , USER	R	-	-	+	+	+	+	+	+	N
K007[1]	PTRM , PRNM , BCAP , LTRM , USER , REA7	R	-	-	+	+	+	+	+	+	N
K008[1]	PTRM , PRNM , BCAP , LTRM , USER	R	-	-	+	+	+	+	+	-	Y
K009	PTRM , PRNM , BCAP , LTRM , USER , TAC , RCDC	R	D	D	+	+	+	+	+	+	N
K010	PTRM , PRNM , BCAP , LTRM , USER , TAC	R	D	D	+	+	+	+	+	+	N
K011	PTRM , PRNM , BCAP , LTRM , USER , ATAC1	R	+	+	+	+	+	+	+	+	N
K012	NUMMSGS	R	R	-	-	-	-	-	-	-	N
K013	PTRM , PRNM , BCAP , LTRM , CMD	R	R	-	+	+	+	+	+	+	N
K016	PTRM , PRNM , BCAP , LTRM , USER	R	D	+	+	+	+	+	+	+	N
K017	PTRM , PRNM , BCAP , LTRM , USER , TCVG , RCCC , RCDC , TAC	R	-	D	D	+	+	+	+	+	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K018	PTRM , PRNM , BCAP , LTRM , APPL	R	-	-	+	+	+	+	+	+	N
K019	PTRM , PRNM , BCAP , LTRM , APPL	R	R	-	+	+	+	+	+	+	N
K020	PTRM , PRNM , BCAP , LTRM , USER	R	R	-	+	+	+	+	+	+	N
K021	PTRM , PRNM , BCAP , LTRM	-	-	D	+	+	+	+	+	+	N
K022	PTRM , PRNM , BCAP , LTRM	R	-	+	+	+	+	+	+	+	Y
K023	OMSG2	-	-	+	-	-	-	-	-	-	N
K024	PTRM , PRNM , BCAP , LTRM , USER	R	R	D	+	+	+	+	+	+	N
K025	PTRM , PRNM , BCAP , LTRM	R	-	D	D	+	+	+	+	+	Y
K026	PTRM , PRNM , BCAP , LTRM , USER	R	+	+	+	+	+	+	+	+	N
K027	PTRM , PRNM , BCAP , LTRM	R	+	D	+	+	+	+	+	+	N
K028	PTRM , PRNM , BCAP , LTRM , USER , PAS1	R	-	-	+	+	+	+	+	-	N
K029	PTRM , PRNM , BCAP , LTRM , USER	R	-	-	+	+	+	+	+	-	N
K030[1]	PTRM , PRNM , BCAP , LTRM , USER	R	-	-	+	+	+	+	+	+	N
K031[1]	PTRM , PRNM , BCAP , LTRM , USER	R	-	-	+	+	+	+	+	+	N
K032	CON , PRNM , BCAP , LPAP , USER , RCF1B , RCF2B	-	-	-	D	+	D	+	+	+	N
K033	PTRM , PRNM , BCAP , LTRM , USER , REST , GLOBALSG	-	-	-	D	+	+	+	+	+	N

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K034		R	R	+	-	-	-	-	-	-	N
K035		R	R	+	-	-	-	-	-	-	N
K036	PTRM , PRNM , BCAP , LTRM , RSLT , REA1	-	-	-	D	+	+	+	+	+	N
K038	SYN	-	-	-	-	-	R	+	-	-	Y
K039	STRTPAR	-	-	-	-	-	R	+	-	-	Y
K040	WLEV , PGPOOL	-	-	-	D	+	R	+	+	+	Y
K041	WLEV , PGPOOL	-	-	-	D	+	R	+	+	+	Y
K042		-	-	-	-	-	R	+	-	-	Y
K043	DMSE , FNAM	-	-	-	D	+	R	D	+	+	Y
K044		-	-	-	-	-	R	+	-	-	Y
K045	PTRM , PRNM , BCAP , LTRM , PALTRM , CID	-	-	-	-	+	-	-	-	+	N
K046	PTRM , PRNM , BCAP , LTRM , PALTRM , CID , DPID , ERPRT , IMSG2	-	-	-	+	+	+	+	+	+	N
K047	OPCD1 , RTCD	-	-	-	-	-	R	+	-	-	Y
K049	RCCC2	-	-	-	-	-	D	D	-	-	Y
K050	APPL , VERS , AMOD , TERM , OST1 , BMD1 , ATYP , FNOD	-	-	-	D	+	R	+	+	+	Y
K051	APPL , VERS , AMOD , TERM , OST1 , BMD1 , ATYP , FNOD	-	-	-	D	+	R	+	+	+	Y
K052	TASK , APPL , PID , PRGVERS , STSK	-	-	-	D	+	R	+	+	+	Y
K053	CNTR	-	-	-	D	+	R	+	+	+	Y
K054		-	-	-	D	+	R	+	+	+	Y
K055	ATAC1 , RCCC , RCDC , USER , LTRM	-	-	-	D	+	R	+	+	+	Y
K056	TASK , PID , RSLT	-	-	-	D	+	D	+	+	+	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K057		-	-	-	D	-	D	+	+	-	Y
K058	TASK , PID	-	-	-	+	+	D	+	+	+	Y
K059		-	-	-	+	-	D	+	+	-	Y
K060	TRMA	-	-	-	D	-	R	D	+	-	Y
K061	FNAM	-	-	-	D	-	R	D	+	-	Y
K062		-	-	-	D	-	R	D	+	-	Y
K064	PTRM , PRNM , BCAP , LTRM , DEVC , FIL1A , FIL2A , FIL3 , VTRC , IMSG2 , REA1 , CBRC	-	-	-	D	+	+	+	+	+	Y
K065	PTRM , PRNM , BCAP , LTRM , FIL1B , FIL2B	-	-	-	D	+	+	+	+	+	N
K066		-	-	-	-	-	R	+	-	-	Y
K067	MOD , ERCD1 , ERCD2 , OPCD2	-	-	-	-	-	R	+	-	-	Y
K068	DBCON , DBV1 , DBV2	-	-	-	-	-	R	+	-	-	Y
K069	PTRM , PRNM , BCAP , LTRM , COTM , REA4 , REA6	-	-	-	D	+	+	+	+	+	N
K070	PTRM , PRNM , BCAP , LTRM , USER , COTM , CPTM , GLOBALSG	-	-	-	D	+	+	+	+	+	N
K071	OPCD1 , ERCD3	-	-	-	-	-	R	+	-	-	Y
K072	STMT	-	-	-	-	-	D	D	-	-	Y
K073	ATTR , STMT , PROG	-	-	-	-	-	D	D	-	-	Y
K074	CTYP , PROG , PVER	-	-	-	D	+	R	+	+	+	Y
K075	CTYP , PROG , PVER , TASK , PID	-	-	-	D	+	D	+	+	+	Y
K076	RCCC , RCDC , ADTC , USER , LTRM	-	-	-	D	+	R	+	+	-	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K077	PTRM , PRNM , BCAP , LTRM , CLSIGT	-	-	-	D	-	D	D	-	-	Y
K078	ERRNAME , ERRCODE , REA3	-	-	-	-	-	R	D	-	-	Y
K079	REA2	-	-	-	D	+	+	+	+	+	Y
K080		-	-	-	D	+	+	+	+	+	Y
K081	IMSG1 , OMSG1 , CONU , ATAC2 , LWRT , HITR , WTBF	-	-	-	D	+	+	+	+	+	N
K082	FNAM	-	-	-	-	-	R	D	-	-	Y
K083	FNAM	-	-	-	-	-	R	D	-	-	Y
K084	OBJ1 , VER1 , OBJ2 , VER2 , OST1 , BMD1 , OST2 , BMD2	-	-	-	-	-	R	D	-	-	Y
K085	FNKT	-	-	-	-	-	R	D	-	-	Y
K086	PTRM , PRNM , BCAP , LTRM , USER , SYSD , USSD , FMH7 , AGUS	-	-	-	D	+	+	+	+	+	N
K088	LSES , RSES , LPAP , SRFG , PSQN , ESQS , EBSS , ESQR , ESRR , EBSR	-	-	-	D	+	+	+	+	+	N
K089	GNDATE , GNTIME , DEST , GNUSER , USER , DLDATE , DLTIME , CHAIN	-	-	-	D	+	+	+	+	+	N
K090	DEST , USER , DLDATE , DLTIME	-	-	-	D	+	+	+	+	+	N
K091	PTRM , PRNM , BCAP , LTRM , USER , REA7	R	+	+	+	+	+	+	+	+	N
K092	PTRM , PRNM , BCAP , LTRM , USER , PAS1 , PAS2 , PAS3	R	-	-	+	+	+	+	+	-	N

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K093	PTRM , PRNM , BCAP , LTRM , USER , HSTACK , MSTACK	R	R	+	+	+	+	+	+	+	Y
K094	PTRM , PRNM , BCAP , LTRM , USER , RCF1B , REA4	-	-	-	D	+	+	+	+	+	N
K095		D	R	-	-	-	-	-	-	-	N
K096		R	R	-	-	-	-	-	-	-	N
K097[1]	PTRM , PRNM , BCAP , LTRM , USER	R	-	-	+	+	+	+	+	+	Y
K098	PTRM , PRNM , BCAP , LTRM , USER , RCF1C , RCF2B	R	R	-	+	+	+	+	+	+	N
K099	MSG	-	-	-	-	-	R	+	-	-	Y
K101	PTRM , PRNM , BCAP , LTRM , USER , REA1 , DEST	R	R	+	D	+	D	+	+	+	Y
K104	UTMDEVT , LSES , LPAP , AGUS , OCVST , OTAST , ACTION , NCVST , NTAST , SESSCNTR, VGCNTR , TACNTR , RCVDANNO, TACINDX , LTACINDX	-	-	-	D	+	+	+	+	+	N
K105	LSES , LPAP , AGUS , SYST	-	-	-	D	+	+	+	+	+	N
K119	OSLPAP , USER , TAC , DIA1 , DIA2 , DIA3 , SESSCNTR, VGCNTR , TACNTR , AAIS , AAID	-	-	-	D	+	+	+	+	+	Y
K120[1]	PTRM , PRNM , BCAP , LTRM , USER	R	+	-	D	+	+	+	+	+	N

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K121	PTRM , PRNM , BCAP , LTRM , USER , PAS1 , PAS2 , PAS3 , NUMDAYS	R	-	-	+	+	+	+	+	-	N
K122	PTRM , PRNM , BCAP , LTRM , USER , NUMDAYS	+	R	-	D	+	+	+	+	-	Y
K123[1]	LTRM , TAC , USER	R	+	-	D	+	+	+	+	+	N
K124	RCXAPTP , PHAXAPTP	-	-	-	D	-	D	D	+	-	Y
K125[1]	PTRM , PRNM , BCAP , LTRM , USER	R	+	-	D	+	+	+	+	+	Y
K126	SATRC	-	-	-	D	+	D	D	+	+	N
K127	ERCD6	-	-	-	-	-	R	+	-	-	Y
K128	CON , PRNM , BCAP , LPAP , LSES , REA1 , RCDC , TAC	-	-	-	D	+	+	+	+	+	N
K129	CMD	-	-	-	-	-	R	+	-	-	Y
K130	TPRIO , TASK	-	-	-	D	+	+	+	+	+	Y
K132	OBJ1 , OBJ2	-	-	-	-	-	R	+	-	-	Y
K133	EXIT	-	-	-	-	-	R	+	-	-	Y
K134	DEST , NMSG	-	-	-	D	+	+	+	+	+	Y
K135	PTRM , PRNM , BCAP , LTRM , UPCREAS , UPCSTAT , UPCPROT , UPVENC1 , UPPENC2	-	-	-	D	+	+	+	+	+	N
K136	FNAM	-	-	-	-	-	D	+	+	+	Y
K137	FNAM	-	-	-	+	+	D	+	+	+	Y
K138	FNAM	-	-	-	-	+	D	+	+	+	Y
K139	FNAM	-	-	-	+	+	D	+	D	+	Y
K140	PTRM , PRNM , BCAP , LTRM , MXP1 , MXP2	-	-	-	D	+	+	+	+	+	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K141	PTRM , PRNM , BCAP , LTRM , MXP1	-	-	-	D	+	+	+	+	+	Y
K142	PTRM , PRNM , BCAP , LTRM , MXPT , MXPR , MXLT	-	-	-	D	+	+	+	+	+	Y
K143	PTRM , PRNM , BCAP , LTRM , STS1 , STS2 , STS3 , STS4	-	-	-	D	+	+	+	+	+	Y
K145[1]	PTRM , PRNM , BCAP , LTRM , USER	R	-	+	+	+	+	+	+	+	N
K146	BCMOPCD , BCMRTCD , STDHEAD , TASK , BCAP	-	-	-	D	+	+	+	+	+	Y
K147	PTRM , PRNM , BCAP , LTRM , USER , REA7 , USRTYPE	-	-	-	D	+	D	+	+	+	Y
K149	DIA5	-	-	-	-	-	R	+	+	-	Y
K151	IDEFRC , DMSE , FNAM	-	-	-	D	+	D	D	+	+	Y
K152	COND , MTYPE , OSLPAP , USER , LTAC , AAIS , AAID , SESSCNTR, VGCNTR , TACNTR	-	-	-	D	+	+	+	+	+	Y
K154	PTRM , PRNM , BCAP , LTRM , TCPCL , TCPRC	-	-	-	D	+	D	+	+	+	N
K155	PTRM , PRNM , BCAP , LTRM , USER , PAS1 , PAS2 , PAS3	R	-	-	+	+	+	+	+	-	N
K156	RSLT	-	-	-	D	+	+	R	+	+	N
K158	PTRM , PRNM , BCAP , LTRM , USER , CPUTEXT , CPUBEGIN, CPUEND , CPUUSED , CPUCLNT , CPUREAS	-	-	-	D	+	+	+	+	+	Y
K159	USER , ENCPW	-	-	-	+	+	+	+	+	-	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K160	PTRM , PRNM , BCAP , LTRM , USER , TCVG , TAC , TACNTR , RBCAUSER, RCCC , RCDC , TASK , SESSCNTR, VGCNTR , LTHGTRID, GTRID , PID	-	-	-	D	+	+	+	+	+	Y
K161	TASK , PID	-	-	-	+	+	+	+	+	-	N
K162	TASK , IOPG , IOMS , PID	-	-	-	D	-	R	D	+	-	Y
K163	TASK , IOPG , IOMS , PID	-	-	-	D	+	R	D	+	-	Y
K169	ACTION , IDX1 , HST1 , STATE , IDX2 , HST2 , IDX3 , HST3 , NNM1 , NNM2 , NNM3	-	-	-	R	-	R	D	+	-	Y
K170	DTTM , STATE	-	-	-	R	-	R	D	+	-	Y
K174	DIA1	-	-	-	R	-	R	D	+	-	Y
K175	FNAM	-	-	-	R	-	R	D	+	-	Y
K176	PRCN , MSG2 , RCHX	-	-	-	R	+	R	D	+	+	Y
K178	ACTION , STATE , STA2 , PGS1 , PGS2 , SWNR	-	-	-	R	-	R	D	+	-	Y
K179		-	-	-	R	+	D	D	+	+	Y
K180		-	-	-	R	+	D	D	+	+	Y
K181	FNAM	-	-	-	R	+	D	D	+	+	Y
K182	FNAM	-	-	-	R	+	D	D	+	+	Y
K183	FNAM	-	-	-	R	+	D	D	+	+	Y
K184		-	-	-	R	+	D	D	+	+	Y
K185		-	-	-	R	+	D	D	+	+	Y
K186		-	-	-	R	+	D	D	+	+	Y
K187	FNAM	-	-	-	R	+	D	D	+	+	Y
K188	OBJ1 , DTTM , OBJ2 , DTM2	-	-	-	R	-	R	D	+	-	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K189	PTRM , PRNM , BCAP , LTRM , USER , HST1	R	+	-	D	+	+	+	+	+	Y
K190	DIA1 , INF1 , INF2 , SUFF	-	-	-	R	-	R	D	+	-	Y
K191	SUFF , RQM	-	-	-	D	-	D	D	+	-	Y
K192	NNM1 , HST1 , RSPTC	-	-	-	-	-	D	D	-	-	Y
K193	PTCID , USER , LPAP , LSES , USTYPPTC	-	-	-	D	-	D	D	+	-	Y
K194	GBLNBR , ULLNBR	-	-	-	D	-	D	D	+	-	Y
K195	MSG2	+	+	+	+	+	+	+	+	+	N
K196	MSG2	+	+	+	+	+	+	+	+	+	N
K197	MSG2	+	+	+	+	+	+	+	+	+	N
K198	MSG2	+	+	+	+	+	+	+	+	+	N
K199	MSG2	+	+	+	+	+	+	+	+	+	N
K200	MSG2	-	-	-	+	+	+	+	+	+	N
K201	TSNPID , XASPEC	-	-	-	-	-	R	D	-	-	Y
K202	TSNPID , INSTNUM , TEXT32 , RMSTAT	-	-	-	-	-	R	D	-	-	Y
K203	TSNPID , TEXT32 , INSTNUM , RTAANZ	-	-	-	-	-	R	D	-	-	Y
K204	TSNPID , INTTAID , XATXT	-	-	-	-	-	R	D	-	-	Y
K205	TSNPID , INTTAID , XATXT	-	-	-	-	-	R	D	-	-	Y
K206	TSNPID , INTTAID , XATXT	-	-	-	-	-	R	D	-	-	Y
K207	TSNPID , INTTAID , INSTNUM	-	-	-	-	-	R	D	-	-	Y
K210	TSNPID , XATXT , TEXT32 , INSTNUM	-	-	-	-	-	R	D	-	-	Y
K211	TSNPID , XATXT , TEXT32 , INSTNUM	-	-	-	-	-	R	D	-	-	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K212	TSNPID , XATXT , XAFLAG , INTTAID	-	-	-	-	-	R	D	-	-	Y
K213	TSNPID , XATXT , XAFLAG , INTTAID	-	-	-	-	-	R	D	-	-	Y
K214	TSNPID , XATXT , INTTAID	-	-	-	-	-	R	D	-	-	Y
K215	TSNPID , XATXT , INTTAID	-	-	-	-	-	R	D	-	-	Y
K216	TSNPID , XATXT , TEXT32 , INSTNUM	-	-	-	-	-	R	D	-	-	Y
K217	TSNPID , XATXT , INTTAID	-	-	-	-	-	R	D	-	-	Y
K218	TSNPID , XATXT , INTTAID	-	-	-	-	-	R	D	-	-	Y
K220	TSNPID , TEXT32	-	-	-	-	-	R	D	-	-	Y
K221	TSNPID , TEXT32	-	-	-	-	-	R	D	-	-	Y
K222	TSNPID , XASPEC , TEXT32	-	-	-	-	-	R	D	-	-	Y
K223	TSNPID	-	-	-	-	-	R	D	-	-	Y
K224	TSNPID , XACALL , XATXT , TEXT32 , INSTNUM	-	-	-	-	-	R	D	-	-	Y
K225	TSNPID , XADBC1 , XADBC2	-	-	-	-	-	R	D	-	-	Y
K230	TSNPID , TEXT32	-	-	-	-	-	R	D	-	-	Y
K231	TSNPID	-	-	-	-	-	R	D	-	-	Y
K232	TSNPID	-	-	-	-	-	R	D	-	-	Y
K233	TSNPID , INSTNUM , XACALL , DBCALL , XATXT , LTHGTRID, GTRID	-	-	-	D	+	+	+	+	+	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K235	TCPCL , PRNM , TCPRC , TCPMS , IPADDR	-	-	-	D	+	D	+	+	+	Y
K251	IMPVER , FBASUPD , DEFVER	-	-	-	D	+	D	D	+	+	Y
K252	UPDERR	-	-	-	D	+	D	D	+	+	Y
K255	DMSE , FNAM	-	-	-	D	+	D	D	+	+	Y
K256	FNAM	-	-	-	D	+	D	D	+	+	Y
K257		-	-	-	D	+	D	D	+	+	Y
K258	FNAM	-	-	-	D	+	D	D	+	+	Y
K260	DEFVER , FBASUPD	-	-	-	D	+	D	D	+	+	Y
K261	FNAM	-	-	-	D	+	D	D	+	+	Y
K262	FNAM	-	-	-	D	+	D	D	+	+	Y
K263	FNAM	-	-	-	D	+	D	D	+	+	Y
K269	OBJ1 , OST1 , BMD1 , OBJ3 , OST2 , BMD2	-	-	-	D	+	D	D	+	+	Y
K273	TRMA , UPDMODUL	-	-	-	D	+	D	D	+	+	Y
K274		-	-	-	D	+	D	D	+	+	Y
K277		-	-	-	D	+	D	D	+	+	Y
K278		-	-	-	D	+	D	D	+	+	Y
K279		-	-	-	D	+	D	D	+	+	Y
K300	UPDPRO , PGPOOL	-	-	-	D	+	D	D	+	+	Y
K303	UKCOP , UKCRN , UPDTYP , UKCLM	-	-	-	D	+	D	D	+	+	N
K304	USER , TACTYPE , UKCHSTA	-	-	-	D	+	D	D	+	+	N
K305	UPDTYP , PGS1 , PGS2 , UKCRN , PGPOOL	-	-	-	D	+	D	D	+	+	Y
K306	UPDTYP , PGS1 , PGPOOL	-	-	-	D	+	D	D	+	+	Y
K310	UPDTYP , UKCRN	-	-	-	D	+	D	D	+	+	N

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
K311	UPDTYP , UKCRN	-	-	-	D	+	D	D	+	+	N
K314		-	-	-	D	+	D	D	+	+	N
K317	UKCOP , UKCRN , UPDTYP , UKCLM , RCCC , RCDC , LTRM , USER	-	-	-	D	+	D	D	+	+	N
K318	UPDTYP , UKCRN , LTRM , USER	-	-	-	D	+	D	D	+	+	N
K320	USER , TACTYPE , UERCODE , UERINFO , RCDC	-	-	-	D	+	D	D	+	+	N
P001	XPFUNC , ACPNT , XPRET , XPERR , XP1INFO , XP2INFO , XPCORR	-	-	-	D	+	D	+	+	+	Y
P002	XPFUNC , ACPNT , OSLPAP , XPRET , XPERR , XP1INFO , XP2INFO , XPCORR	-	-	-	D	+	+	+	+	+	Y
P003	ACPNT , XPRJCT , XPLTH	-	-	-	D	+	+	+	+	+	Y
P004	ACPNT , OSLPAP , XPRJCT	-	-	-	D	+	+	+	+	+	Y
P005	ACPNT , XPNSEL , XPTSEL , XPLSSEL , XPCSSEL , XPHSSEL , XPLPSEL , XPCPSEL , XPHPSEL	-	-	-	D	+	+	+	+	+	Y
P006	ACPNT , OSLPAP , XP0OBID , XP1OBID , XP2OBID , XP3OBID , XP4OBID , XP5OBID , XP6OBID , XP7OBID , XP8OBID , XP9OBID	-	-	-	D	+	+	+	+	+	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
P007	ACPNT , OSLPAP , XPRET , XPERR , XP1INFO , XP2INFO , XPCORR	-	-	-	D	+	+	+	+	+	Y
P008	ACPNT , OSLPAP , XPOSAS	-	-	-	D	+	+	+	+	+	Y
P009	ACPNT , OSLPAP , XPRJCT , XPLTH , XPOSAS	-	-	-	D	+	+	+	+	+	Y
P010	ACPNT , OSLPAP , XPNSEL , XPTSEL , XPLSSEL , XPCSSEL , XPHSSEL , XPLPSEL , XPCPSEL , XPHPSEL , XPOSAS	-	-	-	D	+	+	+	+	+	Y
P011	ACPNT , OSLPAP , XP0OBID , XP1OBID , XP2OBID , XP3OBID , XP4OBID , XP5OBID , XP6OBID , XP7OBID , XP8OBID , XP9OBID , XPOSAS	-	-	-	D	+	+	+	+	+	Y
P012	XPCTYPE , XPCCLS , XPCVAL , XPCORR	-	-	-	D	+	+	+	+	+	Y
P013	ACPNT , OSLPAP , XPCRES , XPSRC , XPNDIA , XP1BOOL , XP2BOOL , XP3BOOL , XP4BOOL , XP5BOOL , XPOSAS	-	-	-	D	+	+	+	+	+	Y
P014	XPFUNC , ACPNT , OSLPAP , XPRET , XPERR , XP1INFO , XP2INFO , XPOSAS , XPCORR	-	-	-	D	+	+	+	+	+	Y

MSG ID	Inserts	Station	SYSLINE	Partner	SYSLOG	MSGTAC	SYSOUT	SYSLST	Console	USERDEST	Compress
P015	XPFUNC , ACPNT , OSLPAP , XPLNK , XPSRC , XPNDIA , XPINI , XP1INFO , XP2INFO , XPOSAS , XPCORR	-	-	-	D	+	+	+	+	+	Y
P016	ACPNT , OSLPAP , XPLNK , XPNDIA , XPOSAS	-	-	-	D	+	+	+	+	+	Y
P017	XPPDU , XP1DIA , XP2DIA , XP3DIA	-	-	-	+	+	+	+	+	+	Y
P018	ACPNT , OSLPAP , XPPTYP , XPFSMN	-	-	-	D	+	+	+	+	+	Y
P019	ACPNT , OSLPAP , XPAPDU , XP3INFO	-	-	-	D	+	+	+	+	+	Y
P020	XPTRFAIL	-	-	-	D	+	+	+	+	+	Y
P021	XPEVT , ACPNT , OSLPAP , XPOSAS , XPASST	-	-	-	D	+	+	+	+	+	Y

[1] If you are working with a sign-on service in a UTM application, then these messages are not generated, i.e. they are also not output to the message destinations MSGTAC or SYSLOG.

Destinations for K251- K322 and K800 - K899 (KDCUPD messages)

K800 and K252 - K300 as well as K802 - K850 are output to *stderr* and *stdout*. K801 is output to *stdout* only. The destinations for K303 - K320 and K851 - K858 can be controlled using the LIST statement in the KDCUPD utility. *stdout* and *stderr* are default values.

Destinations for K400 - K522 (KDCDEF messages)

All messages are output to *stdout.*

Destinations for K600 - K622 (KDCCSYSL/KDCPSYSL messages)

stdout for all messages.

Destinations for K650 - K698 (KDCMMOD/KDCMTXT messages)

K660, K661 and K686 thru K688 are output to *stdout*, all other messages are output to *stdout* and *stderr.*

Destinations for U101 - U511

The Messages with message numbers:

U101 thru U125	are output by the dialog terminal process to *stdout*.
U151 and U154 thru U157	are output by the printer process to *stdout* and *stderr.*
U171 thru U177	are output by the utmlog process to *stdout* and *stderr.*
U181 and U182	are output by various processes to *stdout* and *stderr.*
U184 thru U193	are output by various processes to *stdout* and *stderr.*
U201 thru U206	are output by the timer process to *stdout* and *stderr.*
U221 thru U244	are output by the main process to *stdout* and *stderr.*
U245	is output by the main process to *stdout*.
U246 and U247	are output by the main process to *stderr.*
U251 thru U259	are output by kdcuslog to *stdout* and *stderr.*
U271 thru U281	are output by kdccsysl to *stdout* and *stderr.*
U301 thru U323	are output by the network processes to *stdout* and *stderr.*
U341 and U351 thru U363	are output by kdckaa, kdcshut or kdcrem to *stdout* and *stderr.*

U370 thru U392	are output by kdcprog to *stderr*.
U500 to U511	Generated by KDCDEF support for the TNS generation. The messages only occur if the option CHECKTNS=YES is specified. The messages are self-explanatory and are also entered in the relevant line in the file *utmgentns*.

For detailed information on the error numbers in the messages (&ERRNO) on Unix systems, please refer to the C header file `errno.h`.

In Windows systems the corresponding error numbers can be requested using the `GetLastError()` function.

5.12 Windows event logging messages

Installation, uninstallation and operation of openUTM services are logged in the Event Viewer in the form of events. In the Event Viewer, openUTM events can be found in the Application area, the source is openUTM.

An event message is output every 24 hours (MessageId = 5). This informs you that the openUTM service is still active.

The following events are logged:

MessageId	SymbolicName =	Event text	Description
0	UTM_SERVICE_INF_MAIN	<Service Name>: main() started.	The routine main() has been started.
1	UTM_SERVICE_INF_ READREG	<Service Name>: service_main(): ReadRegistry-Parameters() ok.	The necessary registry entries have been read successfully.
5	UTM_SERVICE_INF_REPORT _STATUS	<Service Name>: SetServiceS-tatus to SERVICE_RUNNING ok.	The service has successfully set its status to SERVICE_RUNNING.
6	UTM_SERVICE_INFO_RUN_ SERVICE	<Service Name>: RunService(): called.	The routine RunService() has been called.
7	UTM_SERVICE_INF_SETDIR	<Service Name>: RunService(): directory changed successful.	A switch to the <filebase> directory has been performed successfully in the routing RunService().
8	UTM_SERVICE_INF_READY	<Service Name>: RunService(): service ready.	The routine RunService() reports that the service is available.
9	UTM_SERVICE_INF_STOP	<Service Name>: ServiceStop() called.	The routine ServiceStop() has been called.
11	UTM_SERVICE_INF_MAIN_ RETURN	<Service Name>: End of run: return from main().	The routine main() has been processed in full.
20	UTM_SERVICE_INF_INSTALL	<Service Name>: service installed sucessful.	The openUTM service has been installed successfully.
21	UTM_SERVICE_INF_REMOVE	<Service Name>: service removed sucessful.	The openUTM service has been removed successfully.
101	UTM_SERVICE_ERR_ READREG	<Service Name>: service_main(): ReadRegistry-Parameters() failed. Additional information: GetLastError(): <error code>	The routine service_main() reports that the read of the necessary registry entries has failed. Additional information is provided by the <error code>.

MessageId	SymbolicName =	Event text	Description
103	UTM_SERVICE_ERR_ REGCNTRL	<Service Name>: service_main(): RegisterServiceCtrlHandler() failed: return. Additional information: GetLastError(): <error code>	The routine service_main() reports that the registration of the service has failed. Additional information is provided by the <error code>.
104	UTM_SERVICE_ERR_MAIN_ REPORT	<Service Name>: service_main(): ReportStatusToSCMgr() failed: return. Additional information: GetLastError(): <error code>	The routine service_main() reports that the reporting of the service to the ServiceControlManager has failed. Additional information is provided by the <error code>.
105	UTM_SERVICE_ERR_ REPORT_STATUS	<Service Name>: <error code> failed. Additional information: GetLastError(): <error code>	Setting of the service status has failed. Additional information is provided by the <error code>.
107	UTM_SERVICE_ERR_SETDIR	<Service Name>: RunService(): directory change failed. Additional information: GetLastError(): <error code>	The routine RunService() reports that the switch to the <filebase> directory has failed. Additional information is provided by the <error code>.
108	UTM_SERVICE_ERR_ DISPATCH	<Service Name>: MainMain(): StartServiceCtrlDispatcher failed. Additional information: GetLastError(): <error code>	The routine MainMain() reports that the start of the ServiceControlDispatcher has failed. Additional information is provided by the <error code>.
120	UTM_SERVICE_ERR_INSTALL	<Service Name>: service not installed Additional information: GetLastError(): <error code>	The installation of the openUTM service has failed. Additional information is provided by the <error code>.
121	UTM_SERVICE_ERR_ REMOVE	<Service Name>: service not removed Additional information: GetLastError(): <error code>	The removal of the openUTM service has failed. Additional information is provided by the <error code>.
122	UTM_SERVICE_ERR_ UTMWORK	<Service Name>: utmwork.exe not found Additional information: GetLastError(): <error code>	The file utmwork.exe was not found. Additional information is provided by the <error code>.
123	UTM_SERVICE_ERR_KDCA	<Service Name>: KDCA file not found Additional information: GetLastError(): <error code>	The KDCA file was not found. Additional information is provided by the <error code>.

6 UTM return codes

6.1 KDCS return codes in KCRCCC

The error categories are as follows:

Code	Category	Response
000	No error	Operation carried out without errors
01Z - 09Z	Comments	Operation carried out
10Z - 19Z	Warning or minor errors	Once suitable action has been taken, the program unit run can be continued. The operation was not carried out.
20Z - 39Z	Special functions	A KDCS special function is signaled. Otherwise as for warnings.
40Z - 69Z	Errors	The operation was not carried out. In most cases it will be pointless to continue the program run. Communication with the dialog terminal is still possible.
70Z - 99Z	Serious errors	The program run cannot be continued. openUTM rolls back the transaction and aborts the conversation. openUTM generates a system-specific error message and outputs it to the dialog terminal in a dialog service.

Please note the following:

- For a detailed explanation of the return codes in KCRCCC see the openUTM manual „Programming Applications with KDCS” under each individual KDCS call.
- If more than one error occurs at any one time, the errors from the highest category are displayed. If an operation was not carried out (return code > 09Z) the contents of <parm2> are not modified.
- Evaluation of a UTM-DUMP is described in chapter “The UTM dump” on page 57.

The following return codes are defined (field name specifications: COBOL name/C name):

Code	Meaning
000	The operation was carried out successfully
01Z	Length conflict in KCLA/kcla or KCLKBPRG/kclcapa
02Z	Length conflict in KCLPAB/kclspa
03Z	Name in KCRN/kcrn invalid
04Z	Name in KCRN changed
05Z	Input formatting not carried out with the format identifier specified in KCMF/kcfn. Line mode: 1st character of KCMF/kcfn not space
06Z	Time entry changed within message (DPUT call)
07Z	Length conflict regarding KCLI/kcli
08Z	When reading with waiting in the DGET call: There is currently no message.
10Z	Message already read completely
11Z	When reading without waiting in the DGET call: There is no message.
12Z	No (more) messages from specified service ID or no service stack with the specified number available
14Z	Name in KCRN/kcrn not found
16Z	Operation is illegal and was not performed
19Z	Function key or special function not generated
20Z to 39Z	KDCS special functions (short messages)
40Z	System cannot perform operation (generation error, system error, deadlock, long-term locks)
41Z	Operation illegal at this point
42Z	Operation modifier invalid
43Z	Length entry in KCLM/kclm, KCLI/kcli or KCLA/kcla or KCWTIME/kcwtime invalid
44Z	Name in KCRN/kcrn invalid
45Z	Format identifier KCMF/kcfn or creation time (DGET) invalid
46Z	Name in KCLT/kclt or KCPA/kcpa or KCLANGID/kclangid, KCTERRID/kcterrid, KCCSNAME/kccsname or KCQMODE/kcqmode is invalid
47Z	Storage area <parm2> missing or invalid area address or area cannot be read/written in the specified length
48Z	Invalid interface version
49Z	Unused parameters are not equal to binary zero
51Z	Order with DPUT call not followed
52Z	Message destination is not permitted in KCRN (DPUT call)

Code	Meaning
53Z	Value in KCDPID/kcdpid or KCGTM/kcgtm is invalid (DGET call).
54Z	Value in KCNORPLY/kcnoreply is invalid (in the case of CTRL PR or PE: Value not equal Y or binary zero).
55Z	Name in KCPI/kcpi is invalid (APRO call)
56Z	Value of KCMOD or time entry in KCTAG/kcday,...,KCSEK/kcsec is invalid (DADM, DPUT call)
57Z	Value in KCPOS/kcpos is invalid (MCOM call)
58Z	Value in KCOF/kcof (APRO call)is invalid or the value in KCNEG/kcneg is invaliid (MCOM call).
70Z	System cannot perform operation (generation error, system error)
71Z	Operation not permitted at this stage, e.g. because INIT has not yet been issued
72Z	Operation modifier invalid
73Z	Length entry in KCLM/kclm, KCLA/kcla or KCLI/kcli invalid
74Z	Name in KCRN/kcrn invalid
75Z	Format ID KCMF/kcfn invalid
77Z	Storage area <parm2> missing or invalid area address
79Z	Operation code cannot be interpreted *
80Z	Generation error prior to start of program
81Z	Destination in PEND conflicts with destination in MPUT
82Z	PEND variant conflicts with operand KCOM/kcom or KCRN/kcrn in the MPUT call
83Z	MPUT not issued prior to a PEND KP,RE,FI,ER,FR in a dialog program or an MPUT was not issued prior to a PEND KP,RE in an asynchronous program or an MPUT was issued prior to a PGWT PR.
84Z	PEND missing
86Z	A message complex was not terminated or no FPUT/DPUT was issued after APRO AM for the addressed job-receiving service
87Z	Operation modifier is illegal
88Z	Invalid interface version
89Z	Unused parameters are not equal to binary zero

* Return code 79Z (operation code cannot be interpreted) should not be assigned to any specific operation.

6.2 Internal return code KCRCDC

The incompatible KDCS return code KCRCDC consists of four printable characters and contains more detailed information on the cause of the error than the compatible return code KCRCCC.

The incompatible return code can be set in the following situations:

– if the KDCS call returns the compatible return code 40Z
– if openUTM terminates a service abnormally with PEND ER (KCRCCC ≥ 70Z)
– if a transaction is implicitly reset by openUTM.

The return code KCRCDC has the following structure:

```
sm ##
```

The first character (s) specifies the part of openUTM which reported the error:

A Administration
K UTM system code or ROOT code.
U UPIC (server part)

For all return codes which start with the letter K, the second letter (m) indicates the openUTM module which reported the error (see the list below).

3 KCSPEND (PEND and RSET operations)
6 KCSMPUT (MPUT operation)
7 KCSFPUT(PEND and RSET operations)
8 KCSSSB (SGET, SPUT, SREL, PTDA, GTDA and UNLK operations)
9 KCSLPUT (LPUT operation)
A KCSWAIT
B KCSDGET (DGET operation)
C KCDCTRL (CTRL operation)
D KCDAPRO (UTM-D operations APRO, MPUT, MGET, ...)
E KCSPADM (PADM operation)
F KCSINFO (INFO operation)
G KCSDADM (DADM operation)
H KCSSIGN (SIGN operation)
I KCSISLP (internal operations)
K General DC code
L KCSSTRT
M KCSTRHD (internal operations)
N KCSDLO
Q KCSCRO (QCRE operation)
R KDCRTMM (ROOT-Code)
S KCDSEFU (UTM-D service functions)
T KCDOSTM (OSI-TP transaction termination)

U KDCRTDB (handling of DB processes)
V KCSTUTM (MGET, MPUT for UPIC and socket clients operations)
W KCDOSSF (special OSI-TP functions)
Y KCDSYPM (LU6.1/OSI TP transaction termination)

The following table describes the error causes and, where applicable, the required recovery action. If TESTMODE=OFF, the PEND ER dump is suppressed in cases marked with ([1]). Dynamic loading is also suppressed. At the same time, loading is suppressed unless PGWT calls are permitted for the current TAC and the KDCS call in which the error occurred was not a PEND call.

Code	Module	Cause of error and action (if any)
A010	KCSSADM	User is not a SAT administrator.
A011	KCSSADM	TAC has no SAT admission.
A012	KCSSADM	SAT subsystem is not available.
A013	KCSSADM	SAT version is not compatible.
A015	KCSSADM	An inverse KDCDEF is running or is to be started.
A016	KCSSADM	The gobal cluster administration is locked.
A100	KCSADMI	When calling the program interface for administration, the address specified for the parameter area is either inaccessible or cannot be accessed for the length of the parameter area or it is not on a word boundary. Response: Checking the address specified for the parameter area in the program unit.
A101	KCSADMI	An invalid return code was issued internally when calling the programming interface for administration. System error. Action: Contact System Support. The PEND ER dump is required for diagnosis.
ABTR	KCSENDE	The code is reported in UTM messages K017 or K055. It occurs whenever a service is forced to terminate by UTM and UTM-STXIT handling is not run (STXIT switched off or STXIT code overwritten). This error is comparable to an ABNORMAL TASK TERMINATION, and should therefore be accurately diagnosed. Generally, this does not adversely affect application execution.
EXIT	KDCROOT	– The exit() function was called illegally in a C program. – The statement STOP RUN was executed in a COBOL program. – The COBOL runtime system has detected a data error, an index error or similar in a COBOL. Hints on the causer can be found by analyzing the calling stack in the corresponding core.
K300	KCSPEND	Incorrect operation modification in the PEND call.

Code	Module	Cause of error and action (if any)
K301	KCSWAIT	(As for KA00) The buffer is too small for the input message. Action: Define larger buffer length with MAX TRMSGLTH=length in KDCDEF generation. No PEND ER dump created.
K302	KCSPEND	The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
K303	KCSPEND	An incorrect operation modification was used in the PGWT call.
K304	KCSPEND	The DB system requests a CLOSE DB call prior to END RE/FI, but the call was not entered.
K305	KCSPEND	A PEND KP or a PEND PA/PR with task switch was called, although the DB system with which the transaction was opened does not permit a PEND KP.
K306	KCSPEND	The transaction and the service had to be reset since the DB system reset the DB transaction when closing down.
K307	KCSPEND	PEND PS is only permitted in the first part of the sign-on process.
K308	KCSPEND	The DB transaction was closed prior to PEND KP or PEND PA/PR with task switch.
K309	KCSPEND	The TAC specified in KCRN is not permitted as a follow-on TAC.
K310	KCSPEND	Error when storing the MPUT message during PEND PA/PR with task switch.
K311	KCSPEND	A temporary end of transaction (PTC) was reached with PEND RE/FI, but the DB system where the transaction is open does not permit the transaction status.
K313	KCSPEND	A SIGN OF/OB was issued in a program unit run, but the output message is for the job-receiver or the program run was terminated with an PEND variant that is not permitted.
K314	KCSPEND	A program unit run of a sign-on service was not terminated with PEND PS following a successful SIGN ON.
K315	KCSPEND	With service stacking, an MPUT PM was issued with KCLM/kclm > 0 prior to PEND FI. The last output from the preceding service was a LINEMODE message and can only be output in unmodified form.

Code	Module	Cause of error and action (if any)
K316	KCSPEND	PEND RS was called in the program unit or a situation (e.g. loss of connection) has occurred which requires UTM to roll back the transaction. However, PEND RS is not permitted because no service restart is possible. This is the case if: – the service was started by a UPIC client or an OSI TP job submitter which has not selected the Functional Unit Commit and no user with the restart attribute is signed on at the connection/association and no local service restart is possible because the last synchronization point was not set with PEND SP or PEND FC (UPIC only) – or the last synchronization point was set with PGWT CM. No PENDER dump is written.
K318	KCSPEND	PGWT was called in a program unit for whose TAC – no TAC class is generated or – if TAC classes have been generated: PGWT=YES is not generated for the TAC class – if TAC-PRIORITIES has been generated: the TAC was not generated with PGWT=YES Action: Correct generation with KDCDEF.
K319	KCSPEND	There are not enough tasks to use the PGWT or PGWT was called from a system task (only possible via a privieged LTERM). Action: Increase number of tasks.
K320	KCSPEND	After a RSET in a distributed transaction, the transaction was not reset, although UTMD RSET = GLOBAL was generated.
K322	KCSPEND	The operation modification FC is not permitted in asynchronous services or server services.
K323	KCSPEND	The operation modification FC is not permitted in the sign-on procedure if a service restart is required.
K324	KCSPEND	KCRN does not contain blanks on PEND RS or PEND FR.
K325	KCSPEND	The operation modification RS is not permitted in the sign-on procedure or in the MSGTAC program, or the Resource Manager of the XA connection requires a reset of the transaction in the case of an xa_end of the sign-on procedure or the MSGTAC program,.
K326	KCSPEND	Switch between dialog transaction code and asynchronous transaction code is not permitted, or the follow-up TAC in PEND FC is not a service TAC, or the follow-up TAC in PEND PA/PR, PS, KP, RE or SP is not a follow-up TAC.
K327	KCSPEND	The operation modification FC is not allowed in the sign-on service if the last sign-on attempt failed.
K328	KCSPEND	The sign-on service cannot be terminated normally when the validity period of the password has expired but the password has not been changed.
K329	KCSPEND	The sign-on service cannot be terminated normally when the password passed with KDCUPD does not meet the complexity level requirement or is too short and the password has not been changed with SIGN CP.

Code	Module	Cause of error and action (if any)
K330	KCSPEND	A SIGN OB in the sign-on service for terminals is only allowed in conjunction with an MPUT NT/NE.
K331	KCSPEND	If a sign-on service with a service restart is terminated without the user signing off, then no MPUT NT/NE is allowed.
K332	KCSPEND	If the sign-on service with a service restart PEND FI is terminated without a preceding MPUT, then openUTM terminates the open service.
K333	KCSPEND	There should be a wait for a DGET message, but the follow-up program unit is not in a TAC class. Action: Generate follow-up TAC with TAC class.
K334	KCSPEND	There should be a wait for a DGET message; only PEND PA/PR/RS/ER/FR and PGWT RB are permitted.
K335	KCSPEND	The process is terminated abnormally as during a timeout on the PGWT bourse it was determined that not enough processes are active to continue the procedure. No PENDER dump is written.
K336	KCSPEND	Although required, no MPUT call was made prior to PEND KP/RE/FI/FR/ER or PGWT KP.
K337	KCSPEND	An MPUT call was made prior to PGWT PR.
K338	KCSPEND	The transaction was reset because the application is being terminated.
K339	KCSPEND	The MSGTAC program unit was terminated without reading a message with FGET.
K340	KCSPEND	On transaction rollback, the database reported an error which makes it necessary to dynamically load the application program.
K341	KCSPEND	Only in cluster applications: A user was generated or deleted in the current transaction. The user file could not be locked because it is currently being accessed by a KDCDEF run.
K342	KCSPEND	Only in UTM cluster applications: A user was generated or deleted in the current transaction. The cluster user file could not be locked because another process belonging to the same node application or another node application has locked the cluster user file.
K343	KCSPEND	PET state rejected because the service is locked be administration.
K344	KCSPEND	CR could not be written because a global ADM lock is set.
K345	KCSPEND	The page pool in KDCFILE is full. Action: Change generation with KDCDEF, enlarge page pool with MAX PGPOOL=(number,...). [1]
K346	KCSPEND	The sign-on service is not permitted to terminate an open service if a transaction in the open service is in the “prepare to commit” (PTC) state. The sign-on service is terminated abnormally.

Code	Module	Cause of error and action (if any)
K347	KCSPEND	In a UTM cluster application, a transaction wants to go to the status “prepare to commit” (PTC) but an error occurred while writing the status to the cluster user file.
K348	KCSPEND	The cluster page pool in the KDCFILE is full. Action: Generation with KDCDEF; increase size of cluster page pool with CLUSTER PGPOOL=(number,...).
K349	KCSPEND	The connection to the partner has been cleared down.
K350	KCSPEND	Invalid operation modification after the abnormal termination with CTRL AB of a dialog in which the commit functional unit was selected.
K351	KCSPEND	The transaction is marked to be rolled back but a PEND/PGWT call was used to set a transaction forward.
K360	KCSPEND	The transaction was rolled back with PGWT RB because KCSPEND was called from KDCROOT with PGWT RB.
K361	KCSPEND	The transaction was reset with PEND RS because KCSPEND was called by KDCROOT with PEND RS.
K362	KCSPEND	The transaction was reset with PEND ER because KCSPEND was called by KDCROOT with PEND ER.
K363	KCSPEND	The transaction was reset with RSET because KCSPEND was called by KDCROOT with RSET.
K601	KCSMPUT	Buffer for dialog messages generated too small. Action: Modify generation, define larger buffer with MAX NB=length. [1]
K602	KCSMPUT	MPUT call demands format output but formatting was not generated. [1]
K603	KCSMPUT	MPUT with screen function KCCARD, but terminal does not have an ID card reader. Action: Change generation or program unit [1]
K604	KCSMPUT	MPUT with screen function KCCARD and user ID has operand CARD=(position,string): the ID card reader cannot be used for KDCSIGN checks and data input at the same time. Action: Change generation or program unit
K605	KCSMPUT	MPUT with screen function KCCARD and output in formatted mode. Action: Change program
K606	KCSMPUT	Caused by a MPUT call with KCDF not equal to binary zero and one of the following conditions: - follow-up partial message in formatted mode - KCMF = name of an EUA format - KCMF = name of an edit profile. Action: Change program unit
K607	KCSMPUT	The MPUT message specified by address and length partly covers an internal ROOT buffer area (the MPUT buffer) Action: Change program (length of MPUT too large)

Code	Module	Cause of error and action (if any)
K608	KCSPEND	(See K345) The page pool in KDCFILE is full. Action: Change generation with KDCDEF, enlarge page pool with MAX PGPOOL=(number,...). [1]
K610	KCSMPUT	The ES operation modification of the MPUT call is only permitted for UPIC and SOCKET partners.
K611	KCSMPUT	In the case of the MPUT ES call, the KCRN field is not filled with blanks.
K612	KCSMPUT	MPUT was called in the MSGTAC program unit.
K613	KCSMPUT	Operation modification illegal.
K614	KCSMPUT	MPUT PM in the asynchronous service.
K615	KCSMPUT	MPUT PM in the first part of the sign-on service.
K616	KCSMPUT	MPUT PM with KCLM/kclm not equal to 0 in the sign-on service.
K617	KCSMPUT	MPUT PM, but the service is neither inserted nor a sign-on service.
K618	KCSMPUT	MPUT PM in the sign-on service for a UPIC connection user.
K701	KCSFPUT	UTM refuses an asynchronous message because level 2 has already been reached in the page pool. Action: See K345
K702	KCSFPUT	The name of a dialog TAC was specified in KCRN. Response: new generation or change the program.
K703	KCSMPUT	(See K602) The MPUT call demands format output but formatting was not generated.
K704	KCSPEND	(See K302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
K705	KCSFPUT	An asynchronous message is to be sent with FPUT to an LTERM or (OSI-)LPAP partner, for which LTERM ...,QAMSG=N is specified in the KDCDEF generation. UTM therefore rejects the message. Action: Set up connection
K706	KCSFPUT	An asynchronous message is to be sent to an dialog terminal with FPUT/DPUT in line mode. The message is longer than the buffer area defined in the MAX NB= parameter in the generation. UTM does not accept the message. Action: Specify a higher value for MAX NB= or short message segments.
K707	KCSFPUT	The target time specified in a DPUT call exceeds the limits set by the generation (MAX DPUTLIMIT1=, DPUTLIMIT2=). Action: Change time entry in DPUT call or in generation.

Code	Module	Cause of error and action (if any)
K708	KCSFPUT	An asynchronous message is to be sent to an LTERM or (OSI-)LPAP partner, an asynchronous TAC, a TAC queue, a user queue, a temporary queue or an (OSI) LPAP with FPUT/DPUT. The number of asynchronous messages for this LTERM partner or TAC has already reached the maximum value defined at generation (KDCDEF: LTERM QLEV=, TAC QLEV=, USER QLEV=, QUEUE QLEV=, LPAP QLEV=, OSI-LPAP QLEV=).
K711	KCSFPUT	A DPUT was issued with a new destination in KCRN although the preceding DPUT to another destination is not yet completed.
K712	KCSFPUT	The name of a UTM-D partner, a MUX connection or a UPIC partner was specified in KCRN. Response: Correct generation or program.
K713	KCSFPUT	Caused by an FPUT/DPUT call to an ID card reader Action: Change program unit.
K714	KCSFPUT	The name of a message complex was specified in KCRN in an FPUT call. Action: Correct program.
K715	KCSFPUT	The name of a message complex was specified in KCRN, but this name does not match the message complex currently open. Action: Correct program.
K716	KCSFPUT	The name of a message complex was specified in KCRN, but no message complex is currently open. Action: Correct program.
K717	KCSFPUT	The name of a TAC was specified in KCRN, but the specification is incompatible with the specification in KCOM. Action: Correct program.
K718	KCSFPUT	An asynchronous service running under a deleted LTERM partner attempted to issue an FPUT or DPUT.
K719	KCSFPUT	An asynchronous service running under a deleted user attempted to issue an FPUT or DPUT.
K720	KCSFPUT	In the case of a DPUT call to a USER queue: There is no USER with the name specified in KCRN, or the USER was deleted.
K721	KCSFPUT	In the case of a DPUT call to a USER queue: The KSET of the user and the write ACL of the USER queue do not have a common key.
K722	KCSFPUT	Time-controlled DPUT calls for USER queues are not possible (KCMOD not equal to blank).
K723	KCSFPUT	In the case of a DPUT call to a QUEUE object: There is no QUEUE object with the name specified in KCRN.
K724	KCSFPUT	In the case of a DPUT call to a USER queue: The KSET of the LTERM and the write ACL of the USER queue do not have a common key.
K725	KCSFPUT	Time-controlled DPUT calls are not possible for QUEUE objects. (KCMOD not equal to blank)

Code	Module	Cause of error and action (if any)
K726	KCSFPUT	Invalid value in KCQTYP
K731	KCSFPUT	No asynchronous messages can be sent to a TAC generated with CALL=NEXT.
K732	KCSFPUT	No asynchronous messages may be sent to KDCMSGLT.
K733	KCSFPUT	No asynchronous message may be sent to an LTERM that has been implicitly generated for the internal cluster communication.
K801	KCSPEND	(See 345) The page pool in KDCFILE is full. Action: Change generation with KDCDEF, enlarge page pool with MAX PGPOOL=(number,...). [1]
K802	KCSPEND	(See 302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
K804	KCSSSB	No GSSBs were generated or more GSSBs were generated than permitted in the generation.
K805	KCSSSB	SPUT created more LSSBs than specified at generation.
K810	KCSSSB	When accessing a GSSB, TLS or ULS: The area cannot be occupied by the transaction following a certain wait period. Action: Increase wait period by increasing RESWAIT=(time1,...) parameter in KDCDEF generation.
K811	KCSSSB	When accessing a GSSB, TLS or ULS: The area is currently locked by another transaction for an "indeterminate" period, i.e. it has occupied the area and then issued PEND KP or PGWT KP call.
K812	KCSSSB	When accessing a ULS: The user ID for the user whose ULS is to be accessed will be deleted.
K813	KCSSSB	The application is terminated.
K820	KCSSSB	Waiting for a global scondary storage area would lead to a deadlock.
K822	KCSPEND	(See K348) The cluster page pool in the KDCFILE is full. Action: Generation with KDCDEF; increase size of cluster page pool with CLUSTER PGPOOL=(number,...).
K823	KCSSSB	Only in cluster applications: A timeout occurred on a request for a file lock for the administration file for GSSB or ULS.

Code	Module	Cause of error and action (if any)
K824	KCSSSB	When accessing a GSSB, TLS or ULS: The area is currently locked and the task cannot wait for the lock to be released because there are already too many tasks waiting. Action: – Start more tasks. – Limit the number of tasks for the TACs that access GSSB or ULS and then start more tasks than the specified limit value. For information on how to distribute UTM services that use GSSB or ULS memory areas to the tasks in a UTM cluster application, see openUTM manual “Using openUTM Applications under BS2000 Systems”.
K825	KCSSSB	Only in cluster applications when accessing a GSSB or ULS: The queue at the requested area has already reached the maximum length; the service cannot currently wait for this area.
K826	KCSSSB	Only in cluster applications when accessing a GSSB or ULS: A transient error occurred on a request for an internal resource; access to the requested area is not currently possible.
K827	KCSSSB	Only in cluster applications when accessing a GSSB or ULS: The area is currently locked by a failed node; there is at present no point in waiting for this area.
K901	KCSLPUT	UTM rejects an LPUT call because level 2 has already been reached in the page pool . Action: Change program or increase page pool in generation.
K902	KCSPEND	(See 302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
K903	KCSLPUT	UTM rejects an LPUT call because a DMS error occurred during the last write procedure to the user log file(s). The DMS error was output with message K043. The records buffered in the page pool remain. Action: Analyze the K043 message and the DMS error code, restore or recreate the user log file(s) and then continue with the KDCLOG administration command (or via the corresponding program interface). The LPUT records in the page pool are then output. The lock for the LPUT call is removed.
KA00	KCSWAIT	The buffer is too small for the input message. Action: Define larger buffer length with MAX TRMSGLTH=length in KDCDEF generation. No PEND ER dump created.

Code	Module	Cause of error and action (if any)
KA01	KCSWAIT	A serious error ocuured during communication between job submitter and job receiver (UTM-D). Communication cannot be continued; UTM aborts the job-receiving service with PEND ER. Possible cause: - PEND ER by the job submitter - timeout on the connection to the job submitter. Diagnostic aids: – analyze VTV error message K086 – for PEND ER in job-submitting service, look for cause in dump.
KA02	KCSWAIT	The page pool in the KDCFILE is full, consequently the messages could not be passed to the task waiting in the PGWT. Action: Generation in KDCDEF: increase page pool using MAX PGPOOL=(number,...)
KA03	KCSWAIT	UTM has received an unencrypted message although an encrypted message was expected.
KA04	KCSWAIT	The connection to the partner was shut down.
KA05	KCSWAIT	A transaction in PET state was reset following an administration request.
KB01	KCSDGET	DGET calls are not permitted in the first part of the sign-on service.
KB02	KCSDGET	A DGET message must be waited for; no further DGET calls are therefore permitted.
KB03	KCSDGET	Invalid value of KCQTYP
KB04	KCSDGET	There is no object for the name of the type KCQTYP specified in KCRN, or this object has been deleted.
KB05	KCSDGET	Messages to dialog or asynchronous TACs cannot be read by means of a DGET call.
KB06	KCSDGET	The read ACL or the TAC or USER queue and the KSET of the user's LTERM do not have a common key.
KB07	KCSDGET	The read ACL or the TAC or USER queue and the KSET of the user do not have a common key.
KB08	KCSPEND	(See K302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
KB10	KCSDGET	Negative wait time in the case of a DGET FT /BF call, or wait time is not equal to zero in the case of a DGET NT/BN/PF/PN call.
KB11	KCSDGET	DGET calls with waiting are not permitted for the MSGTAC program.
KB12	KCSDGET	In the case of the call DGET NT/BN/PN, the name or type of the specified queue does not suit the previous DGET call of the current program unit run.

Code	Module	Cause of error and action (if any)
KB13	KCSDGET	An attempt was made to read from a USER queue although the application was generated without any users.
KB14	KCSDGET	An attempt was made to read from a TAC queue with STATUS = HALT or KEEP.
KB15	KCSDGET	An attempt was made to read from a TAC queue with ADMIN=Y, but the user is not an administrator.
KB16	KCSDGET	Operation modifier in KCOM is invalid.
KB17	KCSDGET	For DGET BF/PF: Value in KCDPID does not contain a valid DPUT ID.
KB18	KCSDGET	For DGET FT/NT: KCMF/kcfn does not contain blanks.
KB19	KCSDGET	For DGET NT/BN/PN (next): KCOM does not match the preceding DGET call, or no DGET FT/BF/PF (first) has been issued yet in this program run, or a PGWT call was made in the intervening period.
KB20	KCSDGET	For DGET NT/BN/PN: Since the last DGET call the DGET queue deleted and regenerated.
KB21	KCSDGET	For DGET BN/PN (next): Since the last DGET call the USER specified in the KCRN and KCQTYP or the temporary queue was deleted and regenerated.
KB22	KCSDGET	For DGET BN: There is no message with the creation time specified in KCGTM, or processing has taken place in the meantime.
KB23	KCSDGET	For DGET PF: There is no message with the creation time specified in KCGTM, or processing has taken place in the meantime.
KB24	KCSDGET	For DGET BF/PF: KCDPID does not match the specifications in KCRN and KCQTYP.
KB25	KCSDGET	DGET calls with waiting are not permitted in the sign-on service.
KB26	KCSDGET	Attempts were made with DGET FT/NT/PF/PN to read from the dead letter queue KDCDLETQ.
KC01	KCDCTRL	VTV has not been generated.
KC02	KCDCTRL	The first character in the service ID specified in KCRN is not '>'.
KC03	KCDCTRL	The call was issued for an asynchronous service, i.e. the service ID specified in KCRN was defined with an APRO AM call.
KC04	KCDCTRL	The VGID specified in KCRN is invalid.
KC06	KCDCTRL	The CTRL call was addressed to a partner which is not communicated with via the OSI-TP protocol
KC07	KCDCTRL	CTRL PR was intended for a partner for which the commit functional unit has not been selected.
KC08	KCDCTRL	CTRL PE was intended for a partner for which the commit functional unit has not been selected.
KC09	KCDCTRL	CTRL AB was intended for a partner to whom a message has already been sent with MPUT.

Code	Module	Cause of error and action (if any)
KC10	KCDCTRL	The operation modification OM is nether PR nor PE nor AB.
KC11	KCDCTRL	KCLA/kcla is not zero.
KC12	KCDCTRL	KCLM/kclm is not zero.
KC13	KCDCTRL	KCMF/kcfn is not filled with spaced (blanks).
KC14	KCDCTRL	KCDF/kcdf does not contain binary zeros.
KC15	KCDCTRL	The extended parameter area (EXTENT) does not contain binary zeros.
KC16	KCDCTRL	The call was intended for a partner for whom an MPUT HM has already been issued.
KC17	KCDCTRL	The call CTRL PR was intended for a partner in a middle node, but the local service had not already received a Prepare from its job submitter.
KC18	KCDCTRL	The call CTRL PE was intended for a partner in a middle node, but the local service had not already received a Prepare from its job submitter.
KC19	KCDCTRL	The CTRL AB call was intended for a partner for which the commit functional unit has not been selected and the service status is not O.
KC20	KCDCTRL	The CTRL AB call was intended for a partner for which the commit functional unit has not been selected and the service status is neither O nor C.
KC21	KCDCTRL	The CTRL PR call is not permitted because the service status is not O.
KC22	KCDCTRL	The CTRL PE call is not permitted because the service status is not O.
KC23	KCDCTRL	The CTRL PR call is not permitted because the transaction status is P.
KC24	KCDCTRL	The CTRL PE call is not permitted because the transaction status is P.
KC25	KCDCTRL	Binary zeros have not been entered for the unused part of the extended parameter area (EXTENT).
KC26	KCDCTRL	The KCNORPLY field does not have either the value Y or binary zero.
KD00	KCDAPRO	KCRN does not contain a valid service ID (MGET, MPUT, FPUT, DPUT).
KD01	KCDAPRO	LTAC is not defined.
KD02	KCDAPRO	LTAC cannot be used; LTAC is locked.
KD03	KCDAPRO	User does not have a key in his/her key set for the lock on the LTAC.
KD04	KCDAPRO	APRO DM was called, but the LTAC is generated as an asynchronous LTAC, or APRO AM was called and LTAC is generated as a dialog LTAC.
KD05	KCDAPRO	APRO created more job-receiving services than specified at generation time (KDCDEF generation: UTMD MAXJR=)
KD06	KCDAPRO	MPUT to job-receiving service and KCDF not binary 0
KD08	KCSPEND	(See K345) The page pool in KDCFILE is full. Action: Change generation with KDCDEF, enlarge page pool with MAX PGPOOL=(number,...). [1]

Code	Module	Cause of error and action (if any)
KD09	KCDAPRO	APRO DM was called, but the virtual connections to the remote application were cleared down with the administration command KDCLPAP ACT=QUIET.
KD10	KCDAPRO	APRO DM was called, but no virtual connection was set up to the remote application.
KD11	KCDAPRO	APRO DM was called, but the remote application was generated as "contention winner" and the wait period for session occupancy is 0 (only for LU6.1).
KD12	KCDAPRO	Following an MPUT NE/HM to a job-receiving service, another MPUT was issued to the job-receiving service.
KD13	KCDAPRO	No connection exists to the remote application with FPUT (or DPUT with KCMOD='_') to a job-receiving service and the wait period for session or association occupancy is 0.
KD14	KCDAPRO	An asynchronous service in another application is to be addressed with APRO. The number of asynchronous jobs for this application has already reached the maximum value defined at generation (LPAP QLEV= or OSI-LPAP ALEV=).
KD15	KCDAPRO	With an MPUT EM or MPUT HM the destination in KCRN/kcrn is not an OSI TP communication partner.
KD16	KCDAPRO	No MPUT NT was given prior to an MPUT HM.
KD17	KCDAPRO	The RTAC was generated for an OSI TP partner.
KD18	KCDAPRO	A selected functional unit is not supported by this UTM version.
KD22	KCDAPRO	The commit functional unit was selected, but the abstract syntax CCR was not generated for the partner.
KD23	KCDAPRO	Mixed operation of LU6.1 and OSI TP within a distributed transaction.
KD24	KCDAPRO	Mixed operation of LU6.1 and OSI TP within a distributed transaction.
KD25	KCDAPRO	More than one ACCESS-POINT was used in a distributed transaction with OSI TP.
KD26	KCDAPRO	An abstract syntax which has not been generated was specified for an OSI/TP partner for MPUT/FPUT/DPUT in KCMF/kcfn. The syntax names "CCR" and "OSITP" are not permitted.
KD27	KCDAPRO	KCLM/kclm must be 32.
KD28	KCDAPRO	KCLM/kclm must be zero
KD29	KCDAPRO	KCLM/kclm must be equal to the length of the data structure in COPY KCAPROC or include kcapro.h.
KD30	KCDAPRO	Illegal values in the APRO data area.
KD31	KCDAPRO	KCFUCHN must contain blanks for KCFUCOM = 'N'.
KD32	KCDAPRO	Security type "Same" or "Program" was selected, but the abstract syntax UTMSEC has not been generated for the partner.

Code	Module	Cause of error and action (if any)
KD33	KCDAPRO	An invalid length was specified for the user ID or the password with the security type "Program".
KD34	KCDAPRO	Error on encoding the security data.
KD35	KCDAPRO	KCFUHSH = 'Y' is only permitted in the case of dialog partners.
KD36	KCDAPRO	After a CTRL PR or PE, an MPUT HM was issued to the same partner.
KD37	KCDAPRO	After a CTRL AB, an MPUT was issued to the same partner.
KD38	KCDAPRO	MPUT to job submitter, but KCSEND = NO.
KD39	KCDAPRO	The unused fields for the security function when KCSEC TYP is not equal to 'P' are not deleted. (CHAR fields to blank, numeric fields to 0).
KD40	KCDAPRO	There is no active connection assigned to the OSI-LPAP in the APRO call.
KD41	KCDAPRO	The master LPAP is locked.
KD42	KCSPEND	(See K348) The cluster page pool in the KDCFILE is full. Action: Generation with KDCDEF; increase size of cluster page pool with CLUSTER PGPOOL=(number,...).
KE01	KCSPADM	No authorization for the call: The user is not an administrator and the terminal is not a printer control station or is different from the terminal specified in the KDCS parameter area (KCLT/kclt).
KE02	KCSPADM	No printer (PTERM) is assigned to the printer control terminal. Check generation.
KE03	KCSPADM	A PADM call for printer acknowledgment or print repeat (KCOM = OK/PR) was issued even though there was no printout to be acknowledged for the specified printer.
KE04	KCSPEND	(See K302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
KE05	KCSPADM	A printer is to be assigned to another LTERM (KCOM = CA), but is linked with the application.
KE06	KCSPADM	Connection to a printer is to be set up, but the printer is locked.
KE07	KCSPADM	An inverse KDCDEF is running or is to be started.
KE08	KCSPADM	Administration applying globally to the cluster is not currently possible because node applications with different generations are running.
KF01	KCSINFO	The user has been generated without an ID card and no Kerberos dialog was executed for the LTERM client (INFO CD).

Code	Module	Cause of error and action (if any)
KF02	KCSINFO	The information is no longer available, e.g. because of a loss of connection (INFO CD).
KF05	KCSINFO	The service is not running under a “genuine” user ID.
KF06	KCSINFO	Error on encrypting the password.
KG01	KCSDADM	No authorization for the call: The user is not an administrator and the terminal is not a printer control terminal or is different from the terminal specified in the KDCS parameter area (KCLT/kclt)
KG02	KCSPEND	(See K302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
KG03	KCSDADM	A DADM call to delete all asynchronous messages was made (KCOM = DA) even though messages were being processed for the specified destination.
KG04	KCSDADM	An attempt was made to administer a message currently being processed.
KG05	KCSDADM	A call to delete messages (KCOM = DL/DA) was followed by another delete job or a job to modify concatenation of a message (KCOM = DL/DA/CS).
KG06	KCSDADM	An attempt was made to modify the concatenation of a time-driven message (KCOM = CS), although its start time had not yet been reached.
KG07	KCSDADM	A call to read information about messages was issued (KCOM = RQ), although there were no messages for the specified destination (or only messages currently being processed).
KG08	KCSDADM	KCQTYP invalid
KG09	KCSDADM	In the case of DADM MV with blanks in KCLT/kclt, the original destination of the message in the dead letter queue no longer exists. This message must be assigned a new destination.
KG10	KCSDADM	In the case of DADM MV, the destination specified in KCLT/kclt no longer exists.
KH01	KCSPEND	(See K302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
KH02	KCSSIGN	With SIGN CP: The new password does not satisfy the requested complexity level.
KH03	KCSSIGN	With SIGN CP: The new password is too short.
KH04	KCSSIGN	With SIGN CP: The new password is identical to the old password or to a password in the password history.

Code	Module	Cause of error and action (if any)
KH08	KCSSIGN	An inverse KDCDEF is running or is to be started.
KH09	KCSSIGN	With SIGN CP: The password may not be changed because the minimum validity period has not been exceeded yet.
KH10	KCSSIGN	With SIGN CP: A password cannot be assigned or changed for a user generated with a certificate.
KH11	KCSSIGN	The service was terminated abnormally because the page list for the cluster service data could not be written when the user signed off due to the fact that the cluster page pool was full.
KH12	KCSSIGN	An open cluster service cannot be continued because the sequence of TAC statements has changed following a regeneration of the node KDCFILE, or the properties of teh follow-up TAC have been changed, or the follow-up TAC or the service TAC no longer exist.
KJ01	KCCUSF	The service was terminated abnormally because it was marked for abnormal termination by another node application. This may have been performed via the administration functions or could be due to the fact that, even though there is a service bound to this node application, the user has signed on at another node application.
KK01	KCSSVCS	The second parameter in the KDCS call is missing.
KK02	KCSSVCS	Address validation of the second parameter in the KDCS call failed.
KK03	KCSSVCS	The KDCS call was issued from a program unit in an asynchronous service.
KK04	KCSSVCS	The KDCS operation code is unknown.
KK07	KCSSVCS	The KDCS operation code APRO is not permitted.
KK08	KCSSVCS	The call is not permitted in UTM cluster applications.
KL00	KCSSTRT	During warm start of the application no job-receiver session in the PET state was found.
KL01	KCSSTRT	During warm start of the application no valid NODE table entry was found for the OSI TP service.
KL02	KCSSTRT	The user who started this service has been deleted.
KL03	KCSSTRT	During warm start it was determined that the service cannot be continued. OSI TP job revceiving services are possibly terminated not before the association to OSI TP partner has been established and the subsequent recovery has been processed.
KL04	KCSSTRT	PEND ER triggered by KCSSTRT
KL05	KCSSTRT	PEND ER triggered by KCSSTRT
KL06	KCSSTRT	PEND ER triggered by KCSSTRT
KL07	KCSSTRT	PEND ER triggered by KCSSTRT
KL08	KCSSTRT	PEND ER triggered by KCSSTRT
KM01	KCSTRHD	TAC is not defined. [1]

Code	Module	Cause of error and action (if any)
KM02	KCSTRHD	TAC cannot be used, TAC locked. [1]
KM03	KCSTRHD	There is no key in the key set of the LTERM or (OSI-)LPAP partner for the lock on the TAC. [1]
KM04	KCSTRHD	The user does not have a key in his/her key set for the lock on the TAC. [1]
KM05	KCSTRHD	The TAC is generated as an administration TAC, but the user is not an administrator. [1]
KM07	KCSTRHD	The program unit associated with the transaction code is not linked or could not be loaded. [1]
KM08	KCSTRHD	The transaction code is completely locked. It cannot be specified as a follow-on TAC in a PEND call, nor can new jobs be specified for this TAC. [1]
KM09	KCSTRHD	The transaction code is protected by an encryption level and the caller does not support this encryption level or the input message was not encrypted with the appropriate level.
KM99	KCSTRHD	A TAC which is generated with XOPEN-API may only be used by a job submitter with whom communication takes place via the OSI TP, LU6.1 or UPIC protocol.
KN01	KCSDLO	The user who started the service has been deleted.
KQ01	KCSCRO	In the case of the QCRE WN call, the name in KCRN begins with a digit.
KQ02	KCSCRO	In the case of the QCRE NN call, KCRN was not supplied with blanks.
KQ03	KCSPEND	(See K302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
KQ04	KCSCRO	The tables for temporary queues reserved at generation by means of the QUEUE statement have been used up. Action: Delete or regenerate QUEUE objects that are no longer required.
KQ05	KCSCRO	In the case of the QCRE WN call, KCRN contains an invalid character or is not filled with blanks.
KR01	KDCRTMM	A dialog program unit with which a service was meant to continue is missing (not linked). [1]
KR02	KDCRTMM	An asynchronous program unit for which a message is awaiting processing is missing (not linked). [1]
KR04	KDCRTMM	A program unit run prior to the last KDCS call continued writing beyond the end of the KB. Action: Change program unit or generation.
KR05	KDCRTMM	A program unit run prior to the last KDCS call continued writing beyond the end of the SPAB. Action: Change program unit or generation.

Code	Module	Cause of error and action (if any)
KR06	KDCRTMM	A KDCS call was made by the VORGANG exit program. The service aborts with PEND ER.
KR09	KDCRTMM	Error when loading a program unit. The service aborts with PEND ER.
KR10	KDCRTMM	Program for VORGANG exit is missing. The service aborts with PEND ER.
KS00	KCDSEFU	In connection with a transaction end request, MPUTs were made for two or more partners.
KS01	KCDSEFU	In connection with a transaction end request, at least one partner has an illegal transaction or service status.
KS02	KCDSEFU	In connection with a transaction end request, at least one session was not used (too many APRO calls).
KS03	KCDSEFU	MPUT was issued to a service ID by the job submitterand teh transaction was with subsequently terminated with PEND RE even though all job submitters had not yet reported PEND RE/FI.
KS04	KCDSEFU	A message was sent with PEND KP to an LU6.1 partner which has already initiated transaction end.
KS05	KCDSEFU	If the session is both job receiver and job submitter, then it may not send PEND RE to its job receiver until after receiving a PEND RE message from its job submitter.
KS06	KCDSEFU	No asynchronous message was issued for the addressed service following an APRO AM.
KS07	KCDSEFU	PEND SP was issued even though there is at least one partner with an open transaction.
KS08	KCDSEFU	The OSI TP client sent a message to the OSI TP server without terminating the transaction, or has requested the OSI TP server to terminate the transaction although the server has already initiated transaction termination.
KS09	KCDSEFU	The OSI TP server has not terminated the transaction even though it was requested to do so by the OSI TP client.
KS10	KCDSEFU	Contrary to the request of the OSI TP client, the OSI TP server terminated the transaction with PEND SP.
KS11	KCDSEFU	The OSI TP client sent a message to the OSI TP server and terminated the transaction, although it does not have the send authorization for transaction termination, since it is an OSI TP server in a different dialog and its OSI TP client has not relinquished the send authorization for transaction termination.
KS12	KCDSEFU	Contrary to the request of the OSI TP client, the OSI TP server terminated the transaction with PEND RE.
KS13	KCDSEFU	The OSI TP client terminated the service with PEND FI although it still has an open server conversation without COMMIT functionality.
KS14	KCDSEFU	Contrary to the request of the OSI TP client, the OSI TP server terminated the transaction with PEND FI.

Code	Module	Cause of error and action (if any)
KS15	KCDSEFU	The OSI TP client terminated the service with PEND FC although it still has an open server conversation without COMMIT functionality.
KS16	KCDSEFU	The OSI TP server sent a message to the OSI TP client although it does not possess send authorization on this dialog.
KS17	KCDSEFU	The OSI TP server terminated the dialog step, but did not send a message to the OSI TP client although it possesses send authorization on this dialog.
KS18	KCDSEFU	The OSI TP client relinquished send authorization for transaction termination to more than one OSI TP server.
KS19	KCDSEFU	The OSI TP client requested the OSI TP server to terminate the transaction or service with CTRL PR or PE, sent a message to this partner and then initiated transaction termination itself.
KS20	KCDSEFU	The local service has called PGWT CM or PGWT RB although a partner is involved in the distributed transaction and the LU6.1 protocol is used to communicate with the partner.
KS21	KCDSEFU	The OSI TP server has terminated the transaction with PGWT CM contrary to the request of the OSI TP client.
KS22	KCDSEFU	The OSI TP client terminated the service with PEND FI although it only requested a server conversation to terminate the transaction with CTRL PR.
KS23	KCDSEFU	The OSI TP client terminated the service with PEND FC although it only requested a server conversation to terminate the transaction with CTRL PR.
KS24	KCDSEFU	The OSI TP client requested a server conversation to terminate the transaction or service with CTRL PR/PE even though no message has been sent to the server conversation.
KS25	KCDSEFU	Session restart has failed.
KT01	KCSPEND	(See K302) The task-specific buffer area for restart information is too small (see UTM manual Generating appllications, restart area). Action: Define a larger buffer area with MAX RECBUF=(...,length) in KDCDEF generation. [1]
KT02	KCSPEND	(See K345) The page pool in KDCFILE is full. Action: Change generation with KDCDEF, enlarge page pool with MAX PGPOOL=(number,...). [1]
KT03	KCSPEND	(See K306) The transaction and the service had to be reset since the DB system reset the DB transaction when closing down.
KT04	KCDOSTM	A log record is too large for the input buffer. Response: address fewer OSI TP job-receivers in this service [1]
KT05	KCDOSTM	The transaction must be rolled back because the database transaction was rolled back.

Code	Module	Cause of error and action (if any)
KT06	KCDOSTM	The database reported that the transaction was rolled back when the transaction was rolled terminated after a Prepare statement.
KT07	KCDOSTM	An OSI TP server reported a heuristic decision (heuristic mixed).
KT08	KCDOSTM	An OSI TP server reported a heuristic decision (heuristic hazard).
KU04	KDCRTDB	The DB transaction had to be reset. UTM also resets the UTM transaction: Control is restored to the program unit run, as with the UTM call RSET.
KU08	KDCRTDB	UTM has to terminate the service with PEND ER. The terminal user is sent a message announcing the termination of the service. The first 4 bytes of the DB error message are included in the display.
KU0C	KDCRTDB	The DB system (the DBH) is not or no longer available. UTM terminates the application abnormally or aborts application startup.
KU10	KDCRTDB	The DB system is no longer available due to DB administrator action. UTM terminates application abnormally.
KU14[1]	KDCRTDB	The DB system is momentarily not available. A new attempt to set up the connection is made after 5 seconds.
KU18	KDCRTDB	A (possibly recoverable) DB system error occurred.
KU1C	KDCRTDB	An unrecoverable DB system error occurred. UTM terminates the application abnormally.
KU20	KDCRTDB	The DB system detected a user error. The error occurs at application startup when the DB-specific start parameters are checked. The DB error message is output to SYSOUT.
KU24	KDCRTDB	UTM behavior unexpected by the DB system. The application is terminated with PEND06.
KV01	KCSTUTM	Socket partner sends with USP version 1.0. Buffer for dialog messages generated too small or MPUT length has exceeded the maximum value. Action: Change generation; select larger buffer with MAX NB=length.
KV02	KCSPEND	(See K345) The page pool in KDCFILE is full. Action: Change generation with KDCDEF, enlarge page pool with MAX PGPOOL=(number,...). [1]
KV03	KCSPEND	(See K348) The cluster page pool in the KDCFILE is full. Action: Generation with KDCDEF; increase size of cluster page pool with CLUSTER PGPOOL=(number,...).
KW01	KCDOSSF	TP-HANDSHAKE-IND received.
KW02	KCDOSSF	TP-BEGIN-DIALOGUE-CNF received (negative).
KW03	KCDOSSF	TP-U-ERROR-IND received from a partner which does not have send authorization.

Code	Module	Cause of error and action (if any)
KW04	KCDOSSF	Timer expired.
KW05	KCDOSSF	TP-U-ABORT-IND or TP-P-ABORT-IND received.
KW06	KCDOSSF	TP-ROLLBACK-IND received.
KW07	KCDOSSF	openUTM does not support reception of TP-END-DIALOGUE-IND from a client.
KW08	KCDOSSF	The dialog with the client was not terminated with the first transaction in the unchained functions functional unit.
KY00	KCDSYPM	A negative acknowledgment was received from the partner.
KY01	KCDSYPM	Timeout when reserving a session.
KY02	KCDSYPM	The job-receiving service was terminated with PEND ER.
KY03	KCDSYPM	The job-submitting service was terminated with PEND ER.
KY04	KCDSYPM	Mismatch with the partner, or the local service is in the PET status.
KY05	KCDSYPM	Mismatch with the partner.
KY06	KCDSYPM	Mismatch with the database.
KY07	KCDSYPM	Loss of connection to the partner.
NOTA	KCDRTDB	Transaction is unknown at the XA Resource Manager. (May occur, for example, after timeouts with the Oracle database).
UPCP	KCSPEND	(See K348) The cluster page pool in the KDCFILE is full. Action: Generation with KDCDEF; increase size of cluster page pool with CLUSTER PGPOOL=(number,...).
UPDE	KCSUPIC	The UPIC partner has required a service abort.
UPPG	KCSPEND	(See K345) The page pool in KDCFILE is full. Action: Change generation with KDCDEF, enlarge page pool with MAX PGPOOL=(number,...). [1]
UPRS	KCSUPIC	The UPIC partner has suppressed a service restart. This occurs when a service restart would have been possible, but was not required by the UPIC partner. UTM then resets the service.
UPXC	KCSUPIC	In a program unit with X/OPEN-API, the message to the UPIC job submitter is to be concluded even though a message to a job receiver service has already been concluded.
UPXM	KCSUPIC	In a program unit with X/OPEN-API, the message to the UPIC job submitter is to be concluded even though a message to the UPIC job submitter has already been concluded.
XTnn	KDCROOT	The STXIT routine was called in KDCROOT, nn = event code of the STXIT macro. XT80[1] see footnote.

[1] When TESTMODE=OFF, these PENDER dumps are suppressed. Dynamic loading is suppressed at the same time unless for the current TAC, PGWT calls are permitted and the KDCS call at which the error occurred was not a PEND call.

Glossary

A term in *italic* font means that it is explained somewhere else in the glossary.

abnormal termination of a UTM application
: Termination of a *UTM application*, where the *KDCFILE* is not updated. Abnormal termination is caused by a serious error, such as a crashed computer or an error in the system software. If you then restart the application, openUTM carries out a *warm start*.

abstract syntax (OSI)
: Abstract syntax is defined as the set of formally described data types which can be exchanged between applications via *OSI TP*. Abstract syntax is independent of the hardware and programming language used.

acceptor (CPI-C)
: The communication partners in a *conversation* are referred to as the *initiator* and the acceptor. The acceptor accepts the conversation initiated by the initiator with Accept_Conversation.

access list
: An access list defines the authorization for access to a particular *service*, *TAC queue* or *USER queue*. An access list is defined as a *key set* and contains one or more *key codes*, each of which represent a role in the application. Users or LTERMs or (OSI) LPAPs can only access the service or *TAC queue/USER queue* when the corresponding roles have been assigned to them (i.e. when their *key set* and the access list contain at least one common *key code)*.

access point (OSI)
: See *service access point.*

ACID properties
: Acronym for the fundamental properties of *transactions*: atomicity, consistency, isolation and durability.

administration
: Administration and control of a *UTM application* by an *administrator* or an *administration program*.

administration command

Commands used by the *administrator* of a *UTM application* to carry out administration functions for this application. The administration commands are implemented in the form of *transaction codes*.

administration journal

See *cluster administration journal*.

administration program

Program unit containing calls to the *program interface for administration*. This can be either the standard administration program *KDCADM* that is supplied with openUTM or a program written by the user.

administrator

User who possesses administration authorization.

AES

AES (Advanced Encryption Standard) is the current symmetric encryption standard defined by the National Institute of Standards and Technology (NIST) and based on the Rijndael algorithm developed at the University of Leuven (Belgium). If the AES method is used, the UPIC client generates an AES key for each session.

Apache Axis

Apache Axis (Apache eXtensible Interaction System) is a SOAP engine for the design of Web services and client applications. There are implementations in C++ and Java.

Apache Tomcat

Apache Tomcat provides an environment for the execution of Java code on Web servers. It was developed as part of the Apache Software Foundation's Jakarta project. It consists of a servlet container written in Java which can use the JSP Jasper compiler to convert JavaServer pages into servlets and run them. It also provides a fully featured HTTP server.

application cold start

See *cold start*.

application context (OSI)

The application context is the set of rules designed to govern communication between two applications. This includes, for instance, abstract syntaxes and any assigned transfer syntaxes.

application entity (OSI)

An application entity (AE) represents all the aspects of a real application which are relevant to communications. An application entity is identified by a globally unique name (“globally” is used here in its literal sense, i.e. worldwide), the *application entity title* (AET). Every application entity represents precisely one *application process*. One application process can encompass several application entities.

application entity qualifier (OSI)

Component of the *application entity title*. The application entity qualifier identifies a *service access point* within an application. The structure of an application entity qualifier can vary. openUTM supports the type “number”.

application entity title (OSI)

An application entity title is a globally unique name for an *application entity* (“globally” is used here in its literal sense, i.e. worldwide). It is made up of the *application process title* of the relevant *application process* and the *application entity qualifier.*

application information

This is the entire set of data used by the *UTM application*. The information comprises memory areas and messages of the UTM application including the data currently shown on the screen. If operation of the UTM application is coordinated with a database system, the data stored in the database also forms part of the application information.

application process (OSI)

The application process represents an application in the *OSI reference model*. It is uniquely identified globally by the *application process title*.

application process title (OSI)

According to the OSI standard, the application process title (APT) is used for the unique identification of applications on a global (i.e. worldwide) basis. The structure of an application process title can vary. openUTM supports the type *Object Identifier.*

application program

An application program is the core component of a *UTM application*. It comprises the main routine *KDCROOT* and any *program units* and processes all jobs sent to a *UTM application*.

application restart

see *warm start*

application service element (OSI)

An application service element (ASE) represents a functional group of the application layer (layer 7) of the *OSI reference model*.

application warm start

see *warm start*.

association (OSI)

An association is a communication relationship between two application entities. The term “association” corresponds to the term *session* in *LU6.1*.

asynchronous conversation

CPI-C conversation where only the *initiator* is permitted to send. An asynchronous transaction code for the *acceptor* must have been generated in the *UTM application*.

asynchronous job

Job carried out by the job submitter at a later time. openUTM includes *message queuing* functions for processing asynchronous jobs (see *UTM-controlled queue* and *service-controlled queue)*. An asynchronous job is described by the *asynchronous message*, the recipient and, where applicable, the required execution time. If the recipient is a terminal, a printer or a transport system application, the asynchronous job is a *queued output job*. If the recipient is an *asynchronous service* of the same application or a remote application, the job is a *background job*. Asynchronous jobs can be *time-driven jobs* or can be integrated in a *job complex*.

asynchronous message

Asynchronous messages are messages directed to a *message queue*. They are stored temporarily by the local *UTM application* and then further processed regardless of the job submitter. Distinctions are drawn between the following types of asynchronous messages, depending on the recipient:

– In the case of asynchronous messages to a *UTM-controlled queue,* all further processing is controlled by openUTM. This type includes messages that start a local or remote *asynchronous service* (see also *background job*) and messages sent for output on a terminal, a printer or a transport system application (see also *queued output job*).
– In the case of asynchronous messages to a *service-controlled queue*, further processing is controlled by a *service* of the application. This type includes messages to a *TAC queue*, messages to a *USER queue* and messages to a *temporary queue*. The USER queue and the temporary queue must belong to the local application, whereas the TAC queue can be in both the local application and the remote application.

asynchronous program

Program unit started by a *background job*.

asynchronous service (KDCS)

Service which processes a *background job*. Processing is carried out independently of the job submitter. An asynchronous service can comprise one or more program units/transactions. It is started via an asynchronous *transaction code*.

audit (BS2000 systems)

During execution of a *UTM application*, UTM events which are of relevance in terms of security can be logged by *SAT* for auditing purposes.

authentication

See *system access control*.

authorization

See *data access control*.

Axis

See *Apache Axis*.

background job

Background jobs are *asynchronous jobs* destined for an *asynchronous service* of the current application or of a remote application. Background jobs are particularly suitable for time-intensive processing or processing which is not time-critical and where the results do not directly influence the current dialog.

basic format

Format in which terminal users can make all entries required to start a service.

basic job

Asynchronous job in a *job complex*.

browsing asynchronous messages

A *service* sequentially reads the *asynchronous messages* in a *service-controlled queue*. The messages are not locked while they are being read and they remain in the queue after they have been read. This means that they can be read simultaneously by different services.

bypass mode (BS2000 systems)

Operating mode of a printer connected locally to a terminal. In bypass mode, any *asynchronous message* sent to the printer is sent to the terminal and then redirected to the printer by the terminal without being displayed on screen.

cache

Used for buffering application data for all the processes of a *UTM application*. The cache is used to optimize access to the *page pool* and, in the case of UTM cluster applications, the *cluster page pool*.

CCS name (BS2000 systems)

See *coded character set name*.

client

Clients of a *UTM application* can be:

– terminals
– UPIC client programs
– transport system applications (e.g. DCAM, PDN, CMX, socket applications or UTM applications which have been generated as *transport system applications*).

Clients are connected to the UTM application via LTERM partners.
openUTM clients which use the OpenCPIC carrier system are treated just like *OSI TP partners*.

client side of a conversation

This term has been superseded by *initiator*.

cluster

A number of computers connected over a fast network and which in many cases can be seen as a single computer externally. The objective of clustering is generally to increase the computing capacity or availability in comparison with a single computer.

cluster administration journal

The cluster administration journal consists of:

– two log files with the extensions JRN1 and JRN2 for global administration actions,
– the JKAA file which contains a copy of the KDCS Application Area (KAA). Administrative changes that are no longer present in the two log files are taken over from this copy.

The administration journal files serve to pass on to the other node applications those administrative actions that are to apply throughout the cluster to all node applications in a UTM cluster application.

cluster configuration file

File containing the central configuration data of a *UTM cluster application*. The cluster configuration file is created using the UTM generation tool *KDCDEF*.

cluster filebase

Filename prefix or directory name for the *UTM cluster files*.

cluster GSSB file

File used to administer GSSBs in a *UTM cluster application*. The cluster GSSB file is created using the UTM generation tool *KDCDEF*.

cluster lock file

File in a *UTM cluster application* used to manage cross-node locks of user data areas.

cluster page pool

The cluster page pool consists of an administration file and up to 10 files containing a *UTM cluster application's* user data that is available globally in the cluster (service data including LSSB, GSSB and ULS). The cluster page pool is created using the UTM generation tool *KDCDEF*.

cluster start serialization file

Lock file used to serialize the start-up of individual node applications (only in Unix systems and Windows systems).

cluster ULS file

File used to administer the ULS areas of a *UTM cluster application*. The cluster ULS file is created using the UTM generation tool *KDCDEF*.

cluster user file

File containing the user management data of a *UTM cluster application*. The cluster user file is created using the UTM generation tool *KDCDEF*.

coded character set name (BS2000 systems)

If the product *XHCS* (e**X**tended **H**ost **C**ode **S**upport) is used, each character set used is uniquely identified by a coded character set name (abbreviation: "CCS name" or "CCSN").

cold start

Start of a *UTM application* after the application terminates normally (*normal termination*) or after a new generation (see also *warm start*).

communication area (KDCS)

KDCS *primary storage area*, secured by transaction logging and which contains service-specific data. The communication area comprises 3 parts:

– the KB header with general service data
– the KB return area for returning values to KDCS calls

– the KB program area for exchanging data between UTM program units within a single *service*.

communication resource manager

In distributed systems, communication resource managers (CRMs) control communication between the application programs. openUTM provides CRMs for the international OSI TP standard, for the LU6.1 industry standard and for the proprietary openUTM protocol UPIC.

configuration

Sum of all the properties of a *UTM application*. The configuration describes:

– application parameters and operating parameters
– the objects of an application and the properties of these objects. Objects can be *program units* and *transaction codes*, communication partners, printers, *user IDs*, etc.
– defined measures for controlling data and system access.

The configuration of a UTM application is defined at generation time (*static configuration*) and can be changed dynamically by the administrator (while the application is running, *dynamic configuration*). The configuration is stored in the *KDCFILE*.

confirmation job

Component of a *job complex* where the confirmation job is assigned to the *basic job*. There are positive and negative confirmation jobs. If the *basic job* returns a positive result, the positive confirmation job is activated, otherwise, the negative confirmation job is activated.

connection bundle

see *LTERM bundle*.

connection user ID

User ID under which a *TS application* or a *UPIC client* is signed on at the *UTM application* directly after the connection has been established. The following applies, depending on the client (= LTERM partner) generation:

– The connection user ID is the same as the USER in the LTERM statement (explicit connection user ID). An explicit connection user ID must be generated with a USER statement and cannot be used as a “genuine” *user ID*.

– The connection user ID is the same as the LTERM partner (implicit connection user ID) if no USER was specified in the LTERM statement or if an LTERM pool has been generated.

In a *UTM cluster application*, the service belonging to a connection user ID (RESTART=YES in LTERM or USER) is bound to the connection and is therefore local to the node.
A connection user ID generated with RESTART=YES can have a separate service in each *node application*.

contention loser

Every connection between two partners is managed by one of the partners. The partner that manages the connection is known as the *contention winner*. The other partner is the contention loser.

contention winner

A connection's contention winner is responsible for managing the connection. Jobs can be started by the contention winner or by the *contention loser*. If a conflict occurs, i.e. if both partners in the communication want to start a job at the same time, then the job stemming from the contention winner uses the connection.

conversation

In CPI-C, communication between two CPI-C application programs is referred to as a conversation. The communication partners in a conversation are referred to as the *initiator* and the *acceptor*.

conversation ID

CPI-C assigns a local conversation ID to each *conversation*, i.e. the *initiator* and *acceptor* each have their own conversation ID. The conversation ID uniquely assigns each CPI-C call in a program to a conversation.

CPI-C

CPI-C (Common Programming Interface for Communication) is a program interface for program-to-program communication in open networks standardized by X/Open and CIW (**C**PI-C **I**mplementor's **W**orkshop).
The CPI-C implemented in openUTM complies with X/Open's CPI-C V2.0 CAE Specification. The interface is available in COBOL and C. In openUTM, CPI-C can communicate via the OSI TP, *LU6.1* and UPIC protocols and with openUTM-LU62.

Cross Coupled System / XCS

Cluster of BS2000 computers with the *Highly Integrated System Complex* Multiple System Control Facility (HIPLEX® MSCF).

data access control

In data access control openUTM checks whether the communication partner is authorized to access a particular object belonging to the application. The access rights are defined as part of the configuration.

dead letter queue

The dead letter queue is a TAC queue which has the fixed name KDCDLETQ. It is always available to save queued messages sent to transaction codes or TAC queues but which could not be processed. The saving of queued messages in the dead letter queue can be activated or deactivated for each message destination individually using the TAC statement's DEAD-LETTER-Q parameter.

DES

DES (Data Encryption Standard) is an international standard for encrypting data. One key is used in this method for encoding and decoding. If the DES method is used, the UPIC client generates a DES key for each session.

dialog conversation

CPI-C conversation in which both the *initiator* and the *acceptor* are permitted to send. A dialog transaction code for the *acceptor* must have been generated in the *UTM application*.

dialog job, interactive job

Job which starts a *dialog service*. The job can be issued by a *client* or, when two servers communicate with each other (*server-server communication*), by a different application.

dialog message

A message which requires a response or which is itself a response to a request. The request and the response both take place within a single service. The request and reply together form a dialog step.

dialog program

Program unit which partially or completely processes a *dialog step*.

dialog service

Service which processes a *job* interactively (synchronously) in conjunction with the job submitter (*client* or another server application) . A dialog service processes *dialog messages* received from the job submitter and generates dialog messages to be sent to the job submitter. A dialog service comprises at least one *transaction.* In general, a dialog service encompasses at least one dialog step. Exception: in the event of *service chaining,* it is possible for more than one service to comprise a dialog step.

dialog step

A dialog step starts when a *dialog message* is received by the *UTM application*. It ends when the UTM application responds.

dialog terminal process (Unix systems/Windows systems)

A dialog terminal process connects a terminal of a Unix system or a Windows system with the work processes of the *UTM application*. Dialog terminal processes are started either when the user enters utmdtp or via the LOGIN shell. A separate dialog terminal process is required for each terminal to be connected to a UTM application.

Distributed Lock Manager / DLM (BS2000 systems)

Concurrent, cross-computer file accesses can be synchronized using the Distributed Lock Manager.
DLM is a basic function of HIPLEX® MSCF.

distributed processing

Processing of *dialog jobs* by several different applications or the transfer of *background jobs* to another application. The higher-level protocols *LU6.1* and *OSI TP* are used for distributed processing. openUTM-LU62 also permits distributed processing with LU6.2 partners. A distinction is made between distributed processing with *distributed transactions* (transaction logging across different applications) and distributed processing without distributed transactions (local transaction logging only). Distributed processing is also known as server-server communication.

distributed transaction

Transaction which encompasses more than one application and is executed in several different (sub)-transactions in distributed systems.

distributed transaction processing

Distributed processing with *distributed transactions.*

dynamic configuration

Changes to the *configuration* made by the administrator. UTM objects such as *program units*, *transaction codes*, *clients*, *LU6.1 connections,* printers or *user IDs* can be added, modified or in some cases deleted from the configuration while the application is running. To do this, it is necessary to create separate *administration programs* which use the functions of the *program interface for administration*. The WinAdmin administration program or the WebAdmin administration program can be used to do this, or separate *administration programs* must be created that utilize the functions of the *administration program interface.*

encryption level

The encryption level specifies if and to what extent a client message and password are to be encrypted.

event-driven service

This term has been superseded by *event service*.

event exit

Routine in an application program which is started automatically whenever certain events occur (e.g. when a process is started, when a service is terminated). Unlike *event services*, an event exit must not contain any KDCS, CPI-C or XATMI calls.

event function

Collective term for *event exits* and *event services*.

event service

Service started when certain events occur, e.g. when certain UTM messages are issued. The *program units* for event-driven services must contain KDCS calls.

filebase

UTM application filebase
In BS2000 systems, filebase is the prefix for the *KDCFILE*, the *user log file* USLOG and the *system log file* SYSLOG.
In Unix and Windows systems, filebase is the name of the directory under which the KDCFILE, the user log file USLOG, the system log file SYSLOG and other files relating to to the UTM application are stored.

generation

Static configuration of a *UTM application* using the UTM tool KDCDEF and creation of an application program.

global secondary storage area

See *secondary storage area.*

hardcopy mode

Operating mode of a printer connected locally to a terminal. Any message which is displayed on screen will also be sent to the printer.

heterogeneous link

In the case of *server-server communication:* a link between a *UTM application* and a non-UTM application, e.g. a CICS or TUXEDO application.

Highly Integrated System Complex / HIPLEX®

Product family for implementing an operating, load sharing and availability cluster made up of a number of BS2000 servers.

HIPLEX® MSCF

(MSCF = **M**ultiple **S**ystem **C**ontrol **F**acility)
Provides the infrastructure and basic functions for distributed applications with HIPLEX®.

homogeneous link

In the case of *server-server communication*: a link between two *UTM applications.* It is of no significance whether the applications are running on the same operating system platforms or on different platforms.

inbound conversation (CPI-C)

See *incoming conversation.*

incoming conversation (CPI-C)

A conversation in which the local CPI-C program is the *acceptor* is referred to as an incoming conversation. In the X/Open specification, the term “inbound conversation” is used synonymously with “incoming conversation”.

initial KDCFILE

In a *UTM cluster application*, this is the *KDCFILE* generated by *KDCDEF* and which must be copied for each node application before the node applications are started.

initiator (CPI-C)

The communication partners in a *conversation* are referred to as the initiator and the *acceptor*. The initiator sets up the conversation with the CPI-C calls Initialize_Conversation and Allocate.

insert

Field in a message text in which openUTM enters current values.

inverse KDCDEF

A function which uses the dynamically adapted configuration data in the *KDCFILE* to generate control statements for a *KDCDEF* run. An inverse KDCDEF can be started “offline” under *KDCDEF* or “online” via the *program interface for administration*.

JDK

Java Development Kit
Standard development environment from Sun Microsystems for the development of Java applications.

job

Request for a *service* provided by a *UTM application*. The request is issued by specifying a transaction code. See also: *queued output job*, *dialog job*, *background job*, *job complex*.

job complex

Job complexes are used to assign *confirmation jobs* to *asynchronous jobs*. An asynchronous job within a job complex is referred to as a *basic job*.

job-receiving service (KDCS)

A job-receiving service is a *service* started by a *job-submitting service* of another server application.

job-submitting service (KDCS)

A job-submitting service is a *service* which requests another service from a different server application (*job-receiving service*) in order to process a job.

KDCADM

Standard administration program supplied with openUTM. KDCADM provides administration functions which are called with transaction codes (*administration commands*).

KDCDEF

UTM tool for the *generation* of *UTM applications*. KDCDEF uses the configuration information in the KDCDEF control statements to create the UTM objects *KDCFILE* and the ROOT table sources for the main routine *KDCROOT*.
In UTM cluster applications, KDCDEF also creates the *cluster configuration file,* the *cluster user file*, the *cluster page pool*, the *cluster GSSB file* and the *cluster ULS file*.

KDCFILE

One or more files containing data required for a *UTM application* to run. The KDCFILE is created with the UTM generation tool *KDCDEF*. Among other things, it contains the *configuration* of the application.

KDCROOT

Main routine of an *application program* which forms the link between the *program units* and the UTM system code. KDCROOT is linked with the *program units* to form the *application program*.

KDCS message area

For KDCS calls: buffer area in which messages or data for openUTM or for the *program unit* are made available.

KDCS parameter area

See *parameter area*.

KDCS program interface

Universal UTM program interface compliant with the national DIN 66 265 standard and which includes some extensions. KDCS (compatible data communications interface) allows dialog services to be created, for instance, and permits the use of *message queuing* functions. In addition, KDCS provides calls for *distributed processing*.

Kerberos

Kerberos is a standardized network authentication protocol (RFC1510) based on encryption procedures in which no passwords are sent to the network in clear text.

Kerberos principal

Owner of a key.
Kerberos uses symmetrical encryption, i.e. all the keys are present at two locations, namely with the key owner (principal) and the KDC (Key Distribution Center).

key code

Code that represents specific access authorization or a specific role. Several key codes are grouped into a *key set*.

key set

Group of one or more *key codes* under a particular a name. A key set defines authorization within the framework of the authorization concept used (lock/key code concept or *access list* concept). A key set can be assigned to a *user ID*, an *LTERM partner* an (OSI) *LPAP partner*, a *service* or a *TAC queue*.

linkage program

See *KDCROOT*.

local secondary storage area

See *secondary storage area*.

Log4j

Log4j is part of the Apache Jakarta project. Log4j provides information for logging information (runtime information, trace records, etc.) and configuring the log output. *WS4UTM* uses the software product Log4j for trace and logging functionality.

lock code

Code protecting an LTERM partner or transaction code against unauthorized access. Access is only possible if the *key set* of the accesser contains the appropriate *key code* (lock/key code concept).

logging process

Process in Unix and Windows systems that controls the logging of account records or monitoring data.

LPAP bundle

LPAP bundles allow messages to be distributed to LPAP partners across several partner applications. If a UTM application has to exchange a very large number of messages with a partner application then load distribution may be improved by starting multiple instances of the partner application and distributing the messages across the individual instances. In an LPAP bundle, *open*UTM is responsible for distributing the messages to the partner application instances. An LPAP bundle consists of a master LPAP and multiple slave LPAPs. The slave LPAPs are assigned to the master LPAP on generation. LPAP bundles exist for both the OSI TP protocol and the LU6.1 protocol.

LPAP partner

In the case of *distributed processing* via the *LU6.1* protocol, an LPAP partner for each partner application must be configured in the local application. The LPAP partner represents the partner application in the local application. During communication, the partner application is addressed by the name of the assigned LPAP partner and not by the application name or address.

LTERM bundle

An LTERM bundle (connection bundle) consists of a master LTERM and multiple slave LTERMs. An LTERM bundle (connection bundle) allows you to distribute queued messages to a logical partner application evenly across multiple parallel connections.

LTERM group

An LTERM group consists of one or more alias LTERMs, the group LTERMs and a primary LTERM. In an LTERM group, you assign multiple LTERMs to a connection.

LTERM partner

LTERM partners must be configured in the application if you want to connect clients or printers to a *UTM application*. A client or printer can only be connected if an LTERM partner with the appropriate properties is assigned to it. This assignment is generally made in the *configuration*, but can also be made dynamically using terminal pools.

LTERM pool

The TPOOL statement allows you to define a pool of LTERM partners instead of issuing one LTERM and one PTERM statement for each *client*. If a client establishes a connection via an LTERM pool, an LTERM partner is assigned to it dynamically from the pool.

LU6.1

Device-independent data exchange protocol (industrial standard) for transaction-oriented *server-server communication*.

LU6.1-LPAP bundle

LPAP bundle for *LU6.1* partner applications.

LU6.1 partner

Partner of the *UTM application* that communicates with the UTM application via the *LU6.1* protocol.

Examples of this type of partner are:

– a UTM application that communicates via LU6.1
– an application in the IBM environment (e.g. CICS, IMS or TXSeries) that communicates via LU6.1

main process (Unix systems / Windows systems)

Process which starts the *UTM application*. It starts the *work processes*, the *UTM system processes*, *printer processes*, *network processes*, *logging process* and the *timer process* and monitors the *UTM application*.

main routine KDCROOT

See *KDCROOT*.

management unit

SE Servers component; in combination with the *SE Manager*, permits centralized, web-based management of all the units of an SE server.

mapped host name

Mapping of the partner application's UTM host name to a real host name or vice versa.

message definition file

The message definition file is supplied with openUTM and, by default, contains the UTM message texts in German and English together with the definitions of the message properties. Users can take this file as a basis for their own message modules.

message destination

Output medium for a *message*. Possible message destinations for a message from the openUTM transaction monitor include, for instance, terminals, *TS applications*, the *event service* MSGTAC, the *system log file* SYSLOG or *TAC queues, asynchronous TACs, USER queues*, SYSOUT/SYSLST or stderr/stdout.
The message destinations for the messages of the UTM tools are SYSOUT/SYSLST and stderr/stdout.

message queue

Queue in which specific messages are kept with transaction management until further processed. A distinction is drawn between *service-controlled queues* and *UTM-controlled queues*, depending on who monitors further processing.

message queuing

Message queuing (MQ) is a form of communication in which the messages are exchanged via intermediate queues rather than directly. The sender and recipient can be separated in space or time. The transfer of the message is independent of whether a network connection is available at the time or not. In openUTM there are *UTM-controlled queues* and *service-controlled queues*.

message router (BS2000 systems)

Device in a central host or a communication computer which distributes queued input messages to different *UTM applications* which can be located on different computers. The message router also allows you to work with *multiplex connections*.

MSGTAC

Special event service that processes messages with the message destination MSGTAC by means of a program. MSGTAC is an asynchronous service and is created by the operator of the application.

multiplex connection (BS2000 systems)

Special method of connecting terminals to a *UTM application*. A multiplex connection enables several terminals to share a single transport connection.

multi-step service (KDCS)

Service carried out in a number of *dialog steps*.

multi-step transaction
Transaction which comprises more than one *processing step*.

Network File System/Service / NFS
Allows Unix systems to access file systems across the network.

network process (Unix systems / Windows systems)
A process in a *UTM application* for connection to the network.

network selector
The network selector identifies a service access point to the network layer of the *OSI reference model* in the local system.

node
Individual computer of a *cluster*.

node application
UTM application that is executed on an individual *node* as part of a *UTM cluster application*.

node bound service
A node bound service belonging to a user can only be continued at the node application at which the user was last signed on. The following services are always node bound:
– Services that have started communications with a job receiver via LU6.1 or OSI TP and for which the job-receiving service has not yet been terminated
– Inserted services in a service stack
– Services that have completed a SESAM transaction
In addition, a user's service is node bound as long as the user is signed-on at a node application.

node filebase
Filename prefix or directory name for the *node application's KDCFILE*, *user log file* and *system log file*.

node recovery
If a node application terminates abnormally and no rapid warm start of the application is possible on its associated *node computer* then it is possible to perform a node recovery for this node on another node in the UTM cluster. In this way, it is possible to release locks resulting from the failed node application in order to prevent unnecessary impairments to the running *UTM cluster application*.

normal termination of a UTM application

Controlled termination of a *UTM application*. Among other things, this means that the administration data in the *KDCFILE* are updated. The *administrator* initiates normal termination (e.g. with KDCSHUT N). After a normal termination, openUTM carries out any subsequent start as a *cold start*.

object identifier

An object identifier is an identifier for objects in an OSI environment which is unique throughout the world. An object identifier comprises a sequence of integers which represent a path in a tree structure.

open terminal pool

Terminal pool which is not restricted to clients of a single computer or particular type. Any client for which no computer- or type-specific terminal pool has been generated can connect to this terminal pool.

online import

In a *UTM cluster application*, online import refers to the import of application data from a normally terminated node application into a running node application.

online update

In a *UTM cluster application*, online update refers to a change to the application configuration or the application program or the use of a new UTM revision level while a *UTM cluster application* is running.

OpenCPIC

Carrier system for UTM clients that use the *OSI TP* protocol.

OpenCPIC client

OSI TP partner application with the *OpenCPIC* carrier system.

openSM2

The openSM2 product line offers a consistent solution for the enterprise-wide performance management of server and storage systems. openSM2 offers the acquisition of monitoring data, online monitoring and offline evaluation.

openUTM application

See *UTM application*.

openUTM cluster

From the perspective of UPIC clients, **not** from the perspective of the server: Combination of several node applications of a UTM cluster application to form one logical application that is addressed via a common symbolic destination name.

openUTM-D

openUTM-D (openUTM distributed) is a component of openUTM which allows *distributed processing*. openUTM-D is an integral component of openUTM.

OSI-LPAP bundle

LPAP bundle for *OSI TP* partner applications.

OSI-LPAP partner

OSI-LPAP partners are the addresses of the *OSI TP partners* generated in openUTM. In the case of *distributed processing* via the *OSI TP* protocol, an OSI-LPAP partner for each partner application must be configured in the local application. The OSI-LPAP partner represents the partner application in the local application. During communication, the partner application is addressed by the name of the assigned OSI-LPAP partner and not by the application name or address.

OSI reference model

The OSI reference model provides a framework for standardizing communications in open systems. ISO, the International Organization for Standardization, described this model in the ISO IS7498 standard. The OSI reference model divides the necessary functions for system communication into seven logical layers. These layers have clearly defined interfaces to the neighboring layers.

OSI TP

Communication protocol for distributed transaction processing defined by ISO. OSI TP stands for Open System Interconnection Transaction Processing.

OSI TP partner

Partner of the UTM application that communicates with the UTM application via the OSI TP protocol.

Examples of such partners are:

– a UTM application that communicates via OSI TP
– an application in the IBM environment (e.g. CICS) that is connected via openUTM-LU62
– an application of the OpenCPIC carrier system of the openUTM client
– applications from other TP monitors that support OSI TP

outbound conversation (CPI-C)

See *outgoing conversation*.

outgoing conversation (CPI-C)

A conversation in which the local CPI-C program is the *initiator* is referred to as an outgoing conversation. In the X/Open specification, the term “outbound conversation” is used synonymously with “outgoing conversation”.

page pool

Part of the *KDCFILE* in which user data is stored.
In a *standalone application* this data consists, for example, of *dialog messages*, messages sent to *message queues*, *secondary memory areas*.
In a UTM cluster application, it consists, for example, of messages to *message queues, TLS*.

parameter area

Data structure in which a program unit passes the operands required for a UTM call to openUTM.

partner application

Partner of a UTM application during *distributed processing*. Higher communication protocols are used for distributed processing (*LU6.1*, *OSI TP* or LU6.2 via the openUTM-LU62 gateway).

postselection (BS2000 systems)

Selection of logged UTM events from the SAT logging file which are to be evaluated. Selection is carried out using the SATUT tool.

prepare to commit (PTC)

Specific state of a distributed transaction
Although the end of the distributed transaction has been initiated, the system waits for the partner to confirm the end of the transaction.

preselection (BS2000 systems)

Definition of the UTM events which are to be logged for the *SAT audit*. Preselection is carried out with the UTM-SAT administration functions. A distinction is made between event-specific, user-specific and job-specific (TAC-specific) preselection.

presentation selector

The presentation selector identifies a service access point to the presentation layer of the *OSI reference model* in the local system.

primary storage area

Area in main memory to which the *KDCS program unit* has direct access, e.g. *standard primary working area*, *communication area*.

print administration

Functions for *print control* and the administration of *queued output jobs*, sent to a printer.

print control

openUTM functions for controlling print output.

printer control LTERM

A printer control LTERM allows a client or terminal user to connect to a UTM application. The printers assigned to the printer control LTERM can then be administered from the client program or the terminal. No administration rights are required for these functions.

printer control terminal

This term has been superseded by *printer control LTERM*.

printer group (Unix systems)

For each printer, a Unix system sets up one printer group by default that contains this one printer only. It is also possible to assign several printers to one printer group or to assign one printer to several different printer groups.

printer pool

Several printers assigned to the same *LTERM partner*.

printer process (Unix systems)

Process set up by the *main process* for outputting *asynchronous messages* to a *printer group*. The process exists as long as the printer group is connected to the *UTM application*. One printer process exists for each connected printer group.

process

The openUTM manuals use the term “process” as a collective term for processes (Unix systems / Windows systems) and tasks (BS2000 systems).

processing step

A processing step starts with the receipt of a *dialog message* sent to the *UTM application* by a *client* or another server application. The processing step ends either when a response is sent, thus also terminating the *dialog step*, or when a dialog message is sent to a third party.

program interface for administration

UTM program interface which helps users to create their own *administration programs*. Among other things, the program interface for administration provides functions for *dynamic configuration*, for modifying properties and application parameters and for querying information on the configuration and the current workload of the application.

program unit

UTM *services* are implemented in the form of one or more program units. The program units are components of the *application program*. Depending on the employed API, they may have to contain KDCS, XATMI or CPIC calls. They can be addressed using *transaction codes*. Several different transaction codes can be assigned to a single program unit.

queue

See *message queue*.

queued output job

Queued output jobs are *asynchronous jobs* which output a message, such as a document, to a printer, a terminal or a transport system application.
Queued output jobs are processed by UTM system functions exclusively, i.e. it is not necessary to create program units to process them.

Quick Start Kit

A sample application supplied with openUTM (Windows systems).

redelivery

Repeated delivery of an *asynchronous message* that could not be processed correctly because, for example, the *transaction* was rolled back or the *asynchronous service* was terminated abnormally. The message is returned to the message queue and can then be read and/or processed again.

reentrant program

Program whose code is not altered when it runs. In BS2000 systems this constitutes a prerequisite for using *shared code*.

request

Request from a *client* or another server for a *service function*.

requestor

In XATMI, the term requestor refers to an application which calls a service.

resource manager

Resource managers (RMs) manage data resources. Database systems are examples of resource managers. openUTM, however, also provides its own resource managers for accessing message queues, local memory areas and logging files, for instance. Applications access RMs via special resource manager interfaces. In the case of database systems, this will generally be SQL and in the case of openUTM RMs, it is the KDCS interface.

restart

See *screen restart*,
see *service restart.*

RFC1006

A protocol defined by the IETF (Internet Engineering Task Force) belonging to the TCP/IP family that implements the ISO transport services (transport class 0) based on TCP/IP.

RSA

Abbreviation for the inventors of the RSA encryption method (Rivest, Shamir and Adleman). This method uses a pair of keys that consists of a public key and a private key. A message is encrypted using the public key, and this message can only be decrypted using the private key. The pair of RSA keys is created by the UTM application.

SAT audit (BS2000 systems)

Audit carried out by the SAT (Security Audit Trail) component of the BS2000 software product SECOS.

screen restart

If a *dialog service* is interrupted, openUTM again displays the *dialog message* of the last completed *transaction* on screen when the service restarts provided that the last transaction output a message on the screen.

SE manager

Web-based graphical user interface (GUI) for the SE series of Business Servers. SE Manager runs on the *management unit* and permits the central operation and administration of server units (with /390 architecture and/or x86 architecture), application units (x86 architecture), net unit and peripherals.

SE server

A Business Server from Fujitsu's SE series.

secondary storage area

Memory area secured by transaction logging and which can be accessed by the KDCS *program unit* with special calls. Local secondary storage areas (LSSBs) are assigned to one *service*. Global secondary storage areas (GSSBs) can be accessed by all services in a *UTM application*. Other secondary storage areas include the *terminal-specific long-term storage (TLS)* and the *user-specific long-term storage (ULS)*.

selector

A selector identifies a service access point to services of one of the layers of the *OSI reference model* in the local system. Each selector is part of the address of the access point.

semaphore (Unix systems / Windows systems)

Unix systems and Windows systems resource used to control and synchronize processes.

server

A server is an *application* which provides *services*. The computer on which the applications are running is often also referred to as the server.

server-server communication

See *distributed processing*.

server side of a conversation (CPI-C)

This term has been superseded by *acceptor*.

service

Services process the *jobs* that are sent to a server application. A service of a UTM application comprises one or more transactions. The service is called with the *service TAC*. Services can be requested by *clients* or by other servers.

service access point

In the OSI reference model, a layer has access to the services of the layer below at the service access point. In the local system, the service access point is identified by a *selector*. During communication, the *UTM application* links up to a service access point. A connection is established between two service access points.

service chaining (KDCS)

When service chaining is used, a follow-on service is started without a *dialog message* specification after a *dialog service* has completed .

service-controlled queue

Message queue in which the calling and further processing of messages is controlled by *services*. A service must explicitly issue a KDCS call (DGET) to read the message. There are service-controlled queues in openUTM in the variants *USER queue*, *TAC queue* and *temporary queue*.

service restart (KDCS)

If a service is interrupted, e.g. as a result of a terminal user signing off or a *UTM application* being terminated, openUTM carries out a *service restart*. An *asynchronous service* is restarted or execution is continued at the most recent *synchronization point*, and a *dialog service* continues execution at the most recent *synchronization point*. As far as the terminal user is concerned, the service restart for a dialog service appears as a *screen restart* provided that a dialog message was sent to the terminal user at the last synchronization point.

service routine

See *program unit*.

service stacking (KDCS)

A terminal user can interrupt a running *dialog service* and insert a new dialog service. When the inserted *service* has completed, the interrupted service continues.

service TAC (KDCS)

Transaction code used to start a *service*.

session

Communication relationship between two addressable units in the network via the SNA protocol *LU6.1*.

session selector

The session selector identifies an *access point* in the local system to the services of the session layer of the *OSI reference model*.

shared code (BS2000 systems)

Code which can be shared by several different processes.

shared memory

Virtual memory area which can be accessed by several different processes simultaneously.

shared objects (Unix systems / Windows systems)

Parts of the *application program* can be created as shared objects. These objects are linked to the application dynamically and can be replaced during live operation. Shared objects are defined with the KDCDEF statement SHARED-OBJECT.

sign-on check

See *system access control*.

sign-on service (KDCS)

Special *dialog service* for a user in which *program units* control how a user signs on to a UTM application.

single-step service

Dialog service which encompasses precisely one *dialog step*.

single-step transaction

Transaction which encompasses precisely one *dialog step*.

SOA

(Service-Oriented Architecture)
SOA is a system architecture concept in which functions are implemented in the form of re-usable, technically independent, loosely coupled *services*. Services can be called independently of the underlying implementations via interfaces which may possess public and, consequently, trusted specifications. Service interaction is performed via a communication infrastructure made available for this purpose.

SOAP

SOAP (Simple Object Access Protocol) is a protocol used to exchange data between systems and run remote procedure calls. SOAP also makes use of the services provided by other standards, XML for the representation of the data and Internet transport and application layer protocols for message transfer.

socket connection

Transport system connection that uses the socket interface. The socket interface is a standard program interface for communication via TCP/IP.

standalone application

See *standalone UTM application*.

standalone UTM application

Traditional *UTM application* that is not part of a *UTM cluster application*.

standard primary working area (KDCS)

Area in main memory available to all KDCS *program units*. The contents of the area are either undefined or occupied with a fill character when the program unit starts execution.

start format

Format output to a terminal by openUTM when a user has successfully signed on to a *UTM application* (except after a *service restart* and during sign-on via the *sign-on service*).

static configuration
Definition of the *configuration* during generation using the UTM tool *KDCDEF*.

SYSLOG file
See *system log file*.

synchronization point, consistency point
The end of a *transaction*. At this time, all the changes made to the *application information* during the transaction are saved to prevent loss in the event of a crash and are made visible to others. Any locks set during the transaction are released.

system access control
A check carried out by openUTM to determine whether a certain *user ID* is authorized to work with the *UTM application*. The authorization check is not carried out if the UTM application was generated without user IDs.

system log file
File or file generation to which openUTM logs all UTM messages for which SYSLOG has been defined as the *message destination* during execution of a *UTM application*.

TAC
See *transaction code*.

TAC queue
Message queue generated explicitly by means of a KDCDEF statement. A TAC queue is a *service-controlled queue* that can be addressed from any service using the generated name.

temporary queue
Message queue created dynamically by means of a program that can be deleted again by means of a program (see *service-controlled queue*).

terminal-specific long-term storage (KDCS)
Secondary storage area assigned to an *LTERM, LPAP* or *OSI-PAP partner* and which is retained after the application has terminated.

time-driven job
Job which is buffered by openUTM in a *message queue* up to a specific time until it is sent to the recipient. The recipient can be an *asynchronous service* of the same application, a *TAC queue*, a partner application, a terminal or a printer. Time-driven jobs can only be issued by KDCS *program units*.

timer process (Unix systems / Windows systems)

Process which accepts jobs for controlling the time at which *work processes* are executed. It does this by entering them in a job list and releasing them for processing after a time period defined in the job list has elapsed.

TNS (Unix systems / Windows systems)

Abbreviation for the Transport Name Service. TNS assigns a transport selector and a transport system to an application name. The application can be reached through the transport system.

Tomcat

see *Apache Tomcat*

transaction

Processing section within a *service* for which adherence to the *ACID properties* is guaranteed. If, during the course of a transaction, changes are made to the *application information*, they are either made consistently and in their entirety or not at all (all-or-nothing rule). The end of the transaction forms a *synchronization point*.

transaction code/TAC

Name which can be used to identify a *program unit*. The transaction code is assigned to the program unit during *static* or *dynamic configuration*. It is also possible to assign more than one transaction code to a program unit.

transaction rate

Number of *transactions* successfully executed per unit of time.

transfer syntax

With *OSI TP*, the data to be transferred between two computer systems is converted from the local format into transfer syntax. Transfer syntax describes the data in a neutral format which can be interpreted by all the partners involved. An *Object Identifier* must be assigned to each transfer syntax.

transport selector

The transport selector identifies a service access point to the transport layer of the *OSI reference model* in the local system.

transport system application

Application which is based directly on a transport system interface (e.g. CMX, DCAM or socket). When transport system applications are connected, the partner type APPLI or SOCKET must be specified during *configuration*. A transport system application cannot be integrated in a *distributed transaction*.

TS application

See *transport system application.*

typed buffer (XATMI)

Buffer for exchanging typed and structured data between communication partners. Typed buffers ensure that the structure of the exchanged data is known to both partners implicitly.

UPIC

Carrier system for openUTM clients. UPIC stands for Universal Programming Interface for Communication.

UPIC Analyzer

Component used to analyze the UPIC communication recorded with *UPIC Capture*. This step is used to prepare the recording for playback using *UPIC Replay.*

UPIC Capture

Used to record communication between UPIC clients and UTM applications so that this can be replayed subsequently (*UPIC Replay*).

UPIC client

The designation for openUTM clients with the UPIC carrier system.

UPIC Replay

Component used to replay the UPIC communication recorded with *UPIC Capture* and prepared with *UPIC Analyzer.*

user exit

This term has been superseded by *event exit.*

user ID

Identifier for a user defined in the *configuration* for the *UTM application* (with an optional password for *system access control*) and to whom special data access rights (*system access control*) have been assigned. A terminal user must specify this ID (and any password which has been assigned) when signing on to the UTM application. In BS2000 systems, system access control is also possible via *Kerberos*.

For other clients, the specification of a user ID is optional, see also *connection user ID*.

UTM applications can also be generated without user IDs.

user log file

File or file generation to which users write variable-length records with the KDCS LPUT call. The data from the KB header of the *KDCS communication area* is prefixed to every record. The user log file is subject to transaction management by openUTM.

USER queue

Message queue made available to every user ID by openUTM. A USER queue is a *service-controlled queue* and is always assigned to the relevant user ID. You can restrict the access of other UTM users to your own USER queue.

user-specific long-term storage

Secondary storage area assigned to a *user ID*, a *session* or an *association* and which is retained after the application has terminated.

USLOG file

See *user log file.*

UTM application

A UTM application provides *services* which process jobs from *clients* or other applications. openUTM is responsible for transaction logging and for managing the communication and system resources. From a technical point of view, a UTM application is a process group which forms a logical server unit at runtime.

UTM cluster application

UTM application that has been generated for use on a cluster and that can be viewed logically as a **single** application.
In physical terms, a UTM cluster application is made up of several identically generated UTM applications running on the individual cluster *nodes*.

UTM cluster files

Blanket term for all the files that are required for the execution of a UTM cluster application. This includes the following files:

- *Cluster configuration file*
- *Cluster user file*
- Files belonging to the *cluster page pool*
- *Cluster GSSB file*
- *Cluster ULS file*
- Files belonging to the *cluster administration journal**
- *Cluster lock file**
- Lock file for start serialization* (only in Unix systems and Windows systems)

The files indicated by * are created when the first node application is started. All the other files are created on generation using KDCDEF.

UTM-controlled queue

Message queues in which the calling and further processing of messages is entirely under the control of openUTM. See also *asynchronous job, background job* and *asynchronous message*.

UTM-D

See *openUTM-D*.

UTM-F

UTM applications can be generated as UTM-F applications (UTM fast). In the case of UTM-F applications, input from and output to hard disk is avoided in order to increase performance. This affects input and output which *UTM-S* uses to save user data and transaction data. Only changes to the administration data are saved.

In UTM cluster applications that are generated as UTM-F applications (APPLI-MODE=FAST), application data that is valid throughout the cluster is also saved. In this case, GSSB and ULS data is treated in exactly the same way as in UTM cluster applications generated with UTM-S. However, service data relating to users with RESTART=YES is written only when the relevant user signs off and not at the end of each transaction.

UTM message

Messages are issued to *UTM message destinations* by the openUTM transaction monitor or by UTM tools (such as *KDCDEF*). A message comprises a message number and a message text, which can contain *inserts* with current values. Depending on the message destination, either the entire message is output or only certain parts of the message, such as the inserts).

UTM page

A UTM page is a unit of storage with a size of either 2K, 4K or 8 K. In *standalone UTM applications*, the size of a UTM page on generation of the UTM application can be set to 2K, 4K or 8 K. The size of a UTM page in a *UTM cluster application* is always 4K or 8 K. The *page pool* and the restart area for the KDCFILE and *UTM cluster files* are divided into units of the size of a UTM page.

utmpath (Unix systems / Windows systems)

The directory under which the openUTM components are installed is referred to as *utmpath* in this manual.

To ensure that openUTM runs correctly, the environment variable UTMPATH must be set to the value of *utmpath*. On Unix systems, you must set UTMPATH before a UTM application is started. On Windows systems, UTMPATH is set on installation.

UTM-S

In the case of UTM-S applications, openUTM saves all user data as well as the administration data beyond the end of an application and any system crash which may occur. In addition, UTM-S guarantees the security and consistency of the application data in the event of any malfunction. UTM applications are usually generated as UTM-S applications (UTM secure).

UTM SAT administration (BS2000 systems)

UTM-SAT administration functions control which UTM events relevant to security which occur during operation of a *UTM application* are to be logged by *SAT*. Special authorization is required for UTM-SAT administration.

UTM system process

UTM process that is started in addition to the processes specified via the start parameters and which only handles selected jobs. UTM system processes ensure that UTM applications continue to be reactive even under very high loads.

UTM terminal

This term has been superseded by *LTERM partner*.

virtual connection

Assignment of two communication partners.

warm start

Start of a *UTM-S* application after it has terminated abnormally. The *application information* is reset to the most recent consistent state. Interrupted *dialog services* are rolled back to the most recent *synchronization point*, allowing processing to be resumed in a consistent state from this point (*service restart*). Interrupted *asynchronous services* are rolled back and restarted or restarted at the most recent *synchronization point*.

For *UTM-F* applications, only configuration data which has been dynamically changed is rolled back to the most recent consistent state after a restart due to a preceding abnormal termination.

In UTM cluster applications, the global locks applied to GSSB and ULS on abnormal termination of this node application are released. In addition, users who were signed on at this node application when the abnormal termination occurred are signed off.

WebAdmin

Web-based tool for the administration of openUTM applications via a Web browser. WebAdmin includes not only the full function scope of the *administration program interface* but also additional functions.

Web service

Application which runs on a Web server and is (publicly) available via a standardized, programmable interface. Web services technology makes it possible to make UTM program units available for modern Web client applications independently of the programming language in which they were developed.

WinAdmin

Java-based tool for the administration of openUTM applications via a graphical user interface. WinAdmin includes not only the full function scope of the *administration program interface* but also additional functions.

work process (Unix systems / Windows systems)

A process within which the *services* of a *UTM application* run.

workload capture & replay

Family of programs used to simulate load situations; consisting of the main components *UPIC Capture*, *UPIC Analyzer* and *Upic Replay* (on Unix and Windows systems) the utility program *kdcsort*. Workload Capture & Replay can be used to record UPIC sessions with UTM applications, analyze these and then play them back with modified load parameters.

WS4UTM

WS4UTM (**W**eb**S**ervices for open**UTM**) provides you with a convenient way of making a service of a UTM application available as a Web service.

XATMI

XATMI (X/Open Application Transaction Manager Interface) is a program interface standardized by X/Open for program-program communication in open networks.
The XATMI interface implemented in openUTM complies with X/Open's XATMI CAE Specification. The interface is available in COBOL and C. In openUTM, XATMI can communicate via the OSI TP, *LU6.1* and UPIC protocols.

XHCS (BS2000 systems)

XHCS (Extended Host Code Support) is a BS2000 software product providing support for international character sets.

XML

XML (eXtensible Markup Language) is a metalanguage standardized by the W3C (WWW Consortium) in which the interchange formats for data and the associated information can be defined.

Abbreviations

Please note: Some of the abbreviations used here derive from the German acronyms used in the original German product(s).

ACSE	Association Control Service Element
AEQ	Application Entity Qualifier
AES	Advanced Encryption Standard
AET	Application Entity Title
APT	Application Process Title
ASCII	American Standard Code for Information Interchange
ASE	Application Service Element
Axis	Apache eXtensible Interaction System
BCAM	Basic Communication Access Method
BER	Basic Encoding Rules
BLS	Binder - Loader - Starter (BS2000)
CCP	Communication Control Program
CCR	Commitment, Concurrency and Recovery
CCS	Coded Character Set
CCSN	Coded Character Set Name
CICS	Customer Information Control System
CID	Control Identification
CMX	Communication Manager in Unix Systems
COM	Component Object Model
CPI-C	Common Programming Interface for Communication
CRM	Communication Resource Manager
CRTE	Common Runtime Environment (BS2000)
DB	Database
DC	Data Communication
DCAM	Data Communication Access Method

DES	Data Encryption Standard
DLM	Distributed Lock Manager (BS2000)
DMS	Data Management System
DNS	Domain Name Service
DP	Distribted Processing
DSS	Terminal (Datensichtstation)
DTD	Document Type Definition
DTP	Distributed Transaction Processing
EBCDIC	Extended Binary-Coded Decimal Interchange Code
EJB	Enterprise JavaBeans™
FGG	File Generation Group
FHS	Format Handling System
FT	File Transfer
GSSB	Global Secondary Storage Area
HIPLEX®	Highly Integrated System Complex (BS2000)
HLL	High-Level Language
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IFG	Interactive Format Generator
ILCS	Inter-Language Communication Services (BS2000)
IMS	Information Management System (IBM)
IPC	Inter-Process Communication
IRV	International Reference Version
ISO	International Organization for Standardization
Java EE	Java Platform, Enterprise Edition
JCA	Java EE Connector Architecture
JDK	Java Development Kit
KAA	KDCS Application Area
KB	Communication Area
KBPRG	KB Program Area
KDCADMI	KDC Administration Interface
KDCS	Compatible Data Communication Interface

KTA	KDCS Task Area
LAN	Local Area Network
LCF	Local Configuration File
LLM	Link and Load Module (BS2000)
LSSB	Local Secondary Storage Area
LU	Logical Unit
MQ	Message Queuing
MSCF	Multiple System Control Facility (BS2000)
NB	Message Area
NEA	Network Architecture for BS2000 Systems
NFS	Network File System/Service
NLS	Native Language Support
OLTP	Online Transaction Processing
OML	Object Module Library
OSI	Open System Interconnection
OSI TP	Open System Interconnection Transaction Processing
OSS	OSI Session Service
PCMX	Portable Communication Manager
PID	Process Identification
PIN	Personal Identification Number
PLU	Primary Logical Unit
PTC	Prepare to commit
RAV	Computer Center Accounting Procedure
RDF	Resource Definition File
RM	Resource Manager
RSA	Encryption algorithm according to Rivest, Shamir, Adleman
RSO	Remote SPOOL Output (BS2000)
RTS	Runtime System
SAT	Security Audit Trail (BS2000)
SECOS	Security Control System
SEM	SE Manager
SGML	Standard Generalized Markup Language
SLU	Secondary Logical Unit

SM2	Software Monitor 2
SNA	Systems Network Architecture
SOA	Service-oriented Architecture
SOAP	Simple Object Access Protocol
SPAB	Standard Primary Working Area
SQL	Structured Query Language
SSB	Secondary Storage Area
SSO	Single Sign-On
TAC	Transaction Code
TCEP	Transport Connection End Point
TCP/IP	Transport Control Protocol / Internet Protocol
TIAM	Terminal Interactive Access Method
TLS	Terminal-Specific Long-Term Storage
TM	Transaction Manager
TNS	Transport Name Service
TP	Transaction Processing (Transaction Mode)
TPR	Privileged Function State in BS2000 (Task Privileged)
TPSU	Transaction Protocol Service User
TSAP	Transport Service Access Point
TSN	Task Sequence Number
TU	Non-Privileged Function State in BS2000 (Task User)
TX	Transaction Demarcation (X/Open)
UDDI	Universal Description, Discovery and Integration
UDS	Universal Database System
UDT	Unstructured Data Transfer
ULS	User-Specific Long-Term Storage
UPIC	Universal Programming Interface for Communication
USP	UTM Socket Protocol
UTM	Universal Transaction Monitor
UTM-D	UTM Variant for Distributed Processing in BS2000
UTM-F	UTM Fast Variant
UTM-S	UTM Secure Variant
UTM-XML	openUTM XML Interface

VGID	Service ID
VTSU	Virtual Terminal Support
WAN	Wide Area Network
WS4UTM	Web-Services for openUTM
WSDD	Web Service Deployment Descriptor
WSDL	Web Services Description Language
XA	X/Open Access Interface (X/Open interface for acess to the resource manager)
XAP	X/OPEN ACSE/Presentation programming interface
XAP-TP	X/OPEN ACSE/Presentation programming interface Transaction Processing extension
XATMI	X/Open Application Transaction Manager Interface
XCS	Cross Coupled System
XHCS	eXtended Host Code Support
XML	eXtensible Markup Language

Related publications

You will find the manuals on the internet at *http://manuals.ts.fujitsu.com*. You can order printed copies of those manuals which are displayed with an order number.

PDF files of all openUTM manuals are included on the openUTM Enterprise DVD with open platforms and on the openUTM WinAdmin DVD (for BS2000 systems).

openUTM documentation

openUTM
Concepts and Functions
User Guide

openUTM
Programming Applications with KDCS for COBOL, C and C++
Core Manual

openUTM
Generating Applications
User Guide

openUTM
Using openUTM Applications under BS2000 Systems
User Guide

openUTM
Using openUTM Applications under Unix Systems and Windows Systems
User Guide

openUTM
Administering Applications
User Guide

openUTM
Messages, Debugging and Diagnostics in BS2000 Systems
User Guide

openUTM
Messages, Debugging and Diagnostics in Unix Systems and Windows Systems
User Guide

openUTM
Creating Applications with X/Open Interfaces
User Guide

openUTM
XML for openUTM

openUTM Client (Unix systems)
for the OpenCPIC Carrier System
Client-Server Communication with openUTM
User Guide

openUTM Client
for the UPIC Carrier System
Client-Server Communication with openUTM
User Guide

openUTM WinAdmin
Graphical Administration Workstation for openUTM
Description and online help system

openUTM WebAdmin
Web Interface for Administering openUTM
Description and online help system

openUTM, openUTM-LU62
Distributed Transaction Processing
between openUTM and CICS, IMS and LU6.2 Applications
User Guide

openUTM (BS2000)
Programming Applications with KDCS for Assembler
Supplement to Core Manual

openUTM (BS2000)
Programming Applications with KDCS for Fortran
Supplement to Core Manual

openUTM (BS2000)
Programming Applications with KDCS for Pascal-XT
Supplement to Core Manual

openUTM (BS2000)
Programming Applications with KDCS for PL/I
Supplement to Core Manual

WS4UTM (Unix systems and Windows systems**)**
WebServices for openUTM

openUTM
Master Index

Documentation for the openSEAS product environment

BeanConnect
User Guide

JConnect
Connecting Java Clients to openUTM
User documentation and Java docs

WebTransactions
Concepts and Functions

WebTransactions
Template Language

WebTransactions
Web Access to openUTM Applications via UPIC

WebTransactions
Web Access to MVS Applications

WebTransactions
Web Access to OSD Applications

Documentation for the BS2000 environment

AID
Advanced Interactive Debugger
Core Manual
User Guide

BCAM
BCAM Volume 1/2
User Guide

BINDER
User Guide

BS2000 OSD/BC
Executive Macros
User Guide

BS2000
BLSSERV
Dynamic Binder Loader / Starter
User Guide

DCAM
COBOL Calls
User Guide

DCAM
Macros
User Guide

DCAM
Program Interfaces
Description

FHS
Format Handling System for openUTM, TIAM, DCAM
User Guide

IFG for FHS
User Guide

HIPLEX AF
High-Availability of Applications in BS2000/OSD
Product Manual

HIPLEX MSCF
BS2000 Processor Networks
User Guide

IMON
Installation Monitor
User Guide

MT9750 (MS Windows)
9750 Emulation under Windows
Product Manual

OMNIS/OMNIS-MENU (BS2000)
Functions and Commands
User Guide

OMNIS/OMNIS-MENU (BS2000)
Administration and Programming
User Guide

OSS (BS2000)
OSI Session Service
User Guide

RSO
Remote SPOOL Output
User Guide

SECOS
Security Control System
User Guide

SECOS
Security Control System
Ready Reference

SESAM/SQL
Database Operation
User Guide

openSM2
Software Monitor
Volume 1: Administration and Operation

TIAM
User Guide

UDS/SQL
Database Operation
User Guide

Unicode in BS2000/OSD
Introduction

VTSU
Virtual Terminal Support
User Guide

XHCS
8-Bit Code and Unicode Support in BS2000/OSD
User Guide

Documentation for the Unix system environment

CMX V6.0 (Unix systems)
Betrieb und Administration (only available in German)
User Guide

CMX V6.0
Programming CMX Applications
Programming Guide

OSS (UNIX)
OSI Session Service
User Guide

PRIMECLUSTER*TM*
Concepts Guide (Solaris, Linux)

openSM2
The documentation of openSM2 is provided in the form of detailed online help systems, which are delivered with the product.

Other publications

XCPI-C (X/Open)
Distributed Transaction Processing
X/Open CAE Specification, Version 2
ISBN 1 85912 135 7

Reference Model Version 2 (X/Open)
Distributed Transaction Processing
X/Open Guide
ISBN 1 85912 019 9

TX (Transaction Demarcation) (X/Open)
Distributed Transaction Processing
X/Open CAE Specification
ISBN 1 85912 094 6

XTAMI (X/Open)
Distributed Transaction Processing
X/Open CAE Specification
ISBN 1 85912 130 6

XML
W3C specification (www consortium)
Web page: *http://www.w3.org/XML*

Index