ADAUTM (Universal Transaction Monitor Support)

Software AG provides ADAUTM to coordinate Adabas database operations with the SNI BS2000 systems running the Universal Transaction Monitor (UTM).

This section describes ADAUTM function, installation and operation.

Common Synchronization Points

To safely operate SNI’s Universal Transaction Monitor (UTM) and a database, both systems must be able to run separately and coordinate their restarts. That is the only way to synchronize a database transaction running inside a UTM transaction.

ADAUTM makes it possible to have a common point of synchronization and thereby to set the whole transaction back to the last common synchronization point, if needed.

Both Adabas and UTM define points of synchronization (“sync points”):

- Adabas defines a sync point at the end of the database transaction;
- UTM defines a sync point at the end of a UTM transaction.

In some cases, these sync points are common to both Adabas and UTM. If an emergency restart should be necessary, both of the systems can set the interrupted transactions back to the last common sync point.

A database connection module (DBCON) is used for communication between UTM, its user program, and Adabas. The use of synchronization mode is controlled by a parameter in the UTM utility KDCDEF.

The UTM Transaction Concept

The UTM transaction concept is to be referenced to the database access here only.

Each UTM user program starts a transaction using an INIT call and ends with a PEND *nn*. When *nn* takes on the value RE or FI, UTM writes a synchronization point followed by an end-of-transaction for the database transaction. Normally, a user program determines the end of a database transaction. In an Adabas environment, the DBCON can take over that task. After an end-of-transaction ET for a database, no more calls to this database are allowed within the current UTM transaction.

Comments and Limitations

DBCON is invoked from UTM only if it occurs on a TP/DB transaction. If the last transaction within a UTM process is a pure TP transaction, the start parameter VG-ENDE=CL cannot be carried out at the end of the UTM process.

A 16-byte prefix of ET data is written at every sync point, in the order of synchronization. This prefix is transparent to the user. This means that when a UTM process contains more than one UTM transaction and the user wants the ET data to be available after the end of the process, the ET data must be written at every ET or with the last ET/CL for this DB transaction. Otherwise, the former ET data will be overwritten by the prefix.

A UTM process imposes only a few limitations on ADAUTM at this time. Those limitations refer only to the Adabas commands OP,ET, and BT. When using an OP command, the R option cannot be used if a process contains more than one UTM transaction because the record buffer is not available after the end of a database transaction. In case of ET and BT, the ISN hold option cannot be used.

The length of ET data that can be used is limited to 1984 bytes for users.

ADAUTM Functions

In the database transaction process, all calls to the database are transferred to DBCON to be checked. The start, update, and end of transaction are the most important parts.

At the start, a transaction is checked to see if the first call is an OP command and if so, whether the Additions 1 field of the Adabas control block contains a valid value. If the Additions 1 field does not contain a valid string, a generic one is created and inserted. If the first command is not an OP command, such a command is carried out implicitly; after that, the user’s order is handed over to Adabas.

All commands that have, either explicitly or implicitly, an attribute update are checked. The database ID of the first commands of that category of a transaction is marked as a update database for this transaction. That means that it is not allowed to carry out updates for more than one database within one database transaction.

Usually, the user initiates the end of transaction by issuing an ET/CL command. If that is not the case, the DBCON generates the UTM end of transaction. At this point, calls to the database are not permitted until the end of the UTM transaction (otherwise, the whole transaction must be set back to the last sync point).

When processing a UTM transaction, the main task is to handle the end of transaction procedure. Here, the synchronization with the database is done. In any case, ET data written for synchronization purposes is transparent to the user. Another listed function is the backout of a TP/DB transaction, as well as the functions with special positions, such as those to process status information. That special function is used during emergency restart before UTM is active.

- Establishing and Terminating a Connection
- User Call

- UTM End of Transaction
- UTM Transaction Backout
- Coordinated Restart
- User Exit 1

Establishing and Terminating a Connection

- Connecting to Adabas
- Disconnecting from Adabas

Connecting to Adabas

The initial link may be subdivided into the following parts:

- setting default values that are not defined at session startup, using the appropriate parameters

  All parameters that did not acquire a value during the start of the UTM session are now initialized by a default value. These defaults are described in a later section.

- creating the internal administration tables

  A table is built that contains information about the user and the user’s transactions. The table is installed as a common memory pool. The value of the parameter APPLI-ID is used as part of the pool’s name.

- making the database and the running mode available

Disconnecting from Adabas

When disconnecting from Adabas, the administration pool is closed in accordance with BS2000 conventions.

User Call

All user calls to Adabas pass through this function. Here, all the necessary checks are carried out. These checks refer to the synchronization of the TP and DB transaction, but not to the syntax of the Adabas commands or their buffers. The functionality is described with an example of a UTM transaction that contains accesses to a database.

At the beginning, DBCON determines whether the current database call is the first within the UTM transaction. If it is the first call, DBCON determines whether it is an OP (open) command call to initiate an Adabas transaction. If it is not an OP command, an internal OP is performed after the user’s call is completed.

DBCON also determines whether a command leads or could lead to an update in any form. If so, the database ID is recorded. With the occurrence of the first update-type command, DBCON marks the transaction as an update transaction. From that point, updates may only be done on this database during this transaction. Modifications to any other databases are rejected as errors, and the whole transaction is set back.

At the end of a database transaction, the ET/CL (end-of-transaction/close) commands are not passed immediately to Adabas, but instead are delayed until the end of the UTM transaction. This means that between the end of a database transaction and the end of the related UTM transaction, no more calls to the database can be carried out. Any attempt to execute such calls leads to a backout of that transaction. At the end of the UTM transaction, it is determined whether the user has determined the end of the transaction, or if DBCON must end it internally.

- User Open
- Internal Open

User Open

If the field ACBADD1 contains a value equal to zero or blank, DBCON tries to find an ET data ID in the administration pool (subsequent transaction within a UTM process). When available, this ID is taken over into ACBADD1; otherwise, an ID is built by the constant ADAU, the APPLI-ID, and an internal number. If a user exit is available, it gets control in order to modify the passed Adabas control block. The only change the exit can make is to modify the value in ACBADD1. After the DBCON gets back control, it overwrites the field with the old value if the exit has changed it to zero or blank, or the ET data ID was changed after the first transaction of this process.

Internal Open

If the first database call is not an OP command and ET-MODE=AUTO (the default) is activated, DBCON determines whether or not this is a subsequent transaction within the current UTM process. If it is, the call will be executed. If Adabas returns the call with a response code 9 (ADARSP009), an internal OP command is created and executed; after that, the user's call is repeated. At the beginning of a UTM process, an internal OP command is created, carried out, and after that, the user call. The internally generated OP command is passed to the user exit in both cases. See also the section *User Exit 1*.

UTM End of Transaction

The function PEND with one of the attributes RE, FI, or FC determines the end of transaction of UTM. The TP transaction end also contains an end of a DB transaction. That means that now the commands ET/CL are carried out. These were delivered either by the user, or must be created by the DBCON now. ET/CL commands always get an "E" option in the Adabas control block. Before Adabas is called, the DBCON passes the command through to the exit. After a successful execution of the command with a PEND-RE, data about this transaction is stored in the administration pool.

UTM Transaction Backout

This function is called by UTM if a user executes a RESET or PEND-ER call. The DBCON calls this for itself if the database is not available or an error occurs during the end of transaction procedure. Adabas return codes that occur during a transaction are passed directly back to the user and do not lead to a backout of the transaction.

Coordinated Restart

The coordinated restart contains two functions which are called during the phase of emergency restart and/or the normal end of the application. An emergency restart indicates an abnormal end of the last UTM session. At a new start of the application, one of two conditions may occur:

- there are open transactions
- there are no open transactions

While the UTM transaction is still open, it must be determined whether or not the database transaction is also still open. This occurs as described below.

In the case of emergency restart, the DBCON is called by the order Check DB Status for update transaction. At first, the ET data belonging to the update database are read. Then all read-only database operations are handled in the same way. The following situations may occur:

- if the synchronization data of UTM are equal to that read from the update database ET data header, the transaction is marked as finished for UTM.
- if the synchronization data of UTM are not equal, the complete DB transaction is backed out (BT) and UTM is informed with “transaction canceled”. If Adabas returns a response code unequal to 9 (ADARSP009) or 22 (ADARSP022), the process is stopped. The UTM utility KDCDEF may be required.

If there are no open transactions, UTM requires the DBCON to delete all information needed for synchronization. In addition, after a normal end of the last UTM session, the DBCON is required to delete all information.

User Exit 1

User Exit 1 can be used to influence the Adabas commands OP,ET, and CL.

However, User Exit 1 can only modify the commands which are offered; it cannot execute its own Adabas command because control is not passed back after executing such a command.

The Adabas control block and the record buffer may be modified depending on the commands being used.

The field E1PARB contains the address of the ET data area provided by ADAUTM. The address at E1PARB points to a field which contains the record length and is a part of a 16-byte header. With the exception of the length field, the header cannot be modified. The current length of the new data, plus 16 (the length of the header) must be set in the length field.

- Processing at the Start of a UTM Process
- Processing at the End of a UTM Update Transaction/Process
- User Exit Parameter List

Processing at the Start of a UTM Process

The read ET data option with the OP command cannot be inserted by the exit. The value that can be modified is the ET data ID in the Additions 1 field of the Adabas control block.

The ET data ID can only be modified at the beginning of a UTM process. It cannot be modified after a PEND RE if the DBCON has to issue an OP (after an Adabas error 9).

When using an OP command, only the fields E1PFB and E1PARB can be modified.

The ET data ID in the Additions 1 field can only be modified as follows:

- If bit E1PAR is on, the ET data ID cannot be modified.
- Bit E1POP and E1PIO allow the ET data ID to be modified; however, they cannot be changed to either zero or blank.

Processing at the End of a UTM Update Transaction/Process

An update transaction at the end of a UTM transaction or process can exchange ET/CL commands with each other and/or make ET data available.

At the end of a UTM transaction/process, the following conventions apply:

- User Exit 1 gets control only for an update transaction
- the contents of the Adabas command field may only be modified by the exit according to the start parameter VG-ENDE
- checks must not occur after returning from the exit
- the ET data ID may not be modified because its range of validity is the UTM process. ET data itself may be changed; the exit must provide the new data in its own area.
- the current length must be delivered in the field Length of the record buffer in the Adabas control block

If the exit sets the bit E1PFBRE, ADAUTM moves the data into its own area; otherwise, no user ET data are written.

A maximum of 1984 bytes of user ET data are allowed.

If there was an internal ET/CL, the field E1PARB contains low-value.

User Exit Parameter List

The following description contains the action fields of the parameter list. The complete list can be taken from the Assembler DSECT EX1PARM.

```
E1PTAS1
E1POP: User with its own OPEN command.
E1PAR: If the first command is not an OPEN after PEND RE, it is done after response code 9 (ADARSP009).
E1PIO: It is provided an internal OPEN command.
E1PFB
E1PFBCO : EXIT1 has modified the ACB during an OPEN
E1PFBCE : EXIT1 has modified the ACB during an ET/CL command.
E1PFBRE : EXIT1 has modified the RB during an ET/CL command.
E1PACB Address of ACB, either of the user or of ADAUTM.
E1PARB Address of RB; will be delivered by ADAUTM or is equal to zero using an internal ET/CL.
The exit 1 provides an address of its own record buffer, if an internal ET/CL occurs.
```

The exit is called by the standard BALR interface of the Assembler language. The following registers are used:

```
R01 : Address of the parameter list
R13 : Address of the save area
R14 : Return address
R15 : Start address of the modules
```

ADAUTM Installation

To make a coordinated restart possible, the DBCON (ADAUTM) must be specified in the UTM root module of the application. This is done by using the DATABASE statement in the UTM utility KDCDEF. The DATABASE statement defines the database in the source of the UTM root. In addition, the ADAUTM modules and the Adabas link module must be provided for the linkage editor’s run.

- Installation Procedure
- ADAUTM Parameters
- ADAUTM Parameter Example

Installation Procedure

The following procedure must be used to install ADAUTM.

To install ADAUTM:

1. Run KDCDEF with:

   ```
   DATABASE TYPE=DB,ENTRY=xyz
   ```

   —where ENTRY=*xyz* must match the name used in Adabas calls (usually “ADABAS”). If ENTRY is omitted or specified as “DB”, ENTRY=ADABAS is generated.

   The LIB parameter of the DATABASE statement does not apply for ADAUTM because ADAUTM must be statically linked with KDCROOT.

2. Assemble KDCROOT with the macro library AUT*nnn*.MAC where *nnn* is the ADAUTM version, revision, and SP level numbers).

3. Link the UTM application with:

   - AUTDB*mm* from AUT*nnn*.MOD where *mm* is the TM version (31, 32, 33, 34, and 40 are supported);
   - AUTNUC from AUT*nnn*.MOD; and
   - ADAUSER from ADA*nnn*.MOD

     Usually, KDCROOT will contain an entry point ADABAS for Adabas calls. ADAUSER also has a symbol ADABAS. It is therefore necessary to rename this symbol on binding ADAUSER. In $TSOLINK, specify:

     ```
     RENAME ADABAS,ADABASX
     INCLUDE ADAUSER,SAG.ADALIB
     ```

     Alternatively, if using the Binder, specify:

     ```
     INCLUDE-MODULES LIB=&ADA,ELE=ADAUSER,TYP=R
     RENAME-SYMBOL SYMBOL-NAME=ADABAS,NEW-NAME=ADABASX
     ```

4. Add ADAUTM parameters to the UTM startup parameters in the UTM startup procedure.

ADAUTM Parameters

This section describes the ADAUTM parameters.

ADAUTM Parameter Syntax

The syntax of an ADAUTM parameter is

```
.DB ADABAS parameter = value
```

If the prefix “.DB ADABAS” is omitted, UTM sends the message "K38 -Parameter Error-". The application ends abnormally.

ADAUTM Parameters

Parameter	Description	Possible Values	Default
DATABASE DA DB	Default database ID that is the update database for the current session.	1 - 65536	1
APPLI-ID AID	Required. The short name of the UTM application. The application ID identifies the UTM application running against Adabas. This value becomes part of the name of the common memory pool. That means that, in accordance with the parameter SCOPE, a default value may lead to an error of the UTM process administration.	1 - 9999	1
ET-MODE ETM	ADAUTM generates the OP, ET,CL commands if the application does not do it. ETM=AUTO will cause an OP command to be generated at the beginning of a database transaction, and an ET/CL command is automatically executed at the end of a UTM or database transaction, when necessary. Both are required for synchronization. ETM=MAN disables this function.	AUTO | MAN	AUTO
VG-ENDE VGE	Generate a CL instead of an ET command at the end of the UTM process. VGE=CL generates a CL command for the Adabas transaction if the UTM transaction finishes with PEND FI/FC; otherwise, an ET is generated.	CL | ET	ET

Parameter	Description	Possible Values	Default
UEX1	The name of the user exit that gets control during OP/ET/CL commands. Name of the module that is linked from a library with the link name DDLIB.	module name	(none)
SCOPE	Accessibility of the common memory pool for ADAUTM. SCOPE applies only for TASKTYPE “BATCHTASK” and “TP”. If TASKTYPE is “INTERACTIV”, the scope of the common memory pool is always “TASK”.	USERID | SYSTEM | TASK | USER_GROUP	USERID

UID-ADA	How the last 8 bytes of the Adabas communication ID are built by ADAUTM.
	
	UID-ADA=KCBENID: the KB field KCBENID is used
	
	UID-ADA=KCLOGTER: the KB field KCLOGTER is used
	
	UID-ADA=VGNR: the rightmost 4-bytes are built using the UTM conversation ID, as in releases of Adabas prior to 6.1, and the leftmost 4 bytes (the prefix) are specified by the UID-PRF parameter.
		KCBENID | KCLOGTER | VGNR	VGNR

UID-PRF	If the UID-ADA parameter value is “VGNR”, this parameter specifies the leftmost 4 bytes.	abcd	(none)

ADAUTM Parameter Example

The following is an example of ADAUTM parameter usage:

```
.DB ADABAS DB = 002 , AID = 80
.DB ADABAS VG-ENDE = CL , SCOPE = USERID
.DB ADABAS ET-MODE = AUTO
.DB ADABAS UEX1 = ADAEX1
```

ADAUTM Diagnostic Information

ADAUTM diagnostic information is written to UTM’s DB-DIAGAREA and to SYSOUT.

- UTM DB-DIAGAREA
- Messages to SYSOUT

- ADAUTM Message Codes

UTM DB-DIAGAREA

The general layout and use of the DB-DIAG-AREA area are described in the appropriate UTM manual. The DB-DIAGAREA is written as follows:

- *primary*: DB trace information provides essential information in an ADAUTM response code
- *secondary*: DB trace information includes more detail; see the DSECT DDBTRAC generated by the macro DDBTRAC for the layout of the information

Messages to SYSOUT

ADAUTM writes messages to SYSOUT in the format:

```
AUTxxxx date time OP=yyyy UID=abcdefgh DBID=nnnnn RSP=mmm
```

where

```
xxxx is the ADAUTM message code
yyyy is the UTM primary opcode; see the UTM macro DBCONPAA
abcdefgh is the last 8 bytes of the Adabas communication ID
nnnnn is the Adabas database ID
mmm is the Adabas nucleus response code
```

ADAUTM Message Codes

This section describes the ADAUTM messages.

- ADAUTM Message Format
- ADAUTM Messages
- Handling Adabas Nucleus Response Codes

ADAUTM Message Format

ADAUTM message codes are 4 characters long in the following format:

```
xnnn
```

where

x	is the type identifier. Possible values are: ● D (database): ADAUTM received a response code *nnn* from Adabas. With the exception of D148, Adabas response codes starting with D are not reported. ● I (internal): an error occurred in ADAUTM’s handling of ET data ● P (parameter): an error occurred in ADAUTM’s startup parameters ● S (system): ADAUTM received an error from a BS2000 executive macro; or, the installation is not correct ● U (user): ADAUTM received unsupported Adabas calls
nnn	is the Adabas response code received by ADAUTM; see value D above.

ADAUTM Messages

The following table contains the ADAUTM response codes and messages:

Response Code	Message / Description
I100	Internal area for ET data exhausted
P100	Statement format invalid
P101	Unknown parameter
P102	Invalid continuation
P103	Prefix “.DB ADABAS” not correct
P104	Invalid length of statement
P105	Invalid value
P120	Value not numeric
P121	Numeric value out of range
S100	ENAMP error AUTPOOL
S101	REQMP error AUTPOOL
S102	DISMP error AUTPOOL
S106	ESQUTM not linked AUTCONC
S107	TRACE open error AUTCONC
S108	ENQAR error AUTPOOL
S109	DEQAR error AUTPOOL
S110	ADALNQ not linked AUTMAIN
S111	Unsupported BS2000 version AUTMAIN
S112	Unsupported HSI version AUTMAIN
S113	HSITYPE error AUTMAIN
S114	TABLE error AUTMAIN
U100	More than four (4) DBIDs used in a single transaction.
U101	Update command issued between ET and end of UTM transaction.
U102	OP command issued, but ET or CL required.
U103	More than one update DBID used in a single transaction.

Handling Adabas Nucleus Response Codes

When ADAUTM receives response code 148 (ADARSP148) from the Adabas nucleus, a “DBMS down” condition is reported to UTM.