Restart and Recovery

- Recovery Records
- Suspect Transaction Records
- Adabas Resource Locks
- Handling unplanned outages
- Recovery with the CICS Resource Manager Interface
- Recovery with RRMS

Recovery Records

The TM records details of incomplete, prepared transactions in its recovery file. You can use the Online Services application to check for incomplete transactions in the system.

Caution:
The recovery file is a critical resource where the TM persists information about itself and the systems and applications it is working amongst. You must *not*

- change the Node ID of the Adabas System Coordinator daemon within which the TM runs as a service;
- change any of the following transaction manager parameter values: `TMETDATA, TMSYNCMGR`, or `TMTCIDPREF`.

If you wish to change these things you must make sure there are no incomplete transactions, all ET data are preserved so it is possible to recreate it again, etc.

Suspect Transaction Records

The TM uses suspect transaction records (STJ) in the recovery file to record all known details of incomplete transactions that have been purged from the system as a result of intervention by the operator or database administrator.

Incomplete transactions can be purged as follows:

- using the Stop Transaction function provided by Online Services, or
- as a result of a forced restart using the runtime parameter `TMRESTART`.

Online Services can be used to browse through the suspect transaction records..

Alternatively, you may use the sample program ATMSPRNT in the supplied JOBS library to produce a readable printout of the suspect transaction records. See Print Suspect Transaction Records for more information. Use the comments in the job when modifying it to conform to site requirements.

There is no automatic housekeeping of suspect transaction records. It is intended for emergency use only. The database administrator should purge these records from time to time, after making sure that the information contained in it is no longer required.

Adabas Resource Locks

Distributed transactions require more management to make sure they are conducted safely so they will have a tendency to take longer to complete than transactions that are not distributed. Distributed transactions take longer because Adabas must preserve information about held records in its Work when the transaction is prepared (for restart/recovery purposes). This can have knock-on effects on hold queue sizes etc, which need to be considered.

Handling unplanned outages

- Transaction Manager outage and failover
- Undetected Database Restarts

Transaction Manager outage and failover

Quite obviously, if the transaction management service fails (this usually implies the System Coordinator daemon has also failed) it should be restarted immediately. In most systems automatic restart management makes sure this happens immediately and the TM negotiates recovery with all databases, etc automatically and immediately without intervention. Therefore the general rule is to restart TM as soon as possible after a failure.

Unplanned outage in a single system

When TM becomes unavailable applications will run according to their *Continuous operation mode* setting. This may mean their transactions continue to run or that errors are issued. As soon as the TM returns restart/recovery takes place as appropriate and things quickly return to normal.

Unplanned outage in a multi-system

Multiple TMs running as peers in a group across multi-systems can collaborate and provide failover assistance to each other. In-flight transactions being managed by a TM that suffers outage are automatically and dynamically dispersed among TM peers based upon the failover capabilities of the applications that are running

Some applications support dynamic transaction routing in a multi-systems environment (such as CICS/PLEX). These applications can be configured to dynamically adopt failover by dispersing client sessions across peers in the same or other systems. The TM peers react automatically to client sessions being dispersed by negotiating transfer of transaction management to the TM that is local to where the client session is itself transferred (this demands a TM runs in each eligible system).

Other applications either do not support or are not configured for dynamic transaction routing so they will run according to their *Continuous operation mode* setting as previously described, and peer TM may takeover some clean-up if that is possible. Similar to single-systems, the general rule is that a failed TM service should be restarted immediately. When it is restarted it will negotiate its restart/recovery with its TM peers and continue normally.

Note:
Given that the general rule is to restart a failed TM immediately (and that most large systems do this automatically anyway) then a default action is that no TM peers will pre-empt any failover duties until 60 seconds have passed. This gives time for a failed TM to resume its duties without others doing unnecessary failover work.

Undetected Database Restarts

With or without Adabas Transaction Manager it can happen that:

- A client session has an in-flight transaction with Adabas.
- Adabas is recycled without the client session being made aware (for example it may be busy communicating with a different database).
- During this recycle Adabas undoes (backs out) the in-flight transaction.
- When Adabas returns, the client performs more modifications (not knowing the previous were undone).
- The client commits (applies) the latest modifications.

The result is inconsistency from the perspective of the client but not from the perspective of Adabas.

This situation can arise where a) ETID are not used and b) Adabas runs without `OPENRQ=YES`. In this situation the client session will not receive a response code 9 so the recycle of Adabas will go undetected.

To avoid this situation (with or without using Adabas Transaction Manager):

- Set the ADARUN parameter `OPENRQ=YES` for all databases, so that a restarted database will give response code 9, subcode 66, indicating that a new OP is required.
- Use ETIDs for all client sessions, so that "backout during system open" sets a pending response 9 for the affected session.

Recovery with the CICS Resource Manager Interface

For a system which has been configured to use the CICS Resource Manager Interface, the following recovery process occurs at CICS startup (or soon after):

1. The Adabas Transaction Manager CICS re-synchronization driver program (ATMRMIRS) obtains from the local TM a list of all prepared (but incomplete) transactions that were controlled by this CICS system.
2. CICS is then instructed to re-synchronize each of these transactions.
3. During this process, CICS indicates whether each of these transactions should be backed out or unconditionally committed.
4. When the last incomplete transaction has been processed, the TM writes a console message indicating that the re-synchronization process is complete.

In order to re-synchronize incomplete transactions in this way, CICS logging must be active and CICS must be warm started. If CICS logging is not in use or if CICS is cold started when there are incomplete transactions in the system, transaction integrity cannot be guaranteed. Consult your CICS documentation for more information.

Recovery with RRMS

When RRMS is already active at TM start (this is the normal case), the TM re-synchronizes in cooperation with RRMS to resolve any incomplete transactions that were under RRMS control.

When RRMS is unavailable at TM start, the TM issues a warning message to the console and waits until RRMS becomes available. Then it re-synchronizes.

If a critical component of RRMS becomes unavailable while the TM is operating, a warning message is issued to the console. In some cases, the TM is able to continue processing and initiates re-synchronization processing as soon as the missing component is reactivated.