# Syncpoint Processing Options

This section describes the available Syncpoint processing options.

- Overview of Syncpoint Processing Options
- Adabas Commands and External Syncpoints

---

# Overview of Syncpoint Processing Options

The normal rules of transactional programming under CICS and other TP systems are different from those that are familiar to Adabas programmers. Specifically, in a standard CICS application, a screen I/O normally means that pending changes are committed, and locks are freed, whereas Adabas allows a transaction to span screen I/Os. Indeed, Natural utilities such as SYSMAIN could behave differently when the ATM CICS Resource Manager Interface is in use, because they are written in such a way that they expect transactions to remain open across screen I/Os.

With the advent of ATM, this difference in programming styles led to two requirements for the ATM CICS RMI implementation:

- it should be possible for pending Adabas changes to remain uncommitted across pseudo-conversational task end;
- it should be possible to have ISNs released from held status, automatically, when a CICS syncpoint occurs.

The first requirement makes the introduction of ATM easier for sites that have taken advantage of the possibility of keeping Adabas transactions open across screen I/Os. It also allows Natural utilities to execute as before. The second brings Adabas behavior into line with standard behavior in CICS and other TP environments, and is ATM's normal mode of operation. The two are, of course, incompatible, and are therefore implemented as options through the `TransactionModel` client runtime control.

### Message-based Transaction Model

This is the default option, and defines ATM's standard way of processing syncpoints. Processing of a message always terminates with a syncpoint. Normally this means that a screen I/O causes pending changes to be committed. If an external transaction coordinator is in control of a transaction, an `ET` or `CL` command will trigger a commit syncpoint by the external coordinator; a `BT` command, or an `OP` to a changed database, will trigger a rollback syncpoint by the external coordinator. Held ISNs will be released, or will remain held, according to the setting of the extended hold option, and any P or M command options. An unsolicited syncpoint from an external transaction coordinator will cause ATM to commit or back out pending changes to all databases and release all held ISNs. In any case, the commit process will be synchronized for all `DTP=RM` databases; changes to other databases will be committed or backed out after completion of the syncpoint processing.

### Dynamic Transaction Model

This option allows existing Adabas applications (including Natural utilities such as SYSMAIN) to execute under the ATM CICS RMI without being affected by the unsolicited CICS syncpoints that occur at pseudo-conversational task end. ATM will honor all rollback syncpoints, whether they originate from `BT` or `OP` commands, from a CICS command, or from CICS itself. ATM will also honor commit syncpoints triggered by `ET` or `CL` commands, but it will ignore other commit syncpoints.

This option is appropriate for the Natural utilities, and for applications which keep transactions open across screen I/O operations. However, it might not be suitable for applications that execute under the CICS RMI and change both Adabas and non-Adabas resources. If such an application encounters an unsolicited commit syncpoint (when a screen I/O occurs, for example), its non-Adabas changes will be committed, but the Adabas changes will remain uncommitted until an `ET` or `CL` command is executed. That is, all syncpoints will continue to affect other resource managers (such as DB2) exactly as they did before, regardless of the behavior of ATM.

**Note:**
If you use the message-based transaction model under the CICS RMI, and execute Natural using an `ADAMODE` setting that causes Natural to execute two parallel Adabas sessions, you must make sure that Natural's system session always begins with an `OP` command, otherwise response code 9, subcode 97, will occur frequently. You can do this by always supplying a non-blank ETID or by using Natural's `DBOPEN` parameter.

The `Transaction Model` setting has no effect for IMS/TM systems whose transactions are coordinated by RRMS, and whose local ATM runs with `TMSYNCMGR=RRMS`.

# Adabas Commands and External Syncpoints

ATM allows Adabas changes to be committed synchronously with non-Adabas changes, by interacting with external transaction coordinators. When an external transaction coordinator takes a syncpoint, ATM ensures that changed Adabas databases take part in the commit or rollback operation. By default, ATM also causes the external transaction coordinator to take a syncpoint whenever it detects that pending changes are to be committed or backed out.

Usually, this means that every `ET` or `CL` command causes an external commit syncpoint, and any `BT` command causes an external rollback syncpoint to take place. However, there are cases in which this behavior might be different from what is required. For example, consider a CICS environment in which a Cobol program changes DB2, then starts a Natural session, expecting Natural to return control before a decision to commit or back out is taken. Natural can issue an `ET` command during LOGON processing, and `CL` commands at session end. By default, each of these commands (if issued under the same Communications ID as the Cobol program's commands) would cause a CICS SYNCPOINT to take place, and the first of these would cause the pending DB2 changes to be committed.

Client runtime controls are provided which can be used to change this behavior. See the descriptions of the `GenerateExternalSyncpoint` client runtime controls. In the example described above, it would be appropriate to specify NO for `GenerateExternalSyncpointOnCL` and `GenerateExternalSyncpointOnET`, so that the `CL` and `ET` commands generated by Natural would not cause a CICS SYNCPOINT.