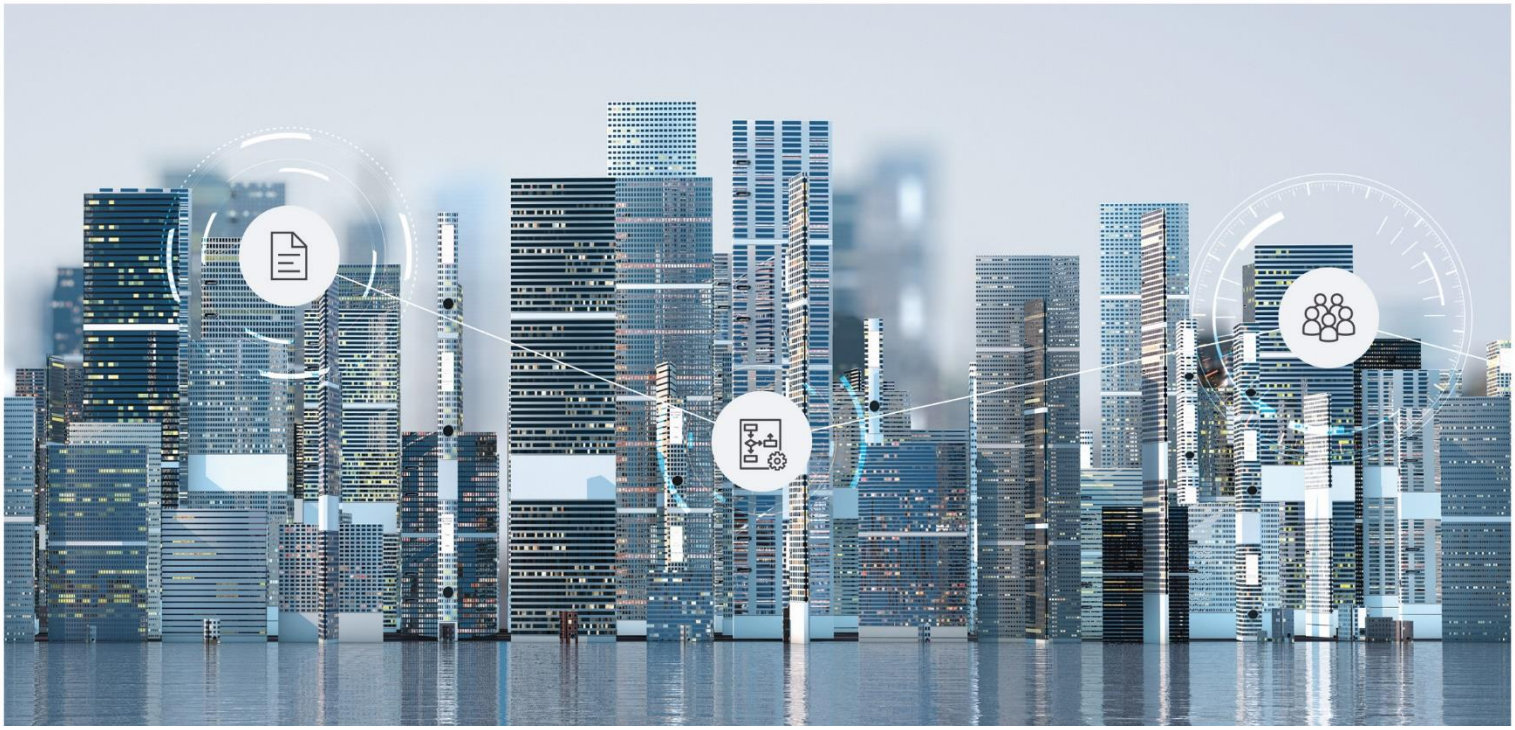
Fabasoft



White Paper

Fabasoft Folio Distributed Transaction Manager

Fabasoft Folio 2022 Update Rollup 1

Contents

1 Introduction 4

2 Software Requirements 4

3 Functionality and Architecture 4

3.1 Functionality 4

3.2 Architecture 5

3.3 Log File 6

3.4 Example: Recovery 8

4 Settings for Using Fabasoft Folio Distributed Transaction Manager 8

5 Monitoring of Fabasoft Folio Distributed Transaction Manager 10

5.1 Event Log and Fabasoft Folio Distributed Transaction Manager Log Files 10

5.2 Recovery Mode 10

5.3 Performance Indicators 10

1 Introduction

This document describes the functionality and use of Fabasoft Folio Distributed Transaction Manager.

If a transaction is performed on several Fabasoft Folio COO Services it is called distributed transaction. A distributed transaction is a transaction, in which different resources are involved. Distributed transactions must, like all other transactions too, fulfill so-called “ACID”-properties. This means, that each transaction has to be atomic, consistent, isolated and durable. A transaction manager is used to ensure this behavior also for distributed transactions.

Up to Fabasoft software products version 6.1 P2 the Microsoft Distributed Transaction Coordinator (MSDTC) was used in a Microsoft Windows environment, independent of using Microsoft SQL Server or Oracle Database. In a Linux environment with Oracle Database the Fabasoft Folio Distributed Transaction Manager (FSCDTM) was used.

From Fabasoft software products version 6.1 SP1 when using Oracle Database the Fabasoft Distributed Transaction Manager is used (also in a Microsoft Windows system environment). For the combination of Microsoft Windows and Microsoft SQL server the Microsoft Distributed Transaction Coordinator is used further on.

2 Software Requirements

System environment: All information contained in this document implicitly assumes a Microsoft Windows environment or Linux environment.

Supported platforms: For detailed information on supported operating systems and software see the software product information on the Fabasoft distribution media.

Descriptions in this document are based on the following software:

- Oracle Database 19c Enterprise Edition (from 19.3.0.0.0) for Linux x86-64 with RAC

3 Functionality and Architecture

3.1 Functionality

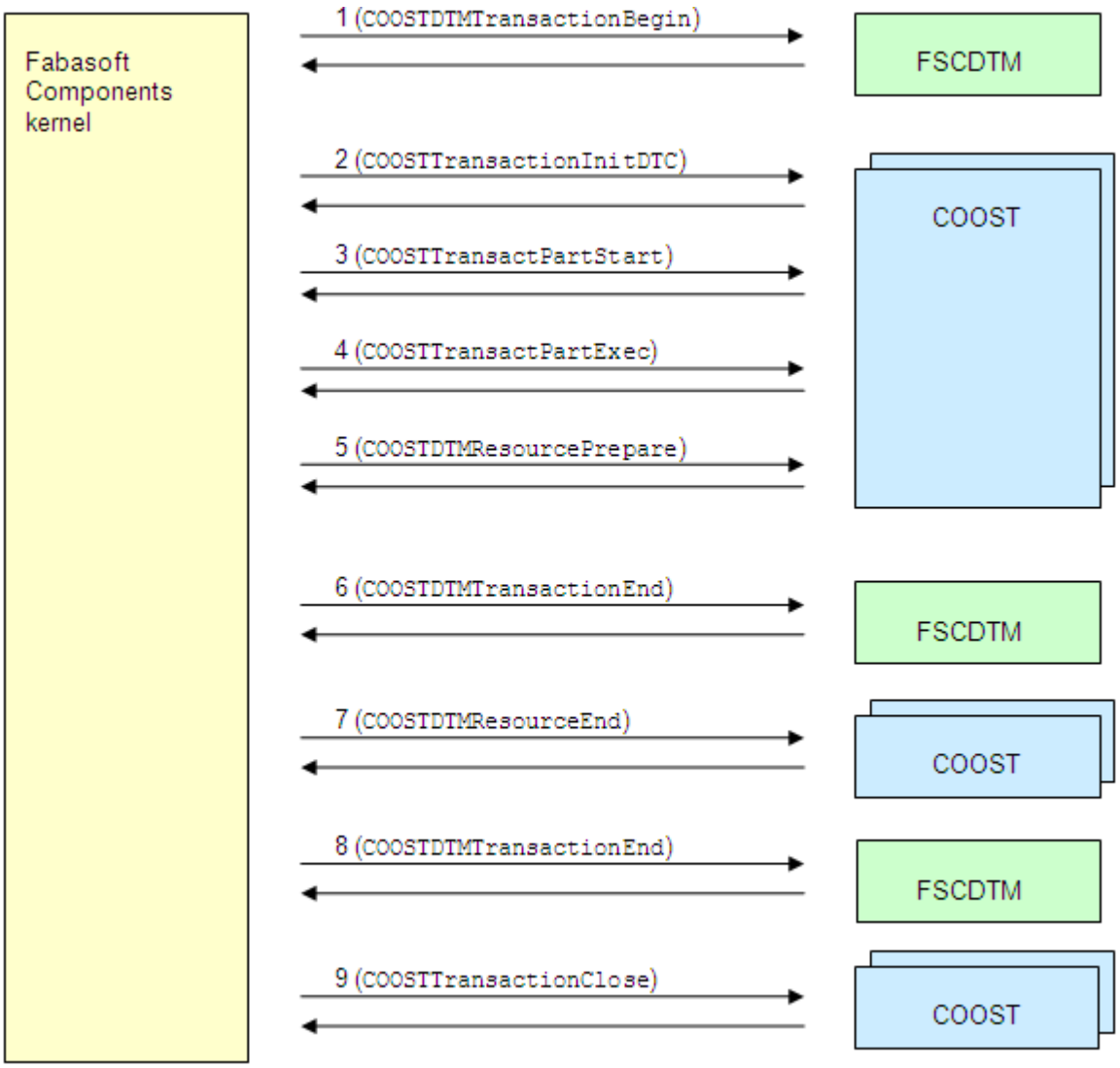
Exactly one Fabasoft Folio Distributed Transaction Manager is needed for one distributed transaction. In a Fabasoft Folio installation each Fabasoft Folio COO Service can also be a Fabasoft Folio Distributed Transaction Manager (1:1 relationship).

The Fabasoft Folio Kernel is responsible for performing distributed transactions. At the beginning of a distributed transaction the Fabasoft Folio Kernel chooses exactly that Fabasoft Folio COO Service as Fabasoft Folio Distributed Transaction Manager which is involved in the distributed transaction and has the highest instance number.

Example: If the Fabasoft Folio COO Services with the instance numbers 1, 2 and 4 are involved in the distributed transaction, the Fabasoft Folio Distributed Transaction Manager of the Fabasoft Folio COO Service with the instance number 4 is used.

The Fabasoft Folio Distributed Transaction Manager is implemented on the basis of the XA protocol.

3.2 Architecture



(1) `COOSTDTMTransactionBegin`
Starts a distributed transaction. Input parameter: `instancelist`; Output parameter: `XID`

(2) `COOSTTransactionInitDTC`
Starts a transaction with MSDTC/FSCDTM

(3) `COOSTTransactPartStart`
Starts a new part of a transaction.

(4) `COOSTTransactPartExec`
Runs the defined part of a transaction since the last call of `COOSTTransactPartStart`.

(5) `COOSTDTMResourcePrepare`
Prepares an involved resource for a "Commit".

(6) `COOSTDTMTransactionEnd`
Informs the Fabasoft Folio Distributed Transaction Manager about a successful "Prepare"

(7) `COOSTDTMResourceEnd`
Closes the local transaction of a resource with "Commit" or "Rollback". Update of the cache.

(8) `COOSTDTMTransactionEnd`
Informs the Fabasoft Folio Distributed Transaction Manager about a successful "Commit" or "Rollback"

(9) `COOSTTransactionClose`
Transaction is marked in the service as "closed". RPC-transaction will be closed.

3.3 Log File

In order to close open distributed transactions accordingly after a system crash, a log about all performed transactions is created. The path to this log file can be configured explicitly for each Fabasoft Folio COO Service. The Fabasoft Folio Distributed Transaction Manager performs periodically a "Recover" of open transactions.

The filename of the log file is created the following way:
`FSCDTM_<cooservice>.dtm` (e.g. `FSCDTM_COOService_1_1190_2.dtm`)

The log file starts with a header containing among others the creation date of the log:

Example: `FSCDTM 1.0 Transaction Log 2006-07-25T09:44:32`

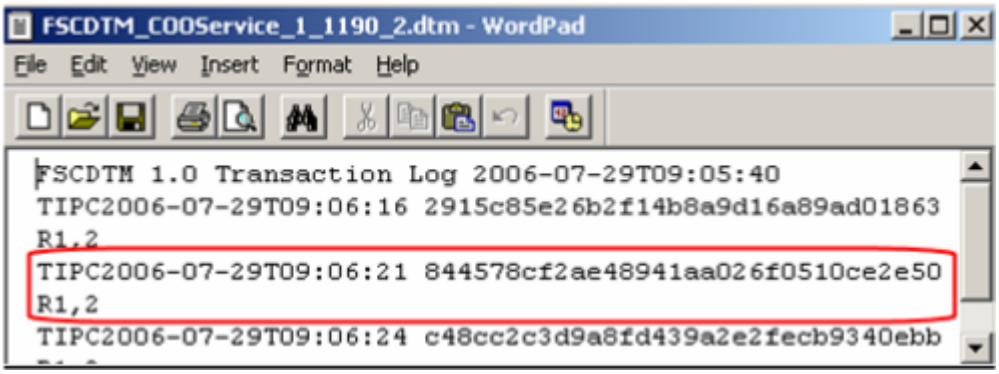
After this entry a list of distributed transactions follows. Each entry from the list can contain the following information:

- Status flags about the transaction
  - T...transaction
  - I...initialized (transaction was started)
  - P...prepared (transaction was prepared)
    or
    O...transaction was prepared but one or more resources answered with "read-only"
  - C...committed (transaction has been finished)
    or
    R...Rolled back (transaction has been reset)

A transaction is finished if it is closed or reset.

- Start time of the transaction
  UTC time displaying when the transaction has been started. It is also used to calculate the timeouts.
- XID
  The global identification of the transaction.
- Involved resources
  All resources (instance numbers of Fabasoft Folio COO Services) involved in the transaction. The resources are written in an own line.

Example for a log file:



The color bordered part represents an entry for a transaction in the log file. The entry displays a transaction (T), which was initialized (I), prepared (P) and finally closed (C). Then the start time of the transaction is displayed followed by the global identification, the XID. In the line below the involved resources are mentioned. In this case these are the Fabasoft Folio COO Services with the instance numbers 1 and 2.

Syntax of the log file:

```
Log =
  EntryWelcome { EntryTransaction EntryResource { EntryResource } }.
EntryWelcome =
  Prolog Blank TimeStamp Padding.
Prolog =
  "FSCDTM 1.0 Transaction Log".
EntryTransaction =
  "T" Initiated Prepared Completed TimeStamp Blank GlobalTransationId Padding
Initiated =
  1  x (I | Blank).
Prepared =
  1  x (P | O | Blank).  -- P prepared, O prepared with read-only votes
Completed =
  1  x (C | R | Blank).  -- C committed, R rolled back
TimeStamp =
  19 x (0123456789T:-). -- Date formatted YYYY-MM-DDThh:mm:ss
GlobalTransationId =
  32 x (0-9A-F).  -- typedef struct { int gtrid[4];  } DTMGTRID;   BinHex coded
EntryResource =
  "R" Resources Padding.
Resources =
  Resource { ',' Resource };
Resource =
  (0-9)*.
Blank =
  ' '.
```

```
Padding =
  Blank till LENGTH_LOGENTRY-1 bytes are reached, followed by '\n'
-- Required minimal size of entry: 1 + 1 + 1 + 1 + 19 + 1 + 32 = 56 bytes
```

3.4 Example: Recovery

Assuming that a distributed transaction has successfully prepared all involved transaction parts. Suddenly at "Commit" an involved database is not available anymore.

In the log file the following entry is created:

```
TIP 2006-07-26T10:15:34 9d080d46066d9145adbe4f55d2cb3765
R1,2
```

The first four letters show that the transaction (T) was initialized (I) and prepared (P), but neither a rollback (R) nor a commit (C) were performed.

It is also possible to access information about open distributed transactions via the `DBA_2PC_PENDING` view in the Oracle database instance:

```
SQL> select global_tran_id, state from dba_2pc_pending;

GLOBAL_TRAN_ID                                   STATE
------------------------------------------------ ----------------
17100.02000000.9D080D46066D9145ADBE4F55D2CB3765  prepared
```

Comparing the XID in the log file and the one in the `DBA_2PC_PENDING` view shows, that it is the same transaction and that this one has the "prepared" state in the database too.

The "commit" failed during the transaction, so a periodically running recovery thread tries to close the transaction now. After a successful transaction closing the entry in the log file changes:

```
TIC2006-07-26T10:15:34 9d080d46066d9145adbe4f55d2cb3765
R1,2
```

Now the "C" shows, that the transaction has been closed.

The `DBA_2PC_PENDING` view does not contain entries about open distributed transactions any longer.

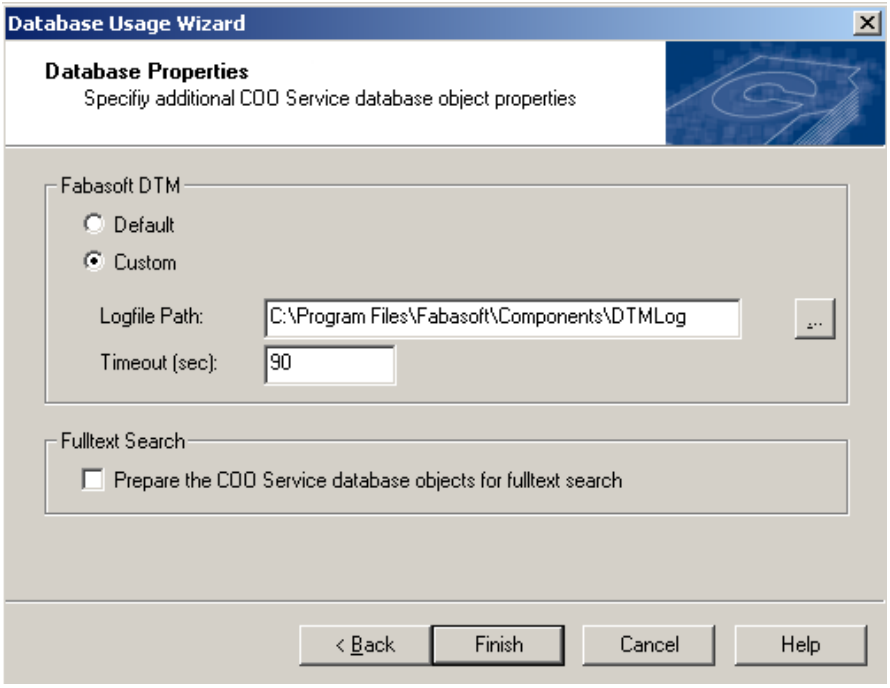
```
SQL> select global_tran_id, state from dba_2pc_pending;

Es wurden keine Zeilen ausgewählt
```

4 Settings for Using Fabasoft Folio Distributed Transaction Manager

The Fabasoft Folio Distributed Transaction Managers is used automatically when using Oracle Database.

It can be configured where the log file of Fabasoft Folio Distributed Transaction Manager should be placed and how long the Fabasoft Folio Distributed Transaction Manger timeout should be.

These values can be specified when creating a Fabasoft Folio COO Service. If the values are not specified, the default values are used.



After an installation it is still possible to change the values manually in the registry.

- Path: `HKEY_LOCAL_MACHINE\SOFTWARE\Fabasoft\Fabasoft Components Server\Domain <x.y>\Service <z>\Datasource\Default`
  Note: When using Linux the registry path is mapped to a directory structure in `/etc/fabasoft/settings`. For more information about managing registry keys under Linux consult the white paper "Fabasoft Folio Environment Variables".
- `FSCDTM_TXLOG_PATH`:
  path, where the log file is placed (default value: value from `COOROOT`)
- `FSCDTM_TXLOG_TIMEOUT`:
  Timeout in seconds (default value: 90)
- `SesTM`
  Session Timeout (default value: 99; if the `FSCDTM_TXLOG_TIMEOUT` is set during Fabasoft Folio COO Service creation, the `SesTm` is automatically set too (`SesTm` = `FSCDTM_TXLOG_TIMEOUT +10`))

Note:

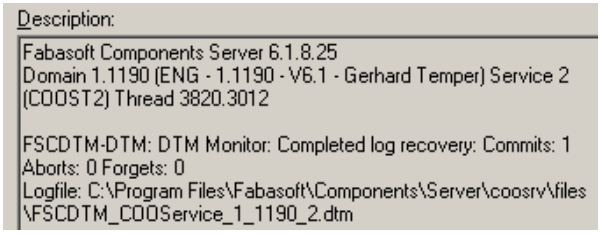
- Using a cluster, the log files have to be available on the appropriate cluster resources (Fabasoft Folio Distributed Transaction Manager log files may exist only once).
- For the `DISTRIBUTED_LOCK_TIMEOUT` oracle database parameter a value of 300 is recommended. If this value is not used (default value 60) errors ("ORA-02049: timeout: distributed transaction waiting for lock", probably also "ORA-01591: lock held by in-doubt distributed transaction") are possible.
  The timeouts should be configured to ensure:
  `FSCDTM_TXLOG_TIMEOUT < SesTm < DISTRIBUTED_LOCK_TIMEOUT`

5 Monitoring of Fabasoft Folio Distributed Transaction Manager

5.1 Event Log and Fabasoft Folio Distributed Transaction Manager Log Files

Errors, problems and information about the work of Fabasoft Folio Distributed Transaction Manager create an entry in the event log.

After a "Recover" the following entry is created:



When problems occur, the log file of Fabasoft Folio Distributed Transaction Manager can be opened. The transaction list shows which transactions are closed or still open.

Attention: The log file must not be modified manually.

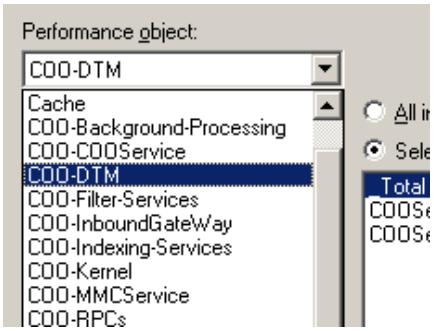
5.2 Recovery Mode

Starting a Fabasoft Folio COO Service in the "recovery mode" the log file of the Fabasoft Folio COO Service is deleted. An appropriate note is displayed.

Thus, before starting in "recovery mode" it has to be ensured that the data is consistent. After the start in the "recovery mode" the log file does not exist any longer and a transaction recovery is not possible anymore.

5.3 Performance Indicators

To monitor the Fabasoft Folio Distributed Transaction Manager during operation different performance indicators and SNMP performance indicators are available:



For the "COO-DTM" performance object the following performance indicators exist:

- "Transactions/sec"
  Number of started distributed transactions per second.
- "Active Transactions"
  This value indicates the number of active distributed transactions (recovered transactions are also included in this value).
- "Transactions committed/sec"
  Number of successfully realized distributed transactions per second.

- “Transactions rolled back/sec”
  Number of rolled back distributed transactions per second.
- “Recovered (commit) Transactions/sec”
  Number of successfully closed distributed transactions after recovery per second.
- “Recovered (rolled back) Transactions/sec”
  Number of rolled back distributed transactions after recovery per seconds. “