[MS-TIPP]:
Transaction Internet Protocol (TIP) Extensions

Revision Summary

Date	Revision History	Revision Class	Comments
07/20/2007	0.1	Major	MCPP Milestone M5 Initial Availability
09/28/2007	1.0	Major	Updated and revised the technical content.
10/23/2007	2.0	Major	Added new content.
11/30/2007	3.0	Major	Updated and revised the technical content.
01/25/2008	3.0.1	Editorial	Revised and edited the technical content.

Table of Contents

1 Introduction 6
1.1 Glossary 6
1.2 References 7
1.2.1 Normative References 7
1.2.2 Informative References 8
1.3 Protocol Overview (Synopsis) 8
1.3.1 Protocol Roles 8
1.3.1.1 The TIP Application Role 9
1.3.1.2 The Transaction Manager Role 10
1.3.1.2.1 The TIP Superior Transaction Manager Facet 10
1.3.1.2.2 The TIP Subordinate Transaction Manager Facet 10
1.3.1.2.3 The TIP Transaction Manager Communicating with an Application Facet 10
1.3.2 Common Scenarios 11
1.3.2.1 Starting and Completing a Transaction 11
1.3.2.2 Pulling a Transaction 12
1.3.2.3 Pushing a Transaction 13
1.3.2.4 TIP Two-Phase Commit 14
1.4 Relationship to Other Protocols 16
1.5 Prerequisites/Preconditions 16
1.6 Applicability Statement 16
1.7 Versioning and Capability Negotiation 16
1.8 Vendor-Extensible Fields 17
1.9 Standards Assignments 17

2 Messages 18
2.1 Transport 18
2.2 Message Syntax 18
2.2.1 ALREADYPUSHED 19
2.2.2 BEGUN 19
2.2.3 IDENTIFY 19
2.2.4 PULL 19
2.2.5 PUSH 19
2.2.6 PUSHED 19
2.2.7 QUERY 19
2.2.8 RECONNECT 19

3 Protocol Details 20
3.1 Common Details 20
3.1.1 Abstract Data Model 20
3.1.1.1 Data Elements 20
3.1.1.2 TIP Connection Object 21
3.1.1.3 TIP Connection Management Operations 21
3.1.1.3.1 GetTipConnection Operation 21
3.1.1.3.2 GetTipConnectionFromAddress Operation 22
3.1.1.3.3 HasPartnerTransaction Operation 23
3.1.1.3.4 FreeTipConnection Operation 23
3.1.1.3.5 TerminateTipConnection Operation 23
3.1.1.4 TIP Command Object 23
3.1.1.5 Transaction Identifier Converter Operations 24
3.1.1.5.1 Convert TIP Transaction Identifier to Transaction Identifier Operation 24
3.1.1.5.2 Convert Transaction Identifier to TIP Transaction Identifier Operation 24
3.1.1.6 Primary State Transition Table 24

3.1.1.7 Secondary State Transition Table 25
3.1.2 Timers 25
3.1.3 Initialization 25
3.1.4 Higher-Layer Triggered Events 26
3.1.5 Message Processing Events and Sequencing Rules 26
3.1.5.1 Receiving BEGUN TIP Command 26
3.1.5.2 Receiving CANTMULTIPLEX TIP Command 26
3.1.5.3 Receiving CANTTLS TIP Command 26
3.1.5.4 Receiving IDENTIFIED TIP Command 26
3.1.5.5 Receiving IDENTIFY TIP Command 27
3.1.5.6 Receiving MULTIPLEX TIP Command 28
3.1.5.7 Receiving MULTIPLEXING TIP Command 28
3.1.5.8 Receiving NEEDTLS TIP Command 29
3.1.5.9 Receiving NOTBEGUN TIP Command 29
3.1.5.10 Receiving TLS TIP Command 29
3.1.5.11 Receiving TLSING TIP Command 29
3.1.6 Timer Events 29
3.1.7 Other Local Events 29
3.1.7.1 Invalid TIP Command Event 29
3.1.7.2 Transport Events 30
3.1.7.2.1 Received Message 30
3.1.7.2.2 Transport Connection Down 30
3.2 TIP Superior Transaction Manager Facet Details 30
3.2.1 Abstract Data Model 30
3.2.1.1 TIP Superior Transaction Manager Facet State Transition Table 31
3.2.2 Timers 33
3.2.3 Initialization 33
3.2.4 Higher-Layer Triggered Events 33
3.2.4.1 Push Transaction 33
3.2.5 Message Processing Events and Sequencing Rules 34
3.2.5.1 Receiving ABORTED TIP Command 34
3.2.5.2 Receiving ALREADYPUSHED TIP Command 35
3.2.5.3 Receiving COMMITTED TIP Command 35
3.2.5.4 Receiving NOTPUSHED TIP Command 35
3.2.5.5 Receiving NOTRECONNECTED TIP Command 36
3.2.5.6 Receiving PREPARED TIP Command 36
3.2.5.7 Receiving PULL TIP Command 37
3.2.5.8 Receiving PUSHED TIP Command 38
3.2.5.9 Receiving QUERY TIP Command 39
3.2.5.10 Receiving READONLY TIP Command 40
3.2.5.11 Receiving RECONNECTED TIP Command 40
3.2.5.12 Receiving ERROR TIP Command 40
3.2.6 Timer Events 41
3.2.7 Other Local Events 41
3.2.7.1 Invalid TIP Command Event 41
3.2.7.2 Process Error 41
3.2.7.3 Events Signaled by the Core Transaction Manager Facet 42
3.2.7.3.1 Begin Commit 42
3.2.7.3.2 Begin Phase One 43
3.2.7.3.3 Begin Rollback 43
3.2.7.3.4 Create Subordinate Enlistment Failure 44
3.2.7.3.5 Create Subordinate Enlistment Success 44
3.2.7.4 Transport Events 45
3.2.7.4.1 Transport Connection Down 45
3.3 TIP Subordinate Transaction Manager Facet Details 45

3.3.1 Abstract Data Model 46
3.3.1.1 TIP Subordinate Transaction Manager Facet State Transition Table 46
3.3.2 Timers 48
3.3.2.1 Query Timer 48
3.3.3 Initialization 48
3.3.4 Higher-Layer Triggered Events 48
3.3.4.1 Pull Transaction 48
3.3.5 Message Processing Events and Sequencing Rules 49
3.3.5.1 Receiving ABORT TIP Command 49
3.3.5.2 Receiving COMMIT TIP Command 49
3.3.5.3 Receiving NOTPULLED TIP Command 50
3.3.5.4 Receiving PREPARE TIP Command 50
3.3.5.5 Receiving PULLED TIP Command 51
3.3.5.6 Receiving PUSH TIP Command 51
3.3.5.7 Receiving QUERIEDEXISTS TIP Command 52
3.3.5.8 Receiving QUERIEDNOTFOUND TIP Command 53
3.3.5.9 Receiving RECONNECT TIP Command 53
3.3.5.10 Receiving ERROR TIP Command 55
3.3.6 Timer Events 55
3.3.6.1 Query Timer Expired Event 55
3.3.7 Other Local Events 55
3.3.7.1 Invalid TIP Command Event 55
3.3.7.2 Process Error 56
3.3.7.3 Events Signaled by the Core Transaction Manager Facet 57
3.3.7.3.1 Commit Complete 57
3.3.7.3.2 Create Superior Enlistment Success 57
3.3.7.3.3 Create Superior Enlistment Failure 58
3.3.7.3.4 Phase Zero Complete 58
3.3.7.3.5 Phase One Complete 59
3.3.7.3.6 Recover In Doubt Transaction 60
3.3.7.3.7 Rollback Complete 61
3.3.7.3.8 Unilaterally Aborted 61
3.3.7.4 Transport Events 61
3.3.7.4.1 Transport Connection Down 61
3.4 TIP Transaction Manager Communicating with an Application Facet Details 62
3.4.1 Abstract Data Model 62
3.4.1.1 TIP Transaction Manager Communicating with an Application Facet State Transition Table 62
3.4.2 Timers 63
3.4.3 Initialization 63
3.4.4 Higher-Layer Triggered Events 63
3.4.5 Message Processing Events and Sequencing Rules 63
3.4.5.1 Receiving ABORT TIP Command 63
3.4.5.2 Receiving BEGIN TIP Command 64
3.4.5.3 Receiving COMMIT TIP Command 64
3.4.5.4 Receiving ERROR TIP Command 65
3.4.6 Timer Events 65
3.4.7 Other Local Events 65
3.4.7.1 Invalid TIP Command Event 65
3.4.7.2 Events Signaled by the Core Transaction Manager Facet 66
3.4.7.2.1 Create Transaction Failure 66
3.4.7.2.2 Create Transaction Success 66
3.4.7.2.3 Phase Zero Complete 66
3.4.7.2.4 Phase One Complete 67
3.4.7.2.5 Rollback Complete 68

3.4.7.2.6 Unilaterally Aborted 68
3.4.7.3 Transport Events 68
3.4.7.3.1 Transport Connection Down 68
4 Protocol Examples 70
4.1 Transaction Processing Scenario 70
4.1.1 Creating the TIP Connection 70
4.1.2 Propagating the Transaction 71
4.1.2.1 Pull Propagation 71
4.1.2.2 Push Propagation 72
4.1.3 Committing the Transaction 73
4.1.3.1 Two Phase Commit 73
4.1.3.1.1 Read Only 73
4.1.3.1.2 Phase One 73
4.1.3.1.3 Recovery 74
4.1.3.1.4 Phase Two 75
4.1.3.2 Single Phase Commit 76
4.2 Begin Scenario 76
4.2.1 Creating the TIP Connection 76
4.2.2 Beginning the Transaction 76
4.2.3 Committing the Transaction 77
5 Security 78
6 Appendix A: Windows Behavior 79
7 Appendix B: Summary of Extensions 80
8 Index 83

1 Introduction

This document specifies a set of extensions to the standard Transaction Internet Protocol (TIP) Version 3.0, as specified in [RFC2371]. This specification assumes that the reader has familiarity with the concepts and requirements specified in [RFC2371]. Concepts and requirements specified in [RFC2371] are repeated in this specification when needed to provide clarity.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

Computer Name
Core Transaction Manager Facet
Facet
IPv4 Address in String Format
Signal
Subordinate Transaction Manager
Superior Transaction Manager
Single-Phase Commit
Transport Layer Security (TLS)
Transaction
Transaction Manager
Two-Phase Commit

The following terms are specific to this document:

Higher-Layer Business Logic: The application functionality that invokes the functionality that is specific to this protocol.

OleTx Transaction Manager (OleTx TM): A **transaction manager** that implements the OleTx Transaction Protocol.

Partner Transaction Manager: A **transaction manager** that plays the opposite role in an enlistment. When the **TIP Subordinate Transaction Manager Facet** is communicating with the Partner Transaction Manager, the Partner Transaction Manager acts a **Superior Transaction Manager**. When the **TIP Superior Transaction Manager Facet** is communicating with the Partner Transaction Manager, the Partner Transaction Manager acts as a **Subordinate Transaction Manager**. The **TIP Transaction Manager Communicating with an Application Facet** does not communicate with a Partner Transaction Manager.

TIP: An acronym for the Transaction Internet Protocol, which is specified in [RFC2119] section 13.

TIP Command: A **TIP** request or reply, including action and parameters, as specified in [RFC2371] section 13.

TIP Command Line: That part of a **TIP** message that contains a single **TIP Command**. This is specified in the **TIP** standard [RFC2371] section 11 as a "line of ASCII text, using only octets with values in the range 32 through 126 inclusive, followed by either a CR (an octet with value 13) or an LR (an octet with value 10)".

TIP Communication: An exchange of **TIP commands** and responses that follows message exchange patterns that conform to the **TIP** specification, as specified in [RFC2371].

TIP Connection: A connection that is initiated and used, as specified in [RFC2371] section 4.

TIP Pipelining: The process of concatenating more than one TIP Command lines into a single TCP message, as specified in [RFC2371] section 12.

TIP Subordinate Transaction Manager: A **Subordinate Transaction Manager** that implements the transaction management functionality that is specified in TIP.

TIP Superior Transaction Manager: A **Superior Transaction Manager** that implements the transaction management functionality that is specified in TIP.

TIP Transaction Manager: A **transaction manager** that implements the **transaction** management functionality that is specified in **TIP**.

TIP Transaction Manager Communicating with an Application Facet: The **Facet** that accepts requests to create and complete a transaction from an application.

TIP Transaction Manager Facets: The **Facets** that constitute the **transaction manager** role, namely the **TIP Superior Transaction Manager Facet**, the **TIP Subordinate Transaction Manager Facet**, and the **TIP Transaction Manager Communicating with an Application Facet**.

TIP Superior Transaction Manager Facet: The **Facet** that accepts requests to pull a transaction from the **Partner Transaction Manager**, sends requests to push a transaction to the **Partner Transaction Manager**, drives the **Two-Phase Commit** protocol with the **Partner Transaction Manager**, and after a failure, performs recovery.

TIP Subordinate Transaction Manager Facet: The **Facet** that accepts requests to push a transaction from the **Partner Transaction Manager**, sends requests to pull a transaction from the **Partner Transaction Manager**, and participates as a subordinate in the **Two-Phase Commit** protocol.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-DTCO] Microsoft Corporation, "MSDTC Connection Manager: OleTx Transaction Protocol Specification", July 2007.

[MS-DTYP] Microsoft Corporation, "Windows Data Types", January 2007.

[MS-ERREF] Microsoft Corporation, "Windows Error Codes", January 2007.

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt

[RFC2371] Lyon, J., Evans, K., and Klein, J., "Transaction Internet Protocol Version 3.0", RFC 2371, July 1998, http://www.ietf.org/rfc/rfc2371.txt

[RFC4234] Crocker, D., Ed. and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005, http://www.ietf.org/rfc/rfc4234.txt

1.2.2 Informative References

[GRAY] Gray, G. and A. Reuter, "Transaction Processing: Concepts and Techniques", San Mateo, CA: Morgan Kaufmann Publishers, 1993, ISBN: 1558601902.

[RFC2372] Lyon, J., Evans, K., and Klein, J., "Transaction Internet Protocol – Requirements and Supplemental Information", RFC 2372, July 1998, http://www.ietf.org/rfc/rfc2372.txt

1.3 Protocol Overview (Synopsis)

This protocol represents an extension to the **TIP** protocol, as specified in [RFC2371], and it is assumed to operate in an environment in which a **OleTx Transaction Manager** is present. In this context, the protocol provides concrete mechanisms for associating a OleTx Transaction and a TIP **Transaction**. These include mechanisms for creating the association, coordinating agreement on a single atomic outcome, and reliably distributing that outcome to the **Transaction Managers** involved in the overall transaction.

- It provides a way to group multiple actions across different nodes to define the next state.
- It guarantees that all the nodes agree on the same outcome, so that:
  - All of these actions complete and all the nodes move together to the next state.
  - All the nodes remain in their previous state.

For multiple platforms to participate in this, it is important to have a standard protocol for reaching this agreement. The TIP standard protocol [RFC2371] specifies such a standard. This document defines an extension of the TIP standard protocol.

The TIP standard protocol [RFC2371] specifies **TIP Connection** initialization, push and pull enlistment, distributed agreement and remote transactions. These are summarized in the following subsections and specified in sections 2 and 3.

1.3.1 Protocol Roles

This protocol comprises the following self-contained classes of functionality or protocol roles:

- The TIP Application role.
- The TIP Transaction Manager role, which can be further divided into three sub-roles or facets:
  - The TIP Superior Transaction Manager facet (section 1.3.1.2.1).
  - The TIP Subordinate Transaction Manager facet (section 1.3.1.2.2).
  - The TIP Transaction Manager Communicating with an Application facet (section 1.3.1.2.3).

The following figure shows the protocol roles:

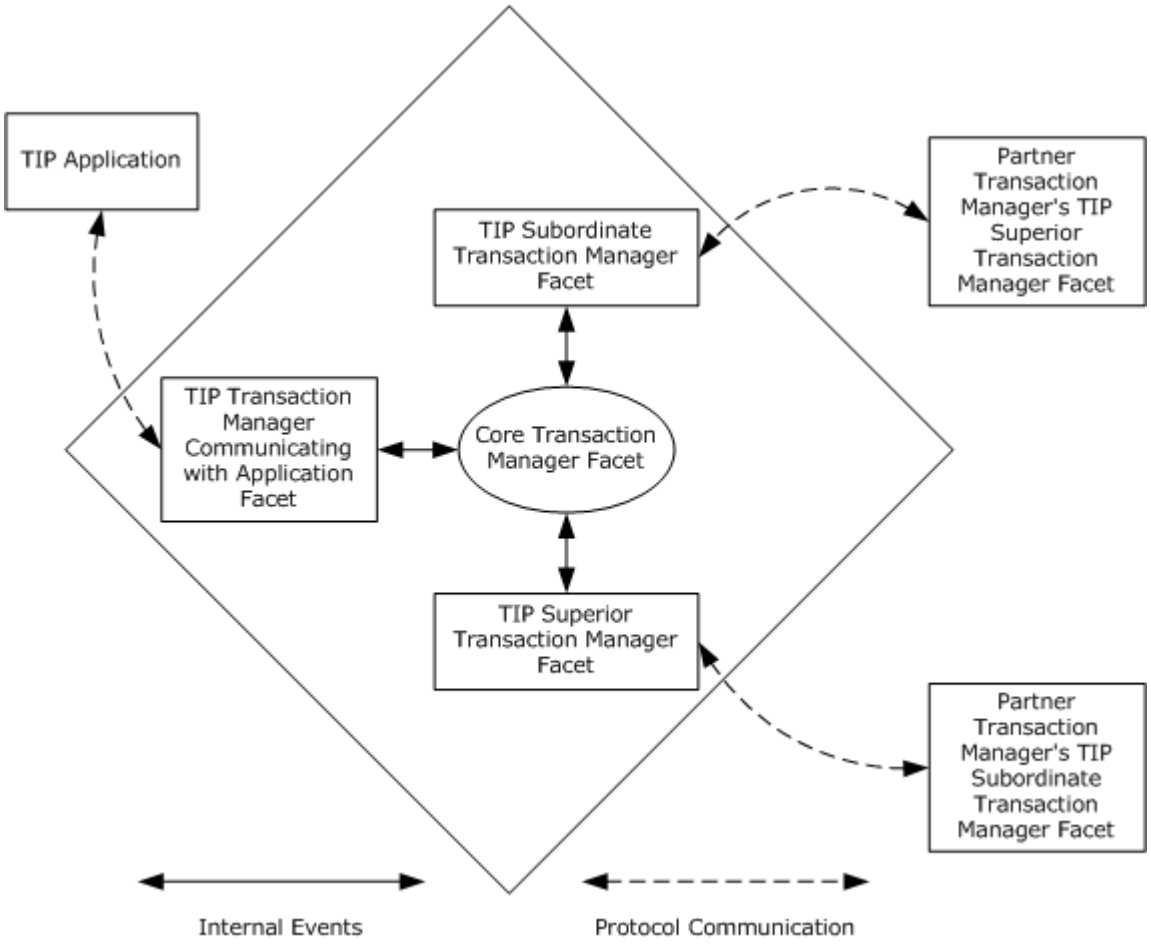

Figure 1: Protocol roles

These facets MUST communicate with each other by using a set of events. Each facet MUST define the set of events that the facet supports.

An event MUST consist of the following data elements:

- The name of the event.
- The list of arguments with which the event MUST be signaled.

This protocol assumes the existence of an implementation-specific communication mechanism used to signal events and access data elements between facets inside a transaction manager.

1.3.1.1 The TIP Application Role

The TIP Application role performs the following tasks:

- Establishes a TIP Connection with the TIP Transaction Manager Communicating with an Application Facet.
- Requests the creation of a transaction on the TIP Transaction Manager Communicating with an Application Facet and obtains an identifier for the created transaction.

- Requests the Commit or Rollback of a transaction it created on the TIP Transaction Manager Communicating with an Application Facet and obtains the transaction outcome.

1.3.1.2 The Transaction Manager Role

1.3.1.2.1 The TIP Superior Transaction Manager Facet

The TIP Superior Transaction Manager facet performs the following tasks:

- Establishes a TIP Connection with the **Partner Transaction Manager**'s **TIP Subordinate Transaction Manager Facet**.
- Accepts requests to pull a transaction from the Partner Transaction Manager's TIP Subordinate Transaction Manager Facet.
- Sends requests to push a transaction to the Partner Transaction Manager’s TIP Subordinate Transaction Manager Facet.
- Drives the **Two-Phase Commit** Protocol with its Partner Transaction Manager’s TIP Subordinate Transaction Manager Facet.
- After a failure, it performs transaction recovery and provides transaction outcome notifications to its Partner Transaction Manager’s TIP Subordinate Transaction Manager Facet.

1.3.1.2.2 The TIP Subordinate Transaction Manager Facet

The TIP Subordinate Transaction Manager facet performs the following tasks:

- Establishes a TIP Connection with the Partner Transaction Manager’s **TIP Superior Transaction Manager Facet**.
- Sends requests to pull a transaction from the Partner Transaction Manager’s TIP Superior Transaction Manager Facet.
- Accepts requests to push a transaction from the Partner Transaction Manager’s TIP TIP Superior Transaction Manager Facet.
- Participates in the Two-Phase Commit Protocol with its Partner Transaction Manager’s TIP Superior Transaction Manager Facet.
- After a failure, it participates in recovery and accepts transaction outcome notifications from its Partner Transaction Manager's TIP Superior Transaction Manager Facet.

1.3.1.2.3 The TIP Transaction Manager Communicating with an Application Facet

The TIP Transaction Manager Communicating with an Application Facet performs the following tasks:

- Accepts requests to create a transaction from the TIP Application role, and responds with the identifier for the created transaction.
- Accepts requests to Commit or Rollback a transaction from the TIP Application role, and responds with the transaction outcome.

1.3.2 Common Scenarios

1.3.2.1 Starting and Completing a Transaction

In this scenario, an application (playing the TIP Application role) creates a transaction with a **TIP Transaction Manager** (that implements this protocol), performs some work by using that transaction, and eventually it completes (commits or aborts) the transaction.

The following figure illustrates the scenario (TIP protocol messages are illustrated with dashed arrows):

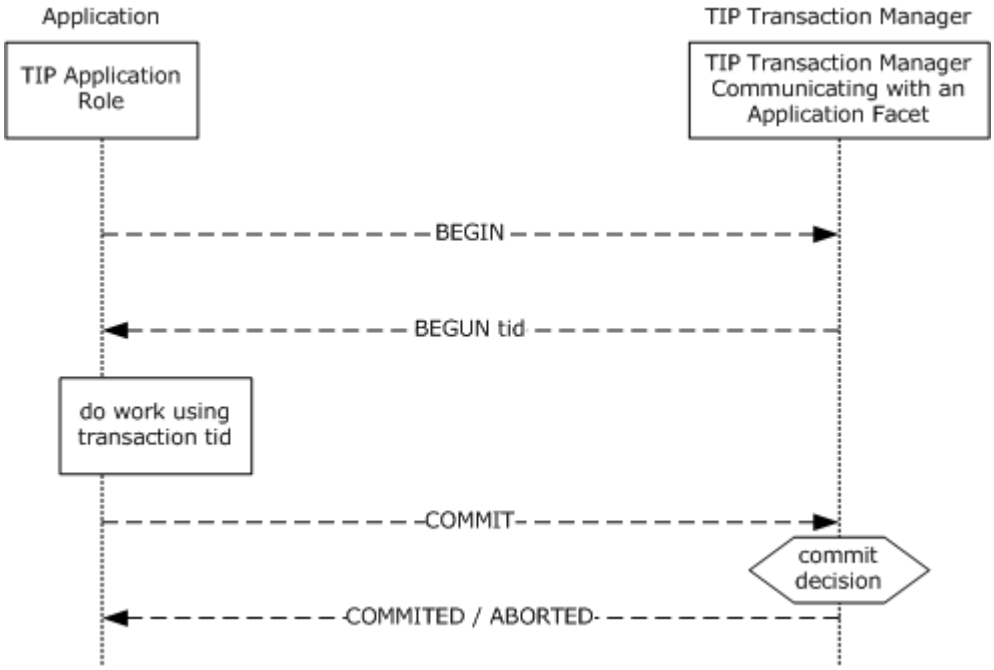

Figure 2: Starting and completing a transaction

1. The TIP Application requests the TIP Transaction Manager Communicating with an Application Facet of a TIP Transaction Manager to create a transaction by sending the BEGIN **TIP Command**.

2. TIP Transaction Manager Communicating with an Application Facet replies with a BEGUN TIP Command, passing in the identifier of the transaction created by the TIP Transaction Manager.

3. The TIP Application performs work using the transaction.

4. When completing all transacted work associated with the transaction, the TIP Application requests the TIP Transaction Manager Communicating with an Application Facet of the TIP Transaction Manager to commit the transaction by sending the COMMIT TIP Command.

5. The TIP Transaction Manager makes the appropriate commit decision and notifies TIP Application of the transaction's outcome by using either the COMMITTED or ABORTED TIP Command.

1.3.2.2 Pulling a Transaction

In this scenario, Application A sends a request to Application B to pull a local transaction that it creates with its TIP Transaction Manager A, and do some work as part of the pulled Transaction. The following figure illustrates the scenario (TIP protocol messages are illustrated with dashed arrows):

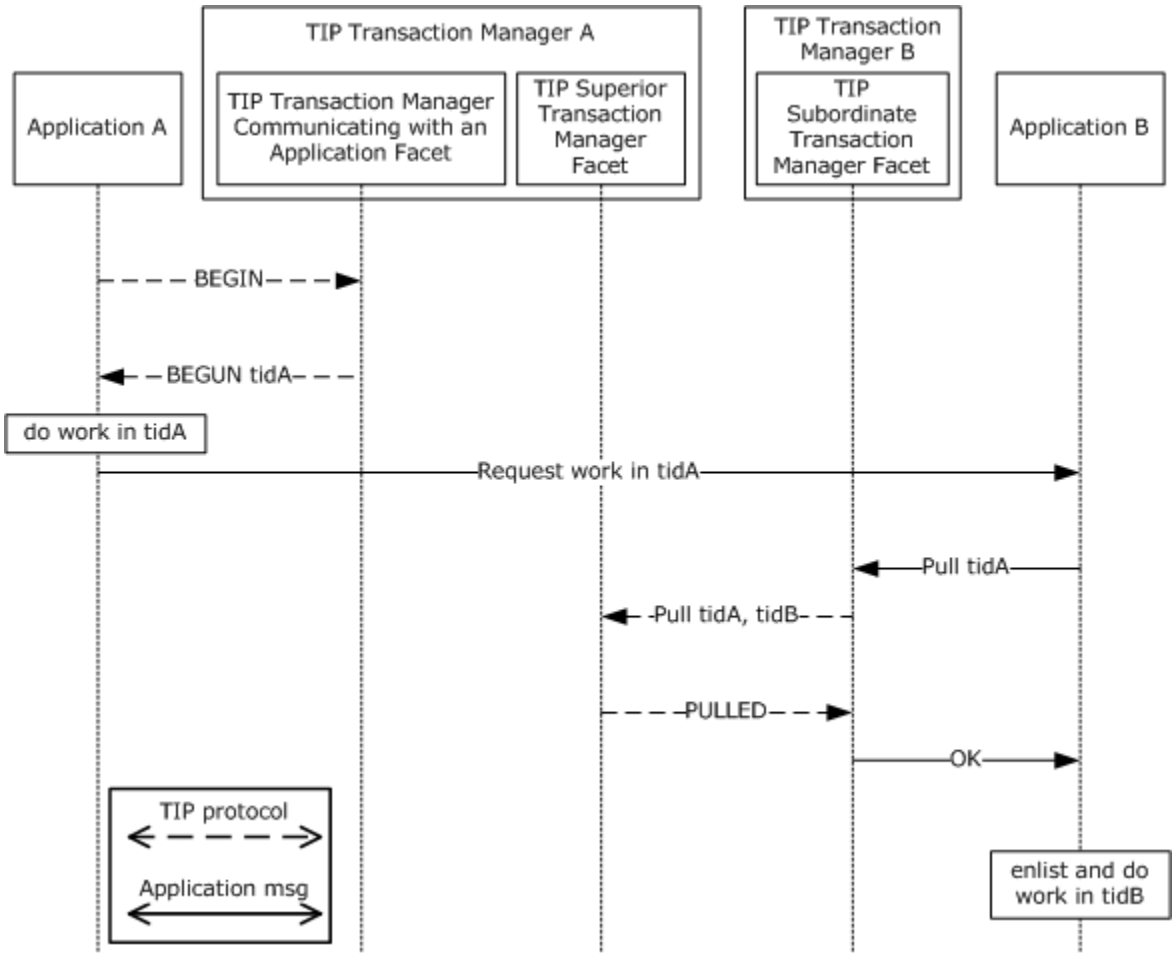

Figure 3: Pulling a Transaction

1. Application A requests the TIP Transaction Manager Communicating with an Application Facet of TIP Transaction Manager A to create a transaction by sending the BEGIN TIP Command.
2. TIP Transaction Manager Communicating with an Application Facet replies with a BEGUN TIP Command, passing in the identifier of the transaction (tidA) created by TIP Transaction Manager A.
3. Application A does some local work in the Transaction.
4. Application A requests Application B to do some work within the same Transaction.
5. Application B requests TIP Transaction Manager B to pull this Transaction.

6. The TIP Subordinate Transaction Manager Facet of TIP Transaction Manager B sends a PULL TIP Command to the TIP Superior Transaction Manager Facet of TIP Transaction Manager A, passing in parameters tidA and tidB (its local identifier for the Transaction).

7. TIP Transaction Manager A agrees by responding with the PULLED TIP Command. At this point, TIP Transaction Manager B has an Enlistment in the Transaction, and the Transaction is bound to the TIP Connection.

8. TIP Transaction Manager B returns to Application B.

9. Application B does the requested work using the pulled Transaction.

1.3.2.3 Pushing a Transaction

In this scenario, Application A requests its Transaction Manager A to push a Transaction to TIP Transaction Manager B, and then sends a request to application B to do some work as a part of the pushed Transaction.

The following figure illustrates the scenario (TIP protocol messages are illustrated with dashed arrows):

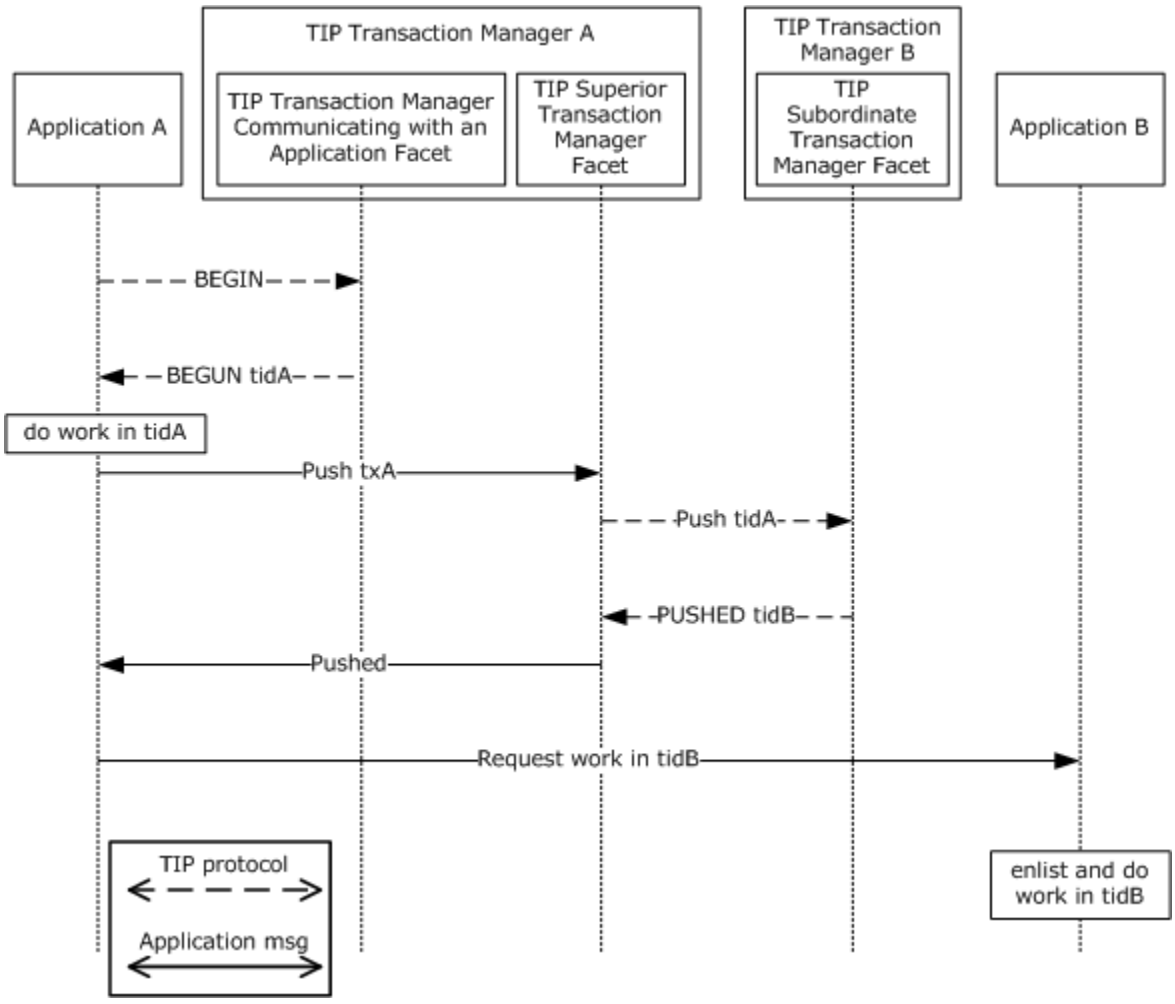

Figure 4: Pushing a transaction

1. Application A requests the TIP Transaction Manager Communicating with an Application Facet of TIP Transaction Manager A to create a transaction by sending the BEGIN TIP Command.

2. TIP Transaction Manager Communicating with an Application Facet replies with a BEGUN TIP Command, passing in the identifier of the transaction (tidA) created by TIP Transaction Manager A.

3. Application A does some local work in the Transaction.

4. Application A asks its TIP Transaction Manager A to push the Transaction to TIP Transaction Manager B.

5. The TIP Superior Transaction Manager facet of TIP Transaction Manager A sends a PUSH TIP Command to the TIP Subordinate Transaction Manager Facet of TIP Transaction Manager B, passing as a parameter tidA.

6. The TIP Transaction Manager B agrees by sending the PUSHED TIP Command, passing as a parameter tidB, which is TIP Transaction Manager B's identifier for the Transaction. At this point, TIP Transaction Manager B has an Enlistment in the Transaction, and the Transaction is bound to the TIP Connection.

7. TIP Transaction Manager A returns to Application A.

8. Application A asks Application B to do some work within the same Transaction passing it the identifier of the pushed Transaction, tidB.

9. Application B does the requested work using the pushed Transaction.

1.3.2.4 TIP Two-Phase Commit

Distributed agreement between two Transaction Managers is accomplished using the Two-Phase Commit protocol (see [GRAY]). The following figure illustrates this scenario (TIP protocol messages are illustrated with dashed arrows):

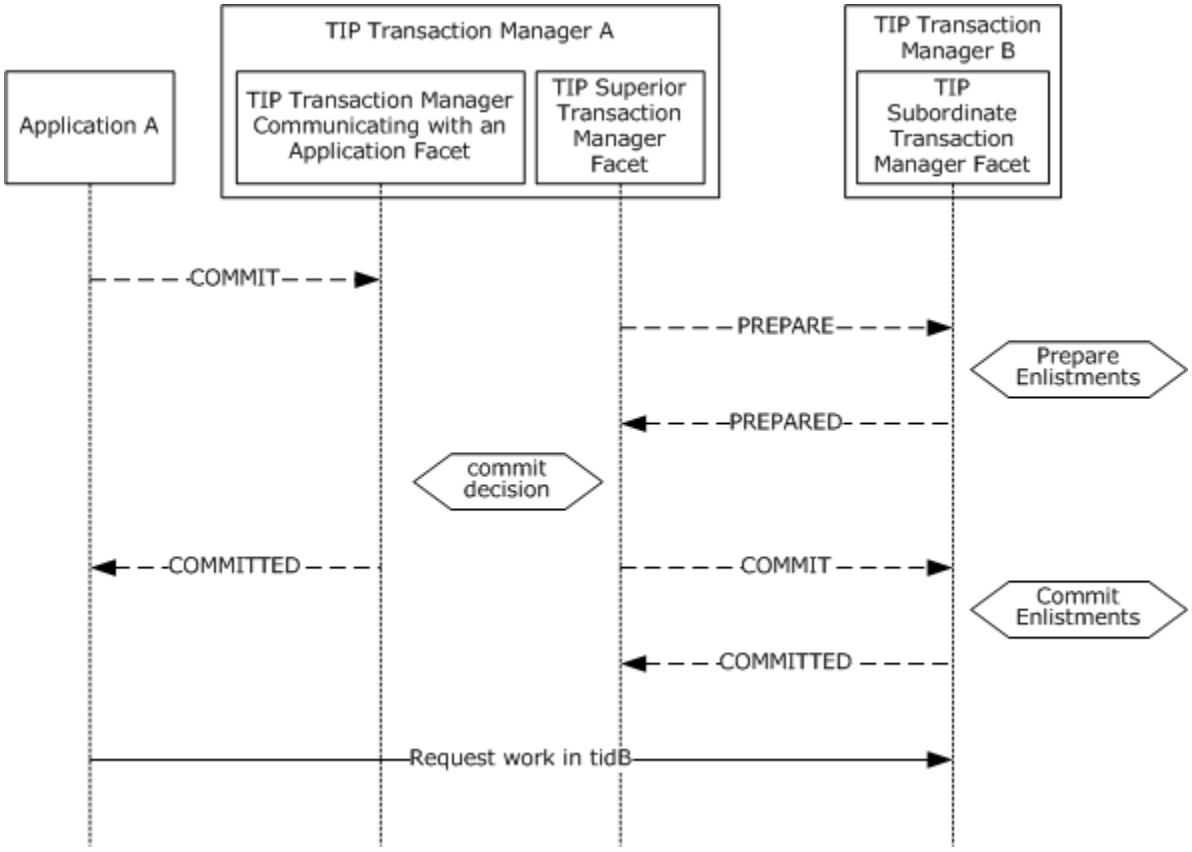

Figure 5: TIP two-phase commit

1. Application A asks the TIP Transaction Manager A to commit the current Transaction.

2. The TIP Superior Transaction Manager facet of the TIP Transaction Manager A initiates the two-phase commit protocol (assuming the transaction has two or more enlistments). As part of that protocol, it sends a PREPARE TIP Command to the TIP Subordinate Transaction Manager facet of TIP Transaction Manager B, which is enlisted as a subordinate in the Transaction.

3. Assuming the TIP Transaction Manager B successfully prepares all its enlistments for this Transaction, it replies with the PREPARED TIP Command.

4. Assuming all enlistments prepare successfully, the TIP Transaction Manager A starts the second phase of the Two-Phase Commit protocol, and asks all Enlistments in the Transaction to commit. In particular, it sends a COMMIT TIP Command to the TIP Subordinate Transaction Manager Facet of TIP Transaction Manager B.

5. After receiving the COMMIT TIP Command, the TIP Transaction Manager B notifies all its enlistments for the respective Transaction to commit, and replies with a COMMITTED TIP Command.

6. After receiving the COMMITTED response from the TIP Transaction Manager B, the TIP Transaction Manager A knows it no longer has any responsibilities with respect to that enlistment and, it frees the associated resources.

1.4 Relationship to Other Protocols

This protocol is an extension of the TIP standard protocol, as specified in [RFC2371].

Figure 6 illustrates its relationship with other protocols:

- The left-side of the layering diagram in Figure 6 illustrates the fact that this protocol uses the extensibility mechanism defined in MSDTC Connection Manager: OleTx Transaction Protocol Specification, as specified in [MS-DTCO] to become a Protocol Extension to an OleTx Transaction Manager implementation.
- The right-side of the layering diagram in Figure 6 illustrates how protocol relies on the Session and Connection transport infrastructure defined in the TCP protocol.

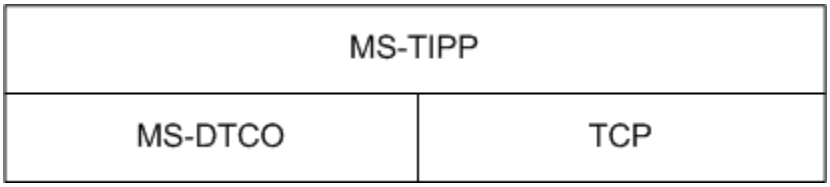

Figure 6: Protocol layering

1.5 Prerequisites/Preconditions

The operation of this protocol assumes the following:

- A TCP/IP implementation MUST be available for use by all of the protocol roles.
- An OleTx Transaction Manager MUST be present and operating so that the implementation of this protocol can use its transaction management services.

1.6 Applicability Statement

This protocol is a distributed transaction management and coordination protocol, and therefore it is applicable in situations in which distributed Transaction management coordination is necessary. Because this protocol is un-secure, an implicit level of trust is required between the parties using the protocol. All the prerequisites/preconditions specified in section 1.5 MUST also be satisfied for this protocol to be employed successfully.

1.7 Versioning and Capability Negotiation

The TIP standard, as specified in [RFC2371], which this protocol extends, specifies a versioning negotiation mechanism. This protocol complies with that versioning negotiation mechanism, but is restricted to support only version 3.

The TIP standard, [RFC2371], specifies a capability negotiation mechanism for Multiplexing. This protocol complies with that capability negotiation mechanism, but is restricted to not support Multiplexing.

The TIP standard, [RFC2371], specifies a capability negotiation mechanism for TLS. This protocol is restricted to not support TLS.

1.8 Vendor-Extensible Fields

There is a variable-length ASCII string in each TIP Command that can be used for any purpose. It is specified in [RFC2371] section 11 of the TIP Standard.

1.9 Standards Assignments

Parameter	Value	Reference
TCP port default	3372	[RFC2371]

2 Messages

Unless stated otherwise, this protocol complies with the TIP standard as specified in [RFC2371].

2.1 Transport

This protocol restricts the connections specified in [RFC2371] section 4 to TCP connections.

2.2 Message Syntax

This protocol places the following syntax restrictions on [RFC2371] specification:

- **TIP Command Line restrictions**:
  - Messages received by this protocol restricts the **TIP Command Line** specified in [RFC2371] section 11, as follows:
    - The TIP Command Line MUST NOT cross 1,024 character boundaries.
  - Messages sent by this protocol MUST restrict the TIP Command Line specified [RFC2371] section 11, as follows:
    - A message MUST contain at most 1 TIP Command Line.
    - The TIP Command Line MUST NOT exceed 1024 characters.
- **Transaction identifier restrictions**:
  - A Transaction identifier created by this protocol MUST restrict the TIP Transaction Identifier specified in [RFC2371] section 5 to the following ABNF:

```
OleTxTipTransactionIdentifier
    = x74 %x4F %x6C %x65 %x54 %x78 "-" LowerCaseUUID
```

    where *LowerCaseUUID* is defined to be the same as the Glossary term GUID with the restriction that alpha characters MUST be lower-case. For example: OleTx-725d5246-2217-11dc-8314-0800200c9a66.
- **Transaction Manager address restrictions**:
  - A Transaction Manager address created by this protocol MUST restrict the Transaction Manager address specified [RFC2371] section 7 to the following ABNF:

```
%x74 %x69 %x70 %x3A %x2F %x2F HostName %x2F
```

    where *HostName* is defined to be one of the following:
    - A **Computer Name** with restriction that first character cannot be an underscore or a number.
    - An **IPv4 Address in String Format**.

The following sub-sections specify which TIP Command parameters have the preceding syntax restrictions. These sub-sections include only those TIP Commands that place restrictions on [RFC2371].

2.2.1 ALREADYPUSHED

The *subordinate's transaction identifier* parameter specified in [RFC2371] section 13 for this TIP Command MUST adhere to the above Transaction identifier restrictions.

2.2.2 BEGUN

The *transaction identifier* parameter specified [RFC2371] in section 13 for this TIP Command MUST adhere to the above Transaction identifier restrictions.

2.2.3 IDENTIFY

The *primary transaction manager address* and *secondary transaction manager address* parameters specified in [RFC2371] section 13 for this TIP Command MUST adhere to the above Transaction Manager address restrictions.

2.2.4 PULL

The *superior's transaction identifier* and subordinate's transaction identifier parameters specified in [RFC2371] section 13 for this TIP Command MUST adhere to the above Transaction identifier restrictions.

2.2.5 PUSH

The *superior's transaction identifier* parameter specified in [RFC2371] section 13 for this TIP Command MUST adhere to the above Transaction identifier restrictions.

2.2.6 PUSHED

The *subordinate's transaction identifier* parameter specified in [RFC2371] section 13 of this TIP Command MUST adhere to the above Transaction identifier restrictions.

2.2.7 QUERY

The *superior's transaction identifier* parameter specified in [RFC2371] section 13 for this TIP Command MUST adhere to the above Transaction identifier restrictions.

2.2.8 RECONNECT

The *subordinate's transaction identifier* parameter specified in [RFC2371] section 13 for this TIP Command MUST adhere to the above Transaction identifier restrictions.

3 Protocol Details

This section defines the expected behavior of the Transaction Manager role, which consists of three Facets:

- TIP Superior Transaction Manager Facet
- TIP Subordinate Transaction Manager Facet
- TIP Transaction Manager Communicating with an Application Facet

3.1 Common Details

This section contains protocol details that are common to all **TIP Transaction Manager Facets**.

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

Note The abstract data model can be implemented in a variety of ways. This protocol does not prescribe or advocate any specific implementation technique. This abstract data model is an extension of the abstract data model specified in [MS-DTCO].

3.1.1.1 Data Elements

A TIP Transaction Manager Facet MUST maintain the following data elements:

- **A table of TIP Connections**: This is a table of TIP Connection objects.
- A set of flags that allow restrictions to be placed on this protocol:
  - **Allow Begin**: A flag whose true value indicates that the TIP Transaction Manager Facet will accept a BEGIN TIP Command.
  - **Allow Passthrough**: A flag whose true value indicates that the TIP Transaction Manager Facet will allow a transaction to be pushed and then pulled without a local enlistment.
  - **Allow Non Default Port**: A flag whose true value indicates that the TIP Transaction Manager Facet will allow a TCP connection from a port number other than 3372.
  - **Allow Different Partner Address**: A flag whose true value indicates that the TIP Transaction Manager Facet will accept an IDENTIFY (section 2.2.3) TIP Command whose primary transaction manager address parameter does not match the address from which the TCP connection originated.
  - **Transaction Manager Identifier Override**: If the field is set, the TIP Transaction Manager Facet will use it as the primary transaction manager address argument when it sends the IDENTIFY (section 2.2.3) TIP Command.

The TIP Transaction Manager Facet MUST extend the definition of an Enlistment object, as specified in [MS-DTCO] section 3.2.1.3, to include the following data fields:

- **TIP Connection**: This field references the TIP Connection object associated with the Enlistment.
- **Partner Transaction Identifier**: This field contains the transaction identifier that the Partner Transaction Manager uses for the transaction object referenced by the Enlistment.
- **Partner Transaction Manager Address**: This field contains a TransactionManagerAddress (as specified in section 2.2) used to verify and contact the Partner Transaction Manager in case of connection failure.

3.1.1.2 TIP Connection Object

A TIP Connection object MUST contain the following data fields:

- **Partner Transaction Manager Address**: This field contains a Transaction Manager Address (as specified in section 2.2) used to identify the Transaction Manager that the TIP Connection connects to.
- **Enlistment**: This field references an Enlistment object associated with the TIP Connection.
- **Transport Connection**: This field references the TCP connection that the TIP Connection uses to send TIP Commands.
- **Connection Type:** An enumeration that indicates whether the TIP Connection will be used for either sending or receiving requests. This field MUST be set to one of the following values:
  - **Primary**: This value is set to indicate that the TIP Connection will be sending requests.
  - **Secondary**: This value is set to indicate that the TIP Connection will be receiving requests.
- **State**: An enumeration that indicates what state the TIP Connection is in. This field MUST be set to one of the following values or one of the values of an extension to the TIP Connection object:
  - **Initial**: The TIP Connection has not yet identified its Partner Transaction Manager.
  - **Initial Identity**: The TIP Connection is waiting for a reply to an IDENTIFY TIP Command sent while in the initial state.
  - **Idle**: The TIP Connection has identified its Partner Transaction Manager but has no associated transaction.

3.1.1.3 TIP Connection Management Operations

The following operations on the table of TIP Connection are used throughout section 3.

3.1.1.3.1 GetTipConnection Operation

This operation is called when a message is received on the TCP connection.

- The input parameter for this operation MUST be a TCP connection.
- This returns a TIP Connection object whose data fields MUST include:
  - Transport Connection is the provided TCP connection.
- When this operation is called, the TIP Connection Manager MUST perform the following actions:
  - Attempt to find a TIP Connection object corresponding to the provided TCP connection.

- If a TIP Connection is found:
  - Return the TIP Connection.
- Otherwise:
  - Create a new TIP Connection object and initialize it with the following values:
    - The **Connection Type** field is initialized to Secondary.
    - The **State** field is initialized to Initial.
    - The **Transport Connection** field is set to the provided TCP Connection.
  - Return the TIP Connection object.

3.1.1.3.2 GetTipConnectionFromAddress Operation

This operation is called when a TIP Transaction Manager Facet needs to initiate a TIP Connection to send a TIP Command (for example, PUSH).

- The input parameter for this operation MUST be a Partner Transaction Manager address.
- This operation returns a TIP Connection object whose data fields MUST include:
  - Partner Transaction Manager Address is the provided address.
  - Connection Type is Primary.
  - State is Idle.

If there is a TIP Connection to the Partner Transaction Manager for which the IDENTIFY/IDENTIFIED exchange has taken place and the State is Idle, as specified in [RFC2371] section 4, the TIP Connection Manager SHOULD return it.

Otherwise, the TIP Connection Manager MUST perform the following actions:

- Create a new TCP connection to the provided Partner Transaction Manager Address.
- Create a corresponding TIP Connection object and initialize it with the following values:
  - The **Transport Connected** field is set to the TCP connection.
  - The **State** field is set to Initial Identify.
  - The **Partner Transaction Manager Address** field is initialized to the provided Partner Transaction Manager Address.
  - The **Connection Type** field is set to Primary.
- Send an IDENTIFY TIP Command with the following arguments:
  - The *lowest protocol version*: "3"
  - The *highest protocol version*: "3"
  - If the **Transaction Manager Address Override** field is set, the *primary transaction manager address* argument MUST be set to the value of the **Transaction Manager Address Override** field; otherwise, it MUST be set to the address of the TIP Transaction Manager Facet.

- The *secondary transaction manager address* argument SHOULD<1> be set to the value provided Partner Transaction Manager Address to conform with that as specified in [RFC2371].

- Wait indefinitely for a response from the Partner Transaction Manager. The TIP Connection Manager MUST accept messages and the TIP Transaction Manager Facet MUST process events while it is waiting.

  - If the connection is terminated, terminate the processing of this event.

  - If the response from the Partner Transaction Manager is a valid IDENTIFIED TIP Command, return the TIP Connection object.

  - Otherwise, terminate the processing of this event.

3.1.1.3.3 HasPartnerTransaction Operation

This operation is called when a TIP Transaction Manager Facet needs to determine whether a Partner Transaction Manager already enlisted in a particular Transaction:

- The input parameters for this operation MUST be:

  - Partner Transaction Manager Address.

  - Partner Transaction Identifier.

- This operation MUST return true if there exists a TIP Connection whose Enlistment has the provided values and false otherwise.

3.1.1.3.4 FreeTipConnection Operation

This operation is called when a TIP Transaction Manager Facet no longer needs the TIP Connection. The input parameter for this MUST be a TIP Connection object. The TIP Connection Manager MUST perform the following actions:

- If the TIP Connection object's **Enlistment** field references an Enlistment object, clear the Enlistment object's **TIP Connection** field.

- If the TIP Connection Manager initiated the TCP connection corresponding to the TIP Connection, then it SHOULD reuse it as specified in [RFC2371] section 4.

3.1.1.3.5 TerminateTipConnection Operation

The input parameter for this operation MUST be a TIP Connection object.

When this operation is called, the TIP Connection Manager MUST do the following:

- If the TIP Connection object's **Enlistment** field references an Enlistment object, clear the Enlistment object's **TIP Connection** field.

- Close the TCP connection referenced by the **Transport Connection** field of the provided TIP Connection object.

- Discard the TIP Connection object.

3.1.1.4 TIP Command Object

A TIP Command object MUST contain the following data fields:

- **Command Name**: This field contains a TIP Command name.
- **Parameter List**: The list of parameters for this TIP Command.
- **TIP Connection Object**: This field references the TIP Connection object for this TIP Command.

3.1.1.5 Transaction Identifier Converter Operations

This following operations that convert between transaction identifier formats are used throughout section 3.

3.1.1.5.1 Convert TIP Transaction Identifier to Transaction Identifier Operation

This operation MUST be called with the following argument:

- TIP Transaction Identifier.

This operation MUST return the following value:

- Transaction Identifier.

If the Convert TIP Transaction Identifier to Transaction Identifier operation is called, it MUST perform the following actions:

- Remove "OleTx-" from the beginning of the TIP Transaction Identifier.
- Convert the TIP Transaction Identifier string to a Globally Unique Identifier and return it.

3.1.1.5.2 Convert Transaction Identifier to TIP Transaction Identifier Operation

This operation MUST be called with the following argument:

- Transaction Identifier.

This operation MUST return the following value:

- TIP Transaction Identifier.

If the Convert Transaction Identifier to TIP Transaction Identifier operation is called, it MUST perform the following actions:

- Convert the Transaction Identifier from a Globally Unique Identifier to a string
- Prefix "OleTx-" to the string and return it.

3.1.1.6 Primary State Transition Table

The following table summarizes the legal state transitions that are common for all TIP Transaction Manager Facets for a TIP Connection whose **Connection Type** field is set to Primary. The table omits the following transitions:

- In every state the TIP Connection is allowed to send an ERROR TIP Command that changes the state to Error.

The following events trigger a state transition:

- A TIP request is sent to the Partner Transaction Manager.

- A TIP reply is received from the Partner Transaction Manager.

Current state	Event	Next state
Initial	IDENTIFY sent	Initial Identify
Initial Identify	IDENTIFIED received	Idle
Idle	NEEDTLS received	Error
Initial Identify	ERROR received	Error

3.1.1.7 Secondary State Transition Table

The following table summarizes the legal state transitions that are common for all TIP Transaction Manager Facets for a TIP Connection whose Connection Type field is set to Secondary. The table omits the following state transitions:

- In every state, the TIP Connection may receive an ERROR TIP Command that changes the state to Error.
- The state changes when a TIP reply is sent to the Partner Transaction Manager in response to a TIP request.

The "<TIP command A> received/<TIP response B> sent" syntax in the table indicates that the Facet received <TIP command A> and decided to respond to it with <TIP response B>. The state changes from <current state> to the <next state> when <TIP response B> is sent to the Partner Transaction Manager.

Current State	Event	Next State
Initial	IDENTIFY received/IDENTIFIED sent.	Idle
Initial	IDENTIFY received/ERROR sent.	Error
Initial	TLS received/CANTTLS sent.	Initial
Initial	TLS received/Error sent.	Error
Idle	MULTIPLEX received/CANTMULTIPLEX sent.	Idle
Idle	MULITPLEX received/Error sent.	Error

3.1.2 Timers

There are no timers common to all TIP Transaction Manager Facets.

3.1.3 Initialization

The TIP implementation MUST perform the following initialization steps:

- The following flags MUST be set to a value that is obtained from an implementation-specific source.<2>
  - **Allow Begin**

- **Allow Passthrough**
- **Allow Non-Default Port**
- **Allow DifferentPartner Address**

- The **Transaction Manager Address** field SHOULD be set to a value that is obtained from an implementation-specific source.<3>
- If the value of the Allow Network Access flag and the Allow TIP flag is true, the TIP implementation MUST listen for incoming TCP requests on an implementation-specific port.<4>

3.1.4 Higher-Layer Triggered Events

There are no higher-layer triggered events common to all TIP Transaction Manager Facets.

3.1.5 Message Processing Events and Sequencing Rules

This section describes how each received TIP Command is processed. Each of these events is signaled with a TIP Command object as an input argument.

When a TIP Transaction Manager Facet receives a TIP Command that is a response (for example, BEGUN) to a TIP request (for example, BEGIN), that it does not support, the TIP Transaction Manager Facet treats the response as an invalid TIP Command.

3.1.5.1 Receiving BEGUN TIP Command

When the TIP Transaction Manager Facet receives a BEGUN TIP Command object, it MUST perform the following actions:

- Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.

3.1.5.2 Receiving CANTMULTIPLEX TIP Command

When the TIP Transaction Manager Facet receives a CANTMULTIPLEX TIP Command object, it MUST perform the following actions:

- Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.

3.1.5.3 Receiving CANTTLS TIP Command

When the TIP Transaction Manager Facet receives a CANTTLS TIP Command object, it MUST perform the following actions:

- Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.

3.1.5.4 Receiving IDENTIFIED TIP Command

When the TIP Transaction Manager Facet receives an IDENTIFIED TIP Command object, it MUST contain the following parameters in its Parameter List:

- *protocol version*

Upon receipt, the TIP Transaction Manager Facet MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Primary.
  - The **State** field is set to Initial Identify.
- If the receiving TIP Connection does not satisfy these conditions, signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.
- If the value of the provided *protocol version* is not 3, signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.
- Set the **State** field of the TIP Connection to Idle.

3.1.5.5 Receiving IDENTIFY TIP Command

When the TIP Transaction Manager Facet receives an IDENTIFY TIP Command object, it MUST contain the following parameters in its Parameter List:

- *lowest protocol version*
- *highest protocol version*
- *primary transaction manager address*
- *secondary transaction manager address*

Upon receipt, the TIP Transaction Manager Facet MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Secondary.
  - The **State** field is set to Initial Identify.
- If the receiving TIP Connection does not satisfy these conditions, signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.
- If the provided *primary transaction manager address* is not set to "-":
  - Set the **Partner Transaction Manager Address** field of the receiving TIP Connection to the provided *primary transaction manager address*.
- Otherwise:
  - If the value of the **Allow Different Partner Address** flag is set to 0 and the provided *primary transaction manager address* does not match the address from which the connection originated, then signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.
  - If **Allow Non Default Port** is set to 0 and the sender's Port referenced by the Transport Connection is not set to 3372, signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.
- Test if the provided values meets one of the following conditions:

- The provided *lowest protocol version* is set to a value less than or equal to the maximum supported TIP Protocol version of the Local TIP Transaction Manager Facets.<5>
- The provided *highest protocol version* is set to a value greater than or equal to the minimum supported TIP Protocol version of the local TIP Transaction Manager Facets.<6>

- If the provided values do not satisfy one of the conditions:
  - Send an ERROR TIP Command.
  - Terminate the TCP connection. This causes the Transport Connection Down event to be signaled.
- Set the **State** field of the receiving TIP Connection object to Idle.
- Send an IDENTIFIED (as specified in [RFC2371] section 13) TIP Command with the following argument:
  - The lesser between the provided *highest protocol version* and the maximum supported TIP Protocol version of the local TIP Transaction Manager Facets.

3.1.5.6 Receiving MULTIPLEX TIP Command

When the TIP Transaction Manager Facet receives a MULTIPLEX TIP Command object, it MUST contain the following parameters in its Parameter List:

- *protocol-identifier*

Upon receipt, the TIP Transaction Manager Facet MUST perform the following actions:

- The TIP Transaction Manager Facet MUST test that the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Secondary.
- The TIP Transaction Manager Facet SHOULD<7> test that the receiving TIP Connection object meets the following conditions in conformance with the [RFC2371] specification:
  - The **State** field is set to Idle.
- The TIP Transaction Manager Facet MAY<8>test that the receiving TIP Connection object meets the following conditions:
  - The value of the provided <protocol-identifier> is "TMP2.0"
- If the receiving TIP Connection does not satisfies the conditions, signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event
- Send a CANTMULTIPLEX (as specified in [RFC2371] section 13) TIP Command.

3.1.5.7 Receiving MULTIPLEXING TIP Command

When the TIP Transaction Manager Facet receives a MULTIPLEXING TIP Command object, it MUST perform the following actions:

- Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.

3.1.5.8 Receiving NEEDTLS TIP Command

When the TIP Transaction Manager Facet receives a NEEDTLS TIP Command object, it MUST perform following actions:

- Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event. These actions do not conform to the [RFC2371] specification.

3.1.5.9 Receiving NOTBEGUN TIP Command

When the TIP Transaction Manager Facet receives a NOTBEGUN TIP Command object, it MUST perform the following actions:

- Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.

3.1.5.10 Receiving TLS TIP Command

When the TIP Transaction Manager Facet receives a **TLS** TIP Command object, it MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Secondary.
  - The **State** field is set to Initial Identify.
- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.
- The TIP Transaction Manager Facet SHOULD<9> send a CANTTLS TIP Command to conform with the [RFC2371] specification.

3.1.5.11 Receiving TLSING TIP Command

When the TIP Transaction Manager Facet receives a TLSING TIP Command object, it MUST perform the following actions:

- Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.

3.1.6 Timer Events

There are no timer events common to all TIP Transaction Manager Facets.

3.1.7 Other Local Events

3.1.7.1 Invalid TIP Command Event

When a TIP Command is determined to be invalid, the TIP Transaction Manager Facet MUST perform the following actions:

- The TIP Transaction Manager Facet SHOULD<10> send the ERROR TIP Command on the TIP Command's TIP Connection.

- If the TIP Connection's Connection State data field is Primary, terminate the TCP connection. This causes the Transport Connection Down event to be signaled.

3.1.7.2 Transport Events

3.1.7.2.1 Received Message

This event is signaled when a TCP message arrives on the TIP port. When this event is signaled, the TIP Transaction Manager Facet MUST perform the following actions:

- If the value of the **Allow Non Default Port** flag is false and the provided TCP connection did not originate from port 3372, then terminate the connection and terminate the processing of this event.
- Call the **GetTipConnection** with the TCP connection as an input parameter. This returns a TIP Connection object whose data fields include the following:
  - **Transport Connection**: The provided TCP connection.
- Parse the message data into separate TIP Commands according to the ABNF rules as specified in section 2.2. To support Pipelining, the incoming message is parsed into separate TIP Commands.
- If this parsing is not successful, Signal the Invalid TIP Command event (as specified in section 3.1.7.1) and terminate the processing of this event.
- For each of the TIP Commands in this message, do the following:
  - Build a TIP Commands object from the parsed TIP Commands name, parameters, and the TIP Connection object.
  - The TIP Commands object is now ready to be processed as an incoming message event.

3.1.7.2.2 Transport Connection Down

This event is signaled when the TIP Transaction Manager Facet Facet is notified that a TIP Connection has gone down. All TIP Transaction Manager Facets MUST define the behavior for this event.

3.2 TIP Superior Transaction Manager Facet Details

This section contains protocol details that relate to the TIP Superior Transaction Manager Facet.

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with the behavior that is described in this document.

Note that the abstract data model can be implemented in a variety of ways. This protocol does not prescribe or advocate any specific implementation technique.

The Facet MUST extend the definition of the **State** field of the TIP Connection object as specified in section 3.1.1.1 to include the following values:

- **State**: An enumeration that indicates what state the TIP Connection is in. This field MUST be set to one of the values in the extended enumeration. The following are the extension values:
  - Idle Push: The TIP Connection is waiting for a reply to a PUSH TIP Command sent while in the Idle state.
  - Idle Reconnect: The TIP Connection is waiting for a reply to a RECONNECT TIP Command sent while in the idle state.
  - Enlisted: The TIP Connection is associated with a transaction object and will send TIP Commands to notify its Partner Transaction Manager of the Transaction's outcome.
  - Enlisted Prepare: The TIP Connection is waiting for a reply to a PREPARE TIP Command sent while in the Enlisted state.
  - Enlisted Commit: The TIP Connection is waiting for a reply to a COMMIT TIP Command sent while in the Enlisted state.
  - Enlisted Abort: The TIP Connection is waiting for a reply to an ABORT TIP Command sent while in the Enlisted state.
  - Prepared: The transaction associated with the TIP Connection has completed Phase 1.
  - Prepared Commit: The TIP Connection is waiting for a reply to a COMMIT TIP Command sent while in the Prepared state.
  - Prepared Abort: The TIP Connection is waiting for a reply to an ABORT TIP Command sent while in the Prepared state.

3.2.1.1 TIP Superior Transaction Manager Facet State Transition Table

The following table summarizes the state transitions that are legal to the protocol as seen by the TIP Superior Transaction Manager Facet. The states are the TIP Connection states. The table omits the following transitions:

- In every state, the TIP Superior Transaction Manager Facet, acting as a Primary, is allowed to send an ERROR TIP Command, which changes the state to Error.
- In every state, the TIP Superior Transaction Manager Facet, acting as a Secondary, may receive an ERROR TIP Command, which changes the state to Error.

The following events trigger a state transition:

- A TIP request is sent to the Partner Transaction Manager
- A TIP reply is received from the Partner Transaction Manager
- A TIP reply is sent to the Partner Transaction Manager in response to a TIP request.

The "<TIP command A> received/<TIP response B> sent" syntax in the table indicates that the Facet received <TIP command A> and decided to respond to it with <TIP response B>. The state changes from <current state> to the <next state> when <TIP response B> is sent to the Partner Transaction Manager.

Current State	Event	Next State
Idle	PULL received/PULLED sent.	Enlisted

Current State	Event	Next State
Idle	PULL received/NOT PULLED sent.	Idle
Idle	PULL received/ERROR sent.	Error
Idle	PUSH sent.	Idle Push
Idle Push	PUSHED received.	Enlisted
Idle Push	ALREADYPUSHED received.	Idle
Idle Push	NOTPUSHED received.	Idle
Idle Push	ERROR received.	Error
Idle	QUERY received/QUERIEDEXISTS sent.	Idle
Idle	QUERY received/QUERIEDNOTFOUND sent.	Idle
Idle	QUERY received/ERROR sent.	Error
Idle	RECONNECT sent.	Idle Reconnect
Idle Reconnect	RECONNECTED received.	Prepared
Idle Reconnect	NOTRECONNECTED received.	Idle
Idle Reconnect	ERROR received.	Error
Enlisted	ABORT sent.	Enlisted Abort
Enlisted Abort	ABORTED received.	Idle
Enlisted Abort	ERROR received.	Error
Enlisted	COMMIT sent.	Enlisted Commit
Enlisted Commit	ABORTED received.	Idle
Enlisted Commit	COMMITTED received.	Idle
Enlisted Commit	ERROR received.	Error
Enlisted	PREPARE sent	Enlisted Prepare
Enlisted Prepare	PREPARED received.	Prepared
Enlisted Prepare	ABORTED received.	Idle
Enlisted Prepare	READONLY received.	Idle
Enlisted Prepare	ERROR received.	Error
Prepared	ABORT sent.	Prepared Abort
Prepared Abort	ABORTED received.	Idle
Prepared Abort	ERROR sent.	Error

Current State	Event	Next State
Prepared	COMMIT sent	Prepared Commit
Prepared Commit	COMMITTED received.	Idle
Prepared Commit	ERROR received.	Error

3.2.2 Timers

There are no timers specific to the TIP Superior Transaction Manager Facet.

3.2.3 Initialization

The TIP Superior Transaction Manager Facet MUST perform all initialization as specified in section 3.1.3.

3.2.4 Higher-Layer Triggered Events

3.2.4.1 Push Transaction

This event must be triggered by the higher-layer software with the following arguments:

- Partner Transaction Manager Address
- Transaction Identifier

If the Push Transaction event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:

- Attempt to find a Transaction object in the Transaction Table referenced by the Core Transaction Manager Facet that meets the following requirements:
  - The **Transaction Identifier** field is set to the provided Transaction Identifier.
- If a Transaction object is not found, notify the **higher-layer business logic** that the Push request failed and terminate the processing of this event.
- Create a new Enlistment object with the following settings:
  - The Transaction object is set to the Transaction object that was found.
  - The **Partner Transaction Manager** field is set to the provided Partner Transaction Manager Address
- Call the TIP Connection Manager's **GetTipConnectionFromAddress** operation with the following parameter:
  - The **Partner Transaction Manager Address** field of the Enlistment object.
- If a TIP Connection object cannot be obtained, notify the higher layer that the Push request failed and terminate the processing of this event.
- If the value of the **Allow Network Transaction** flag or the **Allow Outbound Transactions** flag is false:

- Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
  - The TIP Connection object.
- Notify the higher layer that the Push request failed and terminate the processing of this event.

- Set the Enlistment object's **TIP Connection** field to the TIP Connection object.
- Set the TIP Connection object's **Enlistment** field to the Enlistment object.
- Set the TIP Connection object's **State** field to Idle Push.
- Call the Transaction Identifier Converter's Convert Transaction Identifier to TIP Transaction Identifier operation with the following argument:
  - The Transaction Identifier field of the Transaction object referenced by the Enlistment.
- Send a PUSH (section 2.2.5)TIP Command with the following argument:
  - Return value from Transaction Identifier Converter's Convert Transaction Identifier to TIP Transaction Identifier operation.

3.2.5 Message Processing Events and Sequencing Rules

This section describes how each received TIP Command is processed. Each of these events is signaled with a TIP Command object as an input argument.

3.2.5.1 Receiving ABORTED TIP Command

When the TIP Superior Transaction Manager Facet receives an ABORTED TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to Primary, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection is not set to either Enlisted Abort, Prepared Abort, Enlisted Prepare or Enlisted Commit, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is set to either Enlisted Abort or Prepared Abort:
  - Signal the Enlistment Rollback Complete event on the Core Transaction Manager Facet with the following arguments:
    - The Enlistment object referenced by the receiving TIP Connection.
- If the **State** field of the receiving TIP Connection is not set to either Enlisted Prepare or Enlisted Commit:
  - Signal the Enlistment Phase One Complete event on the Core Transaction Manager Facet with the following arguments:
    - The Enlistment object referenced by the receiving TIP Connection object.
    - The Phase One outcome set to Aborted.

3.2.5.2 Receiving ALREADYPUSHED TIP Command

This TIP Command MUST be received with the following arguments:

- <subordinate's transaction identifier>

When the TIP Superior Transaction Manager facet receives an ALREADYPUSHED TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Primary.
  - The **State** field is set to Idle Push.
- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- Call the TIP Connection Manager's **FreeTipConnection** operation with the following argument:
  - The receiving TIP Connection object.
- Notify the higher-layer software that the Push request succeeded.

3.2.5.3 Receiving COMMITTED TIP Command

When the TIP Superior Transaction Manager facet receives a COMMITTED TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to Primary, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is not set to either Enlisted Commit or Prepared Commit, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is set to Enlisted Commit, signal the Enlistment Phase One Complete event on the Core Transaction Manager Facet with the following arguments:
  - The TIP Connection object's Enlistment object.
  - The Phase One outcome set to Committed.
- If the **State** field of the TIP Connection object is set to Prepared Commit, signal the Enlistment Commit Complete event on the Core Transaction Manager Facet with the following arguments:
  - The Enlistment object referenced by the receiving TIP Connection object.
  - The Phase One outcome set to Committed.

3.2.5.4 Receiving NOTPUSHED TIP Command

When the TIP Superior Transaction Manager facet receives a NOTPUSHED TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:

- The **Connection Type** field is set to Primary.
- The **State** field is set to Idle Push.

- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- The TIP Superior Transaction Manager facet SHOULD<11> call the TIP Connection Manager's **FreeTipConnection** operation with the following argument, to conform with the [RFC2371] specification.
  - The TIP Connection object referenced by the provided Enlistment object.
- Notify the higher-layer software that the Push request failed.

3.2.5.5 Receiving NOTRECONNECTED TIP Command

When the TIP Superior Transaction Manager Facet receives a NOTRECONNECTED TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Primary.
  - The **State** field is set to Idle Reconnect.
- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- Signal the Enlistment Commit Complete event on the Core Transaction Manager Facet with the following arguments:
  - The TIP Connection object's Enlistment object.
- Call the TIP Connection Manager's **FreeTipConnection** operation with the following argument:
  - The TIP Connection object referenced by the provided Enlistment object.

3.2.5.6 Receiving PREPARED TIP Command

When the TIP Superior Transaction Manager Facet receives a PREPARED TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to Primary, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is not set to Enlisted Prepare, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the Partner Transaction Manager **Address** field of the receiving TIP Connection object is not set, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- Signal the Enlistment Phase One Complete event on the Core Transaction Manager Facet with the following arguments:

- The Enlistment object referenced by the receiving Connection object.
- The Phase One outcome set to Prepared.

3.2.5.7 Receiving PULL TIP Command

This event MUST be received with the following arguments:

- *superior's transaction identifier*
- *subordinate's transaction identifier*

When the TIP Superior Transaction Manager Facet receives a PULL TIP Command, it MUST perform the following actions:

- If the value of the **Allow Network Transactions** flag or the **Allow Outbound Transactions** flag is false:
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
    - The provided TIP Connection object.
  - Terminate the processing of this TIP Command.
- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Secondary.

    The **State** field is set to Idle.
- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the provided *superior's transaction identifier* does not have the OleTxTipTransactionIdentifier format, as specified in section 2.2:
  - Call the TIP Connection Manager's **HasPartnerTransaction** operation with the following arguments:
    - The provided Partner Transaction Manager Address.
    - The provided *superior's transaction identifier*.
  - If **HasPartnerTransaction** returns true, send a NOTPULLED TIP Command and terminate the processing of this TIP Command.
- Call the Transaction Identifier Converter's Convert TIP Transaction Identifier to Transaction Identifier operation with the following arguments:
  - The provided *superior's transaction identifier*.
- Attempt to find a Transaction object in the Transaction Table referenced by the Core Transaction Manager Facet that meets the following conditions:
  - Transaction Identifier is set to the return value from the Transaction Identifier Converter's Convert TIP Transaction Identifier to Transaction Identifier operation.

- If a Transaction object is not found, send a NOTPULLED TIP Command and terminate the processing of this TIP Command.
- Attempt to find a Transaction object in the Transaction Table referenced by the Core Transaction Manager Facet that has an Enlistment object whose **Partner Transaction Identifier** field is not of the OleTxTipTransactionIdentifier format.
- If a Transaction object is found:
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the provided TIP Connection object. This action does not conform to the [RFC2371] specification.
  - Terminate the processing of this event.
- If the value of the **Allow TIP Pass Through** flag is false and the Transaction object has a superior TIP enlistment and no local enlistments, then send a NOTPULLED TIP Command and terminate the processing of this TIP Command.
- Create a new Enlistment Object with the following values:
  - The TIP Connection reference set to the TIP Connection object.
  - The Transaction object reference set to the Transaction object.
  - The **Partner Transaction Identifier** field set to the provided *subordinate's transaction identifier*.
  - Set the TIP Connection object's Enlistment field to the Enlistment object.
  - Signal the Create Subordinate Enlistment event on the Core Transaction Manager Facet with the following arguments:
    - The TIP Connection object's Enlistment object.

3.2.5.8 Receiving PUSHED TIP Command

This event MUST be signaled with the following argument:

- *subordinate's transaction identifier*

When the TIP Superior Transaction Manager Facet receives a PUSHED TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Primary.
  - The **State** field is set to Idle Push.
- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the provided *subordinate's transaction identifier* does not have the OleTxTipTransactionIdentifier format, as specified in section 2.2:
  - Call the TIP Connection Manager's **HasPartnerTransaction** operation with the following parameters:

- The **Partner Transaction Manager Address** field of the Enlistment object referenced by the receiving TIP Connection object.
- The provided *subordinate's transaction identifier*.

- If this operation returns true, notify the higher-layer software that the Push request failed and signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.

- Signal the Create Subordinate Enlistment event on the Core Transaction Manager Facet with the following arguments:
  - The Enlistment object referenced by the receiving TIP Connection.

3.2.5.9 Receiving QUERY TIP Command

This event MUST be signaled with the following argument: *superior's transaction identifier*

When the TIP Superior Transaction Manager Facet receives a QUERY TIP Command, it MUST perform the following actions:

- If the value of the **Allow Network Transactions** flag or the **Allow Outbound Transactions** flag is false:
  - Call the TIP Connection Manager's TerminateTipConnectio operation with the following argument:
    - The provided TIP Connection object.
  - Terminate the processing of this TIP Command.
- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Secondary.
  - The **State** field is set to Idle.
- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- Call the Transaction Identifier Converter's Convert TIP Transaction Identifier to Transaction Identifier operation with the following argument:
  - The provided *superior's transaction identifier*.
- Attempt to find a Transaction object in the Transaction Table of the Core Transaction Manager Facet that meets the following conditions:
  - Transaction Identifier set to the return value from Transaction Identifier Converter's Convert TIP Transaction Identifier to Transaction Identifier operation.
- If a Transaction object is found, send a QUERIEDEXISTS TIP Command.
- Otherwise, send a QUERIEDNOTFOUND TIP Command.

3.2.5.10 Receiving READONLY TIP Command

When the TIP Superior Transaction Manager Facet receives a READONLY TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to Primary, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- If the TIP Connection object's **State** field is not set to Enlisted Prepare, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- Signal the Enlistment Phase One Complete event on the Core Transaction Manager Facet with the following arguments:
  - The Enlistment object referenced by the receiving TIP Connection object.
  - The Phase One outcome set to Read Only.

3.2.5.11 Receiving RECONNECTED TIP Command

When the TIP Superior Transaction Manager Facet receives a RECONNECTED TIP Command object, it MUST perform the following actions:

- Test if the receiving TIP Connection object meets the following conditions:
  - The **Connection Type** field is set to Primary.
  - The **State** field is set to Idle Reconnect.
- If the receiving TIP Connection does not satisfy the conditions, signal the Invalid TIP Command event (as specified in section 3.2.7.1) and terminate the processing of this TIP Command.
- Set the TIP Connection object's **State** field to Prepared Commit.
- Send a COMMIT TIP Command.

3.2.5.12 Receiving ERROR TIP Command

When the TIP Superior Transaction Manager Facet receives an ERROR TIP Command object, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is set to Primary:
  - Reset the **TIP Connection** field of the Enlistment object Referenced by the receiving TIP Connection object.
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the TIP Connection object as the parameter.
  - Signal the Process Error event with the following arguments:
    - The receiving TIP Connection object.
- Otherwise, if the **Connection Type** field of the receiving TIP Connection object is set to Secondary:

- Set the **State** field of the receiving TIP Connection object to Error.

3.2.6 Timer Events

No timer events are defined for TIP Superior Transaction Manager Facet.

3.2.7 Other Local Events

3.2.7.1 Invalid TIP Command Event

This event overrides the event with the same name specified in section 3.1. It is used by the Superior Transaction Manager Facet.

This event must be triggered with the following argument:

- A TIP Connection object.

When this event is signaled, it MUST perform the following actions:

- Send an ERROR TIP Command.
- If the **Connection Type** field of the receiving TIP Connection object is set to Primary:
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
    - The provided TIP Connection object.
  - Signal the Process Error event with the following arguments:
    - The provided TIP Connection object.
- Otherwise, if the **Connection Type** field of the receiving TIP Connection object is set to Secondary:
  - Set the **State** field of the receiving TIP Connection object to Error.

3.2.7.2 Process Error

This event must be triggered with the following argument:

- A TIP Connection object.

If the Process Error event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:

- If the **Connection Type** field of the provided TIP Connection object is set to Primary:
  - If the **State** field of the provided TIP Connection object is set to Idle Push:
    - Notify the higher-layer business logic that the Push request failed.
  - Otherwise, if the State field of the provided TIP Connection object is set to Idle Reconnect:
    - Signal the Begin Commit event on the TIP Superior Transaction Manager Facet with the following argument:
      - The Connection object's Enlistment object.

- Otherwise, if the **State** field of the provided TIP Connection object is set to either Enlisted Prepare or Enlisted Commit:
  - Signal the Phase One Complete event on the Core Transaction Manager Facet with the following arguments:
    - The Enlistment object referenced by the provided TIP Connection object.
    - The Outcome set to Aborted.
- Otherwise, if the **State** field of the provided TIP Connection object is set to either Enlisted Abort or Prepared Abort:
  - Signal the Enlistment Rollback Complete event on the Core Transaction Manager Facet with the following argument:
    - The Enlistment object referenced by the provided TIP Connection object.
- Otherwise, if the **State** field of the provided TIP Connection object is set to Prepared Commit:
  - Signal the Begin Commit event on the TIP Superior Transaction Manager Facet with the following argument:
    - The Enlistment object referenced by the provided TIP Connection object.

3.2.7.3 Events Signaled by the Core Transaction Manager Facet

3.2.7.3.1 Begin Commit

This event must be triggered by the Core Transaction Manager Facet with the following arguments:

- An Enlistment object

If the Begin Commit event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:

- If the TIP Connection object referenced by the provided Enlistment object is set:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment Object to Prepared Commit
  - Send a COMMIT TIP Command
- Otherwise:
  - Call the TIP Connection Manager's **GetTipConnectionFromAddress** operation with the following parameter:
    - The **Partner Transaction Manager Address** field of the provided Enlistment object.
  - If a TIP Connection object cannot be obtained:
    - Rerun this event with the same arguments.
- Otherwise:
  - If the value of the **Allow Network Transactions** flag ot the **Allow Outbound Transactions** flag is false:

- Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
  - The TIP Connection object.
- Terminate the processing of this event.

- Set the **TIP Connection** field of the provided Enlistment object to the TIP Connection
- Set the Enlistment referenced by the TIP Connection to be the provided Enlistment
- Set the State field of the TIP Connection to Idle Reconnect.
- Send a RECONNECT (section 2.2.8) TIP Command with the following argument:
  - The Enlistment object's **Partner Transaction Identifier** field.

3.2.7.3.2 Begin Phase One

This event MUST be signaled by the Core Transaction Manager Facet with the following arguments:

- An Enlistment object.
- A **Single-Phase Commit** flag indicating whether the transaction SHOULD or MUST NOT attempt to perform a Single-Phase commit.

If the Begin Phase One event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:

- If the Single-Phase Commit flag is set to true:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment object to Enlisted Commit.
  - Send a COMMIT TIP Command.
- Otherwise:
  - Set the State field of the TIP Connection object referenced by the provided Enlistment object to Enlisted Prepare.
  - Send a PREPARE TIP Command.

3.2.7.3.3 Begin Rollback

This event MUST be signaled by the Core Transaction Manager Facet with the following argument:

- An Enlistment object.

If the Begin Rollback event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:

- If the **State** field of the TIP Connection object referenced by the provided Enlistment object is set to Enlisted:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment object to Enlisted Abort.

- Otherwise, if the **State** field of the TIP Connection object referenced by the provided Enlistment object is set to Prepared:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment object to Prepared Abort.
- Send an ABORT TIP Command.

3.2.7.3.4 Create Subordinate Enlistment Failure

This event MUST be signaled by the Core Transaction Manager Facet with the following arguments:

- An Enlistment object.
- A Failure Reason.

If the Create Subordinate Enlistment Failure event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:

- Test if the TIP Connection object referenced by the provided Enlistment object meets the following conditions:
  - The **Connection Type** field is set to Primary.
  - The **State** field is set to Idle Push.
- If the TIP Connection object referenced by the provided Enlistment object satisfies both the conditions:
  - Notify the higher-layer business logic that the Push request failed.
  - Send an ABORTED TIP Command. This action does not conform to the [RFC2371] specification.
  - Reset the TIP Connection object referenced by the Enlistment object referenced by the provided TIP Connection object.
  - Call the TIP Connection Manager's **TerminateTIPConnection** operation with the following argument:
    - The provided TIP Connection object.
  - Signal the Process Error event with the following arguments:
    - The TIP Connection object referenced by the provided Enlistment object
- Otherwise, if the **Connection Type** field of the TIP Connection object referenced by the provided Enlistment object is set to Secondary
  - Send a NOTPULLED TIP Command.

3.2.7.3.5 Create Subordinate Enlistment Success

This event MUST be signaled by the Core Transaction Manager Facet with the following arguments:

- An Enlistment object.

If the Create Subordinate Enlistment Success event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:

- Set the **State** field of the TIP Connection object referenced by the provided Enlistment object to Enlisted.
- Test if the TIP Connection object referenced by the provided Enlistment object meets the following conditions:
  - The **Connection Type** field is set to Primary.
  - The **State** field is set to Idle Push.
- If the TIP Connection object referenced by the provided Enlistment object satisfies the conditions:
  - Notify the higher-layer business logic that the Push request succeeded.
- Otherwise, if the **Connection Type** field of the TIP Connection object referenced by the provided Enlistment object is set to Secondary
  - Send a PULLED TIP Command.
  - Set the **Connection Type** field of the TIP Connection object referenced by the provided Enlistment object to Primary.

3.2.7.4 Transport Events

3.2.7.4.1 Transport Connection Down

This event overrides the event with the same name in section 3.1.7.2.2.

This event must be triggered with the following argument:

- A TIP Connection object.
- If the Transport Connection Down event is signaled, the TIP Superior Transaction Manager Facet MUST perform the following actions:
  - Reset the TIP Connection object referenced by the Enlistment object referenced by the provided TIP Connection object.
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
    - The provided TIP Connection object.
  - Signal the Process Error event with the following argument:
    - The provided TIP Connection object.

3.3 TIP Subordinate Transaction Manager Facet Details

This section contains protocol details that relate to the TIP Subordinate Transaction Manager Facet protocol role.

3.3.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

Note that the abstract data model can be implemented in a variety of ways. This protocol does not prescribe or advocate any specific implementation technique.

The TIP Subordinate Transaction Manager Facet MUST extend the definition of the **State** field of the TIP Connection object as specified in section 3.1.1.1 to include the following values:

- **State**: An enumeration that indicates what state the TIP Connection is in. This field MUST be set to one of the values in the extended enumeration. The following are the extension values:
  - **Idle Pull**: The TIP Connection is waiting for a reply to a PULL TIP Command sent while in the idle state.
  - **Idle Query**: The TIP Connection is waiting for a reply to a QUERY TIP Command is sent in the Idle state.
  - **Enlisted**: The TIP Connection is associated with a transaction object and will receive TIP Commands from its Partner Transaction Manager regarding Transaction's Outcome.
  - **Enlisted Prepare**: The TIP Connection is processing the PREPARE TIP Command received while in the Enlisted state.
  - **Enlisted Commit**: The TIP Connection is processing the COMMIT TIP Command received while in the Enlisted state.
  - **Prepared**: The transaction associated with the TIP Connection has completed Phase 1.
  - **Aborted**: The TIP Connection will send the ABORTED TIP Command as the associated Transaction has been unilaterally aborted.

3.3.1.1 TIP Subordinate Transaction Manager Facet State Transition Table

The following table summarizes the state transitions that are legal to the protocol as seen by the TIP Subordinate Transaction Manager Facet. The states are the TIP Connection states. The table omits the following transitions:

- In every state, the TIP Subordinate Transaction Manager Facet, acting as a Primary, is allowed to send an ERROR TIP Command, which changes the state to Error.
- In every state, the TIP Subordinate Transaction Manager Facet, acting as a Secondary, may receive an ERROR TIP Command, which changes the state to Error.

The following events trigger a state transition:

- A TIP request is sent to the Partner Transaction Manager
- A TIP reply is received from the Partner Transaction Manager
- A TIP reply is sent to the Partner Transaction Manager in response to a TIP request.

The "<TIP command A> received/<TIP response B> sent" syntax in the table indicates that the Facet received <TIP command A> and decided to respond to it with <TIP response B>. The state changes from <current state> to the <next state> when <TIP response B> is sent to the Partner Transaction Manager.

Current state	Event	Next state
Idle	PULL sent.	Idle Pull
Idle Pull	PULLED received.	Enlisted
Idle Pull	NOTPULLED received.	Idle
Idle Pull	ERROR received.	Error
Idle	PUSH received/PUSHED sent.	Enlisted.
Idle	PUSH received/ALREADYPUSHED sent.	Idle
Idle	PUSH received/NOTPUSHED sent.	Idle
Idle	PUSH received/ERROR sent.	Error
Idle	QUERY sent.	Idle Query
Idle Query	QUERIEDEXISTS received.	Idle
Idle Query	QUERIEDNOTFOUND received.	Idle
Idle Query	ERROR received.	Error.
Idle	RECONNECT received/RECONNECTED sent.	Prepared
Idle	RECONNECT received/NOTRECONNECTED sent.	Idle
Idle	RECONNECT received/ERROR sent.	Error
Enlisted	ABORT received/ABORTED sent.	Idle
Enlisted	ABORT received/ERROR sent.	Error
Enlisted	COMMIT received/ABORTED sent.	Idle
Enlisted	COMMIT received/COMMITTED sent.	Idle
Enlisted	COMMIT received/ERROR sent	Error
Enlisted	PREPARE received/PREPARED sent.	Prepared
Enlisted	PREPARE received/ABORTED sent.	Idle
Enlisted	PREPARE received/READONLY sent.	Idle
Enlisted	PREPARE received/ERROR sent.	Error
Prepared	ABORT received/ABORTED received.	Idle
Prepared Abort	ABORT received/ERROR sent.	Error
Prepared	COMMIT received/COMMITTED sent.	Idle

Current state	Event	Next state
Prepared Commit	COMMIT received/ERROR sent.	Error

3.3.2 Timers

3.3.2.1 Query Timer

The Query Timer MUST be started whenever Recovery work needs to be performed for an In Doubt Transaction, as specified by the processing of the Receiving QUERIEDEXISTS TIP Command and Process Error events.

The Query Timer MUST be canceled by the processing of the Receiving RECONNECT TIP Command and by the processing of the Query Timer Expired event.

The value of the timer is set to an implementation-specific value.<12>

When the timer is initialized, the TIP Subordinate Transaction Manager Facet MUST provide an Enlistment object to associate with the timer.

When the timer expires, the Query Timer Expired event is signaled.

3.3.3 Initialization

The TIP Subordinate Transaction Manager Facet MUST perform all initialization as specified in section 3.1.3

3.3.4 Higher-Layer Triggered Events

3.3.4.1 Pull Transaction

The Pull Transaction event MUST be signaled by the higher-layer business logic with the following arguments:

- Partner Transaction Manager Address.
- Superior Transaction Identifier.

If the Pull Transaction event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- If the provided Partner Transaction Identifier does not have the OleTxTipTransactionIdentifier format:
  - Call the TIP Connection Manager's HasPartnerTransaction operation with the following arguments:
    - The provided Partner Transaction Manager Address.
    - The provided Superior Transaction Identifier.
  - If HasPartnerTransaction returns true, then the Pull has already been completed, so notify the higher-layer business logic that the Pull request was completed successfully and terminate the processing of this event.
- Otherwise:

- Create a new Transaction object with a new GUID as an identifier.
- Create a new Enlistment object with the following values:
  - The Transaction field is set to the created Transaction object.
  - The Partner Transaction Manager Address is set to the provided Partner Transaction Manager Address.
  - The Partner Transaction Identifier is set of the provided Superior Transaction Identifier.
- Signal the Create Superior Enlistment event on the Core Transaction Manager Facet with the following arguments:
  - The new Enlistment object.

3.3.5 Message Processing Events and Sequencing Rules

This section describes how each received TIP Command is processed. Each of these events is signaled with a TIP Command object as an input argument.

3.3.5.1 Receiving ABORT TIP Command

When the TIP Subordinate Transaction Manager Facet receives an ABORTED TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to **Secondary**, signal the Invalid TIP Command event and terminate the processing of this event.
- If the **State** field of the receiving TIP Connection object is not set to either **Aborted**, **Enlisted**, or **Prepared**, signal the Invalid TIP Command event and terminate the processing of this event.
- If the **State** field of the receiving TIP Connection object is set to **Aborted**:
  - Send an ABORTED TIP Command.
- Otherwise:
  - Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
    - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.

3.3.5.2 Receiving COMMIT TIP Command

When the TIP Subordinate Transaction Manager Facet receives a COMMIT TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to **Secondary**, signal the Invalid TIP Command event and terminate the processing of this event.
- If the **State** field of the receiving TIP Connection object is not set to either **Aborted**, **Enlisted**, or **Prepared**, signal the Invalid TIP Command event and terminate the processing of this event.
- If the **State** field of the receiving TIP Connection object is set to Enlisted:
  - Set the State field of the receiving TIP Connection object to Enlisted Commit.

- Signal the Begin Phase Zero event on the Core Transaction Manager Facet with the following argument:
  - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.

- Otherwise, if the State field of the receiving TIP Connection object is set to Prepared:
  - Signal the Begin Commit event on the Core Transaction Manager Facet with the following argument:
    - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.

- Otherwise:
  - Send an ABORTED TIP Command.

3.3.5.3 Receiving NOTPULLED TIP Command

When the TIP Subordinate Transaction Manager Facet receives a NOTPULLED TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meets any of the following conditions:
  - The **State** field of the receiving TIP Connection object is not set to **Idle Pull**.
  - The **Connection Type** field of the receiving TIP Connection object is not set to **Primary**
- If the receiving TIP Connection satisfies any of the conditions, signal the Invalid TIP Command event and terminate the processing of this event.
- Notify the higher-layer business logic that the Pull request failed.
- Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
  - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.

3.3.5.4 Receiving PREPARE TIP Command

When the TIP Subordinate Transaction Manager Facet receives a PREPARED TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to Secondary, signal the Invalid TIP Command event and terminate the processing of this event.
- If the **State** field of the receiving TIP Connection object is not set to either **Aborted** or **Enlisted**, signal the Invalid TIP Command event and terminate the processing of this event.
- If the **State** field of the receiving TIP Connection object is set to **Enlisted**:
  - Set the **State** field of the receiving TIP Connection object is set to **Enlisted Prepare**.
  - Signal the Begin Phase Zero event on the Core Transaction Manager Facet with the following argument:

        - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.
- Otherwise:
    - Send an ABORTED TIP Command.

3.3.5.5 Receiving PULLED TIP Command

When the TIP Subordinate Transaction Manager Facet receives a PULLED TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meet any of the following conditions:
    - The **State** field of the receiving TIP Connection object is not set to **Idle Pull**.
    - The **Connection Type** field of the receiving TIP Connection object is not set to **Primary**.
- If the receiving TIP Connection satisfies any of the conditions, signal the Invalid TIP Command event and terminate the processing of this event.
- Set the **Connection Type** field of the receiving TIP Connection object to **Secondary**.
- Set the **State** field of the receiving TIP Connection object to **Enlisted**.
- Notify the higher-layer business logic that the Pull request succeeded.

3.3.5.6 Receiving PUSH TIP Command

This TIP Command MUST be received with the following arguments:

- *superior's transaction identifier*.

When the TIP Subordinate Transaction Manager Facet receives a PUSH TIP Command, it MUST perform the following actions:

- If the value of the **Allow Network Transactions** flag or the **Allow Inbound Transactions** flag is false:
    - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
        - The provided TIP Connection object.
    - Terminate the processing of this TIP Command.
- Test if the receiving TIP Connection object meet any of the following conditions:
    - The **State** field of the receiving TIP Connection object is not set to **Idle**
    - The **Connection Type** field of the receiving TIP Connection object is not set to **Secondary**
- If the receiving TIP Connection satisfies any of the conditions, signal the Invalid TIP Command event and terminate the processing of this event.
- If the **Partner Transaction Manager Address** field of the receiving TIP Connection object is not set, send a NOTPUSHED TIP Command. This action does not conform to the [RFC2371] specification.

- If the provided <*superior's transaction identifier*> does not have the OleTxTransactionIdentifier format:
  - Call the TIP Connection Manager's **HasPartnerTransaction** operation with the following arguments:
    - The **Partner Transaction Manager Address** field of the Enlistment object referenced by the receiving TIP Connection object.
    - The provided <*superior's transaction identifier*>.
  - If the **HasPartnerTransaction** returns true:
    - Send a ALREADYPUSHED (section 2.2.1) TIP Command with the following argument:
      - The **Transaction Identifier** field of the Transaction object referenced by the Enlistment object.
    - Terminate the processing for this TIP Command.
- Create a new Transaction object with the following value:
  - The **Transaction Identifier** field is set to a new GUID value.
- Create a new Enlistment object with the following values:
  - The TIP Connection object set to the receiving TIP Connection object.
  - The **Partner Transaction Identifier** field is set to the provided *superior's transaction identifier*.
  - The Transaction object reference set to new Transaction object.
- Set the Enlistment reference of the receiving TIP Connection object to the new Enlistment object.

  Signal the Create Superior Enlistment event on the Core Transaction Manager Facet with the following arguments:
  - The Enlistment object referenced by the receiving TIP Connection object.

3.3.5.7 Receiving QUERIEDEXISTS TIP Command

When the TIP Subordinate Transaction Manager Facet receives a QUERIEDEXISTS TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meet any of the following conditions:
  - The **State** field of the receiving TIP Connection object is not set to **Idle Query**.
  - The **Connection Type** field of the receiving TIP Connection object is not set to **Primary**.
- If the receiving TIP Connection satisfies any of the conditions, signal the Invalid TIP Command event and terminate the processing of this event.
- Create a Query Timer with the following arguments:
  - The Enlistment object referenced by the receiving TIP Connection object.
  - An implementation specific timer value.

- Set the **State** field of the receiving TIP Connection object to **Idle**.
- Call the TIP Connection Manager's **FreeTipConnection** operation with the following argument:
  - The provided TIP Connection object.

3.3.5.8 Receiving QUERIEDNOTFOUND TIP Command

When the TIP Subordinate Transaction Manager Facet receives a QUERIEDNOTFOUND TIP Command, it MUST perform the following actions:

- Test if the receiving TIP Connection object meet any of the following conditions:
  - The **State** field of the receiving TIP Connection object is not set to **Idle Query**.
  - The **Connection Type** field of the receiving TIP Connection object is not set to **Primary**
- If the receiving TIP Connection satisfies any of the conditions, signal the Invalid TIP Command event and terminate the processing of this event.
- Signal the Begin Rollback event on the Core Transaction Manager Facet with the following arguments:
  - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.
- Call the TIP Connection Manager's **FreeTipConnection** operation with the following argument:
  - The provided TIP Connection object.

3.3.5.9 Receiving RECONNECT TIP Command

This TIP Command MUST be received with the following argument:

- *subordinate's transaction identifier*

When the TIP Subordinate Transaction Manager Facet receives a RECONNECT TIP Command, it MUST perform the following actions:

- If the value of the **Allow Network Transactions** flag or the **Allow Inbound Transactions** flag is false:
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
    - The provided TIP Connection object.
  - Terminate the processing of this TIP Command.
- Test if the receiving TIP Connection object meets any of the following conditions:
  - The **State** field of the receiving TIP Connection object is not set to **Idle**.
  - The **Connection Type** field of the receiving TIP Connection object is not set to **Secondary**.
- If the receiving TIP Connection satisfies any of the conditions, signal the Invalid TIP Command event and terminate the processing of this event.

- Test the Transaction Table to see if there is Transaction object that meet the following conditions:
  - Its **Transaction Identifier** field is set to the provided *subordinate's transaction identifier*.
  - Its **Superior Enlistment** field contains an Enlistment object whose **TIP Connection** field contains a TIP Connection object whose Partner Transaction Manager Address is set to the Partner Transaction Manager Address of the receiving TIP Connection object.
- If these conditions are not met, signal the Invalid TIP Command event and terminate the processing of this event.
- The **TIP Connection** field of the Enlistment object referenced by the Superior Enlistment field of this Transaction object indicates whether this TIP Subordinate Transaction Manager Facet is awaiting the reply for a QUERYTIP Command for the same Transaction. Wait until this field is not set before continuing the processing of this event.
- Call the Transaction Identifier Converter's Convert TIP Transaction Identifier to Transaction Identifier operation with the following argument:
  - *subordinate's transaction identifier*.
- Attempt to find the Transaction Identifier returned from the Transaction Identifier Converter's Convert TIP Transaction Identifier to Transaction Identifier operation in the Transaction table of the Core Transaction Manager Facet.
- If a Transaction object is not found:
  - Send a NOTRECONNECTED TIP Command.
- Otherwise:
  - If the **Partner Transaction Manager Address** field of the Superior Enlistment referenced by the Transaction object is not set to **Partner Transaction Manager Address** field of the TIP Connection object:
    - Send a NOTRECONNECTED TIP Command.
- Otherwise:
  - Test if the following conditions are all true, which indicates that there is an outstanding QUERYTIP Command that this TIP Subordinate Transaction Manager Facet sent for this transaction:
    - The **State** field of the receiving TIP Connection is set to **Idle Query**.
    - The TIP Connection object of the Superior Enlistment referenced by the Transaction object is set
  - If the above conditions are all true:
    - Wait for the TIP Connection object referenced by the Superior Enlistment referenced by the Transaction object to be reset when a response to the QUERY TIP Command is received.
  - Otherwise:
    - Cancel the Query Timer associated with the Superior Enlistment referenced by the Transaction object, if started.

- Reset the TIP Connection object referenced by the Superior Enlistment referenced by the Transaction object.
- Set the TIP Connection field of the Superior Enlistment object of the Transaction object to the receiving TIP Connection object.
- Send a RECONNECTED TIP Command.

3.3.5.10 Receiving ERROR TIP Command

When the TIP Subordinate Transaction Manager Facet receives an ERROR TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is set to **Primary**:
  - Reset the **TIP Connection** field of the Enlistment object referenced by the receiving TIP Connection object.
  - Call the TIP Connection Manager's **FreeTipConnection** operation with the following argument:
    - The receiving TIP Connection object.
  - Signal the Process Error event with the following argument:
    - The receiving TIP Connection object

3.3.6 Timer Events

3.3.6.1 Query Timer Expired Event

The Query Timer Expired event MUST be signaled with the following arguments:

- An Enlistment object.

If the Query Timer Expired event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- Signal the Recover In Doubt Transaction event with the provided Enlistment object.

3.3.7 Other Local Events

A TIP Subordinate Transaction Manager Facet MUST be prepared to process the local events specified in the following sections.

3.3.7.1 Invalid TIP Command Event

This event overrides the event with the same name in section 3.1. It is used by the TIP Subordinate Transaction Manager Facet.

This event must be triggered with the following argument:

- A TIP Connection object.

If the Invalid TIP Command event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- Send an ERROR TIP Command.
- If the **Connection Type** field of the receiving TIP Connection object is set to **Primary**:
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
    - The provided TIP Connection object
  - Signal the Process Error event with the following arguments:
    - The provided TIP Connection object

3.3.7.2 Process Error

The Process Error event MUST be signaled with the following argument:

- A TIP Connection object.

If the Process Error event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- If the **Connection Type** field of the provided TIP Connection object is set to **Primary**:
  - If the **State** field of the provided TIP Connection object Idle Pull:
    - Set the **State** field of the provided TIP Connection object to **Error**.
    - Notify the higher-layer business logic that the Pull request failed.
    - Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
      - The Transaction object referenced by the Enlistment object referenced by the provided TIP Connection object.
  - Otherwise, if the **State** field of the provided TIP Connection object is set to **Idle Query**:
    - Reset the TIP Connection referenced by the Enlistment object referenced by the provided TIP Connection object.
    - Create a Query Timer with the following arguments:
      - The Enlistment object referenced by the provided TIP Connection object.
      - An implementation specific timer value.
- Otherwise, if the **Connection Type** field of the provided TIP Connection object is set to **Secondary**:
  - If the **State** field of the provided TIP Connection object is set to either **Initial** or **Idle** or **Initial Indentify**:
    - Set the **State** field of the provided TIP Connection object to **Error**.
  - Otherwise, if the **State** field of the provided TIP Connection object is set to **Enlisted**:
    - Set the **State** field of the provided TIP Connection object to **Error**.

- Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
  - The Transaction object referenced by the Enlistment object referenced by the provided TIP Connection object.
- Otherwise, if the **State** field of the provided TIP Connection object is set to **Prepared**:
  - Reset the TIP Connection referenced by the Enlistment object referenced by the provided TIP Connection object.
  - Create a Query Timer with the following arguments:
    - The Enlistment object referenced by the provided TIP Connection object.
    - An implementation specific timer value.
- Otherwise, if the **State** field of the provided TIP Connection object is set to **Aborted**:
  - Set the **State** field of the provided TIP Connection object to **Error**.

3.3.7.3 Events Signaled by the Core Transaction Manager Facet

3.3.7.3.1 Commit Complete

The Commit Complete event MUST be signaled with the following arguments:

- An Enlistment object.

If the Commit Complete event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- Set the **State** field of the TIP Connection object referenced by the provided Enlistment is to **Idle**.
- Send a COMMITTED TIP Command.

3.3.7.3.2 Create Superior Enlistment Success

The Create Superior Enlistment Success event MUST be signaled with the following arguments:

- An Enlistment object.

If the Create Superior Enlistment Success event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- If the provided Enlistment object does not have a TIP Connection, then a higher-level pull request is pending, so do the following:
  - Call the TIP Connection Manager's **GetTipConnectionFromAddress** operation with the following parameter:
    - The **Partner Transaction Manager Address** field of the provided Enlistment object.
  - If a TIP Connection object cannot be obtained:
    - Notify the higher-layer business logic that the Pull request failed.
    - Terminate the processing of this event.

- If the value of the **Allow Network Transactions** flag or the Allow Inbound Transactions flag is false:
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
    - The TIP Connection object referenced by the provided Enlistment object.
  - Notify the higher-layer business logic that the Pull request failed.
  - Terminate the processing of this event.
- Send a PULLED TIP Command.

- Otherwise, there is a pending remote push message pending, so do the following:
  - Send a PUSHED (section 2.2.6)TIP Command with the following argument:
    - The **Transaction Identifier** field of the Transaction object referenced by the provided Enlistment object.

3.3.7.3.3 Create Superior Enlistment Failure

The Create Superior Enlistment Failure event MUST be signaled with the following arguments:

- An Enlistment object.
- A Failure Reason.

If the Create Superior Enlistment Failure event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- If the provided Enlistment object does not have a TIP Connection, then a pending higher-level pull request is pending, so do the following:
  - Notify the higher-layer business logic that the Pull request failed.
- Otherwise, there is a pending remote push message pending, so do the following:
  - Send a NOTPUSHED TIP Command.

3.3.7.3.4 Phase Zero Complete

The Phase Zero Complete event MUST be signaled with the following arguments:

- An Enlistment object.
- An outcome value. This value MUST be set to one of the following values:
  - Success
  - Failure

If the Phase Zero Complete event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- If the provided outcome is Failure:

- Send an ABORTED TIP Command.

- Otherwise:

  - If the **State** field of the TIP Connection object referenced by the provided Enlistment is set to **Enlisted Prepare**:

    - Signal the Begin Phase One event on the Core Transaction Manager Facet with the following arguments:

      - The transaction object referenced by the provided Enlistment object.

      - A Single-Phase value set to False.

  - If the **State** field of the TIP Connection object referenced by the provided Enlistment is set to **Enlisted Commit**:

    - Signal the Begin Phase One event on the Core Transaction Manager Facet with the following arguments:

      - The transaction object referenced by the provided Enlistment object.

      - A Single-Phase value set to True.

3.3.7.3.5 Phase One Complete

The Phase One Complete event MUST be signaled with the following arguments:

- An Enlistment object.

- A value indicating the outcome of Phase One. This value MUST be set to one of the following values:

  - Read-Only

  - Prepared

  - Committed

  - Aborted

  - In Doubt

If the Phase One Complete event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- If the provided outcome is In Doubt:

  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:

    - The provided TIP Connection object.

- Otherwise:

  - If the provided outcome is Committed:

    - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to **Idle**.

        - Send a COMMITTED TIP Command.
    - If the provided outcome is Aborted:
        - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to **Idle**.
        - Send an ABORTED TIP Command.
    - If the provided outcome is Read-Only:
        - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to **Idle**.
        - Send an READONLY TIP Command.
    - If the provided outcome is Prepared:
        - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to **Prepared**.
        - Send a PREPARED TIP Command.

3.3.7.3.6 Recover In Doubt Transaction

The Recover In Doubt Transaction event MUST be signaled with the following arguments:

- An Enlistment object

If the Recover In Doubt Transaction event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- Cancel the Query Timer associated with the provided Enlistment object.
- Call the TIP Connection Manager's **GetTipConnectionFromAddress** operation with the following parameter:
    - The **Partner Transaction Manager Address** field of the provided Enlistment object.
- If a TIP Connection object cannot be obtained:
    - Create a Query Timer with the following arguments:
        - The provided Enlistment object.
        - An implementation-specific timer value.
- Otherwise:
    - If the value of the **Allow Network Transactions** flag or the Allow Inbound Transactions flag is false:
        - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
            - The provided TIP Connection object.
        - Terminate the processing of this TIP command.

- Set the TIP Connection field of the provided Enlistment object to the TIP Connection.
- Set the Enlistment referenced by the TIP Connection to be the provided Enlistment.
- Set the **State** of the TIP Connection object to Idle.
- Send a QUERY TIP command with the following argument:
  - The **Partner Transaction Identifier** field of the Enlistment object referenced by the TIP Connection object.

3.3.7.3.7 Rollback Complete

The Rollback Complete event MUST be signaled with the following arguments:

- An Enlistment object.

If the Rollback Complete event is signaled, the TIP Subordinate Transaction Manager Facet (section 1.3.1.2.2) MUST perform the following actions:

- If the **State** field of the TIP Connection object referenced by the provided Enlistment is set to Idle:
  - If the TIP Connection object referenced by the provided Enlistment object is set:
    - Call the TIP Connection Manager's **FreeTipConnection** operation with the following argument:
      - The TIP Connection object referenced by the provided Enlistment object.
- If the **State** field of the TIP Connection object referenced by the provided Enlistment is set to either Enlisted or Prepared:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to Idle.
  - Send an ABORTED TIP Command.

3.3.7.3.8 Unilaterally Aborted

The Unilaterally Aborted event MUST be signaled with the following arguments:

- An Enlistment object.

If the Unilaterally Aborted event is signaled, the TIP Subordinate Transaction Manager Facet (section 1.3.1.2.2) MUST perform the following actions:

- If the **State** field of the TIP Connection object referenced by the provided Enlistment is set to Enlisted:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to **Aborted**.

3.3.7.4 Transport Events

3.3.7.4.1 Transport Connection Down

This event overrides the event with the same name as specified in section 3.1.7.2.2.

The Transport Connection Down event MUST be signaled with the following argument:

- A TIP Connection object.

If the Transport Connection Down event is signaled, the TIP Subordinate Transaction Manager Facet MUST perform the following actions:

- Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
  - The provided TIP Connection object.
- Signal the Process Error event with the following argument:
  - The provided TIP Connection object.

3.4 TIP Transaction Manager Communicating with an Application Facet Details

This section contains protocol details that relate to the TIP Transaction Manager Communicating with an Application Facet protocol role.

3.4.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

Note that the abstract data model can be implemented in a variety of ways. This protocol does not prescribe or advocate any specific implementation technique.

The TIP Transaction Manager Communicating with an Application Facet MUST extend the definition of the State field of the TIP Connection object as specified in section 3.1.1.1 to include the following values:

- **State**: An enumeration that indicates what state the TIP Connection is in. This field MUST be set to one of the values in the extended enumeration. The following are the extension values:
  - **Begun**: The TIP Connection is associated with an Active Transaction.
  - **Aborted**: The TIP Connection sends an ABORTED TIP Command as the associated Transaction has been unilaterally aborted.

3.4.1.1 TIP Transaction Manager Communicating with an Application Facet State Transition Table

The following table summarizes the state transitions that are legal to the protocol as seen by the TIP Transaction Manager Communicating with an Application Facet. The states are the TIP Connection states. The table omits the following transitions:

- In every state, the TIP Transaction Manager Communicating with an Application Facet, acting as a Secondary, may receive an ERROR TIP Command, which changes the state to Error.
- The state changes when a TIP reply is sent to the Partner Transaction Manager in response to a TIP request.

The "<TIP command A> received/<TIP response B> sent" syntax in the table indicates that the Facet received <TIP command A> and decided to respond to it with <TIP response B>. The state changes from <current state> to the <next state> when <TIP response B> is sent to the Partner Transaction Manager.

Current state	Event	Next state
Idle	BEGIN received/BEGUN sent.	Begun
Idle	BEGIN received/NOTBEGUN sent.	Idle
Idle	BEGIN received/ERROR sent.	Error
Begun	COMMIT received/COMMITTED sent.	Idle
Begun	COMMIT received/ABORTED sent.	Idle
Begun	COMMIT received/Error sent.	Error
Begun	ABORT received/ABORTED sent.	Idle
Begun	ABORT received/ERROR sent.	Error

3.4.2 Timers

None.

3.4.3 Initialization

The TIP Transaction Manager Communicating with an Application Facet MUST perform all initialization as specified in section 3.1.3.

3.4.4 Higher-Layer Triggered Events

No higher-layer events apply here.

3.4.5 Message Processing Events and Sequencing Rules

This section describes how each received TIP Command is processed. Each of these events is signaled with a TIP Command object as an input argument.

3.4.5.1 Receiving ABORT TIP Command

When the TIP Transaction Manager Communicating with an Application Facet receives an ABORT TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to Secondary, signal the Invalid TIP Command event and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is not set to either **Aborted** or **Begun**, signal the Invalid TIP Command event and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is set to **Aborted**:
  - Send an ABORTED TIP Command.

- Otherwise, if the **State** field of the receiving TIP Connection object is set to **Begun**:
  - Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
    - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.

3.4.5.2 Receiving BEGIN TIP Command

When the TIP Transaction Manager Communicating with an Application Facet receives a BEGIN TIP Command, it MUST perform the following actions:

- If the value of the **Allow Network Transactions** flag or the **Allow Inbound Transactions** flag is false:
  - Call the TIP Connection Manager's **TerminateTipConnection** operation with the following argument:
    - The provided TIP Connection object.
  - Terminate the processing of this TIP Command.
- Test if the receiving TIP Connection object meets any of the following conditions:
  - The **Connection Type** field of the receiving TIP Connection object is not set to Secondary
  - The **State** field of the receiving TIP Connection object is not set to **Idle**
  - The value of the **Allow Begin** flag is false.
- If the receiving TIP Connection object satisfies any of the conditions, signal the invalid TIP Command event and terminate the processing of this TIP Command.
- Create a new Transaction object with the following value:
  - The **Transaction Identifier** field is set to a new GUID value.
- Set the Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object to the new Transaction object.
- Signal the Create Transaction event on the Core Transaction Manager Facet with the following argument:
  - The Enlistment object referenced by the receiving TIP Connection object.

3.4.5.3 Receiving COMMIT TIP Command

When the TIP Transaction Manager Communicating with an Application Facet receives a COMMIT TIP Command, it MUST perform the following actions:

- If the **Connection Type** field of the receiving TIP Connection object is not set to Secondary, signal the Invalid TIP Command event and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is not set to either **Begun** or **Aborted**, signal the Invalid TIP Command event and terminate the processing of this TIP Command.
- If the **State** field of the receiving TIP Connection object is set to **Begun**:

- Signal the Begin Phase Zero event on the Core Transaction Manager Facet with the following argument:
  - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.
- Otherwise, if the **State** field of the receiving TIP Connection object is set to **Aborted**:
  - Send an ABORTED TIP Command.

3.4.5.4 Receiving ERROR TIP Command

When the TIP Transaction Manager Communicating with an Application Facet receives an ERROR TIP Command object, it MUST perform the following actions:

- If the **State** field of the receiving TIP Connection object is set to Begun:
  - Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
    - The Transaction object referenced by the Enlistment object referenced by the receiving TIP Connection object.
  - Set the **State** field of the receiving TIP Connection object to Error.

3.4.6 Timer Events

None.

3.4.7 Other Local Events

A TIP Transaction Manager Communicating with an Application Facet MUST be prepared to process the local events specified in the following sections.

3.4.7.1 Invalid TIP Command Event

This event overrides the event with the same name as specified in section 3.1. It is used by the TIP Transaction Manager Communicating with an Application Facet.

The Invalid TIP Command event must be triggered with the following argument:

- A TIP Connection object.

If the Invalid TIP Command Event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- If the **State** field of the receiving TIP Connection object is set to **Begun**:
  - Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
    - The Enlistment object referenced by the provided TIP Connection object.
- Otherwise, if the **State** field of the receiving TIP Connection object is not set to Error:
  - Send an ERROR TIP Command.

3.4.7.2 Events Signaled by the Core Transaction Manager Facet

3.4.7.2.1 Create Transaction Failure

The Create Transaction Failure event MUST be signaled with the following arguments:

- An Enlistment object.
- A Failure Reason.

If the Create Transaction Failure event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- Send a NOTBEGUN TIP Command.

3.4.7.2.2 Create Transaction Success

The Create Transaction Success event MUST be signaled with the following arguments:

- A Transaction object.

If the Create Transaction Success event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- Set the **State** field of the Superior Enlistment referenced by the provided Transaction object to Begun.
- Signal the Convert Transaction Identifier to TIP Transaction Identifier event with the following argument:
  - The **Transaction Identifier** field of the provided Transaction object.
- Send a BEGUN TIP Command with the following argument:
  - The return value from the Convert Transaction Identifier to TIP Transaction Identifier event.

3.4.7.2.3 Phase Zero Complete

The Phase Zero Complete event MUST be signaled with the following arguments:

- An Enlistment object.
- An outcome value. This value MUST be set to one of the following values:
  - Success
  - Failure

If the Phase Zero Complete event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- If the provided outcome is Failure:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment object to Idle.
  - Send an ABORTED TIP Command.

- If the provided outcome is Success:
  - Signal the Begin Phase One event on the Core Transaction Manager Facet with the following arguments:
    - The transaction object referenced by the provided Enlistment object.
    - A Single-Phase value set to True.

3.4.7.2.4 Phase One Complete

The Phase One Complete event MUST be signaled with the following arguments:

- An Enlistment object.
- A value indicating the outcome of Phase One. This value MUST be set to one of the following values:
  - Read-Only
  - Committed
  - Aborted
  - In Doubt

If the Phase One Complete event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- If the provided outcome is In Doubt:
  - Send an ABORTED TIP Command. This action does not conform to the [RFC2371] specification.
  - Call the connection management's **FreeTipConnection** operation with the following argument:
    - The provided TIP Connection object.
- Otherwise:
  - If the provided outcome is Committed:
    - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to Idle.
    - Send a COMMITTED TIP Command.
  - If the provided outcome is Aborted:
    - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to Idle.
    - Send an ABORTED TIP Command.
  - If the provided outcome is Read Only:
    - Set the **State** field of the TIP Connection object referenced by the provided Enlistment to Idle.

- Send a COMMITTTED TIP Command.

3.4.7.2.5 Rollback Complete

The Rollback Complete event MUST be signaled with the following arguments:

- An Enlistment object.

If the Rollback Complete event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- If the **State** field of the TIP Connection object referenced by the provided Enlistment object is set to Begun:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment object to Idle.
  - Send an ABORTED TIP Command.

3.4.7.2.6 Unilaterally Aborted

The Unilaterally Aborted event MUST be signaled with the following arguments:

- An Enlistment object.

If the Unilaterally Aborted event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- If the **State** field of the TIP Connection object referenced by the provided Enlistment object is set to Begun:
  - Set the **State** field of the TIP Connection object referenced by the provided Enlistment object to Aborted.

3.4.7.3 Transport Events

3.4.7.3.1 Transport Connection Down

This event overrides the event with the same name as specified in section 3.1.7.2.2.

The Transport Connection Down event MUST be signaled with the following argument:

- A TIP Connection object.

If the Transport Connection Down event is signaled, the TIP Transaction Manager Communicating with an Application Facet MUST perform the following actions:

- If the **State** field of the provided TIP Connection object is set to Begun:
  - Signal the Begin Rollback event on the Core Transaction Manager Facet with the following argument:
    - The Enlistment object referenced by the provided TIP Connection object.
- Set the **State** field of the provided TIP Connection object to Error.

- Reset the TIP Connection object referenced by the Enlistment object referenced by the provided TIP Connection object.
- Call the connection management's **TerminateTipConnection** operation with the following argument:
  - The provided TIP Connection object.

4 Protocol Examples

The following sections describe several operations as used in common scenarios to illustrate the function of the Transaction Internet Protocol (TIP) Extensions. These protocol examples generally assume that a TCP transport session has already been established between the two TIP roles. However, some examples exhibit how one TIP role establishes a new TCP transport session with another TIP role, because of the protocol that is being demonstrated.

TIP roles communicate with each other by using TIP Connections that are in turn layered on top of the TCP transport infrastructure.

4.1 Transaction Processing Scenario

4.1.1 Creating the TIP Connection

A TIP Connection can be created by any TIP role that can be a primary:

- TIP Application.
- **TIP Superior Transaction Manager**.
- **TIP Subordinate Transaction Manager**.

This TIP Command sequence is initiated by starting a TIP Connection on a transport session between a Primary and a Secondary. The TIP Connection starts in the Initial State.

The TIP Command sequence starts when the Primary creates the transport session to the Secondary, then starts the TIP Connection by sending an IDENTIFY TIP Command, specifying *lowest protocol version*, *highest protocol version*, *primary transaction manager address*, and *secondary transaction manager address*. For this example, the Primary specifies that the *lowest protocol version* is 3 and the *highest protocol version* is 3, that the *primary transaction manager address* is primary-tm.fabrikam.com:8086/TipTM/ and that the *secondary transaction manager address* is secondary-tm.fabrikam.com:3372/.

Field	**Value**	Value Description
TIP Command	0x49 0x44 0x45 0x4E 0x54 0x49 0x46 0x59	"IDENTIFY"
SP	0x20	" "
lowest protocol version	0x33	"3"
SP	0x20	" "
highest protocol version	0x33	"3"
SP	0x20	" "
primary transaction manager address	0x70 0x72 0x69 0x6D 0x61 0x72 0x79 0x2D 0x74 0x6D 0x2E 0x66 0x61 0x62 0x72 0x69 0x6B 0x61 0x6D 0x2E 0x63 0x6F 0x6D 0x3A 0x38 0x30 0x38 0x36 0x2F 0x54 0x69 0x70 0x54 0x4D 0x2F	"primary-tm.fabrikam.com:8086/TipTM/"
SP	0x20	" "

Field	Value	Value Description
secondary transaction manager address	0x73 0x65 0x63 0x6F 0x6E 0x64 0x61 0x72 0x79 0x2D 0x74 0x6D 0x2E 0x66 0x61 0x62 0x72 0x69 0x6B 0x61 0x6D 0x2E 0x63 0x6F 0x6D 0x3A 0x33 0x33 0x37 0x32 0x2F	"secondary-tm.fabrikam.com:3372/"
TIP Command Line terminator	0x0A	LF

When the Secondary receives the IDENTIFY TIP Command from the Primary it finds the *high protocol version* supported by both itself and the Primary. If such a protocol version exists, it sends an IDENTIFIED TIP Command to the Primary Transaction Manager specifying the *protocol version* (for this example, 3), and the State of the TIP Connection changes to Idle.

Field	Value	Value Description
TIP Command	0x49 0x44 0x45 0x4E 0x54 0x49 0x46 0x49 0x45 0x44	"IDENTIFIED"
SP	0x20	" "
protocol version	0x33	"3"
TIP Command Line terminator	0x0A	LF

4.1.2 Propagating the Transaction

This exchange involves the TIP Superior Transaction Manager and the TIP Subordinate Transaction Manager.

4.1.2.1 Pull Propagation

The TIP Subordinate Transaction Manager, which is Primary, pulls the transaction by sending a PULL TIP Command, specifying the TIP Superior Transaction Manager's Transaction identifier as the *superior's transaction identifier* and its own transaction identifier as the *subordinate's transaction identifier*. For this example, the TIP Superior Transaction Manager's Transaction identifier is OleTx-188b0af9-1c81-43cf-8c2a-0e865540f450 and the *subordinate transaction identifier* is a6441ea1-b68c-48b0-adf9-015a08fd3f2f.

Field	Value	Value Description
TIP Command	0x50 0x55 0x4C 0x4C	"PULL"
SP	0x20	" "
<superior's transaction identifier>	0x4F 0x6C 0x65 0x54 0x78 0x2D 0x31 0x38 0x38 0x62 0x30 0x61 0x66 0x39 0x2D 0x31 0x63 0x38 0x31 0x2D 0x34 0x33 0x63 0x66 0x2D 0x38 0x63 0x32 0x61 0x2D 0x30 0x65 0x38 0x36 0x35 0x35 0x34 0x30 0x66 0x34 0x35 0x30	"OleTx-188b0af9-1c81-43cf-8c2a-0e865540f450"
SP	0x20	" "

Field	Value	Value Description
subordinate's transaction identifier	0x61 0x36 0x34 0x34 0x31 0x65 0x61 0x31 0x2D 0x62 0x36 0x38 0x63 0x2D 0x34 0x38 0x62 0x30 0x2D 0x61 0x64 0x66 0x39 0x2D 0x30 0x31 0x35 0x61 0x30 0x38 0x66 0x64 0x33 0x66 0x32 0x66	"a6441ea1-b68c-48b0-adf9-015a08fd3f2f"
TIP Command Line terminator	0x0A	LF

When the TIP Superior Transaction Manager receives the PULL TIP Command from the TIP Subordinate Transaction Manager, it attempts to locate the transaction in its list of transaction objects by using the *superior's transaction identifier*. If the TIP **Superior Transaction Manager** can successfully locate the transaction object, it will respond to the TIP Subordinate Transaction Manager with the PULLED TIP Command and the Current Transaction of the TIP Connection will be set to be the Transaction. This swtches Connection Type, so the TIP Superior Transaction Manager is Primary and the TIP Subordinate Transaction Manager is Secondary. The State of the TIP Connection changes to Enlisted.

Field	Value	Value Description
TIP Command	0x50 0x55 0x4C 0x4C 0x45 0x44	"PULLED"
TIP Command Line terminator	0x0A	LF

4.1.2.2 Push Propagation

The TIP Superior Transaction Manager pushes the Transaction by sending a PUSH TIP Command, specifying its own Transaction identifier as the *superior's transaction identifier*. For this example, the *superior's transaction identifier* is 1c7edc47-a302-4cae-8829-c0bf87d79ad7.

Field	Value	Value Description
TIP Command	0x50 0x55 0x53 0x48	"PUSH"
SP	0x20	" "
superior's transaction identifier	0x31 0x63 0x37 0x65 0x64 0x63 0x34 0x37 0x2D 0x61 0x33 0x30 0x32 0x2D 0x34 0x63 0x61 0x65 0x2D 0x38 0x38 0x32 0x39 0x2D 0x63 0x30 0x62 0x66 0x38 0x37 0x64 0x37 0x39 0x61 0x64 0x37	"1c7edc47-a302-4cae-8829-c0bf87d79ad7"
TIP Command Line terminator	0x0A	LF

When the TIP Subordinate Transaction Manager receives the PUSH TIP Command, it adds the Transaction to its list of transaction objects with a newly created Transaction identifier and the Transaction will become the Current Transaction of the TIP Connection. The TIP Subordinate Transaction Manager responds with the PUSHED TIP Command, specifying the new transaction identifier as the *subordinate's transaction identifier*. The State of the TIP Connection changes to Enlisted.

Field	Value	Value Description
TIP Command	0x50 0x55 0x53 0x48 0x45 0x44	"PUSHED"

Field	Value	Value Description
SP	0x20	" "
subordinate's transaction identifier ID	0x4F 0x6C 0x65 0x54 0x78 0x2D 0x34 0x39 0x32 0x63 0x33 0x36 0x34 0x32 0x2D 0x39 0x63 0x34 0x63 0x2D 0x34 0x66 0x38 0x63 0x2D 0x61 0x62 0x65 0x65 0x2D 0x37 0x66 0x65 0x31 0x30 0x38 0x33 0x63 0x62 0x65 0x32 0x61	"OleTx-492c3642-9c4c-4f8c-abee-7fe1083cbe2a"
TIP Command Line terminator	0x0A	LF

4.1.3 Committing the Transaction

4.1.3.1 Two Phase Commit

This exchange involves the TIP Superior Transaction Manager and the TIP Subordinate Transaction Manager.

4.1.3.1.1 Read Only

In the first phase of Two-Phase Commit processing, the TIP Superior Transaction Manager sends the PREPARE TIP Command to the TIP Subordinate Transaction Manager.

Field	Value	Value description
TIP Command	0x50 0x52 0x45 0x50 0x41 0x52 0x45	"PREPARE"
TIP Command Line terminator	0x0A	LF

When the TIP Subordinate Transaction Manager receives the PREPARE TIP Command, it iterates through each of the Transaction's subordinate branches and notifies the subordinates that the transaction processing has begun. The TIP Subordinate Transaction Manager waits for reply notifications from each of its subordinates in order to determine the outcome of the transaction.

If each subordinate branch of a Transaction successfully prepares for the Transaction and it is not necessary for the TIP Subordinate Transaction Manager to receive notification of the transaction's outcome, the TIP Subordinate Transaction Manager is allowed to respond to the TIP Superior Transaction Manager with the READONLY TIP Command. The State of the TIP Connection changes to Idle and the Current Transaction of the TIP Connection is cleared.

Field	Value	Value description
TIP Command	0x52 0x45 0x41 0x44 0x4F 0x4E 0x4C 0x59	"READONLY"
TIP Command Line terminator	0x0A	LF

4.1.3.1.2 Phase One

In the first phase of Two-Phase Commit, the TIP Superior Transaction Manager sends the PREPARE TIP Command to the TIP Subordinate Transaction Manager.

Field	Value	Value description
TIP Command	0x50 0x52 0x45 0x50 0x41 0x52 0x45	"PREPARE"
TIP Command Line Terminator	0x0A	LF

When the TIP Subordinate Transaction Manager receives the PREPARE TIP Command, it iterates through each of the Transaction's subordinate branches and notifies the subordinates that the Transaction processing has begun. The TIP Subordinate Transaction Manager waits for reply notifications from each of it's subordinates in order to determine the outcome of the Transaction.

If each subordinate branch of a Transaction successfully prepares for the Transaction, the TIP Subordinate Transaction Manager responds to the TIP Superior Transaction Manager with the PREPARED TIP Command. The State of the TIP Connection changes to Prepared.

Field	Value	Value description
TIP Command	0x50 0x52 0x45 0x50 0x41 0x52 0x45 0x44	"PREPARED"
TIP Command Line terminator	0x0A	LF

4.1.3.1.3 Recovery

While the State of the TIP Connection is Prepared, it is possible that the underlying transport session will fail.

If this occurs, the TIP Subordinate Transaction Manager will establish a TIP Connection to the TIP Superior Transaction Manager as above (for more information, see section 4.1.1). It will then send a QUERY TIP Command, specifying the *superior's transaction identifier*. For this example, the *superior's transaction identifier* is 1c7edc47-a302-4cae-8829-c0bf87d79ad7.

Field	Value	Value description
TIP Command	0x51 0x55 0x45 0x52 0x59	"QUERY"
SP	0x20	" "
superior's transaction identifier	0x31 0x63 0x37 0x65 0x64 0x63 0x34 0x37 0x2D 0x61 0x33 0x30 0x32 0x2D 0x34 0x63 0x61 0x65 0x2D 0x38 0x38 0x32 0x39 0x2D 0x63 0x30 0x62 0x66 0x38 0x37 0x64 0x37 0x39 0x61 0x64 0x37	"1c7edc47-a302-4cae-8829-c0bf87d79ad7"
TIP Command Line terminator	0x0A	LF

When the TIP Superior Transaction Manager receives the QUERY TIP Command from the TIP Subordinate Transaction Manager, it attempts to locate the transaction in its list of transaction objects by using the *superior's transaction identifier*. If the TIP Superior Transaction Manager can successfully locate the transaction object it will respond to the TIP Subordinate Transaction Manager with the QUERIEDEXISTS TIP Command.

Field	Value	Value description
TIP Command	0x51 0x55 0x45 0x52 0x49 0x45 0x44 0x45 0x58	"QUERIEDEXISTS"

Field	Value	Value description
	0x49 0x53 0x54 0x53	
TIP Command Line terminator	0x0A	LF

The TIP Superior Transaction Manager will then establish a TIP Connection to the TIP Subordinate Transaction Manager as above (for more information, see section 4.1.1). It will send a RECONNECT TIP Command, specifying the *subordinate's transaction identifier*. For this example, the *subordinate's transaction identifier* is OleTx-492c3642-9c4c-4f8c-abee-7fe1083cbe2a.

Field	Value	Value description
TIP Command	0x52 0x45 0x43 0x4F 0x4E 0x4E 0x45 0x43 0x54	"RECONNECT"
SP	0x20	" "
subordinate's transaction identifier	0x4F 0x6C 0x65 0x54 0x78 0x2D 0x34 0x39 0x32 0x63 0x33 0x36 0x34 0x32 0x2D 0x39 0x63 0x34 0x63 0x2D 0x34 0x66 0x38 0x63 0x2D 0x61 0x62 0x65 0x65 0x2D 0x37 0x66 0x65 0x31 0x30 0x38 0x33 0x63 0x62 0x65 0x32 0x61	"OleTx-492c3642-9c4c-4f8c-abee-7fe1083cbe2a"
TIP Command Line terminator	0x0A	LF

When the TIP Subordinate Transaction Manager receives the RECONNECT TIP Command from the TIP Superior Transaction Manager , it attempts to locate the transaction in its list of transaction objects by using the *subordinate's transaction identifier*. If the TIP Subordinate Transaction Manager can successfully locate the transaction object it will respond to the TIP Superior Transaction Manager with the RECONNECTED TIP Command and the transaction will become the Current Transaction of the TIP Connection. The state of the TIP Connection changes to Prepared.

Field	Value	Value Description
TIP Command	0x52 0x45 0x43 0x4F 0x4E 0x4E 0x45 0x43 0x54 0x45 0x44	"RECONNECTED"
TIP Command Line terminator	0x0A	LF

4.1.3.1.4 Phase Two

In the second phase of Two Phase Commit processing the TIP Superior Transaction Manager sends the COMMIT TIP Command to the TIP Subordinate Transaction Manager.

Field	Value	Value description
TIP Command	0x43 0x4F 0x4D 0x4D 0x49 0x54	"COMMIT"
TIP Command Line Terminator	0x0A	LF

When the TIP Subordinate Transaction Manager receives the COMMIT TIP Command, it then iterates through each subordinate branch of the transaction and notifies the subordinates that the transaction is committed. It then waits for reply notifications from each of its subordinates in order to complete Phase Two processing.

When each subordinate branch of the Transaction replies that they have committed the Transaction, the TIP Subordinate Transaction Manager responds to the TIP Superior Transaction Manager with the COMMITTED TIP Command. The State of the TIP Connection changes to Idle and the Current Transaction of the TIP Connection is cleared.

Field	Value	Value description
TIP Command	0x43 0x4F 0x4D 0x4D 0x49 0x54 0x45 0x44	"COMMITTED"
TIP Command Line terminator	0x0A	LF

4.1.3.2 Single Phase Commit

The Single Phase Commit TIP Command sequence is the same as above (as specified in section 4.1.3.1.3) except that when the TIP Subordinate Transaction Manager receives the COMMIT TIP Command, it then performs the activities of both Phase One and Phase Two before responding with the COMMITTED TIP Command.

4.2 Begin Scenario

4.2.1 Creating the TIP Connection

The Primary Transaction Manager will establish a TIP Connection to the Superior Transaction Manager as above (for more information, see section 4.1.1)

4.2.2 Beginning the Transaction

The Primary Transaction Manager begins the transaction by sending a BEGIN TIP Command.

Field	Value	Value description
TIP Command	0x42 0x45 0x47 0x49 0x4E	"BEGIN"
TIP Command Line terminator	0x0A	LF

When the TIP Secondary Transaction Manager receives the BEGIN TIP Command from the TIP Primary Transaction Manager, it creates a new Transaction and adds it to its list of Transaction objects. This Transaction will become the Current Transaction of the TIP Connection. The TIP Secondary Transaction Manager will respond with the BEGUN TIP Command, specifying the new Transaction identifier as the *transaction identifier*. The State of the TIP Connection changes to Begun.

Field	Value	Value description
TIP Command	0x42 0x45 0x47 0x55 0x4E	"BEGUN"
SP	0x20	" "
transaction	0x4F 0x6C 0x65 0x54 0x78 0x2D 0x62 0x62 0x65 0x61 0x34 0x36 0x65 0x39 0x2D 0x36 0x62 0x35 0x63 0x2D 0x34	"OleTx-bbea46e9-6b5c-4cb8-bf69-

Field	Value	Value description
identifier	0x63 0x62 0x38 0x2D 0x62 0x66 0x36 0x39 0x2D 0x37 0x61 0x62 0x38 0x33 0x66 0x32 0x66 0x32 0x62 0x35 0x63	7ab83f2f2b5c"
TIP Command Line terminator	0x0A	LF

4.2.3 Committing the Transaction

The TIP Primary Transaction Manager must commit the transaction by using Single Phase Commit as specified in section 4.1.3.2.

5 Security

The following sections specify security considerations for implementers of the Transaction Internet Protocol (TIP) Extensions.

The transaction processing protocol that is defined by this specification is intended for use in an environment where all participants are trusted to collaborate in driving transactions toward a final outcome.

Misuse of the Two-Phase Commit Protocol can enable participants to perform simple denial of service attacks on their transaction managers. Because transaction managers generally communicate with multiple participants simultaneously, this condition represents a denial of service to other participants.

Each participant SHOULD uphold the following principles:

- Every transaction reaches a common outcome for all participants, in accord with a correctly executed Two-Phase Commit Protocol.
- No transaction remains In Doubt for a longer period of time than the application's higher-layer business logic accepts.

An implementation has the option to further restrict its exposure to security vulnerabilities by initializing the following flags specified in the Abstract Data Model in section 3.1 to false:

- Allow TIP
- Allow Begin
- Allow Pass Through
- Allow Non Default Port
- Allow Different Partner Address

6 Appendix A: Windows Behavior

The information in this specification is applicable to the following versions of Windows:

- Windows Server 2008
- Windows Vista
- Windows Server 2003
- Windows XP
- Windows 2000

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Windows behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that Windows does not follow the prescription.

<1> Section 3.1.1.3.2: On Windows 2000, the *secondary transaction manager address* argument of the IDENTIFY TIP Command may not be set to the value of the provided Partner Transaction Manager Address.

<2> Section 3.1.3: The default value of these flags is false on Windows Server 2008, Windows Vista, Windows Server 2003, Windows XP, and Windows 2000.

<3> Section 3.1.3: On Windows 2000, Windows XP and Windows Server 2003, the **Transaction Manager Address Override** field is not set.

<4> Section 3.1.3: TCP requests are accepted on port 3372 on Windows Server 2008, Windows Vista, Windows Server 2003, Windows XP and Windows 2000.

<5> Section 3.1.5.5: The Lowest Supported Version and the Highest Supported Version on all versions of Windows is 3.

<6> Section 3.1.5.5: The Lowest Supported Version and the Highest Supported Version on all versions of Windows is 3.

<7> Section 3.1.5.6: On Windows 2000, the TIP Transaction Manager Facet does not test that the **State** field is set to Idle.

<8> Section 3.1.5.6: On Windows 2000, the TIP Transaction Manager Facet checks that the value of the provided <protocol identifier> is "TMP2.0". On Windows XP, Windows Server 2003, Windows Vista, and Windows Server 2008, the TIP Transaction Manager Facet does not check the value of the provided <protocol identifier>.

<9> Section 3.1.5.10: On Windows 2000, the TIP Transaction Manager Facet signals the Invalid TIP Command event (as specified in section 3.1.7.1) and terminates the processing of this event.

<10> Section 3.1.7.1: On Windows 2000, the TIP Transaction Manager Facet may not send the ERROR TIP Command or may send an invalid message.

<11> Section 3.2.5.4: On Windows 2000, the TIP Superior Transaction Manager facet signals the Invalid TIP Command event (as specified in section 3.2.7.1).

<12> Section 3.3.2.1: Windows sets the value of the timer to 2000 seconds.

7 Appendix B: Summary of Extensions

The following table documents the conformance of this protocol to the [RFC2371] specification against the TIP Commands specified in [RFC2371] section 13. In cases where Transaction Internet Protocol (TIP) Extensions does not conform to the [RFC2371] specification, specific section references are provided.

TIP Commands from [RFC2371] section 13	Transaction Internet Protocol (TIP) Extensions
ABORT	Conforms.
ABORTED	Conforms.
ALREADYPUSHED	Conforms, except for the format restrictions specified in section 2.2.1.
BEGIN	Conforms, except a Transaction Internet Protocol (TIP) Extensions implementation never sends BEGIN TIP Command.
BEGUN	Conforms, except for the format restrictions specified in section 2.2.2.
CANTMULTIPLEX	Conforms.
CANTTLS	Conforms.
COMMIT	Conforms.
COMMITTED	Conforms.
ERROR	Conforms.
IDENTIFY	Conforms, except for the format restrictions specified in section 2.2.3.
IDENTIFIED	Conforms.
MULTIPLEX	Conforms, except a Transaction Internet Protocol (TIP) Extensions implementation never sends MULTIPLEX TIP Command.
MUTLIPLEXING	Conforms, except a Transaction Internet Protocol (TIP) Extensions implementation never sends MULTIPLEXING TIP Command.
NEEDTLS	Conforms, except a Transaction Internet Protocol (TIP) Extensions implementation never sends NEEDTLS TIP Command.
NOTBEGUN	Conforms.
NOTPULLED	Conforms.
NOTPUSHED	Conforms.
NOTRECONNECTED	Conforms.
PREPARE	Conforms.
PREPARED	Conforms.
PULL	Conforms, except for the format restrictions specified in section 2.2.4.
PULLED	Conforms.

TIP Commands from [RFC2371] section 13	Transaction Internet Protocol (TIP) Extensions
PUSH	Conforms, except for the format restrictions specified in section 2.2.5.
PUSHED	Conforms, except for the format restrictions specified in section 2.2.6.
QUERY	Conforms, except for the format restrictions specified in section 2.2.7.
QUEREDEXISTS	Conforms.
QUERIEDNOTFOUND	Conforms.
READONLY	Conforms.
RECONNECT	Conforms, except for the format restrictions specified in section 2.2.8
RECONNECTED	Conforms.
TLS	Conforms, except a Transaction Internet Protocol (TIP) Extensions implementation never sends TLS TIP Command.
TLSING	Conforms, except a Transaction Internet Protocol (TIP) Extensions implementation never sends TLSING TIP Command.

The following is the complete list of sections where this protocol extends the [RFC2371] specification:

- Section 2.2.1.
- Section 2.2.2.
- Section 2.2.3.
- Section 2.2.4.
- Section 2.2.5.
- Section 2.2.6.
- Section 2.2.7.
- Section 2.2.8.
- Section 3.1.1.3.2.
- Section 3.1.5.6.
- Section 3.1.5.9.
- Section 3.1.5.11.
- Section 3.1.7.1.
- Section 3.2.5.4.
- Section 3.2.5.7.
- Section 3.2.7.3.4.

- Section 3.3.5.6.
- Section 3.4.7.2.4.

8 Index

A

ABORT TIP (section 3.3.5.1, section 3.4.5.1)
ABORTED TIP
Abstract data model
  overview
  TIP Subordinate Transaction Manager Facet details
  TIP Superior Transaction Manager Facet details
  TIP Transaction Manager Communicating with an Application Facet details
ALREADYPUSHED
ALREADYPUSHED TIP
Applicability

B

Begin commit
Begin phase one
Begin Rollback
Begin scenario
BEGIN TIP
BEGUN
BEGUN TIP

C

CANTMULTIPLEX TIP
CANTTLS TIP
Capability negotiation
Commit Complete
COMMIT TIP (section 3.3.5.2, section 3.4.5.3)
COMMITTED TIP
Common details
  overview (section 3.1, section 3.1.5)
Common scenarios
Communication
  Application Facet
  TIP Transaction Manager
Create Subordinate Enlistment
  Failure
  Success
Create Superior Enlistment
  Failure
  Success
Create Transaction
  Failure
  Success
Creating the TIP Connection

D

Data model - abstract
  overview
  TIP Subordinate Transaction Manager Facet details
  TIP Transaction Manager Communicating with an Application Facet details

E

ERROR TIP (section 3.2.5.12, section 3.3.5.10, section 3.4.5.4)
Examples
  begin scenario
  transaction processing scenario
Extensions - summary

F

Fields - vendor-extensible
FreeTipConnection

G

GetTipConnection
GetTipConnectionFromAddress
Glossary

H

HasPartnerTransaction
Higher-layer triggered events
  overview
  TIP Subordinate Transaction Manager Facet details
  TIP Superior Transaction Manager Facet details
  TIP Transaction Manager Communicating with an Application Facet details

I

IDENTIFIED TIP
IDENTIFY
IDENTIFY TIP
Informative references
Initialization
  overview
  TIP Subordinate Transaction Manager Facet details
  TIP Superior Transaction Manager Facet details
  TIP Transaction Manager Communicating with an Application Facet details
Introduction

L

Local events
  overview
  TIP Subordinate Transaction Manager Facet details
  TIP Superior Transaction Manager Facet details
  TIP Transaction Manager Communicating with an Application Facet details

M

Message processing
  TIP Subordinate Transaction Manager Facet details

TIP Superior Transaction Manager Facet details
TIP Transaction Manager Communicating with an Application Facet details
Messages
overview
syntax
transport
MULTIPLEX TIP
MULTIPLEXING TIP

N

NEEDTLS TIP
Normative references
NOTBEGUN TIP
NOTPULLED TIP
NOTPUSHED TIP
NOTRECONNECTED TIP

O

Overview
abstract data model
higher-layer triggered events
initialization
local events
message processing
timer events
timers

P

Phase One
Phase One Complete
Phase Two
Phase Zero Complete
Preconditions
PREPARE TIP
PREPARED TIP
Prerequisites
Primary state transition table
Process error (section 3.2.7.2, section 3.3.7.2)
Propagation
pull
push
Protocol roles
PULL
PULL TIP
Pull transaction
PULLED TIP
PUSH
PUSH TIP
Push transaction
PUSHED
PUSHED TIP

Q

QUERIEDEXISTS TIP
QUERIEDNOTFOUND TIP
QUERY
Query Timer
QUERY TIP

R

Read Only
READONLY TIP
Received message
RECONNECT
RECONNECT TIP
RECONNECTED TIP
Recovery
References
informative
normative
overview
Relationship to other protocols
Rollback Complete (section 3.3.7.3.7, section 3.4.7.2.5)

S

Scenarios
begin scenario
common
transaction processing scenario
Secondary state transition table
Security
Sequencing rules
TIP Subordinate Transaction Manager Facet details
TIP Superior Transaction Manager Facet details
TIP Transaction Manager Communicating with an Application Facet details
Single Phase Commit
Standards assignments
Summary of extensions
Syntax

T

TerminateTipConnection
Timer events
overview
TIP Subordinate Transaction Manager Facet details
TIP Superior Transaction Manager Facet details
TIP Transaction Manager Communicating with an Application Facet details
Timers
overview
TIP Subordinate Transaction Manager Facet details
TIP Superior Transaction Manager Facet details
TIP Transaction Manager Communicating with an Application Facet details
TIP Application role
TIP Command - invalid event (section 3.1.7.1, section 3.2.7.1)
TIP Command Event - invalid
TIP Command object
TIP Connection - creating
TIP Connection management operations
TIP Connection object

TIP Subordinate Transaction Manager Facet
TIP Subordinate Transaction Manager Facet details
  abstract data model
  higher-layer triggered events
  initialization
  local events
  message processing
  overview (section 3.1, section 3.3)
  sequencing rules
  timer events
  timers
TIP Superior Transaction Manager facet
TIP Superior Transaction Manager Facet details
  abstract data model
  higher-layer triggered events
  initialization
  local events
  message processing
  overview (section 3.1, section 3.2)
  sequencing rules
  timer events
  timers
TIP Transaction Manager Communicating with an Application Facet details
  abstract data model
  higher-layer triggered events
  initialization
  local events
  message processing
  overview (section 3.1, section 3.4)
  sequencing rules
  timer events
  timers
TIP Transaction Manager Facet data elements
TIP Two-Phase Commit
TLS TIP
TLSING TIP
Transaction
  beginning
  committing
  completing
  pulling
  pushing
  starting
Transaction identifier converter operations
Transaction Manager role
Transaction processing scenario
Transactions
  committing
  propagating
Transport
Transport connection down (section 3.1.7.2.2, section 3.2.7.4.1, section 3.3.7.4.1, section 3.4.7.3.1)
Transport events
Triggered events - higher-layer
  TIP Subordinate Transaction Manager Facet details
  TIP Superior Transaction Manager Facet details
  TIP Transaction Manager Communicating with an Application Facet details
Two Phase Commit

U

Unilaterally Aborted (section 3.3.7.3.8, section 3.4.7.2.6)

V

Vendor-extensible fields
Versioning

W

Windows behavior