

# [MS-NMFTB]: .NET Message Framing TCP Binding Protocol Specification

---

## Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: <http://www.microsoft.com/interop/osp>) or the Community Promise (available here: <http://www.microsoft.com/interop/cp/default.aspx>). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting [iplg@microsoft.com](mailto:iplg@microsoft.com).
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

**Reservation of Rights.** All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

## Revision Summary

Date	Revision History	Revision Class	Comments
03/12/2010	0.1	Major	First release.
04/23/2010	0.1.1	Editorial	Revised and edited the technical content.
06/04/2010	0.1.2	Editorial	Revised and edited the technical content.
07/16/2010	1.0	Major	Significantly changed the technical content.
08/27/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.

# Contents

<b>1 Introduction</b>	<b>5</b>
1.1 Glossary	5
1.2 References	6
1.2.1 Normative References	6
1.2.2 Informative References	7
1.3 Overview	7
1.4 Relationship to Other Protocols	7
1.5 Prerequisites/Preconditions	7
1.6 Applicability Statement	7
1.7 Versioning and Capability Negotiation	7
1.8 Vendor-Extensible Fields	8
1.9 Standards Assignments	8
<b>2 Messages</b>	<b>9</b>
2.1 Transport	9
2.2 Common Message Syntax	9
2.2.1 Namespaces	9
2.2.2 Messages	9
2.2.3 Elements	9
2.2.4 Complex Types	9
2.2.5 Simple Types	9
2.2.6 Attributes	9
2.2.7 Groups	10
2.2.8 Attribute Groups	10
<b>3 Protocol Details</b>	<b>11</b>
3.1 Common Details	11
3.1.1 Abstract Data Model	11
3.1.2 Timers	12
3.1.3 Initialization	12
3.1.4 Message Processing Events and Sequencing Rules	12
3.1.5 Timer Events	12
3.1.6 Other Local Events	12
3.1.6.1 TCP connection Aborted	12
3.1.6.2 Higher-Layer Triggered Events	12
3.1.6.2.1 Abort the TCP connection	12
3.2 Initiator Details	12
3.2.1 Abstract Data Model	12
3.2.1.1 CONNECTED State	13
3.2.1.2 BUSY State	13
3.2.2 Timers	13
3.2.3 Initialization	13
3.2.4 Message Processing Events and Sequencing Rules	13
3.2.5 Timer Events	13
3.2.6 Other Local Events	13
3.2.6.1 End Framing session	13
3.2.6.2 Higher-Layer Triggered Events	14
3.2.6.2.1 Connect	14
3.3 Receiver Details	14
3.3.1 Abstract Data Model	14

3.3.2	Timers .....	14
3.3.3	Initialization .....	14
3.3.4	Message Processing Events and Sequencing Rules .....	14
3.3.5	Timer Events .....	14
3.3.6	Other Local Events .....	15
3.3.6.1	End Framing session .....	15
<b>4</b>	<b>Protocol Examples .....</b>	<b>16</b>
<b>5</b>	<b>Security .....</b>	<b>17</b>
5.1	Security Considerations for Implementers .....	17
5.2	Index of Security Parameters .....	17
<b>6</b>	<b>Appendix A: Full WSDL .....</b>	<b>18</b>
<b>7</b>	<b>Appendix B: Product Behavior .....</b>	<b>19</b>
<b>8</b>	<b>Change Tracking.....</b>	<b>20</b>
<b>9</b>	<b>Index .....</b>	<b>21</b>

# 1 Introduction

The .NET Message Framing TCP Binding Protocol specifies how the .NET Message Framing Protocol [\[MC-NMF\]](#) is used for framing **SOAP messages** over **TCP** [\[RFC793\]](#).

**Note** This specification does not define any SOAP messages. Rather, it specifies how SOAP messages defined by a higher-layer protocol are framed for transport over TCP.

## 1.1 Glossary

The following terms are defined in [\[MS-GLOS\]](#):

**endpoint**  
**.NET Framework**  
**SOAP**  
**SOAP message**  
**Transmission Control Protocol (TCP)**  
**URI**  
**Web Services Description Language (WSDL)**  
**XML**  
**XML namespace**  
**XML schema (XSD)**

The following terms are defined in [\[MC-NMF\]](#):

**initiator**  
**receiver**

The following terms are specific to this document:

**authority:** A hierarchical element in a **URI scheme** used for delegating governance of the name space defined by the remainder of the **URI**, as defined in [\[RFC3986\]](#) section 3.2.

**connection:** A logical communication path identified by a pair of sockets, as defined in [\[RFC793\]](#).

**end record:** A record containing no data that indicates that communication over a **connection**, as defined in [\[MC-NMF\]](#) section 2.2.3.9.

**fragment:** A component of a **URI** that allows for indirect identification of a secondary resource by reference to a primary resource, as defined in [\[RFC3986\]](#) section 3.5.

**framing session:** The communication session established for framing of the .NET message, as defined in [\[MC-NMF\]](#) section 1.3.

**hierarchical URI:** A **URI** expressed using hierarchical syntax, as defined in [\[RFC3986\]](#).

**host:** A subcomponent of the naming **authority** in a **URI scheme**, as defined in [\[RFC3986\]](#) section 3.2.2.

**port:** A subcomponent of the naming **authority** in a **URI scheme**, as defined in [\[RFC3986\]](#) section 3.2.3.

**query:** Contains non-hierarchical data used to identify a resource within the scope of a **URI scheme** and naming **authority**, as defined in [\[RFC3986\]](#) section 3.4.

**scheme:** The name of a specification to refer to when assigning identifiers within a particular **URI scheme**, as defined in [\[RFC3986\]](#) section 3.1.

**TCP connection:** A **connection** established as specified in [\[RFC793\]](#) section 3.4.

**transport session:** A group of messages correlated into conversation by the transport.

**user information:** User information for the **authority** in a **URI scheme**, as defined in [\[RFC3986\]](#) section 3.2.1.

**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](mailto: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.

[MC-NBFS] Microsoft Corporation, "[.NET Binary Format: SOAP Data Structure](#)", July 2007.

[MC-NBFSE] Microsoft Corporation, "[.NET Binary Format: SOAP Extension](#)", July 2007.

[MC-NMF] Microsoft Corporation, "[.NET Message Framing Protocol Specification](#)", October 2007.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981, <http://www.ietf.org/rfc/rfc0793.txt>

[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>

[RFC3986] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, <http://www.ietf.org/rfc/rfc3986.txt>

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

[SOAP1.2-1/2003] Gudgin, M., Hadley, M., Mendelsohn, N., et al., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part1-20030624>

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

[WSDLSOAP] Angelov, D., Ballinger, K., Butek, R., et al., "WSDL 1.1 Binding Extension for SOAP 1.2", W3c Member Submission, April 2006, <http://www.w3.org/Submission/wsdl11soap12/>

[XMLNS-2ED] World Wide Web Consortium, "Namespaces in XML 1.0 (Second Edition)", August 2006, <http://www.w3.org/TR/2006/REC-xml-names-20060816/>

## 1.2.2 Informative References

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

## 1.3 Overview

The .NET Message Framing TCP Binding Protocol specifies how the mechanism for framing messages over any transport protocol, as defined in the .NET Message Framing Protocol [\[MC-NMF\]](#), can be applied over the TCP [\[RFC793\]](#) transport protocol.

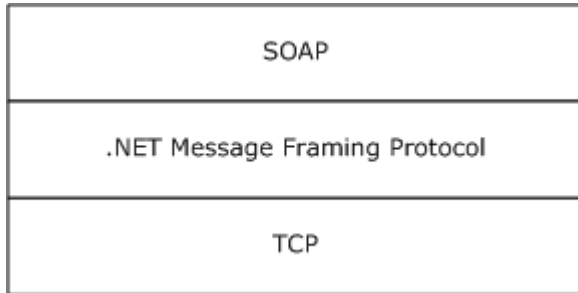
The .NET Message Framing TCP Binding Protocol also defines the net.tcp **URI scheme** and a URI for identifying this protocol as the transport for sending **SOAP** 1.1 messages [\[SOAP1.1\]](#) or SOAP 1.2 messages [\[SOAP1.2-1/2003\]](#).

## 1.4 Relationship to Other Protocols

The .NET Message Framing TCP Binding Protocol uses TCP as the transport [\[RFC793\]](#).

This protocol uses .NET Message Framing [\[MC-NMF\]](#) to send SOAP 1.1 messages [\[SOAP1.1\]](#) or SOAP 1.2 messages [\[SOAP1.2-1/2003\]](#).

The following figure shows the protocol stack:



**Figure 1: .NET Message Framing TCP Binding Protocol transport stack**

## 1.5 Prerequisites/Preconditions

The .NET Message Framing TCP Binding Protocol requires that an **initiator** can connect to a **receiver** over TCP [\[RFC793\]](#).

## 1.6 Applicability Statement

The .NET Message Framing TCP Binding Protocol is applicable in scenarios where an initiator and a receiver require a communication mechanism to send and receive SOAP messages over TCP [\[RFC793\]](#).

## 1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports:** This protocol requires TCP [\[RFC793\]](#) as specified in section [2.1](#).
- **Protocol Versions:** This protocol requires .NET Message Framing Protocol version 1.0 [\[MC-NMF\]](#). When this protocol is implemented using SOAP, the use of SOAP version 1.1 [\[SOAP1.1\]](#) or SOAP version 1.2 [\[SOAP1.2-1/2003\]](#) is required.

- **Capability Negotiation:** This protocol does not support negotiation of the version or capabilities to use.

## **1.8 Vendor-Extensible Fields**

This protocol has no vendor-extensible fields.

## **1.9 Standards Assignments**

There are no standards assignments for this protocol.



## 2 Messages

### 2.1 Transport

The .NET Message Framing TCP Binding Protocol requires TCP [\[RFC793\]](#).

An **endpoint** that uses the .NET Message Framing TCP Binding Protocol with [\[SOAP1.2-1/2003\]](#) MUST set the value of the transport attribute of the wsoap12:binding element [\[WSDLSOAP\]](#) to <http://schemas.microsoft.com/soap/tcp>.

An endpoint that uses the .NET Message Framing TCP Binding Protocol with [\[SOAP1.1\]](#) MUST set the value of the transport attribute of the soap:binding element [\[WSDL\]](#) to <http://schemas.microsoft.com/soap/tcp>.

### 2.2 Common Message Syntax

This section contains common definitions used by this protocol. The syntax of the definitions uses **XML Schema** as defined in [\[XMLSCHEMA1\]](#) and [\[XMLSCHEMA2\]](#), and **Web Services Description Language** as defined in [\[WSDL\]](#).

#### 2.2.1 Namespaces

This specification defines and references various **XML namespaces** using the mechanisms specified in [\[XMLNS-2ED\]](#). Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
soap	<a href="http://schemas.xmlsoap.org/wsdl/soap">http://schemas.xmlsoap.org/wsdl/soap</a>	<a href="#">[WSDL]</a>
soap12	<a href="http://schemas.xmlsoap.org/wsdl/soap12/">http://schemas.xmlsoap.org/wsdl/soap12/</a>	<a href="#">[WSDLSOAP]</a>
wsdl	<a href="http://schemas.xmlsoap.org/wsdl/">http://schemas.xmlsoap.org/wsdl/</a>	<a href="#">[WSDL]</a>

#### 2.2.2 Messages

This specification does not define any common XML Schema message definitions.

#### 2.2.3 Elements

This specification does not define any common XML Schema element definitions.

#### 2.2.4 Complex Types

This specification does not define any common XML Schema complex type definitions.

#### 2.2.5 Simple Types

This specification does not define any common XML Schema simple type definitions.

#### 2.2.6 Attributes

This specification does not define any common XML Schema attribute definitions.

### **2.2.7 Groups**

This specification does not define any common XML Schema group definitions.

### **2.2.8 Attribute Groups**

This specification does not define any common XML Schema attribute group definitions.

## 3 Protocol Details

A participant in this protocol can behave in one of two roles:

- Initiator
- Receiver

An initiator initiates the process by establishing a **TCP connection** to a receiver. The resulting TCP connection is used as the transport for performing the .NET Message Framing Protocol [\[MC-NMF\]](#).

### 3.1 Common Details

#### 3.1.1 Abstract Data Model

A net.tcp URI identifies a resource that listens for TCP connections and assumes the receiver role of this protocol.

A net.tcp URI is a URI that satisfies the following constraints:

- The scheme component of the URI MUST be net.tcp.
- The URI MUST be a **hierarchical URI**.
- The **authority** component of the URI MUST be specified.
- The authority component of the URI MUST NOT include **user information**.
- The URI SHOULD NOT include the **query** URI component. [<1>](#)
- The URI SHOULD NOT include the **fragment** URI component. [<2>](#)

If the authority component of a net.tcp URI does not specify a **port**, then the default port MUST be considered to be 808.

The protocol MUST maintain the following data elements for each TCP connection:

**TCP Protocol Configuration Object (TCPO):** A configuration object as defined in [\[MC-NMF\]](#) section 3.1.1 that satisfies the following constraints:

- The Via, as defined in [\[MC-NMF\]](#) section 2.2.3.3, specified by the **TCPO** MUST be a net.tcp URI as specified in this section.
- The Via, as defined in [\[MC-NMF\]](#) section 2.2.3.3, specified by the **TCPO** MUST be an absolute URI.
- The protocol version specified by the **TCPO** MUST be 1.0.
- The communication mode specified by the **TCPO** MUST NOT be Simplex, as defined in [\[MC-NMF\]](#) section 2.2.3.2. [<3>](#)
- The communication mode specified by the **TCPO** MUST NOT be Singleton-Sized, as defined in [\[MC-NMF\]](#) section 2.2.3.2. [<4>](#)
- If the communication mode is Duplex, as defined in [\[MC-NMF\]](#) section 2.2.3.2, then the encoding specified by the **TCPO** MUST NOT be Binary [\[MC-NBFS\]](#). The transport in this mode uses a **transport session**. [<5>](#)

- If the communication mode is Singleton-Unsigned, as defined in [\[MC-NMF\]](#) section 2.2.3.2, then the encoding specified by the **TPCO** MUST NOT be Binary with in-band dictionary [\[MC-NBFSE\]](#). The transport in this mode uses a transport session. [<6>](#)

### 3.1.2 Timers

None.

### 3.1.3 Initialization

A **TPCO** with an uninitialized transport is made available to the protocol as part of a higher-layer triggered event.

### 3.1.4 Message Processing Events and Sequencing Rules

This specification does not define any common XML Schema operation definitions.

### 3.1.5 Timer Events

None.

### 3.1.6 Other Local Events

#### 3.1.6.1 TCP connection Aborted

If the TCP connection is aborted at any time, then the protocol MUST:

- Close the **framing session** (if one exists)
- Discard any state associated with the TCP connection
- Terminate

#### 3.1.6.2 Higher-Layer Triggered Events

##### 3.1.6.2.1 Abort the TCP connection

The protocol MUST abort the TCP connection.

The protocol MUST discard any state associated with the TCP connection and framing session and then terminate.

## 3.2 Initiator Details

### 3.2.1 Abstract Data Model

The initiator role MUST maintain the following data elements for each TCP connection (in addition to the **TPCO**):

**Connection State:** One of two possible values.

- CONNECTED, see section [3.2.1.1](#)
- BUSY, see section [3.2.1.2](#)

### 3.2.1.1 CONNECTED State

CONNECTED is the initial state. The following events are processed in the CONNECTED state:

- Connect, as specified in section [3.2.6.2.1](#)
- Abort the TCP connection, as specified in section [3.1.6.2.1](#)

### 3.2.1.2 BUSY State

The following events are processed in the BUSY state:

- End the framing session, as specified in section [3.2.6.1](#)
- Abort the TCP connection, as specified in section [3.1.6.2.1](#)

## 3.2.2 Timers

None.

## 3.2.3 Initialization

When a new TCP connection is created, the **Connection State** for the TCP connection MUST be set to [CONNECTED](#).

## 3.2.4 Message Processing Events and Sequencing Rules

This specification does not define any XML Schema operation definitions for the initiator role.

## 3.2.5 Timer Events

None.

## 3.2.6 Other Local Events

### 3.2.6.1 End Framing session

The End Framing session event occurs due to one of the following conditions:

- The initiator has both received an end message and has sent an end message. For details about the end message, see [\[MC-NMF\]](#) section 2.2.3.9.
- The initiator sends a Fault Record. For details about the Fault Record, see [\[MC-NMF\]](#) section 2.2.5.
- The initiator receives a Fault Record.

When the framing session has ended for a given TCP connection, the initiator MUST set the **Connection State** to CONNECTED (see section [3.2.1.1](#)).

## 3.2.6.2 Higher-Layer Triggered Events

### 3.2.6.2.1 Connect

The initiator MUST establish a TCP connection with the **host** that is specified by the authority component of the Via URI from the **TPCO**. The manner in which the TCP connection is established (for example, by creating a new **connection** or by reusing an existing one) is implementation-specific. The initiator MUST NOT reuse a TCP connection in the BUSY **Connection State** (see section [3.2.1.2](#)). Once the TCP connection is established, the initiator MUST set the **Connection State** for that TCP connection to BUSY.

If the initiator fails to establish a TCP connection, then the initiator MUST notify the higher layer of the error and then discard all state for the TCP connection.

An implementation SHOULD NOT leave a connection in the CONNECTED state indefinitely (see section [3.2.1.1](#)).<7>

Once a TCP connection has been established, the initiator MUST set the **TPCO** transport to the TCP connection. The initiator MUST store the **TPCO** for the TCP connection. The initiator MUST then assume the initiator role, as defined in [\[MC-NMF\]](#) in section [3.2](#).

The initiator MUST use the **TPCO** stored for the TCP connection to initialize new framing sessions.

## 3.3 Receiver Details

### 3.3.1 Abstract Data Model

None.

### 3.3.2 Timers

None.

### 3.3.3 Initialization

The following initialization requirements are in addition to the initialization requirements specified in section [3.1.3](#).

The receiver MUST listen for a TCP connection at the host and port specified by the authority component of the Via URI from the **TPCO**.

Once a TCP connection has been established, the receiver MUST set the **TPCO** transport to the TCP connection. The receiver MUST then assume the receiver role as defined in [\[MC-NMF\]](#) section 3.3.

The receiver MUST use the **TPCO** to initialize new sessions.

If the receiver fails to start listening for TCP connections, then the receiver MUST notify the higher layer of the error and terminate.

### 3.3.4 Message Processing Events and Sequencing Rules

This specification does not define any XML Schema operation definitions for the receiver role.

### 3.3.5 Timer Events

None.

## 3.3.6 Other Local Events

### 3.3.6.1 End Framing session

The End Framing session event occurs due to one of the following conditions:

- The receiver has both received an end message and has sent an end message. For more information about the end message, see [\[MC-NMF\]](#) section 2.2.3.9.
- The receiver sends a Fault Record. For more information about the Fault Record, see [\[MC-NMF\]](#) section 2.2.5.
- The receiver receives a Fault Record.

When the framing session has ended for a given TCP connection, the receiver **MUST** reassume the receiver role using the **TPCO** stored with the connection, as defined in [\[MC-NMF\]](#) section 3.3.

## 4 Protocol Examples

The protocol is initialized with the following **TPCO**:

- Transport: <no value>
- Protocol version: 1.0
- Mode: 0x02 (Duplex)
- Via: net.tcp://SampleServer/SampleApp/
- Encoding: Binary

The receiver listens for connections at the address 192.168.2.13 on port 802.

The initiator actively establishes a connection to the host at 192.168.2.13 on port 802 (a TCP connection for the specified Via does not already exist).

Both the initiator and the receiver store the **TPCO** with the established TCP connection as the transport and assume the initiator and receiver roles for the .NET Message Framing Protocol [\[MC-NMF\]](#) respectively.

The protocol exchange proceeds as described in [\[MC-NMF\]](#) section 4.1.

When the initiator receives the final **end record**, the higher layer protocol aborts the connection.



## **5 Security**

### **5.1 Security Considerations for Implementers**

None.

### **5.2 Index of Security Parameters**

None.

## 6 Appendix A: Full WSDL

The following WSDL specifies the WSDL 1.1 binding extension transport URI with the version of SOAP as indicated.

### WSDL 1.1 binding extension transport URI with SOAP 1.2 [\[SOAP1.2-1/2003\]](#)

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/">
  <!-- omitted elements -->
  <wsdl:binding name="MyBinding" type="MyPortType">
    <soap12:binding transport="http://schemas.microsoft.com/soap/tcp"/>
    <wsdl:operation name="MyOperation">
      <!-- omitted elements -->
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="MyService">
    <wsdl:port name="MyPort" binding="MyBinding">
      <soap12:address location="net.tcp://myhost/" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

### WSDL 1.1 binding extension transport URI with SOAP 1.1 [\[SOAP1.1\]](#)

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  <!-- omitted elements -->
  <wsdl:binding name="MyBinding" type="MyPortType">
    <soap:binding transport="http://schemas.microsoft.com/soap/tcp"/>
    <wsdl:operation name="MyOperation">
      <!-- omitted elements -->
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="MyService">
    <wsdl:port name="MyPort" binding="MyBinding">
      <soap:address location="net.tcp://myhost/" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

## 7 Appendix B: Product Behavior

This document specifies version-specific details in the Microsoft® .NET Framework. For information about which versions of .NET Framework are available in each released Microsoft Windows® product or as supplemental software, see [.NET Framework](#).

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® .NET Framework 3.0
- Microsoft® .NET Framework 3.5
- Microsoft® .NET Framework 4.0

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

[<1> Section 3.1.1:](#) The Windows implementation of this protocol ignores the query and fragment components of the Via URI specified by the **TPCO**.

[<2> Section 3.1.1:](#) The Windows implementation of this protocol ignores the query and fragment components of the Via URI specified by the **TPCO**.

[<3> Section 3.1.1:](#) While establishing the framing session as defined in section [3.2.6.2.1](#), if the communication mode specified by the initiator is Simplex, the Windows implementation of the receiver sends an UnsupportedMode Fault Record to the initiator, as specified in [\[MC-NMF\]](#) section 2.2.5.

[<4> Section 3.1.1:](#) While establishing the framing session as defined in section [3.2.6.2.1](#), if the communication mode specified by the initiator is Singleton-Sized, the Windows implementation of the receiver sends an UnsupportedMode Fault Record to the initiator, as specified in [\[MC-NMF\]](#) section 2.2.5.

[<5> Section 3.1.1:](#) While establishing the framing session as defined in section [3.2.6.2.1](#), if the communication mode specified by the initiator is Duplex and the encoding is Binary, the Windows implementation of the receiver sends a ContentTypeInvalid Fault Record to the initiator, as specified in [\[MC-NMF\]](#) section 2.2.5.

[<6> Section 3.1.1:](#) While establishing the framing session as defined in section [3.2.6.2.1](#), if the communication mode specified by the initiator is Singleton-Unsigned and the encoding is Binary with in-band dictionary, the Windows implementation of the receiver sends a ContentTypeInvalid Fault Record to the initiator, as specified in [\[MC-NMF\]](#) section 2.2.5.

[<7> Section 3.2.6.2.1:](#) The Windows implementation of this protocol aborts the TCP connection if the connection is in the CONNECTED state for longer than two minutes, or if a connection in the CONNECTED state has existed for longer than five minutes.

## 8 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

## 9 Index

### A

Abstract data model  
  initiator  
    [BUSY state](#) 13  
    [CONNECTED state](#) 13  
  overview ([section 3.1.1](#) 11, [section 3.2.1](#) 12)  
  receiver ([section 3.1.1](#) 11, [section 3.3.1](#) 14)  
[Applicability](#) 7  
[Attribute groups](#) 10  
[Attributes](#) 9

### C

[Capability negotiation](#) 7  
[Change tracking](#) 20  
[Complex types](#) 9

### D

Data model - abstract  
  initiator  
    [BUSY state](#) 13  
    [CONNECTED state](#) 13  
  overview ([section 3.1.1](#) 11, [section 3.2.1](#) 12)  
  receiver ([section 3.1.1](#) 11, [section 3.3.1](#) 14)

### E

Events  
  local  
    initiator  
      [End Framing session event](#) 13  
      [TCP connection aborted](#) 12  
    receiver  
      [End Framing session event](#) 15  
      [TCP connection aborted](#) 12  
  timer  
    initiator ([section 3.1.5](#) 12, [section 3.2.5](#) 13)  
    receiver ([section 3.1.5](#) 12, [section 3.3.5](#) 14)  
[Examples - overview](#) 16

### F

[Fields - vendor-extensible](#) 8  
[Full WSDL](#) 18

### G

[Glossary](#) 5  
[Groups](#) 10

### H

Higher-layer triggered events  
  initiator  
    [Connect](#) 14  
    [TCP connection](#) 12  
  receiver - [TCP connection](#) 12

### I

[Implementer - security considerations](#) 17  
[Index of security parameters](#) 17  
[Informative references](#) 7  
Initialization  
  initiator ([section 3.1.3](#) 12, [section 3.2.3](#) 13)  
  receiver ([section 3.1.3](#) 12, [section 3.3.3](#) 14)  
Initiator  
  abstract data model  
    [BUSY state](#) 13  
    [CONNECTED state](#) 13  
  overview ([section 3.1.1](#) 11, [section 3.2.1](#) 12)  
  higher-layer triggered events  
    [Connect](#) 14  
    [TCP connection](#) 12  
  initialization ([section 3.1.3](#) 12, [section 3.2.3](#) 13)  
  local events  
    [End Framing session event](#) 13  
    [TCP connection aborted](#) 12  
  message processing ([section 3.1.4](#) 12, [section 3.2.4](#) 13)  
  overview 11  
  sequencing rules ([section 3.1.4](#) 12, [section 3.2.4](#) 13)  
  timer events ([section 3.1.5](#) 12, [section 3.2.5](#) 13)  
  timers ([section 3.1.2](#) 12, [section 3.2.2](#) 13)  
[Introduction](#) 5

### L

Local events  
  initiator  
    [End Framing session event](#) 13  
    [TCP connection aborted](#) 12  
  receiver  
    [End Framing session event](#) 15  
    [TCP connection aborted](#) 12

### M

Message processing  
  initiator ([section 3.1.4](#) 12, [section 3.2.4](#) 13)  
  receiver ([section 3.1.4](#) 12, [section 3.3.4](#) 14)  
Messages  
  [attribute groups](#) 10  
  [attributes](#) 9  
  [complex types](#) 9  
  [elements](#) 9  
  [enumerated](#) 9  
  [groups](#) 10  
  [namespaces](#) 9  
  [simple types](#) 9  
  [syntax](#) 9  
  [transport](#) 9

### N

[Namespaces](#) 9  
[Normative references](#) 6

## O

[Overview \(synopsis\)](#) 7

## P

[Parameters - security index](#) 17  
[Preconditions](#) 7  
[Prerequisites](#) 7  
[Product behavior](#) 19

## R

### Receiver

abstract data model ([section 3.1.1](#) 11, [section 3.3.1](#) 14)  
[higher-layer triggered events - TCP connection](#) 12  
initialization ([section 3.1.3](#) 12, [section 3.3.3](#) 14)  
local events  
    [End Framing session event](#) 15  
    [TCP connection aborted](#) 12  
message processing ([section 3.1.4](#) 12, [section 3.3.4](#) 14)  
    [overview](#) 11  
sequencing rules ([section 3.1.4](#) 12, [section 3.3.4](#) 14)  
timer events ([section 3.1.5](#) 12, [section 3.3.5](#) 14)  
timers ([section 3.1.2](#) 12, [section 3.3.2](#) 14)

### References

[informative](#) 7  
[normative](#) 6

[Relationship to other protocols](#) 7

## S

### Security

[implementer considerations](#) 17  
[parameter index](#) 17

### Sequencing rules

initiator ([section 3.1.4](#) 12, [section 3.2.4](#) 13)  
receiver ([section 3.1.4](#) 12, [section 3.3.4](#) 14)

[Simple types](#) 9

[Standards assignments](#) 8

[Syntax - messages - overview](#) 9

## T

### Timer events

initiator ([section 3.1.5](#) 12, [section 3.2.5](#) 13)  
receiver ([section 3.1.5](#) 12, [section 3.3.5](#) 14)

### Timers

initiator ([section 3.1.2](#) 12, [section 3.2.2](#) 13)  
receiver ([section 3.1.2](#) 12, [section 3.3.2](#) 14)

[Tracking changes](#) 20

[Transport](#) 9

### Triggered events

initiator  
    [Connect](#) 14  
    [TCP connection](#) 12

[receiver - TCP connection](#) 12

### Types

[complex](#) 9  
[simple](#) 9

## V

[Vendor-extensible fields](#) 8

[Versioning](#) 7

## W

[WSDL](#) 18