[MC-SQLR]:
SQL Server Resolution Protocol Specification

Revision Summary

Date	Revision History	Revision Class	Comments
08/10/2007	0.1	Major	Initial Availability
09/28/2007	0.2	Minor	Updated the technical content.
10/23/2007	0.2.1	Editorial	Revised and edited the technical content.
11/30/2007	0.2.2	Editorial	Revised and edited the technical content.
01/25/2008	0.2.3	Editorial	Revised and edited the technical content.

Table of Contents

1 Introduction

This document specifies the SQL Server Resolution Protocol, a Microsoft proprietary protocol. The SQL Server Resolution Protocol is an application-layer request/response protocol that facilitates connectivity to a database **server** and provides for:

- Communication **endpoint** information; for example, the TCP port for connecting to a particular **instance** of the database server on a machine.
- Database instance enumeration.

1.1 Glossary

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

Broadcast
Endpoint
Little Endian

Client: A program that establishes connections for the purpose of sending requests.

Data Store: A repository for data.

Database Server Discovery Service: A database discovery service is a service that allows applications to discover the existence of database **instances**.

Dedicated Administrator Connection (DAC): A special TCP **endpoint** introduced in SQL Server 2005 that provides a special diagnostic connection for administrators when standard connections to the server are not possible.

Instance: A copy of the database server running on a machine. Multiple instances may reside on a single machine.

Multicast: A data transmission to multiple specific recipients at the same time.

Query: A character string expression sent to a **data store** that contains a set of operations that request data from the **data store**.

Server: An application program that accepts connections to service requests by sending back responses. Any program may be capable of being both a **client** and a server. Use of these terms refers only to the role performed by the program for a particular connection rather than to the program's capabilities in general.

Virtual Interface Architecture (VIA): A high-speed interconnect requiring special hardware and drivers provided by third parties.

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

[RFC768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980, http://www.ietf.org/rfc/rfc768.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

1.2.2 Informative References

[MSDN-DAC] Microsoft Corporation, "Using a Dedicated Administrator", http://msdn2.microsoft.com/en-us/library/ms189595.aspx

[MSDN-CS] Microsoft Corporation, "Character Sets", http://msdn2.microsoft.com/en-us/library/ms776430.aspx

1.3 Protocol Overview (Synopsis)

The SQL Server Resolution Protocol is a simple application-level protocol used for the transfer of requests and responses between **clients** and **database server discovery services**.<1> In such a system, the client would either (I) send a single request to a specific machine and expect a single response, or (ii) **broadcast** or **multicast** <2> a request to the network and expect zero or more responses from different discovery services on the network. The first case is used for the purpose of determining the communication endpoint information of a particular database instance, while the second case is used for enumeration of database instances in the network and to obtain the endpoint information of each instance.

The SQL Server Resolution Protocol is a stateless protocol and does not include any facilities for authentication, protection of data, or reliability. The SQL Server Resolution Protocol is always implemented on top of the UDP Transport Protocol [RFC768].

In the case of endpoint determination for a single instance, the following diagram depicts a typical flow of communication:

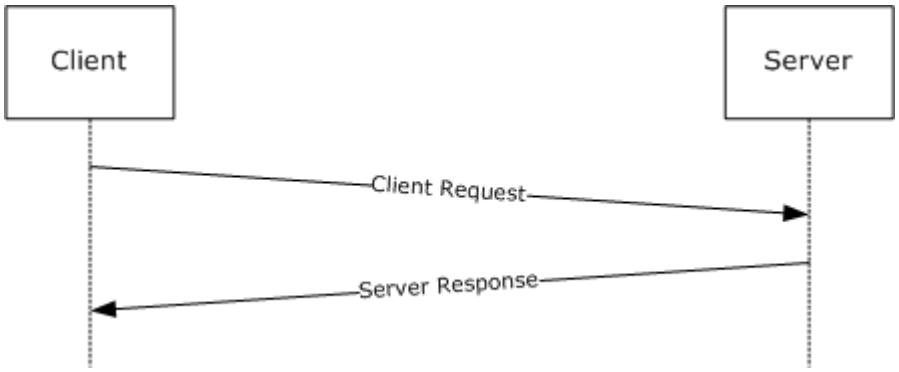


Figure 1: Communication flow for single-instance endpoint discovery

Conversely, in the case of a broadcast/multicast request, the following diagram applies:

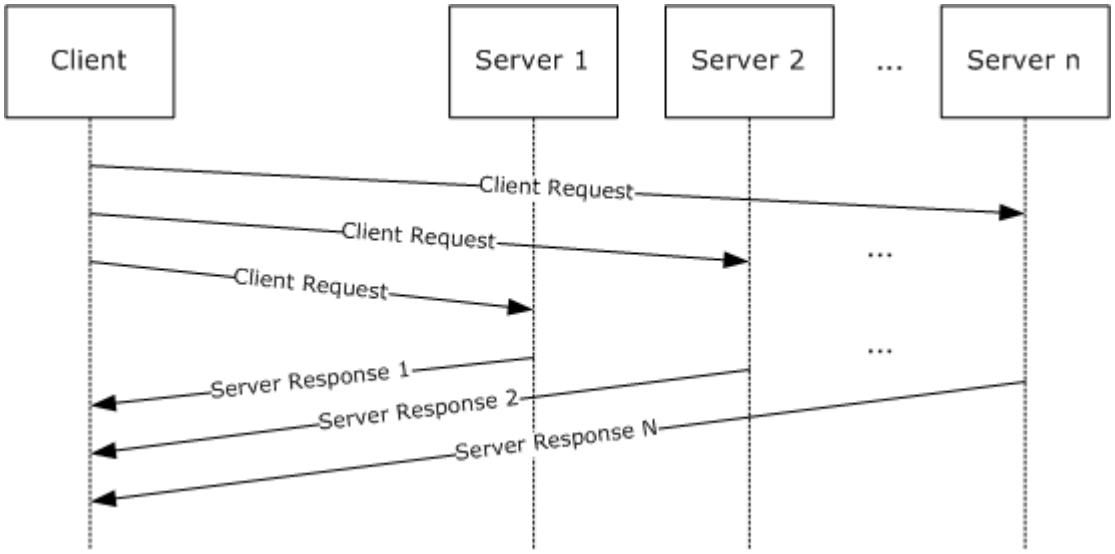


Figure 2: Communication flow for multiple-instance endpoint discovery

In the case of a broadcast or multicast request, the client does not necessarily have any knowledge as to the number of responses that it can expect. As a result, it would be reasonable for the client to enforce a time limitation during which it will wait for responses. Given that some servers may not respond quickly enough, or they may not receive the request (highly dependent upon network topology), the broadcast/multicast request for multiple-instance endpoint information should be considered nondeterministic.

1.4 Relationship to Other Protocols

The SQL Server Resolution Protocol (SSRP) depends upon the UDP Transport Protocol to communicate with the database server machine or to broadcast/multicast its request to the network. No other protocols are involved.

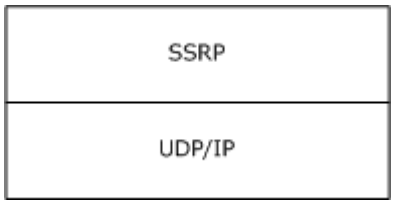


Figure 3: Protocol relationship

1.5 Prerequisites/Preconditions

Aside from the requirement of UDP, there are no prerequisites or preconditions.

1.6 Applicability Statement

The SQL Server Resolution Protocol is appropriate for use to facilitate retrieval of database endpoint information or for database instance enumeration in all scenarios where network or local connectivity is available.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

Supported Transports: This protocol is implemented on top of UDP as discussed in section 2.1.

Protocol Versions: The SQL Server Resolution Protocol supports the following explicit dialect: "SSRP 1.0", as defined in section 2.2.

Security and Authentication Methods: The SQL Server Resolution Protocol does not provide or support any security or authentication methods.

Localization: Localization-dependent protocol behavior is specified in sections 2.2 and 3.1.5.

Capability Negotiation: The SQL Server Resolution Protocol does not support negotiation of the dialect to use. Instead, an implementation must be configured with the dialect to use, as described below in this section.

There is no version or capability negotiation supported by the SQL Server Resolution Protocol. The client will send a request message to the server with the expectation that the server will understand the message and send back a response. If the server does not understand the message, the server will ignore the request and will not send a response back to the client.

1.8 Vendor-Extensible Fields

The SQL Server Resolution Protocol does not include any vendor-extensible fields.

1.9 Standards Assignments

Parameter	Value	Reference
Microsoft-SQL-Monitor (ms-sql-m)	1434/UDP	http://www.iana.com/assignments/port-numbers

The client will always send its request to UDP port 1434 of the server(s).

2 Messages

The following sections specify how SQL Server Resolution Protocol messages are transported and SQL Server Resolution Protocol message syntax.

2.1 Transport

The SQL Server Resolution Protocol uses port 1434 of the UDP Protocol for message transport.

2.2 Message Syntax

All integer fields are represented in **little-endian** format. All text strings are represented as a multiple-byte character set (MBCS) string [MSDN-CS] in the current system Windows ANSI code page of the server and the client (the system codepage on the client and server are assumed to be common). They are not case sensitive.

The list of valid requests from the client are:

- CLNT_BCAST_EX
- CLNT_UCAST_EX
- CLNT_UCAST_INST
- CLNT_UCAST_DAC

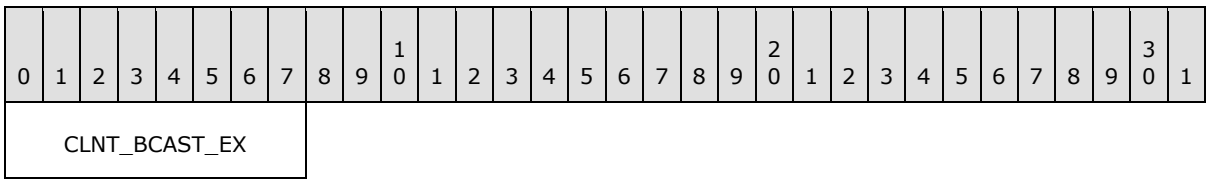
The response from the server is always a SVR_RESP message. The response contains a string data field that is parsed by the client to extract the requested information. The contents of this string are not case sensitive.

These messages will be explained in more detail in the following sections.

2.2.1 CLNT_BCAST_EX

The CLNT_BCAST_EX packet is a broadcast or multicast request generated by clients trying to identify the list of database instances on the network and their network protocol connection information. The server's responses include the following information:

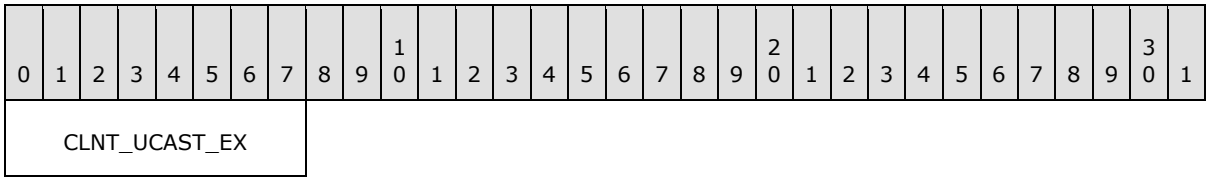
- Servername
- Instancename
- ServerInformation
- ServerVersion
- ProtocolInformation



Name	Value
CLNT_BCAST_EX	0x02

2.2.2 CLNT_UCAST_EX

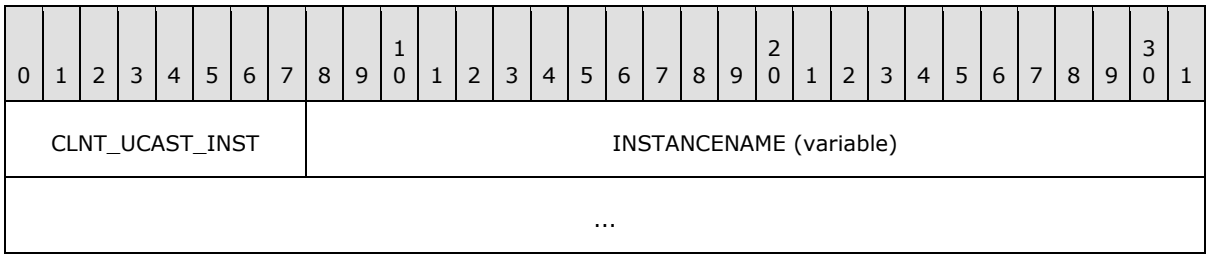
The CLNT_UCAST_EX packet is a unicast request. The client generates a UDP packet with a single byte as shown below. The server responds in the same fashion as it would to a broadcast/multicast request.



Name	Value
CLNT_UCAST_EX	0x03

2.2.3 CLNT_UCAST_INST

The CLNT_UCAST_INST packet is a request for information related to a specific instance. The server responds with information for the requested instance only. The structure of the request is as follows:



Name	Value
CLNT_UCAST_INST	0x04

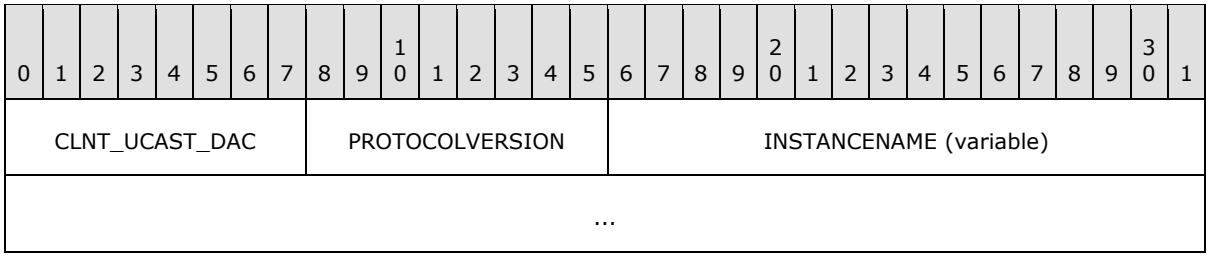
Name	Value
INSTANCENAME	Variable length, null-terminated multiple-byte character set (MBCS) string; does not need to be byte aligned; maximum 32 bytes, not including the null terminator.

INSTANCENAME corresponds to the name of the database instance for which the information is requested. The INSTANCENAME is a sequence of bytes representing either single- or double-byte

characters; that is, MBCS, as determined by the system local code page of the client. The instance name is variable in length up to a 32-byte limit and is null terminated.

2.2.4 CLNT_UCAST_DAC

The CLNT_UCAST_DAC packet request is used to determine the tcp port on which the Microsoft SQL Server™ **Dedicated Administrator Connection (DAC)** [MSDN-DAC] endpoint is listening.



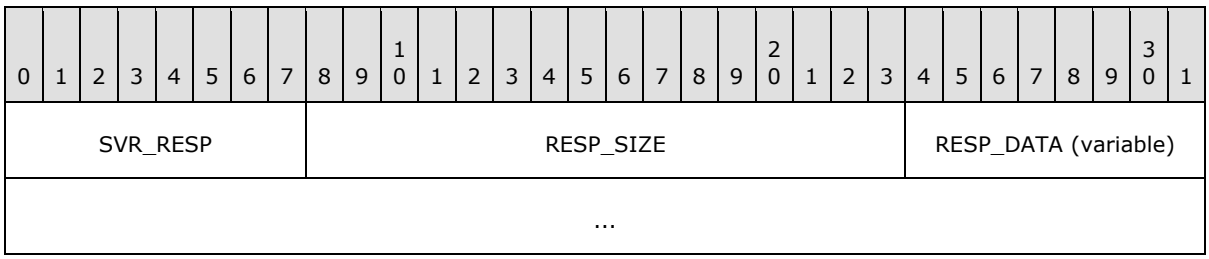
Name	Value
CLNT_UCAST_DAC	0x0F

Name	Value
PROTOCOLVERSION	0x01

Name	Value
INSTANCENAME	Variable length, null-terminated MBCS string; does not need to be byte aligned; maximum 32 bytes, not including the null terminator.

2.2.5 SVR_RESP

The server will respond to all client requests with a SVR_RESP. There is a slight variation in the message format for a response to a CLNT_UCAST_DAC request, as described in the following section.



Name	Value
SVR_RESP	0x05

Name	Value
RESP_SIZE	Two-byte length of subsequent response data.

Name	Value
RESP_DATA	Variable length MBSC string; does not need to be byte aligned.

Note The server will only send back a SVR_RESP if there is at least one instance for which it can send back endpoint information. If no such instances exist, then the server will ignore the client request.

The content of the RESP_DATA string corresponds to the type of request received. For example, a CLNT_UCAST_EX request would be answered with a CLNT_UCAST_EX_RESPONSE string.

```
RESP_DATA = CLNT_BCAST_EX_RESPONSE / CLNT_UCAST_EX_RESPONSE / CLNT_UCAST_INST_RESPONSE

CLNT BCAST EX RESPONSE = CLNT UCAST EX RESPONSE

CLNT UCAST EX RESPONSE = *[CLNT UCAST INST RESPONSE]

CLNT_UCAST_INST_RESPONSE = "ServerName" SEMICOLON SERVERNAME SEMICOLON
"InstanceName" SEMICOLON INSTANCENAME SEMICOLON "IsClustered"
SEMICOLON YES OR NO SEMICOLON "Version" SEMICOLON VERSION STRING
[NP INFO] [TCP INFO] [VIA INFO] SEMICOLON SEMICOLON ; see Note 1

SEMICOLON = ";"

YES_OR_NO = "Yes" / "No"

INSTANCENAME is a text string representing the name of the server
instance being described

VERSION_STRING is a text string conveying the version of the server instance

TCP INFO = SEMICOLON "tcp" SEMICOLON TCP PORT

TCP PORT is a text string representing the decimal value of the TCP
port ; see Note 2

NP INFO = SEMICOLON "np" SEMICOLON PIPENAME

PIPENAME is a text string representing the pipe name

VIA_INFO = SEMICOLON "via" NETBIOS VIALISTENINFO

NETBIOS is a text string representing the NetBios Value, i.e., server name

VIALISTENINFO = 1*["," VIANIC ":" VIAPORT]

VIANIC is a text string representing the VIA network interface card
(NIC) identifier
```

```
VIAPORT is a text string representing the decimal value of the VIA
NIC's port
```

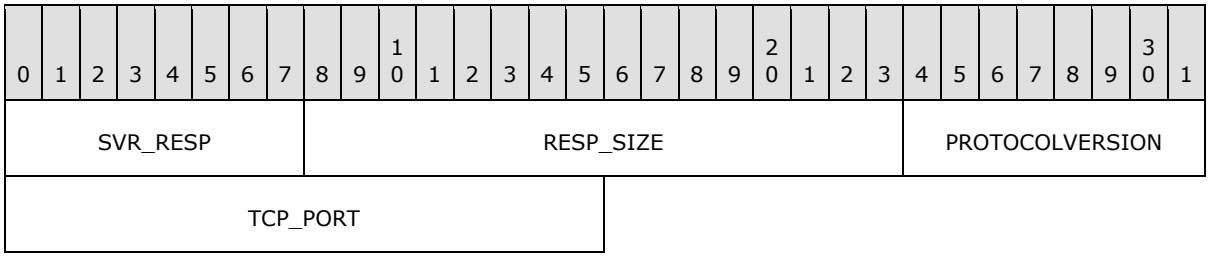
Note 1. NP_INFO, TCP_INFO, and VIA_INFO may be listed in any order.

Note 2. If the request is received on an IPv4 socket, then the response will provide the instance's port associated with an IPv4 address. Likewise for IPv6.

Note that the maximum size of the SQL Server Resolution Protocol response is 1024 bytes per instance. When returning protocol information, the server may include information for a particular protocol as long as the aggregate information for the instance fits within the 1-KB limit. If the information for a protocol would cause the total protocol information to exceed 1 KB—for example, trying to add a 800-KB pipe name when there is already 500 bytes of protocol information collected—that protocol information should not be sent, and the information for the next protocol (if any) should be included in the response (assuming that it does not cause the response to exceed the 1-KB limit). Furthermore, the server should not include protocol information if the information returned would be invalid. For example, if the TCP port is unavailable, TCP should not be included in the response.

2.2.6 SVR_RESP (DAC)

The format of the SVR_RESP is different for a CLNT_UCAST_DAC request only.



Name	Value
SVR_RESP	0x05

Name	Value
RESP_SIZE	0x06 (Two-byte length of the entire response packet.)

Name	Value
PROTOCOLVERSION	0x01

Name	Value
TCP_PORT	Two-byte value of the TCP port number.

3 Protocol Details

This section describes the important elements of the client software and the server software necessary to support the SQL Server Resolution Protocol.

As described in section 1.3, the SQL Server Resolution Protocol is an application-level protocol that is used to **query** information regarding database instance(s) on a server. The protocol defines a limited set of messages through which the client may make a request to the server(s). The SQL Server Resolution Protocol is stateless and involves a single request and a single response from one or more servers.

3.1 Server Details

The following state machine diagram describes the SQL Server Resolution Protocol on the server side.

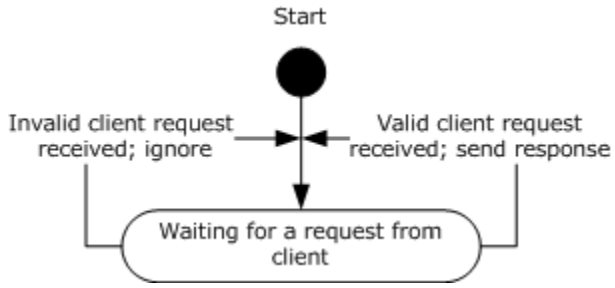


Figure 4: SQL Server Resolution Protocol server state machine

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 provided that their external behavior is consistent with that described in this document.

The SQL Server Resolution Protocol server does not need to maintain any state data. Upon receipt of a valid request, it will send a response, if appropriate, and immediately wait for the next request. If the request cannot be understood, the server will ignore the request and wait for the next request.

3.1.2 Timers

No timers are required by the SQL Server Resolution Protocol server.

3.1.3 Initialization

The SQL Server Resolution Protocol does not perform any initialization on the server side. A UDP socket listening on port 1434 is assumed to have been created by the upper layer.

3.1.4 Higher-Layer Triggered Events

The SQL Server Resolution Protocol server does not have any events triggered by the higher layer.

3.1.5 Message Processing Events and Sequencing Rules

Upon receipt of a message from the client, the server will immediately send a SVR_RESP response back to the client. The data content of the response is dependent upon the request type.

- For CLNT_BCAST_EX and CLNT_UCAST_EX, the server will return information about all available instances.
- For CLNT_UCAST_INST, the server will return information about the specified instance only (if available).
- For CLNT_UCAST_DAC, the server will return information about the Dedicated Administrator Connection only.

In all cases, after a response is returned to the client, the server should immediately wait for the next request. If the request is invalid or not understood, then the server will ignore the request and wait for the next request. Since the SQL Server Resolution Protocol involves a single response per server for each client request, there are no sequencing issues.

3.1.6 Timer Events

No timer events exist for a SQL Server Resolution Protocol server.

3.1.7 Other Local Events

No other local events exist for a SQL Server Resolution Protocol server.

3.2 Client Details

The following state machine diagram describes the SQL Server Resolution Protocol on the client side.

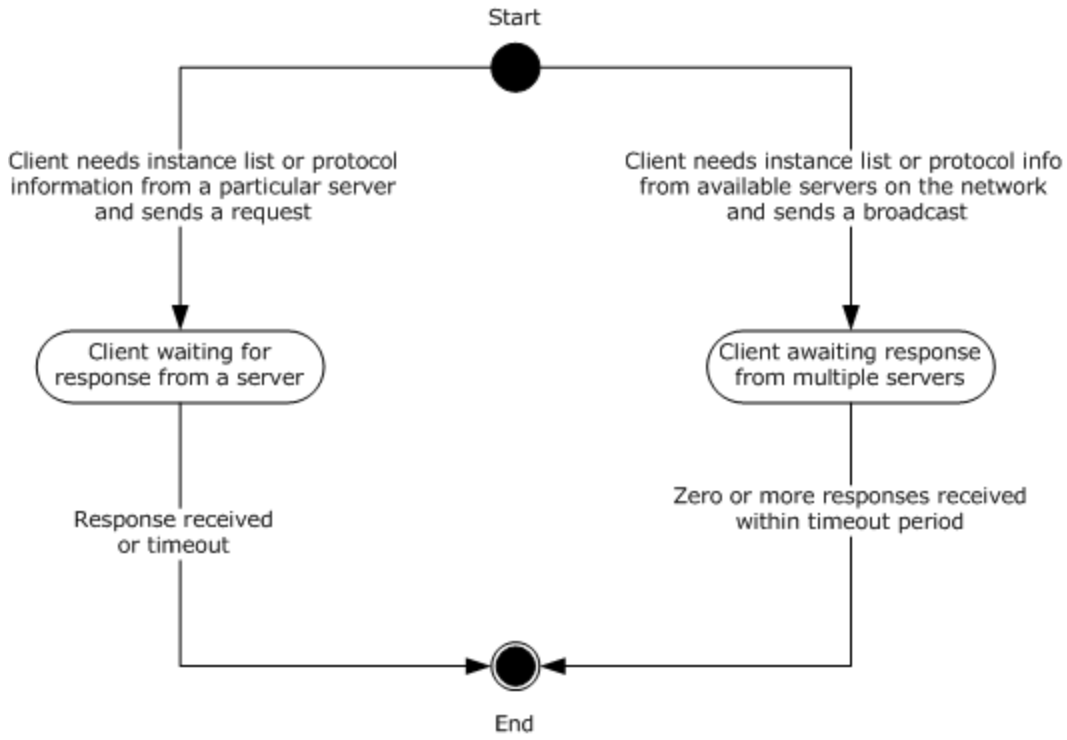


Figure 5: SQL Server Resolution Protocol client state machine

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 provided that their external behavior is consistent with that described in this document.

A SQL Server Resolution Protocol client does not need to maintain any state data except for the knowledge of the request sent to the server. With the exception of the CLNT_BCAST_EX message, the client will wait for a SVR_RESP from the server or else time out, whichever occurs first. For CLNT_BCAST_EX, the client will wait the entire duration of a finite period of time for responses from multiple servers.

3.2.2 Timers

The SQL Server Resolution Protocol client SHOULD implement a timer for the maximum amount of time to wait for a SVR_RESP message from the server once a CLNT_UCAST_EX, CLNT_UCAST_INST, or CLNT_UCAST_DAC request is sent. The timer value may be any value greater than zero, but is recommended to be 10 seconds.

The SQL Server Resolution Protocol client MUST implement a timer for the maximum amount of time to wait for SVR_RESP messages from servers in the network once a CLNT_BCAST_EX request is broadcast/multicast. The timer value may be any value greater than zero, but is recommended to be 15 seconds.

3.2.3 Initialization

The SQL Server Resolution Protocol does not perform any initialization on the client side. A UDP socket is assumed to be created prior to requesting that a SQL Server Resolution Protocol request be sent.

3.2.4 Higher-Layer Triggered Events

The SQL Server Resolution Protocol client implementation MUST support the following event from the higher layer:

- Send client request to server(s). The type of request will dictate if the message is sent to a specific machine or broadcast/multicast to the network. The higher layer must have already created a UDP socket prior to triggering the SQL Server Resolution Protocol client to send a request message. Once the message is sent, the timer SHOULD be started.

The higher layer will be responsible for closing the UDP socket once the response is received or a time-out situation has occurred.

3.2.5 Message Processing Events and Sequencing Rules

As detailed in section 2.2, the server will always respond with a SVR_RESP message. As also previously discussed, the format for the SVR_RESP will be slightly different if it is in response to a CLNT_UCAST_DAC request. If the client receives a message that it does not understand, an error will be returned to the higher level and the socket should be closed. The exception is for a CLNT_BCAST_EX request. In this situation, if the client receives a message that it does not understand, it will ignore the message and continue listening for additional responses until the time-out period has elapsed; at this point the socket should be closed.

Since the SQL Server Resolution Protocol involves a single response per server for each client request, there are no sequencing issues.

3.2.6 Timer Events

If the SQL Server Resolution Protocol client implements a timer for the response to a unicast request, the client MUST close the UDP socket upon expiration of the timer. Likewise, when the timer for the response to a broadcast/multicast request expires, the client MUST close the UDP socket.

3.2.7 Other Local Events

No other local events exist for a SQL Server Resolution Protocol client.

4 Protocol Examples

The following are examples of the binary representation of various client requests and the responses from the server.

4.1 CLNT_UCAST_EX

Request:

```
03
```

Response:

```
05 47 01 53 65 72 76 65 72 4e 61 6d 65 3b 49 4c | .G.ServerName;IL
53 55 4e 47 31 3b 49 6e 73 74 61 6e 63 65 4e 61 | SUNG1;InstanceNa
6d 65 3b 59 55 4b 4f 4e 53 54 44 3b 49 73 43 6c | me;YUKONSTD;IsCl
75 73 74 65 72 65 64 3b 4e 6f 3b 56 65 72 73 69 | ustered;No;Versi
6f 6e 3b 39 2e 30 30 2e 31 33 39 39 2e 30 36 3b | on;9.00.1399.06;
74 63 70 3b 35 37 31 33 37 3b 3b 53 65 72 76 65 | tcp;57137;;Serve
72 4e 61 6d 65 3b 49 4c 53 55 4e 47 31 3b 49 6e | rName;ILSUNG1;In
73 74 61 6e 63 65 4e 61 6d 65 3b 59 55 4b 4f 4e | stanceName;YUKON
44 45 56 3b 49 73 43 6c 75 73 74 65 72 65 64 3b | DEV;IsClustered;
4e 6f 3b 56 65 72 73 69 6f 6e 3b 39 2e 30 30 2e | No;Version;9.00.
31 33 39 39 2e 30 36 3b 6e 70 3b 5c 5c 49 4c 53 | 1399.06;np;\\ILS
55 4e 47 31 5c 70 69 70 65 5c 4d 53 53 51 4c 24 | UNG1\pipe\MSSQL$
59 55 4b 4f 4e 44 45 56 5c 73 71 6c 5c 71 75 65 | YUKONDEV\sql\que
72 79 3b 3b 53 65 72 76 65 72 4e 61 6d 65 3b 49 | ry;;ServerName;I
4c 53 55 4e 47 31 3b 49 6e 73 74 61 6e 63 65 4e | LSUNG1;InstanceN
61 6d 65 3b 4d 53 53 51 4c 53 45 52 56 45 52 3b | ame;MSSQLSERVER;
49 73 43 6c 75 73 74 65 72 65 64 3b 4e 6f 3b 56 | IsClustered;No;V
65 72 73 69 6f 6e 3b 39 2e 30 30 2e 31 33 39 39 | ersion;9.00.1399
2e 30 36 3b 74 63 70 3b 31 34 33 33 3b 6e 70 3b | .06;tcp;1433;np;
5c 5c 49 4c 53 55 4e 47 31 5c 70 69 70 65 5c 73 | \\ILSUNG1\pipe\s
71 6c 5c 71 75 65 72 79 3b 3b                   | ql\query;;
```

The response conveys the instance information for 3 instances named "YUKONSTD", "YUKONDEV", and "MSSQLSERVER".

4.2 CLNT_UCAST_INST

Request:

```
04 59 55 4b 4f 4e 53 54 44 00                   | .YUKONSTD
```

The request is for information for an instance named "YUKONSTD".

Response:

```
05 58 00 53 65 72 76 65 72 4e 61 6d 65 3b 49 4c | .X.ServerName;IL
53 55 4e 47 31 3b 49 6e 73 74 61 6e 63 65 4e 61 | SUNG1;InstanceNa
6d 65 3b 59 55 4b 4f 4e 53 54 44 3b 49 73 43 6c | me;YUKONSTD;IsCl
75 73 74 65 72 65 64 3b 4e 6f 3b 56 65 72 73 69 | ustered;No;Versi
```

```
6f 6e 3b 39 2e 30 30 2e 31 33 39 39 2e 30 36 3b | on;9.00.1399.06;
74 63 70 3b 35 37 31 33 37 3b 3b                | tcp;57137;;
```

4.3 CLNT_UCAST_DAC

Request:

```
0f 01 59 55 4b 4f 4e 53 54 44 00                | ..YUKONSTD
```

The request is for the DAC port of an instance named "YUKONSTD".

Response:

```
05 06 00 01 32 df                               | ....2
```

The port number is 0xDF32 or 57138.

5 Security

5.1 Security Considerations for Implementers

There are no security considerations associated with the SQL Server Resolution Protocol.

5.2 Index of Security Parameters

There are no security parameters for the SQL Server Resolution Protocol.

6 Appendix A: Windows Behavior

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

- Windows NT
- Windows 2000
- Windows XP
- Windows Server 2003
- Windows Vista

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 1.3: SQL Server Browser is an example of a database server discovery service in the Windows environment.

<2> Section 1.3: The IPv6 standard supports multicasting but not broadcasting.

7 Index

T

V

W