X. Fu
B. Schloer
Univ. Goettingen
H. Tschofenig
Siemens
T.Tsenov

Internet-Draft
Expires: September 2006
March 2006

QoS NSLP State Machine
draft-fu-nsis-qos-nslp-statemachine-03.txt

Status of this Memo

Copyright Notice

Abstract

This document describes the state machines for the NSIS Signaling Layer Protocol for Quality-of-Service signaling (QoS NSLP). A set of state machines for QoS NSLP entities at different locations of a flow path are presented in order to illustrate how QoS NSLP may be implemented.

Table of Contents

1. Introduction 3
2. Terminology 3
3. Notational conventions used in state diagrams 3
4. State Machine Symbols 4
5. Common Rules 6
 5.1 Common Procedures 6
 5.2 Common Variables 7
 5.3 Constants 8
 5.4 Assumptions 8
6. State machines 9
 6.1 Diagram notations 9
 6.2 State machine for QNI QoS NSLP node 10
 6.3 State machine for QNE QoS NSLP node 12
 6.4 State machine for QNR QoS NSLP node 15
7. Security Considerations 16
8. Open Issues 16
9. Change History 16
 9.1 Changes in Version -01 16
 9.2 Changes in Version -02 16
10. Acknowledgments 17
11. References 17
 11.1 Normative References 17
 11.2 Informative References 17
Appendix A. ASCII versions of the state diagrams 18
 A.1 State machine for QNI QoS NSLP node (Figure 2,3) 18
 A.2 State machine for QNI QoS NSLP node (Figure 4,5,6) 20
 A.3 State machine for QNI QoS NSLP node (Figure 7) 24
 Authors' Addresses 27
 Intellectual Property and Copyright Statements 28

1. Introduction

This document describes the state machines for QoS NSLP [1], trying to show how QoS NSLP can be implemented to support its deployment. The state machines described in this document are illustrative of how the QoS NSLP protocol defined in [1] may be implemented for the QNI QoS NSLP node, QNE QoS NSLP nodes, and QNR QoS NSLP node in the flow path. Where there are differences [1] are authoritative. The state machines are informative only. Implementations may achieve the same results using different methods.

According to [1], there are several possibilities for QoS NSLP signaling, at least including the following: - end-to-end signaling vs. scoped signaling - sender-initiated signaling vs. receiver-initiated signaling (which need to be incorporated into use scenarios when describing state machine. Note they are represented by way of certain objects/flags in Reserve and Query messages.)

The messages used in the QoS NSLP protocol can be summarized as follows:

```
Requesting message        Responding message
------------------------+--------------------------
RESERVE                 |None or RESERVE or RESPONSE
QUERY                   |RESERVE or RESPONSE
RESPONSE                |NONE
NOTIFY                  |NONE
------------------------+--------------------------
```

We describe a set of state machines for different roles of entities running QoS NSLP to illustrate how QoS NSLP may be implemented.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [2].

3. Notational conventions used in state diagrams

The following text is reused from [3] and the state diagrams are based on the conventions specified in [4], Section 8.2.1. Additional state machine details are taken from [5].

The complete text is reproduced here:

State diagrams are used to represent the operation of the protocol by a number of cooperating state machines each comprising a group of connected, mutually exclusive states. Only one state of each machine can be active at any given time.

All permissible transitions between states are represented by arrows, the arrowhead denoting the direction of the possible transition. Labels attached to arrows denote the condition(s) that must be met in order for the transition to take place. All conditions are expressions that evaluate to TRUE or FALSE; if a condition evaluates to TRUE, then the condition is met. The label UCT denotes an unconditional transition (i.e., UCT always evaluates to TRUE). A transition that is global in nature (i.e., a transition that occurs from any of the possible states if the condition attached to the arrow is met) is denoted by an open arrow; i.e., no specific state is identified as the origin of the transition. When the condition associated with a global transition is met, it supersedes all other exit conditions including UCT. The special global condition BEGIN supersedes all other global conditions, and once asserted remains asserted until all state blocks have executed to the point that variable assignments and other consequences of their execution remain unchanged.

On entry to a state, the procedures defined for the state (if any) are executed exactly once, in the order that they appear on the page. Each action is deemed to be atomic; i.e., execution of a procedure completes before the next sequential procedure starts to execute. No procedures execute outside of a state block. The procedures in only one state block execute at a time, even if the conditions for execution of state blocks in different state machines are satisfied, and all procedures in an executing state block complete execution before the transition to and execution of any other state block occurs, i.e., the execution of any state block appears to be atomic with respect to the execution of any other state block and the transition condition to that state from the previous state is TRUE when execution commences. The order of execution of state blocks in different state machines is undefined except as constrained by their transition conditions. A variable that is set to a particular value in a state block retains this value until a subsequent state block executes a procedure that modifies the value.

On completion of all of the procedures within a state, all exit conditions for the state (including all conditions associated with global transitions) are evaluated continuously until one of the conditions is met. The label ELSE denotes a transition that occurs if none of the other conditions for transitions from the state are met (i.e., ELSE evaluates to TRUE if all other possible exit conditions from the state evaluate to FALSE). Where two or more exit conditions with the same level of precedence become TRUE simultaneously, the choice as to which exit condition causes the state transition to take place is arbitrary.

In addition to the above notation, there are a couple of clarifications specific to this document. First, all boolean variables are initialized to FALSE before the state machine execution begins. Second, the following notational shorthand is specific to this document:

<variable> = <expression1> | <expression2> | ...

> Execution of a statement of this form will result in <variable> having a value of exactly one of the expressions. The logic for which of those expressions gets executed is outside of the state machine and could be environmental, configurable, or based on another state machine such as that of the method.

4. State Machine Symbols

( )

> Used to force the precedence of operators in Boolean expressions and to delimit the argument(s) of actions within state boxes.

;

> Used as a terminating delimiter for actions within state boxes. Where a state box contains multiple

actions, the order of execution follows the normal English language conventions for reading text.

=

Assignment action. The value of the expression to the right of the operator is assigned to the variable to the left of the operator. Where this operator is used to define multiple assignments, e.g., a = b = X the action causes the value of the expression following the right-most assignment operator to be assigned to all of the variables that appear to the left of the right-most assignment operator.

!

Logical NOT operator.

&&

Logical AND operator.

||

Logical OR operator.

if...then...

Conditional action. If the Boolean expression following the if evaluates to TRUE, then the action following the then is executed.

{ statement 1, ... statement N }

Compound statement. Braces are used to group statements that are executed together as if they were a single statement.

!=

Inequality. Evaluates to TRUE if the expression to the left of the operator is not equal in value to the expression to the right.

==

Equality. Evaluates to TRUE if the expression to the left of the operator is equal in value to the expression to the right.

>

Greater than. Evaluates to TRUE if the value of the expression to the left of the operator is greater than the value of the expression to the right.

<=

Less than or equal to. Evaluates to TRUE if the value of the expression to the left of the operator is either less than or equal to the value of the expression to the right.

++

Increment the preceding integer operator by 1.

+

Arithmetic addition operator.

&

Bitwise AND operator.

5. Common Rules

Throughout the document we use terms defi ned in the [1], such as fbw sender, fbw receiver, QUERY, RESERVE or RESPONSE.

5.1 Common Procedures

Tx_RESERVE():
: Transmit RESERVE message

Tx_RESPONSE():
: Transmit RESPONSE message

Tx_QUERY():
: Transmit QUERY message

Tx_NOTIFY():
: Transmit NOTIFY message

Rx_RESPONSE():
: Receive RESPONSE message

Rx_QUERY():
: Receive QUERY message

Rx_RESERVE():
: Receive RESERVE message

Tx_NOTIFY():
: Transmit NOTIFY message

TIMEOUT_StateLifetime:
: State lifetime timer expiration

TIMEOUT_Refresh:
: Refresh interval timer expiration

TIMEOUT_Refresh:
: Wait-Response interval timer expiration

Tg_QUERY:
: External trigger to send a QUERY message (typically triggered by the application).

Tg_RESERVE:

External trigger to send a RESERVE message.

Tg_TEARDOWN:
External trigger to clear previously established QoS state (typically triggered by the application). It is translated to a tx_RESERVE(Ton) message.

Install QoS state:
Install the local QoS state.

Refresh QoS state:
Refresh the local QoS state.

Delete QoS state:
Delete the local QoS state.

Send info to Application:
Report information to the application.

RMF:
Performs Resource Management Function and returns the following values{AVAIL, NO_AVAIL}.

SetRII:
Sets the RII object of the messages e.g. the node requests explicit response to the message being sent. Returns values {0,1}.

CheckRII:
Checks the RII object of received RESPONSE message if it is requested by current node or other upstream node. Returns values {LOCAL, NO_LOCAL}.

ProcessQUERY:
Processes a Query message and provides the requested info

5.2 Common Variables

RII:
Request Identification Information (RII) object. Logical variable representing if the RII is set or not. Takes values {0,1}.

SCOPING:
Scoping flag of common message header. Takes values {"Next_hop","Whole_path"}.

RSN:
Reservation Sequence Number object. Takes values:
- recRSN - RSN object of the received message
- currRSN - Current stored RSN value for installed QoS state. (Assumed to be the one for the direction where the message comes from e.g.Upstream/Downstream)

ACK:

Acknowledgement flag of common message header. Takes values {"On","Off"}.

ReducedRefresh:
Keeps information if Reduced refresh method may be used for refreshing a installed QoS state. Takes value {"On","Off"}.

E_SPEC:
Error_Spec object. Takes values:
- 0x02? - Success values
- 0x04? - Transient Failure values

QSPEC:
QoS specification object.

FlowID:
Flow ID kept by the installed QoS state.

Replace:
Replace flag of common message header. Takes values {"On","Off"}.

SII:
Source Identification Information entry. Takes values:
- CurrSII - SII entry stored for current installed QoS state. (Assumed to be the one for the direction where the message comes from e.g.Upstream/Downstream)
- newSII - SII of the received message is different from the SII stored for the current installed QoS state.

5.3 Constants

5.4 Assumptions

- For simplification not all included objects in a message are showed. Only those that are significant for the case are showed. State machines do not present handling of messages that are not significant for management of the states such as certain NOTIFY and QUERY messages.
- State machines represent handling of messages of the same Session ID and with no protocol errors. Separate parallel instances of the state machines should handle messages for different Session IDs.
- Default message handling should be defined for messages with different Session IDs that have impact on current session state and error messages. This is not included in the current version.
- ACK flag in the common header is set "On" by default.
- Direction of receiving and sending messages is not specified. We assume it is implicit from the context.

6. State machines

6.1 State machine for QNI QoS NSLP node

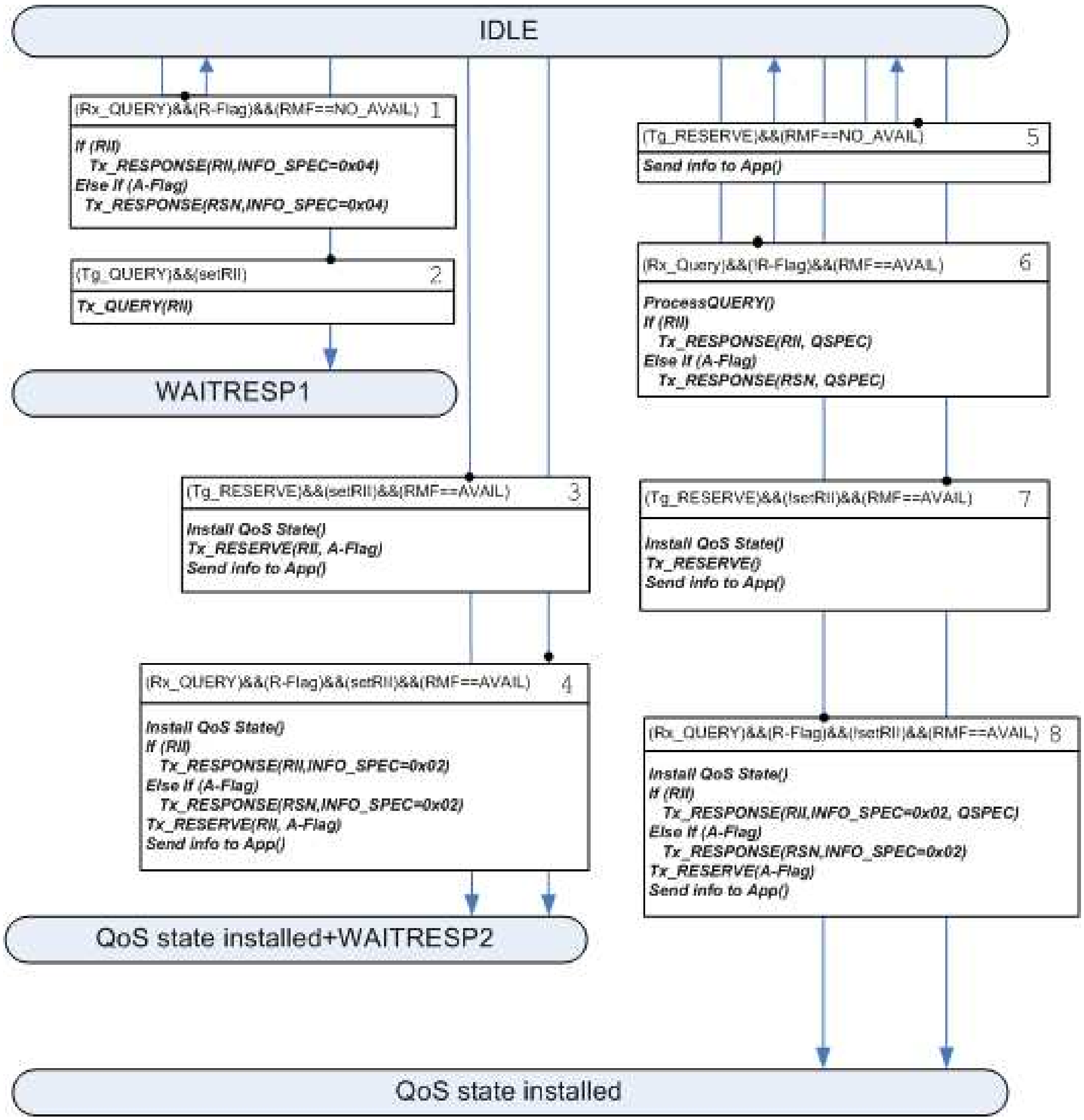

Figure 1: QNI node: "IDLE" State

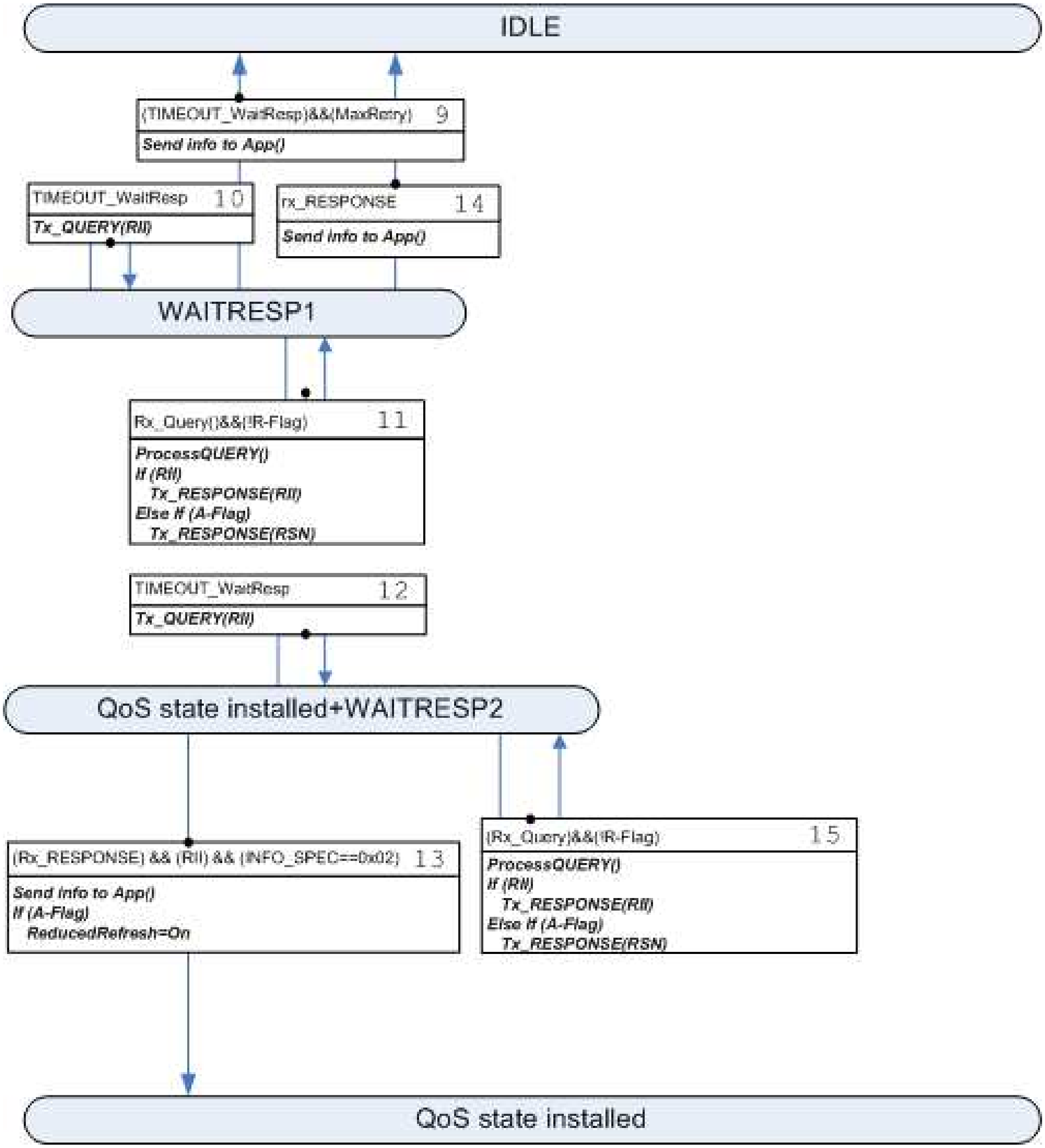

Figure 2: QNI node: "WAITRESP1", "WAITRESP2"

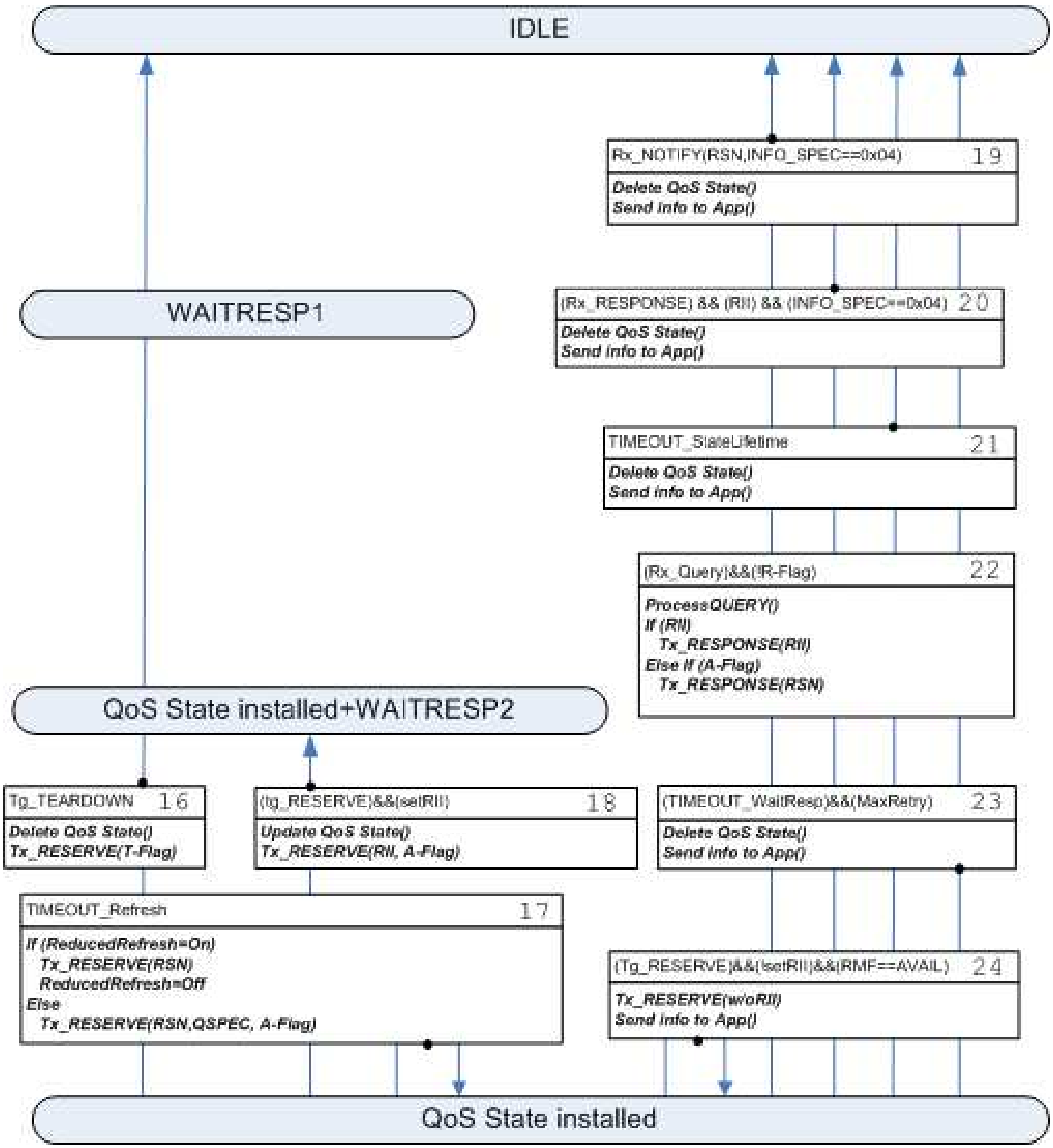

Figure 3: QNI node: "QoS state installed" state

6.2 State machine for QNE QoS NSLP node

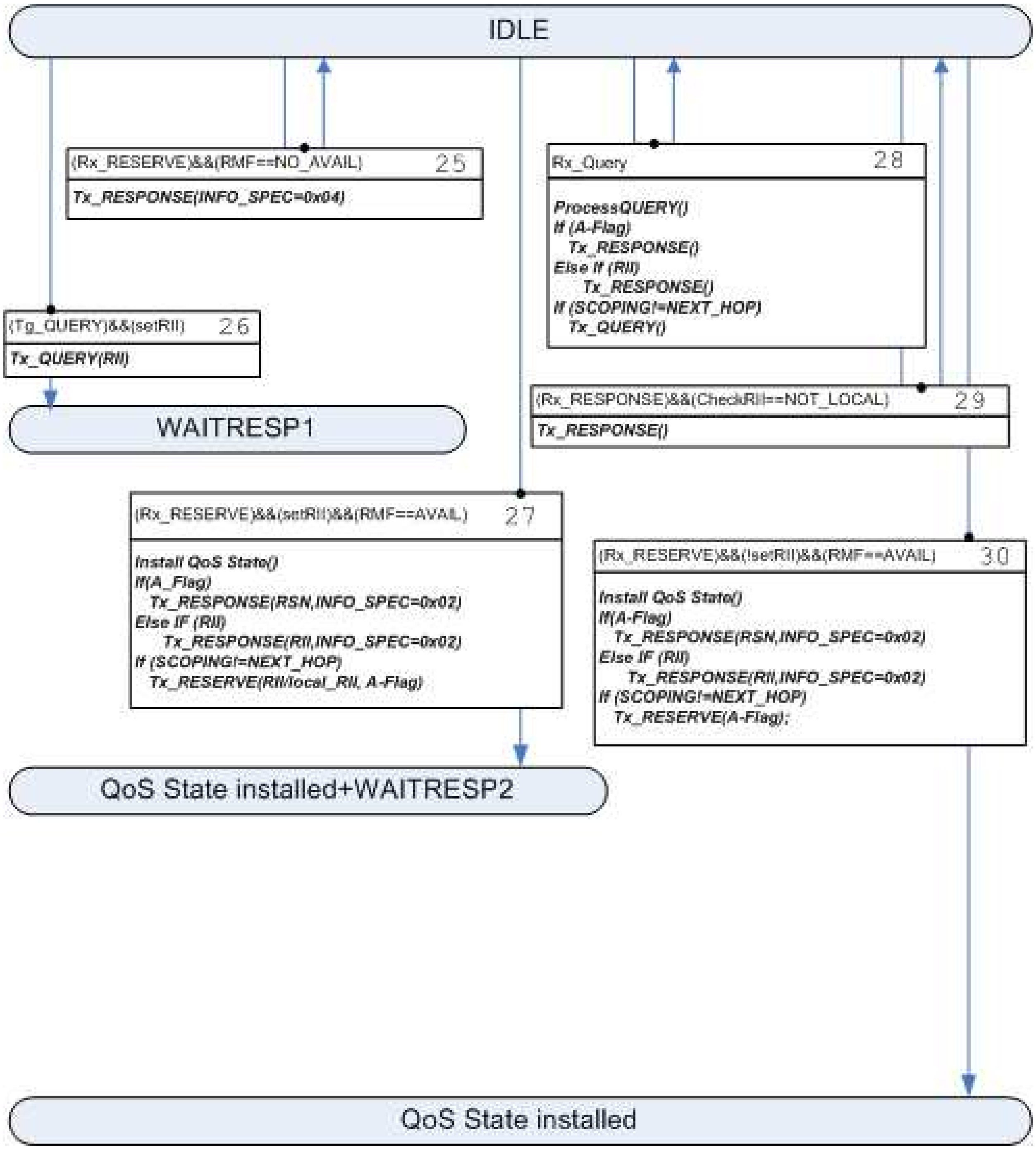

Figure 4: QNE node: "IDLE" state

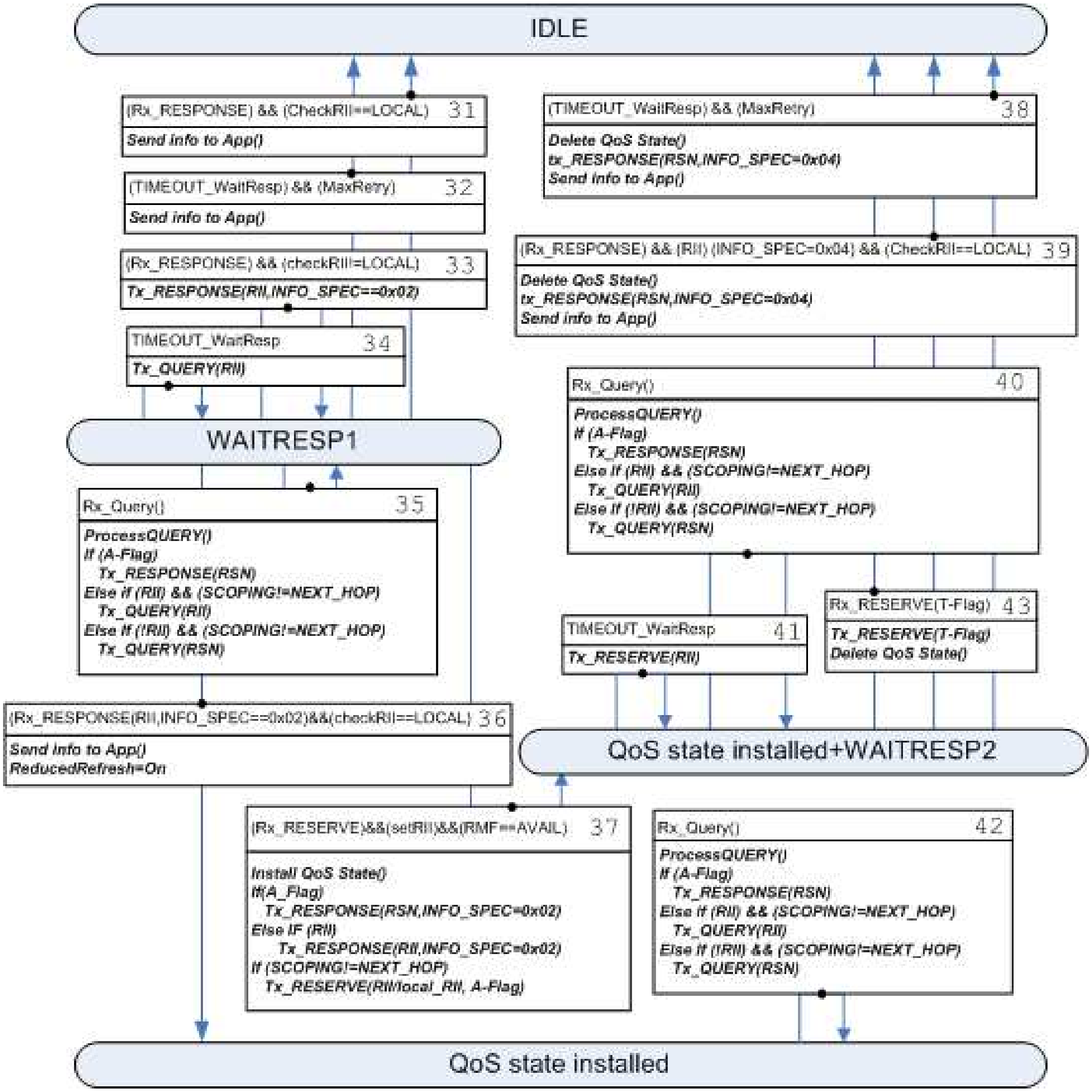

Figure 5: QNE node: "QoS state installed" state

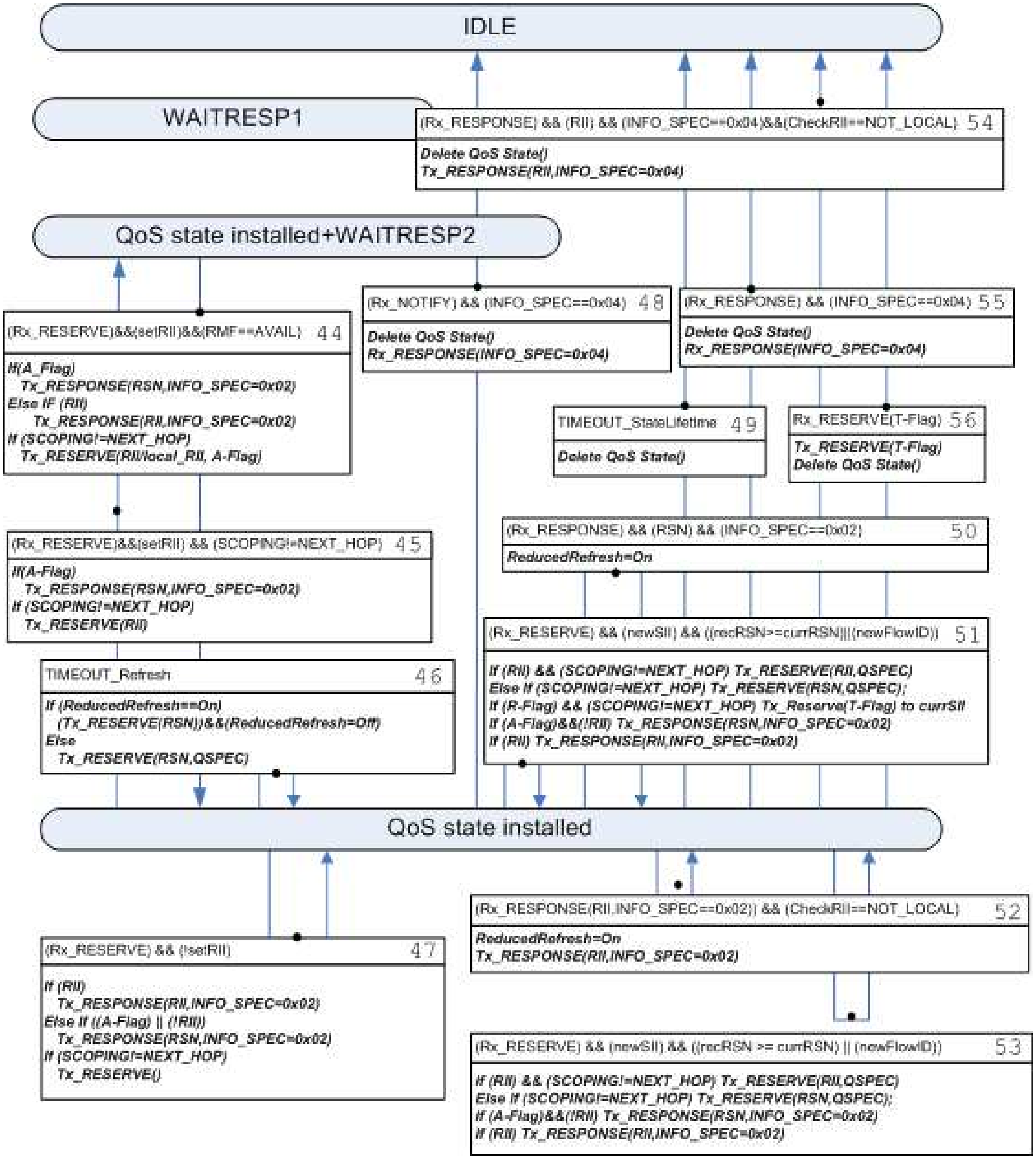

Figure 6: QNE node: "QoS state installed & WaitRESP1" and "WaitRESP2" states

6.3 State machine for QNR QoS NSLP node

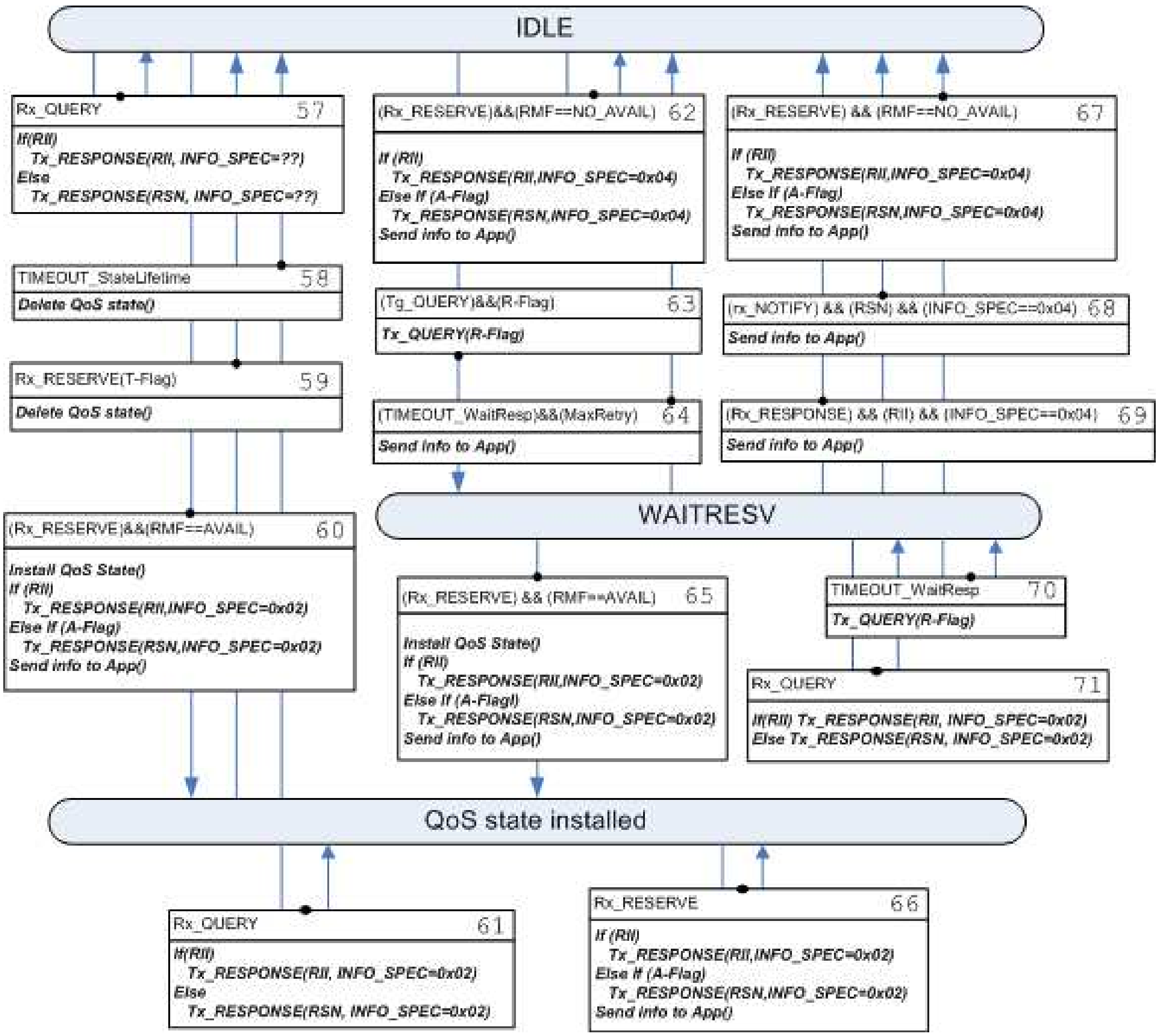

Figure 7: QNR node

7. Security Considerations

This document does not raise new security considerations. Any security concerns with QoS NSLP are likely reflected in security related NSIS work already (such as [1] or [6]).

For the time being, the state machines described in this document do not consider the security aspect of QoS NSLP protocol itself. A future versions of this document will add security relevant states and state transitions.

8. Open Issues

This document tries to describe possible states and transitions for QoS NSLP according to its current specification [1], Section 5. We found some issues during the development of the state machines.

1. Bi-directional reservation is difficult to support as the state machine becomes quite complex (note at one particular point in time the protocol state engine can be only in one state).
2. How to signal unsuccessful reservation for Receiver initiated reservation (No RII included; a resulting Response(RSN) cannot be forwarded further than the next peer). We use NOTIFY message.
3. If QoS state lifetime expires in QNI, should RESERVE(Ton) be sent downstream the path?
4. The case of unsuccessful reservation at a QNE node and no RII specified by upstream nodes. According to the spec RESPONSE(RSN) should not be forwarded further than the next peer. Currently we use NOTIFY(RSN) that is sent further to the upstream nodes.
5. We assume that handling of QoS state lifetime expiration event is based on the local policy of the node. NOTIFY/Reserve(Ton) messages might be sent to other peers.
6. The draft states that RESERVE message MUST be sent only towards the QNR. This is not the case when re-routing procedure is done and RESERVE(Ton) message should be sent from merging QNE node for deleting the old branch. We believe this is towards the QNI.
7. Re-routing functionality described in this document is not complete and need further consideration.

9. Change History

9.1 Changes in Version -01

1. Notation of the nodes changed to QNI, QNE and QNR.
2. Description of soft state refresh functionality.
3. Support of ACK flag in the common header.
4. Include of QoS NSLP objects, flags from the common header and entries stored with the installed QoS state in a node: ACK, Replace, RSN, Error_SPEC, QSPEQ, FlowID, SII.
5. Initial description of Re-routing functionality.
6. For support of all listed changes, some notations are changed.

9.2 Changes in Version -02

1. Switch to .pdf format of the draft and include graphic diagrams.
2. Update notation from "Summary refresh" to "Reduced refresh"

3. Description of QoS reservation update/upgrade

1. Review of the State Machine archtitecure

10. Acknowledgments

The authors would like to thank Sven Van den Bosch for his feedback.

11. References

11.1. Normative References

[1] Manner, J., Karagiannis, G., McDonald, A. and S. Van den Bosch "NSLP for Quality-of-Service signaling", Internet draft, draft-ietf-nsis-qos-nslp-07, July 2005.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

11.2. Informative References

[3] Vollbrecht, J., Eronen, P., Petroni, N., and Y. Ohba, "State Machines for Extensible Authentication Protocol (EAP) Peer and Authenticator", draft-ietf-eap-statemachine-06 (work in progress), December 2004.

[4] Institute of Electrical and Electronics Engineers, "DRAFT Standard for Local and Metropolitan Area Networks: Port-Based Network Access Control (Revision)", IEEE 802-1X-REV/D11, July 2004.

[5] Ohba, Y., "State Machines for Protocol for Carrying Authentication for Network Access (PANA)", draft-ohba-pana-statemachine-01 (work in progress), February 2005.

[6] Tschofenig, H. and D. Kroeselberg, "Security Threats for NSIS", draft-ietf-nsis-threats-06 (work in progress), October 2004.

Appendix A. ASCII versions of state diagrams

This appendix contains the state diagrams in ASCII format. Please use the PDF version whenever possible: it is much easier to understand.

The notation is as follows: for each state there is a separate table that lists in each row:
- an event that triggers a transition,
- actions taken as a result of the incoming event,
- and the new state at which the transitions ends.

A.1. State machine for QNI QoS NSLP node (Figures 2,3)

```
-----------
State: IDLE
-----------

Nr. Condition                Action                     State      Note
---+------------------------+--------------------------+-----------+---
 1 |(Rx_QUERY)&&(R-Flag)&&  |If(RII) tx_RESPONSE(RSN,  |IDLE       |
   |  (RMF==NO_AVAIL)       |   INFO_SPEC=0x04         |           |
   |                        |Else If (A-Flag)          |           |
   |                        |   Tx_RESPONSE(RSN,       |           |
   |                        |    INFO_SPEC=0x04        |           |
   |                        |                          |           |
 2 |(Tg_QUERY)&&(setRII)    |Tx_QUERY (RII)            |WAITRESP1  |
   |                        |                          |           |
 3 |(Tg_RESERVE) &&         |Install QoS state,        |QoS State  |
   |   (setRII) &&          |Tx_RESERVE(RII, A-Flag),  |Instaled   |
   |   (RMF==AVAIL)         |Send info to App()        |           |
   |                        |                          |           |
 4 |(Rx_QUERY) && (R-Flag)  |Install QoS State,        |QoS State  |
   |    && (setRII) &&      |If(RII) Tx_RESPONSE(RII,  |Installed +|
   |    (RMF==AVAIL)        |   INFO_SPEC=0x02)        |WAITRESP2  |
   |                        |Else If (A-Flag)          |           |
   |                        |   Tx_RESPONSE(RSN,       |           |
   |                        |    INFO_SPEC=0x02)       |           |
   |                        |Tx_RESERVE(RII,A-Flag)    |           |
   |                        |                          |           |
 5 |(Tg_RESERVE) &&         |Send info to App()        |IDLE       |
   |   (RMF==NO_AVAIL)      |                          |           |
   |                        |                          |           |
 6 |(Rx_QUERY) && (!R-Flag) |Process Query()           |IDLE       |
   |   && (RMF==AVAIL)      |If(RII) Tx_RESPONSE(RII)  |           |
   |      && (!setRII)      |Else If (A-Flag)          |           |
   |                        |   Tx_RESPONSE(RSN)       |           |
   |                        |                          |           |
 7 |(Tg_RESERVE) &&         |Install QoS State(),      |QoS state  |
```

```
   |   (!setRII) &&         |Tx_RESERVE(),           |Instaled   |
   |   (RMF==AVAIL)         |Send info to App()      |           |
   |                        |                        |           |
 8 |(Rx_QUERY) && (R-Flag)  |Install QoS State(),    |QoS state  |
   |    && (!setRII) &&     |If(RII) Tx_RESPONSE(RII,|Installed  |
   |    (RMF==AVAIL)        |   INFO_SPEC=0x02)      |           |
   |                        |Else If (A-Flag)        |           |
   |                        |   Tx_RESPONSE(RSN,     |           |
   |                        |    INFO_SPEC=0x02)     |           |
   |                        |Tx_RESERVE(RII,A-Flag)  |           |
---+------------------------+------------------------+-----------+---
```

Figure 8

```
----------------
State: WAITRESP1 + WAITRESP2
----------------


Nr. Condition                Action                   State
---+------------------------+------------------------+-----------+
 9 |(TIMEOUT_WaitResp)  &&  |Send info to Application |IDLE       |
   |   (!MaxRetry)          |                        |           |
   |                        |                        |           |
10 |(TIMEOUT_WaitResp)      |Tx_QUERY(RII)           |WAITRESP1  |
   |                        |                        |           |
   |                        |                        |           |
11 |(Rx_QUERY) && (R-Flag)  |ProcessQUERY()          |WAITRESP1  |
   |                        |If(RII) Tx_RESPONSE(RII) |           |
   |                        |Else If (A-Flag)        |           |
   |                        |   Tx_RESPONSE(RSN)     |           |
   |                        |                        |           |
12 |(TIMEOUT_WaitResp)      |Tx_QUERY(RII)           |QoS State  |
   |                        |                        |Installed +|
   |                        |                        |WAITRESP2  |
   |                        |                        |           |
13 |(Rx_RESPONSE) && (RII)  |Send info to App()      |QoS State  |
   |   && (INFO_SPEC==0x02) |If (A-Flag)             |Installed  |
   |                        |   ReducedRefresh=On    |           |
   |                        |                        |           |
14 |Rx_RESPONSE             |Send info to App()      |IDLE       |
   |                        |                        |           |
15 |(Rx_QUERY) && (R-Flag)  |ProcessQUERY()          |QoS State  |
   |                        |If(RII) Tx_RESPONSE(RII) |Installed +|
   |                        |Else If (A-Flag)        |WAITRESP2  |
   |                        |   Tx_RESPONSE(RSN)     |           |
---+------------------------+------------------------+-----------+
```

Figure 9

```
------------------
State: QoS state installed
------------------

Nr. Condition                Action                    State
---+------------------------+-------------------------+-----------+
16 |Tg_TEARDOWN             |Delete QoS State(),      |IDLE       |
   |                        | Tx_RESERVE(T-Flagon)    |           |
   |                        |                         |           |
17 |TIMEOUT_Refresh         |If (ReducedRefresh=On)   |QoS state  |
   |                        |  (Tx_RESERVE(RSN)) &&   |installed  |
   |                        |  (ReducedRefresh=Off)   |           |
   |                        |Else                     |           |
   |                        |  Tx_RESERVE(RSN,QSPEC   |           |
   |                        |  A-Flag)                |           |
   |                        |                         |           |
18 |(Tg_RESERVE) &&         |Tx_RESERVE(RII, A-Flag)  |QoS state  |
   |   (setRII) &&          |                         |Instaled  +|
   |                        |                         |WAITRESP2  |
   |                        |                         |           |
19 |Rx_NOTIFY(RSN,          |Delete QoS state         |IDLE       |
   |    INFO_SPEC=0x04)     |Send info to App()       |           |
   |                        |                         |           |
20 |(Rx_RESPONSE) && (RII)  |Delete QoS state         |IDLE       |
   |   &&(INFO_SPEC==0x04)  |Send info to App()       |           |
   |                        |                         |           |
21 |TIMEOUT_StateLifetime   |Delete QoS state         |IDLE       |
   |                        |Send info to App()       |           |
   |                        |                         |           |
22 |(Rx_QUERY) && (!R-Flag) |ProcessQUERY()           |IDLE       |
   |                        |If(RII) Tx_RESPONSE(RII) |           |
   |                        |Else If (A-Flag)         |           |
   |                        |   Tx_RESPONSE(RSN)      |           |
   |                        |                         |           |
23 |(TIMEOUT_WaitResp) &&   |Delete QoS state         |IDLE       |
   |   (MaxRetry)           |Send info to App()       |           |
   |                        |                         |           |
24 |(Tg_RESERVE) &&         |Tx_RESERVE(),            |QoS State  |
   |   (!setRII)            |Send info to App()       |Instaled   |
---+------------------------+-------------------------+-----------+
```

Figure 10

A.2. State machine for QNE QoS NSLP node (Figures 4,5,6)

```
-----------
State: IDLE
-----------
```

```
Nr. Condition                 Action                    State      Note
---+-------------------------+-------------------------+-----------+---
25 |(Rx_RESERVE) &&          |Tx_RESPONSE              |IDLE       |
   |   (RMF==NO_AVAIL)       |    (INFO_SPEC==0x04)    |           |
   |                         |                         |           |
26 |(Tg_QUERY) && (setRII)   |Tx_QUERY(RII)            |WAITRESP1  |
   |                         |                         |           |
27 |(Rx_RESERVE)&& (setRII)  |Install QoS state,       |QoS State  |
   |  &&  (RMF==AVAIL)       |If(A-Flag)               |Installed +|
   |                         |   Tx_RESPONSE(RSN,      |WAITRESP2  |
   |                         |    INFO_SPEC=0x02)      |           |
   |                         |Else If(RII)             |           |
   |                         |   Tx_RESPONSE(RII,      |           |
   |                         |    INFO_SPEC=0x02)      |           |
   |                         |If(SCOPING!=NEXT_HOP)    |           |
   |                         |   Tx_RESERVE(RII/       |           |
   |                         |    local_RII, A-Flag)   |           |
   |                         |                         |           |
28 |Rx_QUERY                 |ProcessQUERY()           |IDLE       |
   |  &&  (RMF==AVAIL)       |If(A-Flag)               |           |
   |                         |   Tx_RESPONSE(RSN,      |           |
   |                         |    INFO_SPEC=0x04)      |           |
   |                         |Else If(RII)             |           |
   |                         |   Tx_RESPONSE(RII,      |           |
   |                         |    INFO_SPEC=0x04)      |           |
   |                         |If(SCOPING!=NEXT_HOP)    |           |
   |                         |   Tx_RESERVE(RII/       |           |
   |                         |    local_RII, A-Flag)   |           |
   |                         |                         |           |
29 |(Rx_RESPONSE) &&         |Tx_RESPONSE()            |IDLE       |
   |  (CheckRII==NOT_LOCAL)  |                         |           |
   |                         |                         |           |
30 |(Rx_RESERVE)&&(!setRII)  |Install QoS state,       |QoS State  |
   |  &&  (RMF==AVAIL)       |If(A-Flag)               |Installed  |
   |                         |   Tx_RESPONSE(RSN,      |           |
   |                         |    INFO_SPEC=0x02)      |           |
   |                         |Else If(RII)             |           |
   |                         |   Tx_RESPONSE(RII,      |           |
   |                         |    INFO_SPEC=0x02)      |           |
   |                         |If(SCOPING!=NEXT_HOP)    |           |
   |                         |   Tx_RESERVE(RII/       |           |
   |                         |    local_RII, A-Flag)   |           |
   |                         |                         |           |
---+-------------------------+-------------------------+-----------+---
```

Figure 11

```
------------------
State: WAITRESP1 + QoS State Installed
```

```
------------------

31 |(Rx_RESPONSE) &&        |Send info to App()        |IDLE         |
   |  (CheckRII==LOCAL)     |                          |             |
   |                        |                          |             |
32 |(TIMEOUT_WaitResp)  &&  |Send info to Application  |IDLE         |
   |  (MaxRetry)            |   Tx_RESPONSE(RII,       |             |
   |                        |    INFO_SPEC=0x02)       |             |
   |                        |                          |             |
33 |(Rx_RESPONSE) &&        |   Tx_RESPONSE(RII,       |WAITRESP1    |
   |  (CheckRII!=LOCAL)     |    INFO_SPEC=0x04)       |             |
   |                        |                          |             |
34 |(TIMEOUT_WaitResp)      |Tx_QUERY(RII)             |WAITRESP1    |
   |                        |                          |             |
35 |Rx_QUERY                |ProcessQUERY()            |WAITRESP1    |
   |  &&  (RMF==AVAIL)      |If(A-Flag)                |             |
   |                        |   Tx_RESPONSE(RSN,       |             |
   |                        |    INFO_SPEC=0x04)       |             |
   |                        |Else If(RII)              |             |
   |                        |   Tx_RESPONSE(RII,       |             |
   |                        |    INFO_SPEC=0x04)       |             |
   |                        |If(SCOPING!=NEXT_HOP)     |             |
   |                        |   Tx_RESERVE(RII/        |             |
   |                        |    local_RII, A-Flag)    |             |
   |                        |                          |             |
36 |(Rx_RESPONSE) && (RII)  |Delete QoS state()        |QoS State    |
   |   && (INFO_SPEC==0x02) |Send info to App()        |Installed    |
   |   && (CheckRII==LOCAL) |                          |             |
   |                        |                          |             |
   |                        |                          |             |
37 |(Rx_RESPONSE) &&        |Install QoS State(),      |QoS State    |
   |  &&  (RMF==AVAIL)      |If(A-Flag)                |Installed +|
   |                        |   Tx_RESPONSE(RSN,       |WAITRESP2    |
   |                        |    INFO_SPEC=0x04)       |             |
   |                        |Else If(RII)              |             |
   |                        |   Tx_RESPONSE(RII,       |             |
   |                        |    INFO_SPEC=0x04)       |             |
   |                        |If(SCOPING!=NEXT_HOP)     |             |
   |                        |   Tx_RESERVE(RII/        |             |
   |                        |    local_RII, A-Flag)    |             |
   |                        |                          |             |
38 |(TIMEOUT_WaitResp)  &&  |Delete QoS State(),       |IDLE         |
   |  (MaxRetry)            |Tx_RESPONSE(RSN,          |             |
   |                        |   INFO_SPEC=0x04)        |             |
   |                        |                          |             |
   |                        |                          |             |
39 |(Rx_RESPONSE) && (RII)  |Delete QoS State(),       |IDLE         |
   |   && (INFO_SPEC==0x04) |Tx_RESPONSE(RSN,          |             |
```

```
   |   && (CheckRII==LOCAL) |   INFO_SPEC=0x04)        |            |
   |                        | Send info to App()       |            |
   |                        |                          |            |
40 |Rx_QUERY                |ProcessQUERY()            |QoS State   |
   |  &&  (RMF==AVAIL)      |If(A-Flag)                |Installed + |
   |                        |   Tx_RESPONSE(RSN,       |WAITRESP2   |
   |                        |    INFO_SPEC=0x04)       |            |
   |                        |Else If(RII)              |            |
   |                        |   Tx_RESPONSE(RII,       |            |
   |                        |    INFO_SPEC=0x04)       |            |
   |                        |If(SCOPING!=NEXT_HOP)     |            |
   |                        |   Tx_RESERVE(RII/        |            |
   |                        |    local_RII, A-Flag)    |            |
   |                        |                          |            |
41 |(TIMEOUT_WaitResp)      |Tx_QUERY(RII)             |QoS State   |
   |                        |                          |Installed + |
   |                        |                          |WAITRESP2   |
   |                        |                          |            |
42 |Rx_QUERY                |ProcessQUERY()            |QoS State   |
   |  &&  (RMF==AVAIL)      |If(A-Flag)                |Installed   |
   |                        |   Tx_RESPONSE(RSN,       |            |
   |                        |    INFO_SPEC=0x04)       |            |
   |                        |Else If(RII)              |            |
   |                        |   Tx_RESPONSE(RII,       |            |
   |                        |    INFO_SPEC=0x04)       |            |
   |                        |If(SCOPING!=NEXT_HOP)     |            |
   |                        |   Tx_RESERVE(RII/        |            |
   |                        |    local_RII, A-Flag)    |            |
   |                        |                          |            |
43 |Rx_RESERVE(Ton)         |Tx_RESERVE(T-Flag),       |IDLE        |
   |                        |Delete QoS state()        |            |
   |                        |                          |            |
```

Figure 12

```
----------------
State: QoS State Installed
----------------

44 |(Rx_RESERVE)&& (setRII) |If(A-Flag)                |QoS State   |
   |  &&  (RMF==AVAIL)      |   Tx_RESPONSE(RSN,       |Installed   |
   |                        |    INFO_SPEC=0x02)       |            |
   |                        |Else If(RII)              |            |
   |                        |   Tx_RESPONSE(RII,       |            |
   |                        |    INFO_SPEC=0x02)       |            |
   |                        |If(SCOPING!=NEXT_HOP)     |            |
   |                        |   Tx_RESERVE(RII/        |            |
   |                        |    local_RII, A-Flag)    |            |
   |                        |                          |            |
```

```
45 |(Rx_RESERVE)&& (setRII) |If(A-Flag)               |QoS State  |
   |  && (SCOPING!=NEXT_HOP)|   Tx_RESPONSE(RSN,      |Installed +|
   |                        |    INFO_SPEC=0x02)      |WAITRESP2  |
   |                        |Else If(RII)             |           |
   |                        |   Tx_RESPONSE(RII,      |           |
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |Tx_RESERVE(RII)          |           |
   |                        |                         |           |
46 |TIMEOUT_Refresh         |If (ReducedRefresh==On)  |QoS state  |
   |                        |   Tx_RESERVE(RSN)       |Installed  |
   |                        |   ReducedRefresh=Off    |           |
   |                        |Else                     |           |
   |                        |  Tx_RESERVE(RSN,QSPEC,  |           |
   |                        |  A-Flag)                |           |
   |                        |                         |           |
47 |(Rx_RESERVE)&&(!setRII) |If(A-Flag)               |QoS State  |
   |                        |    INFO_SPEC=0x02)      |Installed  |
   |                        |Else If(RII)             |           |
   |                        |   Tx_RESPONSE(RII,      |           |
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |If(SCOPING!=NEXT_HOP)    |           |
   |                        |   Tx_RESERVE(RII/       |           |
   |                        |    local_RII, A-Flag)   |           |
   |                        |                         |           |
48 |(Rx_NOTIFY) &&          |Delete QoS state()       |IDLE       |
   |   (INFO_SPEC==0x04)    |Rx_RESPONSE(RSN,         |           |
   |                        |   INFO_SPEC==0x04)      |           |
   |                        |                         |           |
49 |TIMEOUT_StateLifetime   |Delete QoS state         |IDLE       |
   |                        |                         |           |
50 |(Rx_RESPONSE) && (RSN)  |ReducedRefresh=On        |QoS State  |
   |                        |                         |Installed  |
   |                        |                         |           |
51 |(Rx_RESERVE)&&(newSII)  |If (RII) && (SCOPING!=   |QoS State  |
   | &&(RMF==AVAIL)         |   NEXT_HOP)             |Installed  |
   | &&((recRSN>=currRSN)   |   Tx_RESERVE(RII,QSPEC) |           |
   |    ||(newFlowID))      |Else If (SCOPING!=       |           |
   |                        |   NEXT_HOP) Tx_RESERVE  |           |
   |                        |   (RSN,QSPEC)           |           |
   |                        |If (R-Flag==On) &&       |           |
   |                        |   (SCOPING!=NEXT_HOP    |           |
   |                        |   Tx_Reserve(T-Flag)    |           |
   |                        |   to currSII            |           |
   |                        |If (A-Flag==On)&&(!RII)  |           |
   |                        |   Tx_RESPONSE(RSN,      |           |
   |                        |   INFO_SPEC=0x02)       |           |
   |                        |                         |           |
52 |(Rx_RESPONSE) && (RII)  |ReducedRefresh=On        |QoS State  |
```

```
   |      E_SPEC=0x02))      |Tx_RESPONSE(RII,          |Installed   |
   | &&(CheckRII==NOT_LOCAL)|      E_SPEC==0x02)       |            |
   |                         |                          |            |
53 |(Rx_RESERVE)&&(newSII)   |If (RII) && (SCOPING!=    |QoS State   |
   | &&(RMF==AVAIL)          |    NEXT_HOP)             |Installed   |
   | &&((recRSN>=currRSN)    |    Tx_RESERVE(RII,QSPEC)|            |
   |    ||(newFlowID))       |Else If (SCOPING !=       |            |
   |                         |   NEXT_HOP)              |            |
   |                         |   Tx_RESERVE(RSN,QSPEC);|            |
   |                         |If (A-Flag) && (!RII)     |            |
   |                         |    Tx_RESPONSE(RSN,      |            |
   |                         |    INFO_SPEC=0x02)       |            |
   |                         |If (R-Flag==On)           |            |
   |                         |    Tx_Reserve(T-Flag)    |            |
   |                         |    to currSII            |            |
   |                         |                          |            |
54 |(Rx_RESPONSE) && (RII)   |Delete QoS State()        |            |
   |      E_SPEC==0x02))     |Tx_RESPONSE(RII,          |IDLE        |
   | &&(CheckRII==NOT_LOCAL)|      E_SPEC=0x04)        |            |
   |                         |                          |            |
55 |(Rx_RESPONSE) &&         |Delete QoS State()        |IDLE        |
   |   (INFO_SPEC==0x04)     |Rx_RESPONSE(RSN,          |            |
   |                         |   INFO_SPEC==0x04)       |            |
   |                         |                          |            |
56 |Rx_RESERVE(T-Flag)       |Delete QoS State()        |IDLE        |
   |                         |Tx_RESERVE(T-Flag)        |            |
   |                         |                          |            |
```

Figure 13

A.3. State machine for QNR QoS NSLP node (Figure 7)

```
-----------
State: IDLE + WAITRESV + QoS state installed
-----------

57 |Rx_QUERY                 |If(RII) Tx_RESPONSE(RII,  |IDLE        |
   |                         |    INFO_SPEC==0x02)      |            |
   |                         |Else    Tx_RESPONSE(RSN,  |            |
   |                         |    INFO_SPEC==0x02)      |            |
   |                         |                          |            |
58 |TIMEOUT_StateLifetime    |Delete QoS State()        |IDLE        |
   |                         |                          |            |
59 |Rx_RESERVE(T-Flag)       |Delete QoS State()        |IDLE        |
   |                         |                          |            |
60 |(Rx_RESERVE)             |Install QoS State(),      |QoS State   |
   |  &&  (RMF==AVAIL")      |If(RII)                   |Installed   |
   |                         |   Tx_RESPONSE(RII,       |            |
```

```
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |Else If(A-Flag)          |           |
   |                        |   Tx_RESPONSE(RSN,      |           |
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |Send info to App()       |           |
   |                        |                         |           |
61 |Rx_QUERY                |If(RII) Tx_RESPONSE(RII, |QoS State  |
   |                        |    INFO_SPEC==0x02)     |Installed  |
   |                        |Else    Tx_RESPONSE(RSN, |           |
   |                        |    INFO_SPEC==0x02)     |           |
   |                        |                         |           |
62 |(Rx_RESERVE) &&         |If(RII)                  |IDLE       |
   |  &&  (RMF==NO_AVAIL)   |   Tx_RESPONSE(RII,      |           |
   |                        |    INFO_SPEC=0x04)      |           |
   |                        |Else If(A-Flag)          |           |
   |                        |   Tx_RESPONSE(RSN,      |           |
   |                        |    INFO_SPEC=0x04)      |           |
   |                        |Send info to App()       |           |
   |                        |                         |           |
63 |(Tg_QUERY) && (R-Flag)  |Tx_QUERY(R-Flag)         |WAITRESV   |
   |                        |                         |           |
64 |(TIMEOUT_WaitResp)  &&  |Send info to App()       |IDLE       |
   |    (MaxRetry)          |                         |           |
   |                        |                         |           |
65 |(Rx_RESERVE) &&         |Install QoS State()      |QoS State  |
   |  && (RMF==AVAIL)       |If(RII)                  |Installed  |
   |                        |   Tx_RESPONSE(RII,      |           |
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |Else If(A-Flag)          |Installed  |
   |                        |   Tx_RESPONSE(RSN,      |           |
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |Send info to App()       |           |
   |                        |                         |           |
66 |(Rx_RESERVE)            |If(RII)                  |QoS State  |
   |                        |   Tx_RESPONSE(RII,      |Installed  |
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |Else If(A-Flag)          |           |
   |                        |   Tx_RESPONSE(RSN,      |           |
   |                        |    INFO_SPEC=0x02)      |           |
   |                        |Send info to App()       |           |
   |                        |                         |           |
67 |(Rx_RESERVE) &&         |Install QoS State()      |IDLE       |
   |   && (RMF==NO_AVAIL)   |If(RII)                  |           |
   |                        |   Tx_RESPONSE(RII,      |           |
   |                        |    INFO_SPEC=0x04)      |           |
   |                        |Else If(A-Flag)          |           |
   |                        |   Tx_RESPONSE(RSN,      |           |
   |                        |    INFO_SPEC=0x04)      |           |
```

```
   |                        |Send info to App()        |           |
   |                        |                          |           |
   |                        |                          |           |
   |                        |                          |           |
68 |(Rx_NOTIFY) &&          |Send info to App()        |IDLE       |
   |   (INFO_SPEC==0x04)    |                          |           |
   |                        |                          |           |
69 |(Rx_RESPONSE) && (RII)  |Send info to App()        |IDLE       |
   |   && (INFO_SPEC==0x04) |                          |           |
   |                        |                          |           |
70 |(TIMEOUT_WaitResp)      |Tx_QUERY(R-Flag)          |WAITRESV   |
   |                        |                          |           |
71 |Rx_QUERY                |If(RII) Tx_RESPONSE(RII,  |WAITRESV   |
   |                        |    INFO_SPEC==0x02)      |           |
   |                        |Else    Tx_RESPONSE(RSN,  |           |
   |                        |    INFO_SPEC==0x02)      |           |
---+------------------------+--------------------------+-----------+---
```

Figure 14

Authors' Addresses

Xiaoming Fu
University of Goettingen
Telematics Group
Lotzestr. 16-18
Goettingen 37083
Germany

Email: fu@cs.uni-goettingen.de

Hannes Tschofenig
Siemens
Otto-Hahn-Ring 6
Munich, Bayern 81739
Germany

Email: Hannes.Tschofenig@siemens.com

Tseno Tsenov
Sofia,Bulgaria

Email: tseno.tsenov@mytum.de

Bernd Schloer
University of Goettingen
Telematics Group
Lotzestr. 16-18
Goettingen 37083
Germany

Email: bschloer@cs.uni-goettingen.de

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specifi cation can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.