ANNEX L HIGH-FREQUENCY WIRELESS-TOKEN-RING-PROTOCOL (WTRP) REQUIREMENTS

L.1. INTRODUCTION

This Annex specifies a High-Frequency Wireless Token Ring Protocol (WTRP) for enhanced media access control within single-frequency multi-node radio networks using STANAG 5066. The protocol is in two parts: a message design for the management tokens exchanged by nodes in the radio network, and the algorithms used to create, maintain, and repair the radio-transmission sequence (i.e., the virtual ring) of nodes in the network.

There is a token that represents right to transmit, which is transferred using a STANAG 5066 EOW (Engineering Order Wire) message. This means that the token can be transmitted without any protocol overhead and can be repeated for resilience. Additional messages to manage the token ring are sent using STANAG 5066 EXTENSION D_DPUs as specified in Annex C.

The High Frequency Wireless Token Ring Protocol (WTRP) is a self-organizing Medium Access Control (MAC) protocol for HF wireless networks. The MAC protocol by which mobile stations can share a broadcast channel is crucial in wireless networks, especially for HF wireless networks where bandwidth is considerably lower than other types of wireless networks. WTRP is a MAC protocol tailored for low-speed wireless networks. WTRP ensures that each node gets to transmit at least once per ring traversal, preventing the lock-out that can occur with CSMA. WTRP is efficient in reducing the number of retransmissions due to collisions.

All stations are connected on a single channel with common waveform operating on the same frequency. For waveforms with variable bandwidth a common bandwidth must be used by all stations. The frequency and bandwidth may be fixed or may be negotiated by ALE for all stations, as set out in Annex J. For autobaud waveforms, transmission speed and interleaver may vary during operation or fixed values may be configured.

WTRP requires that stations in a ring take turns to transmit for a specified maximum amount of time, with an order of transmission that includes each node at least once. WTRP is robust against single node failure. WTRP is different from its parent protocol in that it provides the notion of self-rings, that it supports connected networks with arbitrary connectivity, that there is no ring owner, and that stations can transmit more than once per ring cycle.

This Annex of STANAG 5066 is organized as follows.

- Section L.2 gives an overview of WTRP;

- Section L.3 specifies the STANAG 5066 message design for WTRP management messages;
- Section L.4 gives the complete WTRP state diagram and description for each state in WTRP;
- Section L.5 specifies parameters and timer selection for ring-management and operation;
- Section L.6 specifies requirements for token and message transmission.
- Section L7 summarizes changes since Ed3.

L.1.1. Changes in This Edition

The changes since Edition 3 are set out in Section L7 and repeated here for convenience.

The overall model and service provided by Annex L is unchanged. The state machine has some small changes. The protocol in Edition 4 is completely different to Edition 3 and interoperability is not a goal. This change was made because of significant problems with the protocol in Edition 3.

L.2. OVERVIEW: WIRELESS TOKEN RING PROTOCOL

Token-Ring definitions and management concepts are introduced below prior to detailed specification of the protocol in later sections. The original design of WTRP was based on work by Mustafa Ergen, Duke Lee et. al., “Wireless Token Ring Protocol”, University of California, Berkeley, CA 94720, USA.

L.2.1. Definitions

The following terms are used in the specification of the High-Frequency Wireless Token Ring Protocol (WTRP).

L.2.1.1. Stations and Nodes

The terms “station” and “node” are used interchangeably to describe the communication entities on the shared transmission medium.

L.2.1.2. Token, Messages, Rings and Ring Members

The WTRP protocol is a Medium Access Control (MAC) protocol. The task of this protocol is to schedule the access of two or more stations connected to the same physical medium, nominally an HF wireless network. Messages exchanged between stations for WTRP control are called *messages*.

The WTRP protocol organizes stations in such a manner that they rotate a *right-to-transmit Token (*or just *token)* among stations connected to the same physical medium. Only a *station* that receives the *right-to-transmit token* has the right to transmit user data. This *station* is then the *token-holder*. In normal operation there should only be one *token-holder*.

A set of stations sharing the same *right-to-transmit token* is defined as a *ring*, or *virtual ring.* A station participating in a ring is called a *ring member*.

When a station starts initially it is not a member of any ring. It will only be able to become a *ring member* if there is at least already one station connected to the same physical medium. If the station connected to the medium finds out there is no existing ring, it will attempt to set up a new ring with itself as its only member. Such a ring is called a *self ring*. If the station trying to establish a ring finds another station willing to join the ring, the *self ring* becomes a *ring*.

The *right-to-transmit* will be passed in rotation among the ring members, with a cycle that includes every ring member. A complete transit of this ring is referred to as a *ring cycle*.

L.2.1.3. Successors and Predecessors

If station S_A is passing the right to transmit to station S_B, then station S_B shall be called the *successor* of station S_A. The *predecessor* of station S_B shall be station S_A.

In other words, the *successor* of station X is the station to which X sends the *right-to-transmit token* to and the *predecessor* is the station from which X received the *right-to-transmit token*. Note that these stations are member of the same ring.

L.2.1.4. Ring Transmit Order

The order in which the Right-To-Transmit (RTT) token is passed around in the virtual ring is called the *transmit order*, which also defines the *successor* and *predecessor* relationship among *ring members*. For example, if the RTT token is passed from station S_A to station S_B, from station S_B to station S_C, and back to station S_A, then the transmit order is $\{S_A > S_B > S_C > S_A\}$. In a ring whose transmit order equals $\{S_A > S_B > S_C > S_A\}$, station S_B is the *successor* of station S_A, station S_C is the *successor* of station S_B, and station S_A is the *successor* of station S_C.

In a stable WTRP network, the *transmit order* will not vary and will include every *ring member*. In order to accommodate complex ring topologies, a node may appear more than once in the *transmit order*.

In a changing WTRP network, the node holding the token will choose its *successor* to optimize transmission, leading to a modified *transmit order*.

L.2.1.5. **Node States**

The following WTRP states are defined in the protocol:

- Floating State (FLT) - the initial (starting) state where a station is not part of a ring looks for an existing ring that it can join.
- Self Ring State (SFR) - in this state a station assumes there is **no** existing ring to join, and will therefore try to setup a new ring. The new ring will start with this station as the only member and is therefore called a self-ring.
- Seeking State (SEK) - in this state a station considers itself to be in a self-ring and has broadcast an INVITE message as an invitation for other stations to join its ring.
- Solicit Reply State (SRP) - in this state a station tries to join another ring; it intends to respond to a received INVITE message but is waiting for a timeout to avoid congestion before sending its reply.
- Joining State (JON) - In this state a station has replied to the invitation to join (i.e., to the received INVITE message) by sending a JOIN message to the node inviting it to join the net.
- Have-Token State (HVT) - in this state a station is part of a ring. It has received a RTT (right-to-transmit) token from its predecessor and with this token, the right to transmit. From this state a node may optionally transition to SLT in order to invite other nodes, after which it will return to this state. Then it will transmit token, messages and user data and transition to MON.
- Monitoring State (MON) - in this state a station is part of a ring. It passed the RTT token to its successor, but is unsure if the successor has received the right to transmit (i.e., the station is monitoring the channel for an implicit acknowledgement the RTT was successfully passed);
- Idle State (IDL) - In this state a station is part of a ring and knows it has successfully passed the RTT token its successor; it will process any D_PDU messages received from other stations;

- Soliciting State (SLT) - in this state a station is part of a ring, currently has the right to transmit, and is inviting new stations to join the ring by broadcasting an INVITE message and listening for replies;

L.2.1.6. **Primary Timers**

Most states have one or more associated timers. The primary Timers defined within the WTRP state machine are:

- Claim Token Timer (TCLT) - Timer used in the *floating-state* (FLT). Controls the time a station waits while in the *floating state* to claim a token before transiting to another state; a station restarts its TCLT timer when it transits to the FLT state.

- Solicit Successor Timer (TSLS) - This timer is used in the *self-ring-state* (SFR). If it times out the station shall transit to the *seeking-state* (SEK). The timer is specified with a random timeout to reduce the probability of collisions by transmissions from stations attempting to establish different rings at the same time.

- Solicit Reply Timer (TSRP) - Timer used in the *solicit-reply-state* (SRP). A station starts the *solicit-reply timer* when it transits to the SRP state. When it expires the station will send a JOIN message and transition to the JON state.

- Contention Timer (TCON) - Timer used in the *joining-state* (JON). It controls the time a station waits for a response from another station following an attempt to join the network, so-named because failure to receive a response is attributed to contention with other stations attempting to join the network at the same time; a station starts its *contention timer* when it goes to the JON state.

- Idle Timer (TIDL) - Timer used in the *idle-state* (IDL). It controls the time a station waits for its *right-to-transmit* before transiting to the *floating-state* (FLT).

- Solicit Wait Timer (TSLW) - This timer is used in the *soliciting state* (SLT) and the *seeking-state* (SEK) by stations inviting new ring members. When it expires, the inviting station will update the Transmit Order List to include all nodes that have sent JOIN messages.

- Token Pass Timer (TPST) - Used in the *monitoring-state* (MON). It controls the time a station waits after passing an RTT (or other) token to another *station* and failing to hear an implicit acknowledgement before considering the *right-to-transmit* as lost.

L.2.1.7. **Token Contents**

The right-to-transmit-token can be considered as a simple flag that controls the right to transmit. The token is transmitted along with information on the transmitting node's view of current ring size. This enables a joining node to better estimate ring latency and validates that ring members have a consistent view of ring size.

L.2.1.8. **Promiscuous reception**

Promiscuous reception is a receive mode in which a station performs limited processing and information collection on all D_PDUs it receives, whether or not they are addressed to it (i.e., the station takes in all traffic).

Promiscuous reception is the means by which a station discovers and confirms the existence or loss of links in the HF radio network (or that may be used to construct a network), compiles local node adjacency information, which is used to determine the choice of *successor*. This is core to the operation of WTRP.

L.2.1.9. **Receive Table, Connectivity Table, and Next Hop Table**

Each node maintains a *receive table* that records information on all other ring members where transmissions have been recently received. Each ring member will transmit regularly as the token passes around the ring, so it is certain that there will be transmissions from each node in a working ring. For each node from which a transmission has been recently received, the receive table will contain an entry indicating that this node can be heard and the quality of the reception. Where no transmission has been received from a node in the ring, the receive table will implicitly record that no data is being heard by not including the node in the receive table. While receive tables could be updated each ring cycle, it is generally desirable to make calculations based over a number of ring cycles (and reception from each other node) to avoid undue fluctuation of *receive table* values.

In addition, for each node where data can be heard, the *receive table* **shall** record a *transmit speed*. This speed reflects the maximum recommended speed for bulk throughput for data transmitted to the node. The model and encoding are aligned to the Data Rate Selection EOW encodings specified in STANAG 5066 Annex C. This speed will be calculated from the SNR and Frame Error Rate of data received from the node, using information obtained from the modem through the MAC layer.

Each node **shall** send its *receive table* to all other nodes using the TABLE PDU, by transmission around the ring whenever the receive table changes. This means that every node will have the *receive table* for all nodes, which provides the node with information on transmission of data to the node from every other node.

The *receive table* data from each node enables the node to build up an internal *connectivity table* for all nodes. For each node the *connectivity table* will record for each peer of that node:

1. The connectivity status of the node for each peer. One of:
    a. No direct connectivity
    b. Bidirectional transmission
    c. Transmission only from the node to the peer node.
    d. Transmission only from the peer node to the node.
2. Transmit speed from the node to peer node
3. Transmit speed from the peer node to the node

Unidirectional transmission can be useful for non-ARQ data. For ARQ data and token transfer, it is essential that transmission is bidirectional. Because of this, primary connectivity calculations in this specification are based on bidirectional transmission, which is safe for all types of transfer.

The *next hop table* is then built using information from the *connectivity table.* The next hop table is a key information that is passed upwards from WTRP, to enable higher layers to correctly route data.

The *next hop table* contains the following information for every node in the ring:

1. A choice of one of the following two connectivity options:
    a. The node can be reached directly for all transmissions; or
    b. The node can be reached directly for non-ARQ transmission only;
2. For nodes other than 1a, the preferred *data relay node* to be used for other communication.
3. Maximum transmission speed to be used for transmission to the node.
    a. Speed for direct transmission
    b. Speed for transmission to the *data relay node* (for nodes other than 1a)

Nodes with status 1a or 1b can determined directly from the *connectivity table*, and which also supplies the direct transmission speed. For nodes that cannot be reached directly, the *adjacency matrix* can be used to determine the best *data relay node*. For each directly connected node, the adjacency matrix can be used to determine paths to nodes directly connected to these nodes. This can be done iteratively to determine the distance to each node, and a set of paths to each node.

Where several directly connected nodes can act as the *data relay node*, one of them needs to be chosen for the *next hop table*. The usual choice of the preferred *data* *relay node* is the one that is expected to receive the token next, which will usually minimize transfer time. The transmission speeds of each link **may** be considered, and it may be preferable to choose a *data relay node* that leads to faster transmissions on each hop. Note that the next hop table must be recalculated whenever *receive table* data changes.

The *next hop table* is used to communicate to the DTS and CAS layers of STANAG 5066, which nodes can be accessed directly, and which nodes need to be relayed.

To send data to nodes that are not directly connected, STANAG 5066 needs to send data to the *data relay node* for that node. The *data relay node* then needs to relay the data onwards to its final destination or to the next relay hop. The protocol to do this is specified in STANAG 5066 Annex R (Routing Sublayer).

L.2.1.10. **Tour**

A *tour* of a graph is a sequence of nodes from the graph such that each node appears at least once and two nodes are adjacent in the sequence only if they are adjacent in the *connectivity table*. An unconstrained, ordinary tour allows revisits to network nodes.

In the context of the wireless token-ring protocol, the sequence of ownership of the right-to-transmit (RTT) token, should be a *closed tour* of the network that starts and ends with nodes that are adjacent in the network, i.e., every node gets an opportunity to transmit, and passing the RTT token along the closed tour is feasible because nodes that are adjacent in the transmission sequence are adjacent in the network. A ring where the transmission order is stable is always a *closed tour* and may be an *unconstrained closed tour*.

L.2.1.11. **Ring-Cycle Length (RCL)**

The Ring-Cycle Length (RCL) is the length in hops of the tour through the WTRP network
taken by the RTT token as it completes a *closed tour*.

L.2.2. **Concept of Basic Ring Operation**

The basic concept of the Wireless Token Ring Protocol is to provide a mechanism that allows
two or more stations operating on the same single-frequency radio channel to share it in such a way that only one station will transmit at a time. Only the station that has the *right-to-transmit* (*token-holder*) is allowed to transmit on the shared channel. The *right-to-transmit* is passed onward to all stations that joined the ring.

Figure L-1 shows how the *right-to-transmit* is circulated among the *ring-members*.

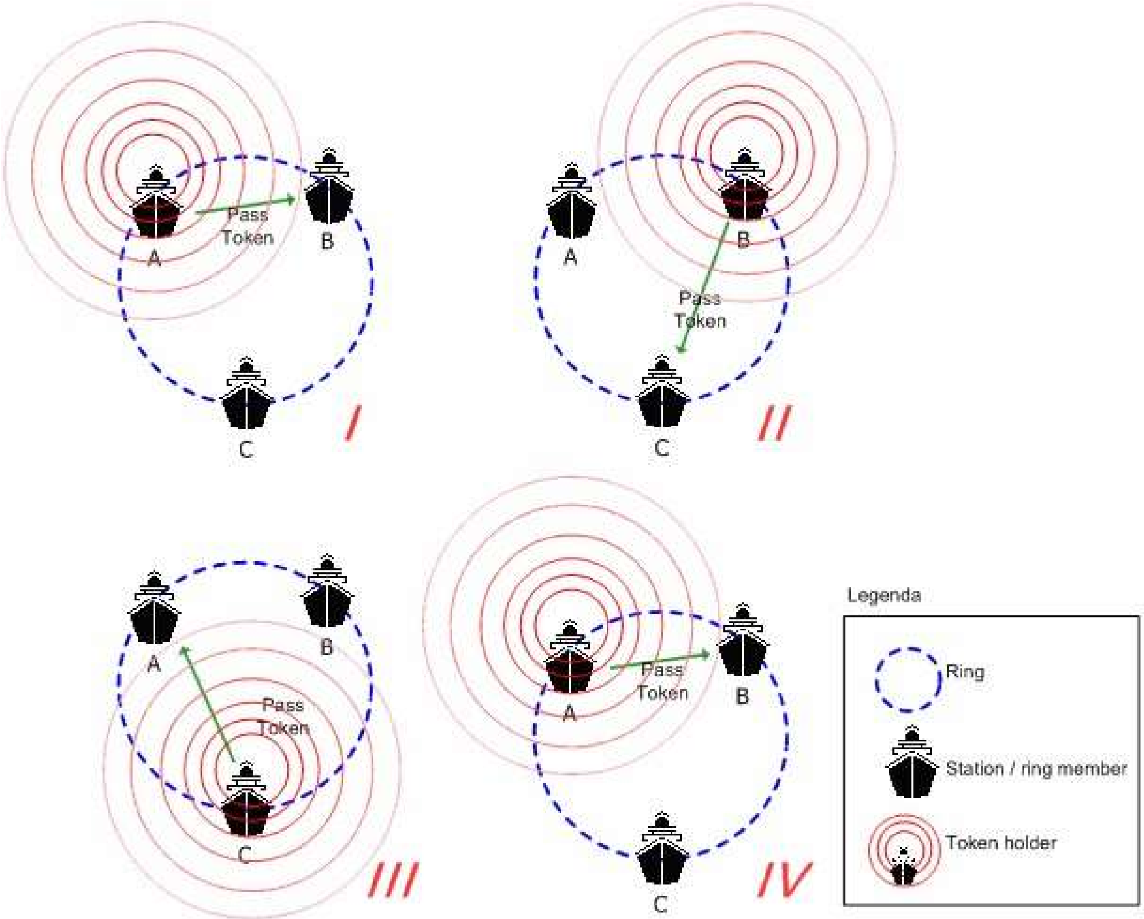

Figure L-1 - Normal Token-Ring Operation

The ring is a closed cycle of stations that transmit each in turn in a prescribed sequence, which will be adapted if new stations join the network or changes in network connectivity force adaptation. When a station receives the *right-to-transmit* from its *predecessor* it will take control of the channel. In addition to the *right-to-transmit token* the station may also receive additional WTRP messages, in particular updated *receive table* messages from one or more nodes. These messages will not be transmitted when the information is stable. *Receive table* messages are passed around the ring to ensure that all nodes have the latest information. A node will send its own *receive table* message if this information has changed. The *right-to-transmit token* is encoded in a standard STANAG 5066 EOW message and will generally be included multiple times in a transmission to minimize risk of token loss. If the station has no data to transmit, then it **shall** pass the *right-to-transmit* and associated messages immediately.

The messages sent by a station are transferred in a set of one or more D_PDUs. The D_PDU **shall** count down the *end of transmit time* (EOT) in accordance with the requirements of Annex C. A station **shall not** exceed the maximum transmission time allowed.

Note that though the transmission sequence and ownership of the *token* in the ring is prescribed, a station with the *right-to-transmit* **may** send data to any other station in the network that it determines from the *next hop table* can receive data and transmit directly back acknowledgements for ARQ data, not just its successor or predecessor in the virtual ring. Any station in the virtual ring may therefore engage in multiple concurrent traffic exchanges with any other station in range. This includes the capability to support concurrent soft-link, physical-link, and ARQ connections in accordance STANAG 5066 Annex A, B, and C, with retransmission timers and other timing parameters selected to allow for the network size.

Possession of the *right-to-transmit* controls access to the radio channel; it does not prescribe the destination(s) to which traffic can be sent. The token-ring protocol controls node access to the transmission medium to preclude collisions and self-interference in the network and thereby increase the efficiency and throughput in the network. To do so, in general, there is no requirement that all nodes in the network be in communications range, only that the network have a closed tour, i.e., that a cyclic sequence of nodes exist where every node is in communications range of its predecessor and successor in the sequence. To support WTRP networks where direct communication is not possible between all nodes, the WTRP layer communicates connectivity to the higher layers of STANAG 5066. The higher layers will only send data to nodes that can be reached directly. Relay to other nodes is provided by STANAG 5066 Annex R (Routing Sublayer).

Each station will retransmit the *token,* passing the *right-to-transmit* to its *successor* until this has been acknowledged by its *successor.* This acknowledgement is implicit, by observing when the successor transmits data.

L.2.3. **Ring formation**

Before operating as a ring, a ring must be formed by stations on the same radio channel. The ring is a self-organizing and self-repairing mechanism, subject to a number of requirements:

- As soon as a station starts to operate it will listen to the radio channel in a floating state, unaffiliated with any ring.
- A station may transmit data (i.e., D_PDU other than ring-management messages) only if it is part of a ring that is not a self ring.

- A station shall first listen for an existing ring on the radio-channel; if it hears a ring it will try to join that ring, otherwise it will form a new ring (a self ring).

- If a station receives the right-to-transmit from another node, it has become a part of the ring.

- A station dropped from a ring must re-join the ring to become again part of it and obtain transmission rights.

There are two possible ways to form a ring:

1. Join an existing one.

2. Start a new ring yourself.

The process of Ring Formation is illustrated below in Figure L-2 - Ring Formation.

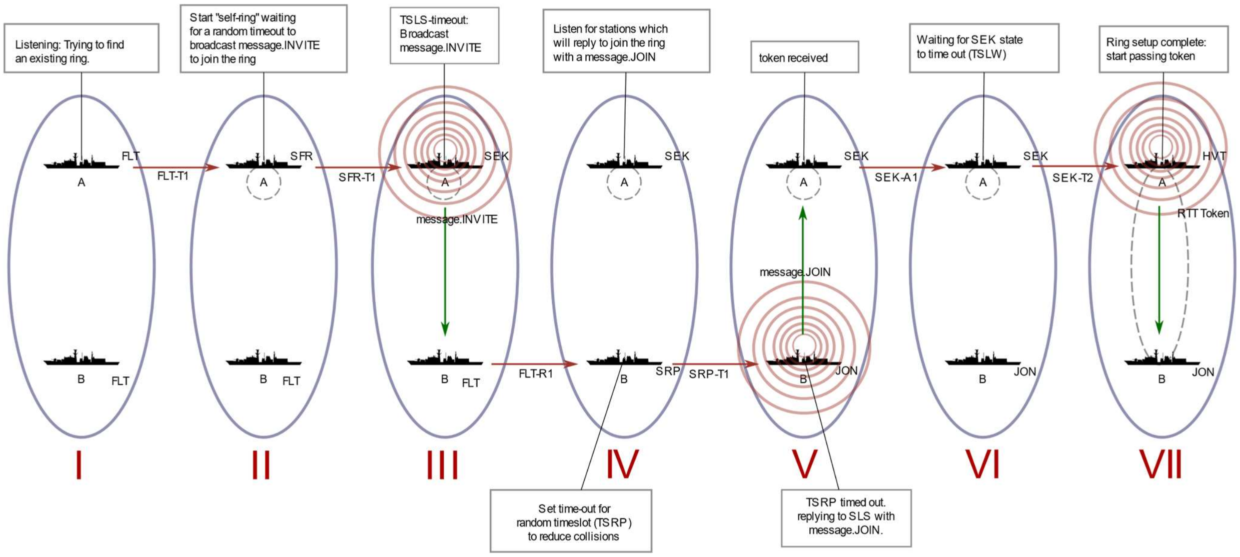
Listening: Trying to find an existing ring.
Start "self-ring" waiting for a random timeout to broadcast message.INVITE to join the ring
TSLS-timeout: Broadcast message.INVITE
Listen for stations which will reply to join the ring with a message.JOIN
token received
Waiting for SEK state to time out (TSLW)
Ring setup complete: start passing token
FLT
A
FLT-T1
SFR
A
SFR-T1
SEK
A
message.INVITE
SEK
A
SEK
A
message.JOIN
SEK-A1
SEK
A
SEK-T2
HVT
A
RTT Token
B
FLT
B
FLT
B
FLT
FLT-R1
B
SRP
SRP-T1
B
JON
B
JON
B
JON
I
II
III
IV
V
VI
VII
Set time-out for random timeslot (TSRP) to reduce collisions
TSRP timed out. replying to SLS with message.JOIN.

Figure L-2 - Ring Formation

Figure L-2 shows a time-sequence evolution of two stations, station A and station B, as they form a two-node ring.

In situation ***I*** station A is listening for an existing ring. After its TCLT timer times out, station A assumes there is no existing ring (otherwise it would have heard it) and it transits from a FLT state to the SFR state, forming a *self ring* as illustrated in situation ***II***.

The transition moment from the SFR state in situation ***II*** to the SEK state in situation ***III*** is controlled by a random timeout that avoids collision between two stations in the same state. The value of the timeout is calculated based on picking an asynchronous timeslot (See TSLS timer details in section L.5).

In the SEK state of situation ***III*** a station invites new members by sending an INVITE message. Each station that is seeking to join will reply with a JOIN (join ring) message. There could be more than one station waiting to reply to an INVITE message. Therefore, to reduce the probability of collision between replies from multiple joining nodes, the joining node's reply shall be transmitted in a randomly chosen time slot among a set of slots available for replies. All nodes in the ring and each node that sends a JOIN **shall** listen for JOIN messages so that nodes other than the one sending the INVITE can add nodes to the ring. The station that sent the INVITE message will add one of the nodes requesting to join to the Ring by passing the token to it. Other nodes that responded to the invite **may** be added by other nodes in the ring or by the node that sent the invite when the token returns to it.

Once the ring is created, each node periodically creates an opportunity to invite new stations to join the ring by broadcasting an INVITE message and waiting for JOIN message replies from nodes wishing to join the ring.

L.2.4. **Ring Entry**

The invitation to join shall be made periodically by a station receiving the right-to-transmit, with invitation frequency controlled by the algorithm specified in Section L.4.9. The opportunity to invite new members needs to be done by all ring members, as new nodes may only be able to connect to one existing member. A configurable maximum ring size **may** be specified, which can be used to stop rings growing too large. This maximum ring size **may** also be used when the number of nodes is known, to prevent taking time to invite new members when there are no possible new members. Invitations can also be configured so that they are less frequent (lower overhead) for stable rings.

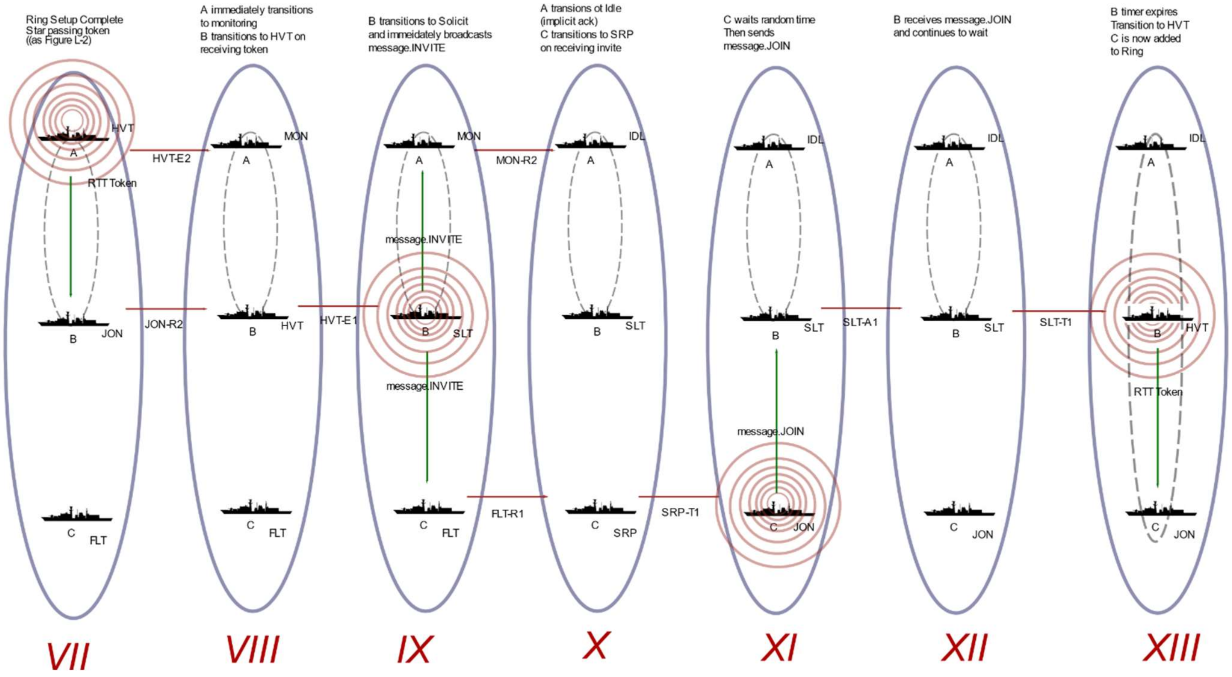

Figure L-3 - Ring Entry

Figure L-3 illustrates adding a station to an existing ring and continues the example of Figure L-2, showing addition of station C to the ring.

In VII, station B has received the token as in HVT state. Station B determines that it needs to issue an invite, so immediately transitions to SLT (IX) and immediately broadcasts a message.INVITE. Station A receives the message.INVITE, which it uses as implicit acknowledgement of token transfer, so it transitions from MON to IDL (X). Node C in FLT plans to accept the invitation and so transitions to SRP (X).

NOTE: If there are multiple responses to the initial Invite, Station A **shall** respond to one of them. Other nodes **may** then respond to other responses.

Station C waits for a random time before responding with message.JOIN (XI). Station B receives the message.JOIN (XII) and continues to wait for potential other message.JOIN from other stations. After a timer, station B transitions to HVT and station C is added to the ring. Station B sends the token to station C, (XIII) and station C will enter HVT state after this.

L.2.5. **Connectivity Update**

Each node monitors the connectivity from all other ring members. This information from other ring members is recorded in the *receive table*, which notes nodes that are heard and an associated recommended maximum transmit speed, determined from SNR, FER and other information.

Nodes will be removed from the *receive table* on either reaching a configurable maximum age or a configurable number of ring cycles with no data heard.

Nodes also monitor for transmissions from nodes not in the ring, which can lead to ring merging where a node is detected is in a different ring.

L.2.6. **Communicating Connectivity**

The calculated receive table is shared with all other nodes. This is done by sending a TABLE message that contains the receive table. TABLE messages are sent along with the token and will progress around the ring and will also be received by other nodes listening in promiscuous mode.

This information is important to optimize ring performance, but is not essential for the ring to operate. The TABLE update approach taken can lead to loss, which is seen as preferable to the additional overhead of a fully reliable mechanism.

TABLE messages are versioned. The model is that when a new version of the *receive table* for a given node is received it will be transmitted exactly once, unless this version of the *receive table* has been received from all nodes to which the node is connected.

Whenever the receive table changes, the version is incremented and a TABLE message sent to share this update with other nodes.

L.2.7. **Changing Ring Transmit Order**

There are four scenarios where the *transmit order* will change.

- Scenario 1 (Joining Scenario): A node joins the network and will be added to the ring.
- Scenario 2 (Leaving Scenario): A node chooses to drop out of the ring, due to operator request. In this scenario, the node does not transmit, leading to the token being transferred to other nodes and the node being dropped.
- Scenario 3 (Token Transfer Fail Scenario): Changes in the network may lead to a situation where token transfer to the intended current successor node fails.

This node will then choose a new successor to transfer the token to. If the token cannot be transmitted to any successor, the node drops out of the ring.

- Scenario 4 (Ring-Optimization Scenario): In this scenario a node determines a better order, with a different successor to the one previously used. To move to this new order, a node will transfer the token to this new successor.

L.2.8. **Service Provision to Higher STANAG 5066 Layers**

STANAG 5066 Annex J defines the MAC model which WTRP provides. There is a clean layer model, with M_ service elements communicating between the layers. The details of these service elements are not specified, but the model and associated protocols are clear. WTRP provides some additional services to the generic MAC functions defined in Annex J.

L.2.8.1. **Data Rate, Interleaver and Transmission Length**

STANAG 5066 DTS layer will control selection of transmission speed, interleaver and transmission length. The DTS layer will also determine which D_PDUs to duplicate, such as sending ACKs multiple times to improve reliability. WTRP provides information to DTS to facilitate this choice.

WTRP introduces new D_PDUs with rules for transmission and duplication. Selection of data rate, interleaver and transmission length must be cognizant of this information. The Annex C Data Rate Selection mechanism remains the primary mechanism for selecting speed. The *next hop table* defined in L.2.1.9 **may** be made available to the DTS to provide maximum transmission speed information for each directly connected node. This information will generally be consistent with the DTS Data Rate Selection information, but could given additional data.

L.2.8.2. **Node Availability and Routing**

WTRP determines which nodes in the ring can be accessed directly and for those nodes which cannot be accessed directly it specified a preferred relay node. This information is generated in the *next hop table*, as specified in L.2.1.9. This information is passed upwards as an Annex J M_ service primitive. This information can be used:

1. By the CAS as specified in STANAG 5066 Annex B to immediately reject messages to nodes that are not in the ring; and
2. By the Routing Sublayer as specified in STANAG 5066 Annex R to relay data to nodes that are not directly connected.

L.2.8.3. Broadcast Retransmission

When a broadcast/multicast PDU is received, information is provided to the higher layers. It is important that broadcast and multicast work correctly for nodes that are not directly connected. This needs to be done in a way that prevents duplicate onward broadcast. Consider four nodes connected in a long trapezoid shape, where the two end nodes cannot communicate directly. If one of the end nodes sends a broadcast message, then one (but not both) of the middle nodes that can communicate with both nodes needs to broadcast the message, so that all nodes receive it.

This is achieved by providing a *Relay Responsible List* to the higher layer for any broadcast or multicast PDU that is received. This tells the higher layers which nodes it needs to re-broadcast to. Handling this is described in STANAG 5066 Annex R (Routing Sublayer), which makes use of next hop table information to correctly.

The *Relay Responsible List* is used by the receiver of a multicast/broadcast PDU to determine which nodes it should relay to, in order to prevent duplicates and loops. There needs to be such a list for each connected node, as the calculation depends on the node broadcasting. All nodes have the same base *receive table* data, and so can perform the same calculation.

To determine the *Relay Responsible List* for a WTRP broadcast sender, take the following steps:

1. Consider each node in the ring. This is the starting list.
2. Determine *Next Hop Table* for the broadcast sender to be used in the next two steps.
3. Eliminate all nodes which can be directly reached by the broadcasting node using non-ARQ.
4. For remaining nodes, if the local node is the preferred relay, add the node to the *Relay Responsible List*.

L.3. WTRP TOKEN AND MESSAGE SPECIFICATION

L.3.1. Node ID

Nodes are identified by a single byte ID (range 0-255). This limits the number of nodes in a ring to 256, which is expected to be higher than the practical limit. This Node ID enables the TABLE PDU, which will typically be used quite a bit, to be more compact. Externally, each node is identified by a variable length STANAG 5066 address (typically 3.5 bytes).

The Node ID will usually be the same as the last byte of the node's STANAG 5066 address, unless there is a conflict in the ring. This enables provisioning of addresses to be done in a way which avoids Node ID conflict.

Each Node will maintain a map between Node ID and STANAG 5066 address for use with TABLE PDUs. The mapping is derived from the set of TABLE PDUs; when distributed the TABLE PDU includes this mapping. It is possible that Node ID conflicts will occur, typically when merging rings. If a node detects that it's Node ID is being used by another node, it will assign a new Node ID and send an updated TABLE message as soon as it can.

L.3.2. **Token Protocol**

The token is transmitted using a standard EOW message. This enables the token to be transmitted along with standard data. EOWs comprise one byte type and one byte content and can be carried in any DTS D_PDU.

The EOW **shall** be sent in at least one D_PDU with destination address of the token recipient. The source address will always be that of the token sender. If any WTRP TABLE PDUs are being sent they can also carry this EOW. The EOW can also be sent with user data going to the correct destination. It is generally desirable to include the EOW multiple times. If there is no D_PDU with the correct destination, ACK-ONLY or PADDING D_PDUs can be used to carry the EOW with low overhead.

The EOW type is value 13.

The single byte value of the EOW content encodes information on number of ring members and ring cycle length. This can be helpful for new nodes to set appropriate timers, prior to receiving full *receive table* information. The encoding is specified in Figure L-1, with values as follows:

- Node Count encodes the number of nodes, with the value reflecting the total number of nodes in the ring minus one (1). So a value of 0 indicates as self ring (one member). A value of 15 indicates 16 or more nodes.
- Hop Delta is used to share the *ring cycle count*: the total number of hops. It is encoded as a delta relative to Node Count. So if Hop Delta is 0, it indicates a fully connected ring without any extra hops. For a ring that has more than 16 nodes, the Hop Delta is the *ring cycle count* minus 16. If Hop Delta is 15, then this is a minimum value.

	MSB							LSB
	7	6	5	4	3	2	1	0
0	Hop Delta				Node Count			

Figure L-4– EOW Content Encoding

This encoding anticipates that the practical upper limit of node count is around 10 nodes.

L.3.3. **WTRP Messages**

WTRP uses a set of messages to communicate information. These use EXTENSION D_PDUs as specified in STANAG 5066 Annex C. The allowed messages and defined extension D_PDU numbers are specified in Table L-1:

Message	Description	Extension Number
INVITE	Invite other nodes to join ring	1
JOIN	Request to join ring	2
TABLE	Receive Table	3
TABLE+	Receive Table (Extended Form)	4

Table L-1 – WTRP Messages

The syntax of these messages is specified in the following sections, and procedures to use them are specified in Section L.4. All of these D_PDUs use D_PDU type of 13 to identify EXTENSION D_PDU, and the extension number is then used to distinguish between each WTRP message.

It is expected that messages will generally be short enough to fit within a STANAG 5066 header, but the TABLE will not always be. Because of this, two forms of the TABLE message are defined. Use of different extension number allows the format to be determined from the extension number in the D_PDU header.

L.3.3.3 INVITE Message

	MSB							LSB
	7	6	5	4	3	2	1	0
0	D_PDU Type = 13				EOW Type			
1	EOW							
2	EOT							
3	Size of Address Field (m)			Size of Header (h)				
3+m	Source and Destination Address							
4+m	MSB	Extended D_PDU Type = 1						LSB
CRC	CRC on Header							
CRC								

Figure L-5 – INVITE Message

The INVITE message is used by a node with the token to invite other nodes to join. It is generally sent to the broadcast address, so that any listening node will receive it.

It may be sent to a specific address, which is referred to as a Targeted INVITE Message. This is used when merging rings, to indicate to the recipient that the sender has heard the node and is inviting a ring merge.

L.3.3.4 JOIN Message

	MSB							LSB
	7	6	5	4	3	2	1	0
0	D_PDU Type = 13				EOW Type			
1	EOW							
2	EOT							
3	Size of Address Field (m)			Size of Header (h)				
3+m	Source and Destination Address							
4+m	MSB	Extended D_PDU Type = 2						LSB
CRC	CRC on Header							
CRC								

Figure L-6– JOIN Message

The JOIN message is used by a node responding to an INVITE message, indicating that it wishes to join the ring.

L.3.3.2 TABLE Message

	MSB							LSB
	7	6	5	4	3	2	1	0
0	D_PDU Type = 13				EOW Type			
1	EOW							
2	EOT							
3	Size of Address Field (m)			Size of Header (h)				
3+m	Source and Destination Address							
4+m	MSB	Extended D_PDU Type = 3						LSB
5+m	MSB			Sending Node Address				
8+m								LSB
9+m	MSB			Sending Node ID				LSB
10+m	MSB			Version				LSB
12+m				Table (max 18 bytes)				
H+m-2								
CRC	CRC on Header							
CRC								

Figure L-7– TABLE Message

	MSB							LSB
	7	6	5	4	3	2	1	0
0	D_PDU Type = 13				EOW Type			
1	EOW							
2	EOT							
3	Size of Address Field (m)			Size of Header (h)				
3+m	Source and Destination Address							
4+m	MSB	Extended D_PDU Type = 4						LSB
5+m	MSB			Sending Node Address				
8+m								LSB
9+m	MSB			Sending Node ID				LSB
10+m	MSB			Version				LSB
11+m	MSB			Number of Table Entries (n)				LSB
CRC	CRC on Header							
CRC								
h+m+1				Table				
h+m+n*2+1	MSB							
				CRC				
h+m+n*2+4								LSB

Figure L-8– TABLE Message (Extended Form)

The TABLE Message communicates a receive table. The Table is a sequence of two byte entries in arbitrary order. Tables of nine entries or less **shall** use the TABLE Message shown in Figure L-7. Longer tables **shall** use the TABLE Message (Extended Form) shown in Figure L-8. The fields are the same, except that the Extended Form has a Number of Table Entries header, which enables the length of the table, and thus the D_PDU length to be determined.

The Sending Node Address field is the STANAG 5066 address of the node that generated the receive table. The Sending Node field is the node ID of the node that generated the receive table.

The table has a version number starting at 0, which is incremented for each change, resetting to zero after 255. This is to ensure that when multiple versions of a receive table are circulating, that a node will ignore a version prior to the one it holds.

Each table entry comprises two bytes. The first byte is the node ID of the node that data can be received from. The second byte is the recommended transmit speed. This is encoded following the encoding specified in Section C.6.5.1 of STANAG 5066 Annex C (Transmission Speed and Interleaver Encoding).

L.4. PROCEDURE OF OPERATION

WTRP operation is defined in terms of the state machine specified here. Specifications and definitions are provided for states and state transitions. This makes use of a number of data structures and procedures defined in this section.

L.4.1. Node Data Structures

A node shall maintain the data structure listed Table L-6. These data structures are all prefixed with the following prefixes to facilitate clarity when they are referenced:

- "Node.": Generic information associated with the node.
- "LocalTable": Information associated with the local receive table.
- "PeerTables.": Information associated with receive tables from other nodes.
.
- "Invite.": Information associated with control of invitations.

Table L-2 – Node Data Structures

Register / Flag	**Type**	**Initial Value**	**Description**
Node.address	STANAG 5066 Address	n/a	The station's address. The address is expected to be configured.
Node.ID	Integer	Last byte of Node.address	Node ID of the station. May be re-assigned in the event of conflict.
Node.cycleCount	Integer	0	The number of times that a complete ring cycle has been completed. This is used as an option to control sending invites only when the ring topology is stable.

Register / Flag	Type	Initial Value	Description
Node.Predecessor	STANAG 5066 Address	n/a	Set to sender of Token, when token is received.
Node.Sucessor	STANAG 5066 Address	*n/a*	Set to the Token receiver, when token is transmitted This is needed for monitoring after the token has been passed.
Node.TokenTransferred	Boolean	false	Acknowledgement is implicit, by monitoring traffic. This variable is set if traffic arrives that indicates token transfer
Node.OperatorDrop	Boolean	false	This variable is set by an operator. If set, it will cause the node to be dropped from the ring.
Node.txTokenCounter	Integer	0	Used to record repeats of token transmission.
Node.NextHopTable	Next Hop Table	empty	This is maintained so that higher layers of STANAG 5066 can select best speed for sending to a given node and can send indirectly to nodes that cannot be reached.
Node.NewNodes	List of Nodes with timestamps	Empty	Used to record JOIN requests, to build list of nodes to be added to ring. Associated with each node is a timestamp
Node.JoinsReceived	Boolean	False	Used to control ignoring of JOINs sent by other nodes
LocalTable.Table	Receive Table	Empty	The value of the local receive table, holding information on receive quality from peer nodes
LocalTable.TableChanged	Boolean	false	True if receive table has been changed since it was last transmitted
LocalTable.UpdateTimes	Array of Times, indexed by node address	empty	For each node in the ring, the time that the entry in LocalTable.Table was last updated.

Register / Flag	Type	Initial Value	Description
PeerTables.Tables	Receive Table List	empty	The active list of receive tables from peers. This list is updated whenever a more recent receive table is received from any node.
PeerTables.Transmitted	Received Table List	empty	List of receive tables indicating the most recent version that has been transmitted. This enables "transmit once" control.
Invite.LastTime	Time	Current time	When the last invite was sent by current node
Invite.Number	Integer	0	Number of invitations issued since joining ring

L.4.2. **Message and Token Notation**

The notation specified in this section is used to refer to tokens and messages and their components. Tokens have the following references, which are used to describe both reading and setting the token.

Table L-3 – Token Notation

Notation	Type	Description
token.sender	STANAG 5066 Address	Node sending the token from D_PDU source address
token.receiver	STANAG 5066 Address	Node to which the token is being sent taken from D_PDU destination address
token.inviterID	NodeID	The node ID of the node which is responsible for sending the next invite and for updating this field to set the subsequent inviter.

Messages are reference by notation of the form message.<message name>, for example message.JOIN. The following references to components of messages are used.

Table L-4 – Message Notation

Notation	Type	Description
message.<message name>.sender	STANAG 5066 Address	Node sending the message from D_PDU source address
message.<message name>.receiver	STANAG 5066 Address	Node to which the token is being sent taken from D_PDU destination address
message.TABLE.Table	Receive Table	Includes sender node ID and length

L.4.3. Node Procedures

The following functions are used in the state machine to specify node behaviour. Note that while this functional notation specified behaviour, this specification imposes no requirement on an implementation to use these functions.

---

isMember(<node address>)

Return type: Boolean

Description: Returns true if the given *node address* is a ring member, otherwise false.

Inspect Node.NextHopTable to determine if the specified node is present.

RingMemberCount()

Return type: Integer

Description: Returns the number of unique nodes in the Node.NextHopTable.

RCL ()

Return type: Integer

Description: Returns the Ring Cycle Length, which can be determined from the *connectivity table*.

BreakLink(<node address>)

Return type: None

Description: When a node cannot be reached, this is used to change network topology. This is used to recalculate the ring topology prior to determining a new successor.

This is achieved by finding the entry for <node address> in PeerTables.Tables and removing the entry for the local node in the identified receive table. This changes the connectivity record, to indicate that the node in question cannot be reached from the local node.

L.4.4. **Processing Inbound Transmissions**

This section, in conjunction with the following sections (L.4.5 – L.4.7) describes the process for handling inbound transmissions. This functionality is driven from the state machine by a single call of Receive(). This will listen for a call and continue processing until a full transmission has been received. At the end of transmission it will return EOT and optionally one of the following as events to the state machine:

1. EOT Event. To indicate end of transmission This is always returned, and the following associated Boolean is set:
    a. D_PDUs received. Set to true if any valid D_PDUs received.

   NOTE: EOT is returned at the end of every received transmission, irrespective of whether or not any D_PDUs are received.

2. One of the following may also be returned with the EOT event:
    a. Token. If a token directed to the local node has been received and none of the following messages.
    b. Message.INVITE. If this has been received, it will be the only message and no Token will be received.
    c. Message.JOIN. If this has been received, it will be the only message and no Token will be received. If this is received, the variable Node.JoinsReceived is set to true.

Processing of Message.TABLE reception is handled by this procedure and transparent to the state machine.

When a transmission is received it is fully processed, prior to any actions being taken in the state machine. D_PDUs other than WTRP messages are processed following the rules of STANAG 5066. If no other event is returned, EOT is returned at end of transmission.

The following WTRP messages may be received:

1. WTRP Message.INVITE is usually sent to the broadcast address, except for a Targeted INVITE that is sent to a specific address. A transmission with a message.INVITE may contain multiple copies of this message. Message.INVITE is returned to the state machine, when the target is a broadcast address or the local address.

2. WRTP Message.JOIN is sent to a single address. A single transmission may contain multiple copies of this message. Note that although Message.JOIN is directed to the inviter other nodes in the ring will also process it. If Message.JOIN.receiver is determined not be a member of the ring, the Message.JOIN is discarded. Otherwise Message.JOIN.sender is added to the Node.NewNodes list.

The sender of the transmission can be identified from any D_PDU in the transmission. Use of isRingMember(sender) determines if the sender is in the current ring. If the sender is not a ring member, the procedure of L.4.10 is followed.

Once a transmission has been processed, it will be possible to determine either the target destination node for the WTRP information, or that the message is a broadcast message.INVITE.

If the target destination node is the local node, the Section L.4.6 is followed. If the target destination is another node then Section L.4.5 is followed.

L.4.5. **Handing Transmissions Directed to Other Nodes**

Transmissions directed to other nodes are not directly handled by the state machine, but may lead to setting of variables that impact the state machine. WTRP must listen for these transmissions (promiscuous mode). Information in these transmissions is used to update local information.

Message.JOIN and Message.INVITE are special transmissions with no user data. Handling these is covered in Section L.4.4.

A message containing a token directed to another node indicates explicitly that the transmission is directed at another node. WTRP messages will always contain a token, so it will be always be possible to determine where a transmission is directed when it contains WTRP messages.

Transfer of tokens **may** be noted. Token.sender and token.receiver may be useful to provide as operator information to monitor progress of the token around the ring.

If the transmission sender of any D_PDU is Node.Successor, set Node.TokenTransferred to true. This setting is used to change out of MON state.

If a monitored D_PDU contains the WTRP token and isMember(token.Sender) is true, set Node.TokenTransferred to true. This setting is used to change out of MON state.

Received Message.TABLE messages are used to update local status. If the message.TABLE sender node ID is the local node, and message.Table.table matches the current or a previous version of LocalTable.Table, it is ignored. If it does not match, this means that another node is using the same Node ID as the local node. This is addressed by assigning a new Node.ID to one that is believed to be unique.

If isMember(message.TABLE.sender) is true the receive table is from the current ring. If it is not the local receive table, update the table in PeerTables.Tables with message.TABLE.table if the version is more recent.

Message.TABLE has version numbers encoded as a single byte. To compare version numbers of current and new, Mod(new – current, 256) < 127 will be true if new is more recent than current.

L.4.6. **Procedure for Receiving the Token**

This section describes how to handle a message with a token that is directed to the local node and therefore needs processing by the local node.

For each entry in LocalTable.Table look at the associated update timestamp in Local.TableUpdateTimes. If it is older than RECEIVE_TABLE_EXPIRY_AGE, remove the entry from LocalTable.Table and set LocalTable.TableChanged to true.

Next, Node.NextHopTable is calculated from the receive tables stored in PeerTables.Tables following the procedure specified in L.2.1.10. This information **shall** be passed up to the higher layers of STANAG 5066.

The Receive() procedure returns Token.

L.4.7. **Determining Successor**

This section sets out the approach for determining the successor, which is used prior to making a general purpose transmission.

The first step is to determine if there are any new nodes to be added to the ring. Node.NewNodes, which contains a list of nodes that have requested to join the ring is considered. First remove any nodes which are determined to already be in the ring. If there are any remaining entries in Node.NewNodes, the oldest one **shall** be chosen as the successor and removed from the Node.NewNodes list.

The basic method of determining successor is by *connected node fairness*. This considers only nodes which can be reached directly. The best way to determine this list is from the *next hop table*, which gives a list of nodes with the two way connectivity needed for token transfer. If this information is not available, a list of nodes which are likely to work can be obtained from the receive table.

The *connected node fairness* algorithm is simply to send the token to the node that received it least recently. Because the local node can hear transmissions from all connected nodes, this is straightforward to determine.

It can be seen that the *connected node fairness* algorithm will work reasonably well for a fully connected ring. A partially connected ring can be considered as a concatenation of sub-rings, each of which are fully connected. It can be seen that this algorithm will ensure that all nodes are reached.

It is expected that the simple *connected node fairness* algorithm will lead to a reasonable ring choice in many scenarios, and in particular for many likely configurations.

This base algorithm is used, as it does not require any *receive table* derived information.

While this base algorithm is expected to be reasonable, it will not be optimal in all scenarios. Consider four nodes deployed in a square using HF surface wave, where optimum transmission speed decreases with distance. The best order will typically be to go around the edges of the square and avoid using the diagonals. The best ring cycle is likely to vary over time, as nodes move and conditions vary.

It is important to adapt to conditions. Consider a fully connected node network, with active transmission sequence ABCD. The *connected node fairness* algorithm will continue with this sequence, as long as all the nodes can communicate. If the best sequence was ABDC, it would need node B to choose to send to node D rather than node C (which is the one which has not had the token for the longest time).

A node can make use of the *connectivity table* to determine the best order. For a small set of nodes, it is practical to find the best order. For a larger set of nodes, determining best order is analogous to the well know “travelling salesman” problem, and seeking a good choice rather than best choice is likely to be computationally preferable.

NOTE: It is anticipated that operational experience will give feedback on the best approach to be taken, It is expected that this feedback will lead to updates to this standard.

L.4.8. **Sending Token, WTRP Messages and User Data**

A node will transmit data using the Transmit() procedure, which invokes the process described in this section. This is called from the state machine. At the end of transmission it will return an EOT-Transmit event to the state machine which indicates end of transmission.Transmission will be made to the node determined by following the procedure of Section L.4.7.

When a node has the token, it will make a transmission that includes the at least one copy of the token and may include user data. The token is encoded as an EOW and will usually be repeated many times in the transmission. Mechanisms to facilitate this are set out in L.6.

A transmission containing WTRP messages **shall** contain the token in each WTRP message. All WTRP messages and tokens in a transmission will have the same source and destination address. Other D_PDUs directed to the successor **may** contain the token. When token is transferred, it **shall** be put into at least one D_PDU. If necessary, a Padding D_PDU can be used for this. The token will generally be repeated many times.

The following WTRP messages **shall** also be transmitted. These messages may be repeated.

1. If LocalTable.TableChanged is true:
    a. message.TABLE is sent containing LocalTable.Table.
    b. LocalTable.TableChanged is set to false.
2. For each received table in PeerTables.Tables where the version is more recent than the one recorded in PeerTables.transmitted and the current version has not been received from all connected peers:
    a. message.TABLE is sent with the received table
    b. the version is updated in PeerTables.transmitted

User data, if available, **may** be sent in this transmission in addition to the messages above which **shall** be sent.

L.4.9. **Controlling Invitations**

This section specifies the algorithm for the node to determine whether or not to issue an invitation and setting the next inviter

Sending invitations needs careful tuning. There are a number of considerations:

1. Invitations should be sent from all ring members, as there may be nodes wishing to join the ring that can only hear transmissions from one ring member.
2. Invitations need to be sent sufficiently frequently so that new ring members can join in a timely manner.
3. In a stable long lived ring, adding new members will commonly be infrequent.
4. The invitation process is quite slow, and sending too many invitations will add unnecessary overhead.
5. In a partially connected network, it may be desirable to send invitations more frequently from the disconnected elements.

The approach taken in this specification is to have nodes independently choose when to send out invitations.

There is also an algorithm and a set of parameters specified which control how often a node issues an invite. This algorithm **may** be used or an implementation **may** choose a different algorithm as the exact algorithm will not impact interoperability although it can impact overall performance,

> NOTE: When operational experience is obtained with the algorithm, a future version of this standard is expected to be updated to reflect this.

The algorithms is specified using two procedures that are called form the state machine.

ReadyToSendInvite ()

Return Type: Boolean

Description: Returns true if the criteria here are met

There is a configurable maximum ring size (MAX_NET_SIZE). If the number of notes in the ring is equal to or greater than this size, no invitation is issued. Procedure returns false.

The following parameters are used to control issuing and invitation:

1. Time since Invite.LastTime. If greater than MIN_INVITE_INTERVAL (configurable), an invitation **should** be issued, and procedure returns true.
2. Number of ring circuits since last invite by this node, determined by Node.CycleCount – Invite.LastCycleCount. If greater than or equal to MIN_INVITE_CYCLES (configurable), invite **should** be issued and procedure returns true
3. InviteNumber. If this is less than or equal to EARLY_INVITE_COUNT (configurable) number and ring circuits less than EARLY_INVITE_CYCLES (configurable), an invite should be issued and procedure returns true. This option **may** be omitted. It is designed to give a higher invitation rate in a new ring.

If none of the above conditions are met, no invite should be issued and procedure returns false.

SendInvite()

Return Type: None

Description: Send an Invite

Send a message.INVITE with:

- message.INVITE.destination = broadcast address

Update variables as follows:

- Invite.LastInviteTime set to current time
- Increment Invite.Number

L.4.10. Handling Transmissions from Nodes not in the Ring

Where a transmission and WTRP messages are received from a node not in the ring and not simply joining the ring, there are three possible scenarios identified.

1. A node forming a self ring that has not yet heard this ring. Strategy is to just let it find the current ring and join.

2. A node joining elsewhere in the current ring. This will sort out without any action.

3. Another formed ring. The approach is to merge the two rings. Care needs to be taken with rings which are on the edge of communication, because of potential instability due to poor links. This algorithm requires repeat hearing.

The definitive indication of another ring is token transfer. If this is not detected, the other transmission is ignored. If token transfer is detected, this will be recorded. If NUM_OTHER_RING_HEARD (configurable) of token transmissions are heard within OTHER_RING_TIME (configurable), this is considered definitive detection of another ring, which is within range.

When another ring is definitively detected, wait until the node has the token and then issue a Targeted INVITE, directed to the node that has been heard. This will signal to the invited node that it has been heard and also that no other nodes should respond to the invite. When the invited node gets the token, it will pass it over and the rings will be merged.

L.4.11. Overall State Diagram

Figure L-9 below shows the complete state diagram of WTRP. Each state is described in detail in the following sections. The starting state is Floating State (FLT). Note that most other states revert to FLT directly on error conditions, which is typically timer expiry. This figure shows all allowed transitions and marks the normal flow for a ring member. A node will be in Idle state (IDL) listening until the token is received. When the token is received, it will transition to Have Token State (HVT) where it will transmit the token and optionally user data. Then it will shift to Monitoring State (MON) where it listens to hear the next node transmit, which is an implicit acknowledgement of token transfer. When it hears this it transitions back to Idle State (IDL).

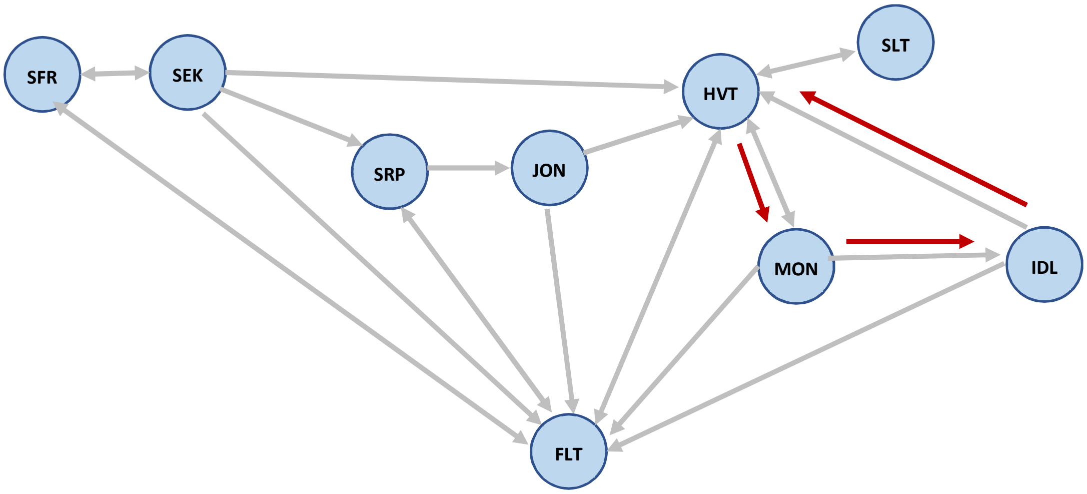

Figure L-9– Overall State Machine: Normal Ring Member Flow

Figure L-10 below illustrates how a node in an active ring will issue invites from time to time, breaking the normal state flow. This happens at intervals when a node is in Have Token State (HVT) it will, based on configurable rules, transition to Soliciting State (SLT) where it broadcasts and invitation to join the ring. It gathers any join requests which are then processed in Have Token State (HVT), either by this node or another one.

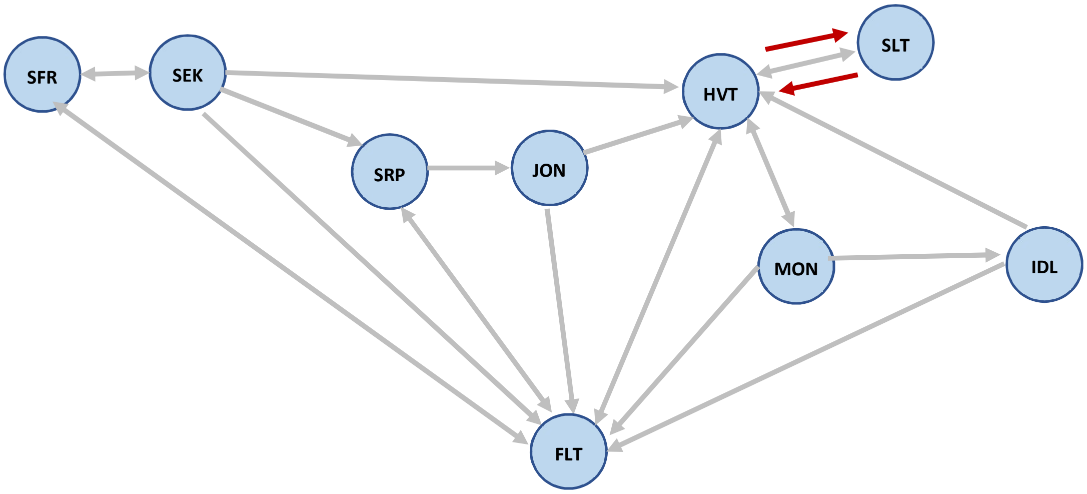

Figure L-10– Overall State Machine: Sending Invites

Figure L-11 below shows the normal path by which a node joins an existing ring. A node in Floating State (FLT) will hear a broadcast message invite. It will then transition to Solicit Reply State (SRP), send a Join message and transition to Join State (JON). The inviting node will send the token to the joining node which will transition to Have Token State (HVT) and become a ring member.

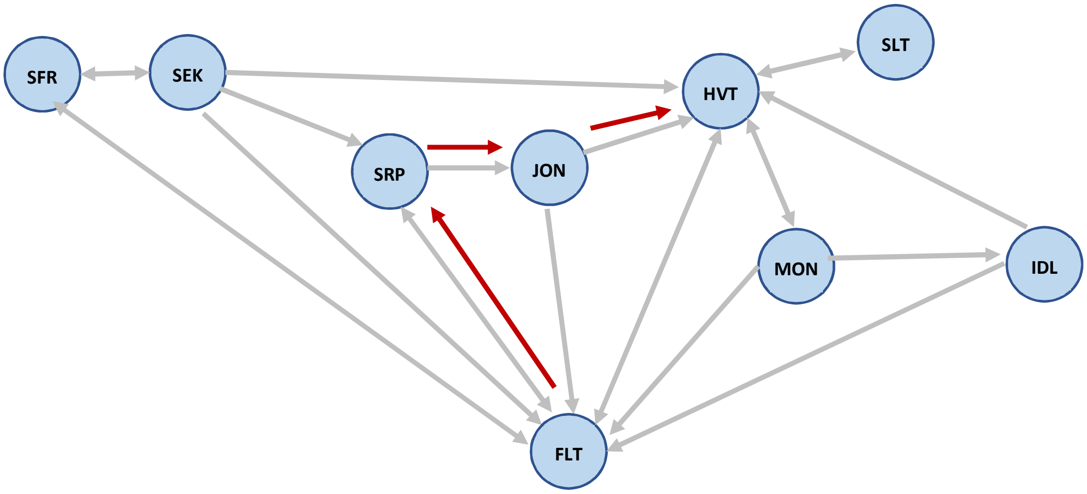

Figure L-11– Overall State Machine: Joining a Ring

Figure L-12 below shows the flow for a node initializing a new ring. A node starting in Floating State (FLT) will transition to Self Ring State (SFR). It will then broadcast a join invitation and move to Seeking State (SEK) where it waits for responses. When it gets one or more responses it will transition to Have Token State (HVT) and build the ring using the responses. It reverts to Floating State (FLT) if no other node responds.

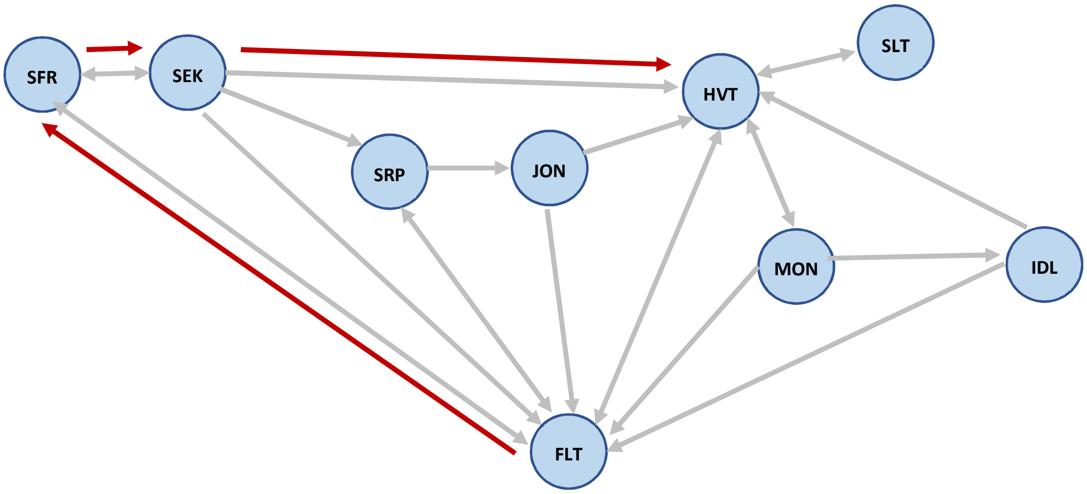

Figure L-12– Overall State Machine: Starting a New Ring

L.4.12. State-Machine Specification

This section and its subsections specify the actions of the WTRP state machine. For every state there are state-entry actions and an outbound-transition table defined. When entering a state, a station first **shall** execute the state-entry actions and then it **shall** wait for an event to occur which triggers one of the transitions to the next state defined in the outbound-transition table. There are two types of event that trigger state transitions:

- Events caused by timeouts; a timeout event is prefixed with the label “Exp” (i.e., for expiry or expired), followed with the name of the timer causing the timeout.
- Events caused by received data; this event is prefixed with the label “Rcv” (i.e., for ‘received’), followed with information to clearly identify what is received.

Only one transition rule **shall** be executed after an event: this **shall** be the first and only the first transition for which the condition is met as the state-machine logic examines the outbound-transitions in the order in which they are listed in the table.

When an action causes data to be transmitted, transition to the next state is expected to happen after the data has been transmitted and before any data is received.

L.4.12.1. Floating State (FLT)

The *Floating State* is the WTRP start-up state and the state in which a station is not part of a ring and waits to join a ring. The floating state is a *listening-only* state. A station **shall** stay in the FLT state until there is a joining opportunity (i.e., a message.INVITE is received inviting the station to solicit membership in the ring) or the TCLT timer expires.

The TCLT timer is used to determine when a station **will** assume that there is no existing ring present. If this timer times out (i.e., expires) the station **shall** proceed to *Self Ring State* (SFR).

If the station receives a message.INVITE it **shall** transit to the *Joining State* (JON) via the *Solicit Reply* (SRP) state,

L.4.12.1.1. Floating State entry actions

On this state entry the station **shall** execute the following actions:

- Start the TCLT timer
- Start Receive()

L.4.12.1.2. Floating State outbound transitions

Outbound transitions from the FLT state are shown in the table below. The most significant outbound transition is to the Solicit-Reply (SRP) State, which occurs when a node receives an invitation to join a ring. For other transitions, the node either remains in the FLT state waiting to receive invitations or transits to the SFR state where it will wait before deciding to send its own invitations for nodes to join its ring.

If any transmission is heard, it will cause the node to wait longer by setting the TCLT timer to TCLT_TRANS_HEARD. TCLT will reset to TCLT_SILENT if no further transmissions are heard.

It is possible that the token will be passed to a node in FLT state. This can happen if node is in ring and transmits token successfully without hearing the onward transmission. In this situation, the node can move to FLT state (from MON) and then later receive a valid token.

Table L-5 - FLT outbound transitions

transition	event	Condition	Action	next state
FLT-R1	EOT	Message.INVITE	Save message.INVITE for processing in SRP state	SRP
FLT-R2	EOT		Set TCLT to TCLT_TRANS_HEARD Start TCLT Timer Receive()	N/A
FLT-R3	EOT	Token		HVT
FLT-T1	Exp: TCLT		SET TCLT to TCLT_SILENT	SFR

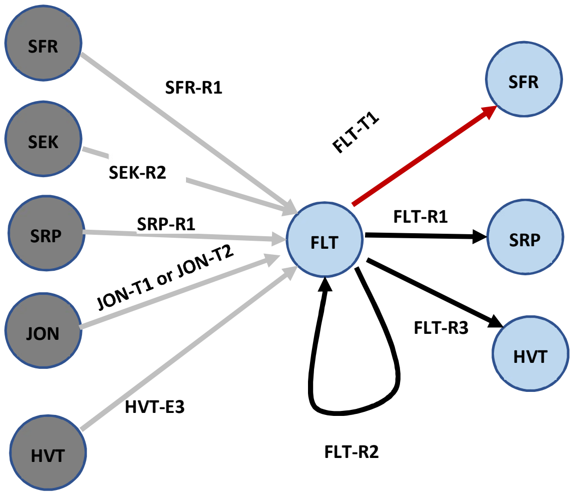

Figure L-13 - FLT Outbound transitions

L.4.12.2. Self Ring State (SFR)

In this state a station has concluded there is no ring to join and therefore will setup a new ring by itself. This condition is called the *self ring*. This state is like the Floating State (FLT) in that it is a listening-only state.

On state entry the TSLS timer **shall** be set to a random timeout value, which is used to avoid collisions with any other station trying to setup a ring. If this timer (i.e., if TSLS) times out the station **shall** transit to the *Seeking State* (SEK), where an invitation to nodes to join its network will be sent as an inbound action.

While waiting for the TSLS timeout, a message.INVITE **might** be received from another station in SEK state; in this case the station **shall** transit to the *Solicit Reply State* (SRP), where it will respond to the invitation.

If any other transmission is received the station **shall** go to the *Floating State* (FLT).

L.4.12.2.1. Self Ring State (SFR) entry actions

On SFR-state entry a station **shall** start its TSLS Timer with a random time-out value over a configurable range. Receive() procedure.

L.4.12.2.2. Self Ring State (SFR) outbound transitions

Outbound transitions for the SFR state are shown in the table below. Transitions from the SFR state are triggered: when the station receives an invitation from another node and then will reply; when the station receives any other D_PDU from another station and thus should wait for an invitation; or when the station hears nothing for some time and then sends its own invitation.

Table L-6 - SFR-State Outbound-Transition Table

transition	event	condition	action	next state
SFR-R1	EOT		Set TCLT to TCLT_TRANS_HEARD	FLT
SFR-R2	EOT	Message.INVITE	Save message.INVITE to be processed in SRP state	SRP
SFR-T1	Exp: TSLS		See L.4.9	SEK

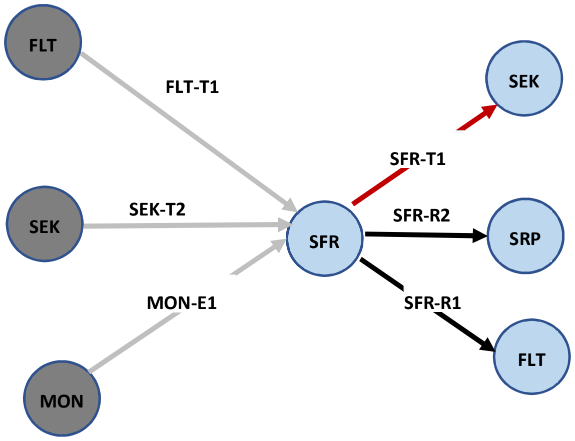

Figure L-14 - Outbound transitions from SFR state

L.4.12.3. Seeking State (SEK)

The *Seeking State* is a state in which a station in a *self ring*, the seeking node, broadcasts a message.INVITE inviting new ring members and listens for replies from solicitors until the TSLW timer times out.

When the TSLW timer expires and no message.JOIN has been received, the node reverts to FLT state. If one or more message.JOIN has been received, they will be recorded to enable addition to the ring. The station will then proceed to the HVT state.

If a message.INVITE is received before the TLSW timer expires, then another ring (possibly a self-ring) is active within radio range of the station. The station in this case **shall** cease its invitations to other nodes to join its self-ring and **shall** transit to the SRP state, with intent to join the ring it has detected.

Reception by a station when it is in the SEK state of any other transmission shall force the station into the FLT state, as reception of such tokens is an indication there is an active ring within radio range, the self-ring condition no longer applies and the station should not be soliciting to form its own ring.

L.4.12.3.1. Seeking State entry actions

On state exit from the SFR state the station has broadcast a message.INVITE, and shall start the TSLW timer waiting for replies. Start Receive() procedure.

L.4.12.3.2. Seeking state outbound transitions

Outbound transitions from the SEK state are shown in the table below. As noted above, reception of message.JOIN does not force an outbound transition; the message.JOIN messages are stored for later processing and the station remains in the SEK state. Reception of a message.INVITE forces a transition to the SRP solicit reply state and reception of any other transmission forces a transition to the FLT floating state; both of these triggering events invalidate the station’s assumption that it can form its own ring but with different responses by the station. As long as the station receives only message.JOIN messages, it will continue to handle them until the TLSW timer expires, at which point the station proceeds to the HVT state and subsequent operation in a multi-node network. If no responses of any kind are heard after waiting for replies, the station returns to the SFR self-ring state.

Table L-7 - SEK state outbound transitions

transition	event	condition	action	next state
SEK-R1	EOT	Message.INVITE	Save message.INVITE for processing in SRP state	SRP
SEK-R2	EOT	D_PDUs Received set to true	Set TCLT to TCLT_TRANS_HEARD	FLT

SEK-R3	EOT	Message.JOIN	Receive() Note that Message.JOIN is processed within Receive() following L.4.4	N/A
SEK-R4	EOT	D_PDUs Received set to false	Receive()	N/A
SEK-T1	Exp: TSLW timer	At least one message.JOIN received		HVT
SEK-T2	Exp: TSLW timer	No message.JOIN received		SFR

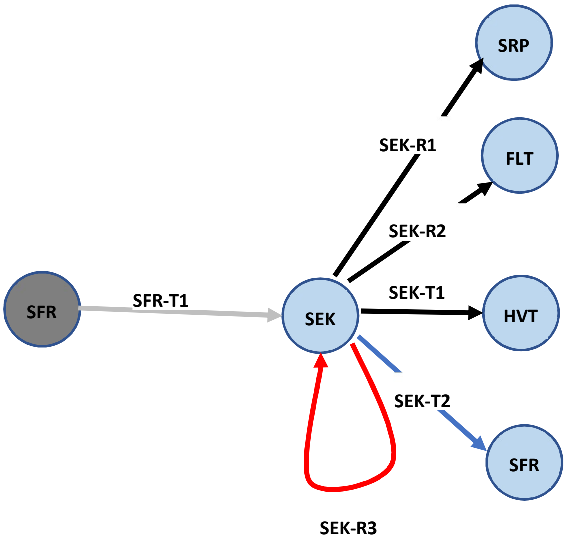

Figure L-15 - Outbound transitions from SEK state

L.4.12.4. Solicit Reply State (SRP)

In the *Solicit Reply State* a station will reply to an invitation to join the network (i.e., to a message.INVITE received from another node) and attempt to join the ring.

The TSRP timer **shall** be started on state entry to determine the moment to reply.

While waiting in the *Solicit Reply State* for the TSRP time to expire, other message.JOIN messages might be received and should be ignored, as they could be expected as replies by other stations to the message.INVITE.

Receipt of other message traffic indicates that the channel is not clear during the protocol's invite-and-solicit dialog for new ring members (an error condition in the protocol). The error condition is best handled by having all solicitors return to a known state (the FLT state) in which they do not transmit.

The soliciting station **shall** reply by sending a message.JOIN to the inviting station, the message.INVITE originator.

L.4.12.4.1. State entry actions:

Set the TSRP timer following the algorithm defined in Section L.5. Start Receive() procedure

L.4.12.4.2. Solicit Reply State Outbound Transitions

Outbound transitions from the solicit transition can be categorized as error recovery or incremental success in joining the network. If any message other than a message.JOIN is heard, the station effectively declares error and transitions to the FLT state, where it will wait for another invitation to join. If the channel remains clear of unexpected traffic, the node sends its solicitation (a message.JOIN) to join the network and transits to the JON state where it will wait to see if its solicitation succeeded.

Table L-8 - SRP outbound transition table

transition	Event	condition	action	next state
SRP-R1	EOT	Node.JoinsReceived set to false and D_PDUSs Receives set to true (Message.JOINs being sent by other nodes in response to invite are ignored)	Set TCLT to TCLT_TRANS_HEARD	FLT
SRP-R2	EOT	(other than SRP-R1)	Receive()	N/A

SRP-T1	Exp: TSRP		Send message.JOIN where
			message.JOIN.sender = Node.address
			message.JOIN.receiver = address of inviter
			message.JOIN.Table = LocalTable.Table
				JON

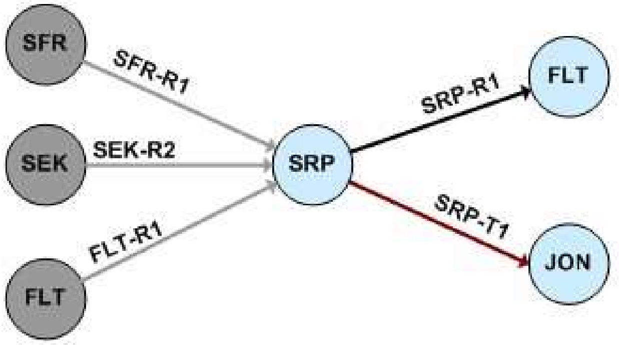

Figure L-16 - Outbound transitions from SRP state

L.4.12.5. Joining State (JON)

In the *Joining State* a station has replied to solicitation opportunity by sending a message.JOIN and is now waiting for the *right-to-transmit token*.
The test that the soliciting station has successfully joined the ring is straightforward; the soliciting station either receives an RTT token (in which case it joins the ring) or it does note.

L.4.12.5.1. Joining State entry actions

Start the TCON timer. Start Receive().

.

L.4.12.5.2. Joining State outbound transitions

Outbound transitions for the JON state are shown in the table below. A station either succeeds with its solicitation to join, and transits to the HVT state as a ring member and able to transmit, or fails and transits to the FLT state where it will wait for the next invitation from a ring member (or its own declaration that there is no ring present and eventual transition to the SFR self-ring state).

Table L-9 - JON-State Outbound-Transition Table

transition	event	condition	action	next state
JON-R1	EOT		Receive()	N/A
JON-R2	EOT	Token		HVT
JON-T1	Exp: TCON timer		Set TCLT to TCLT_TRANS_HEARD	FLT

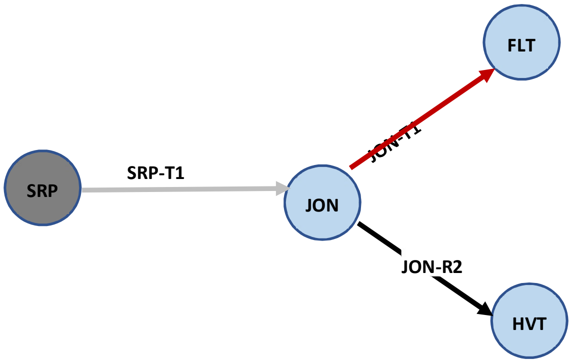

Figure L-17 - Outbound transitions from JON state

L.4.12.6. Have Token State (HVT)

In the *Have Token State* a station has received the *right to transmit token* (RTT) and as part of the state exit action the channel "owner", is allowed to send D_PDUs as long as the length of its transmission does not exceed the *maximum time to transmit*.

L.4.12.6.1. Have Token State entry actions

The node will have received the token in the previous state, which causes the node to enter HVT. The received transmission will have been processed according to L.4.6.

If map or receive table information is needed, but not received, token is sent to predecessor with message.RETRANS and node enters MON state.

The node will verify whether or not an invitation should be sent, following the procedure in L.4.9. If it is determined that an invitation should be sent, and invitation is sent and the node transitions to SLT state. After SLT, the node will return to HVT.

After any invitation has been sent, the token, optional messages and optional user data will be sent following L.4.8, and the node transitions to MON state. L.4.8 will also handle the following special situation:

1. The HVT state is where the node processes operator or other local requests for the node to drop out of the ring. The node will simply not transmit data and return to FLT state, ignoring when the token is sent to it.

L.4.12.6.2. Have Token State outbound transitions

Outbound transitions from the HVT state are shown in the table below.

Transitions from the HVT state to the MON state happens after the transmission has completed, which is indicated by the EOT-Transmit event The TEOT timer protects against missing EOT-Transmit event.

Table L-10 - HVT-State outbound transition table

transaction	event	condition	action	next state
HVT-E1	State entry	ReadyToSendInvite () == true	SendInvite()	SLT
HVT-E2	State entry	ReadyToSendInvite () == false && RingMemberCount() > 1	Set TEOT Transmit() following L.4.8	N/A
HVT-E3	State entry	ReadyToSendInvite () == false && RingMemberCount() <= 1	Set TCLT to TCLT_SILENT	FLT
HVT-R1	EOT-Transmit			MON

HVT-T1	Exp: TEOT			MON

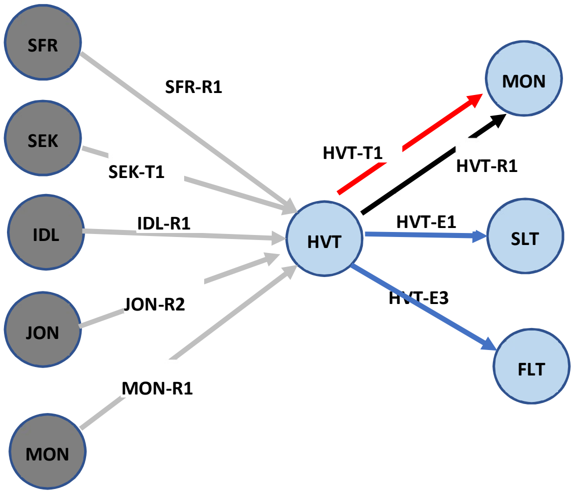

Figure L-18 - Outbound transitions from HVT state

L.4.12.7. Monitoring State (MON)

The *Monitoring State* is a state in which a station has finished transmitting data, passed the RTT token to its successor and is waiting for an acknowledgement. An implicit acknowledgement mechanism is used. Any D_PDU received in the MON state with source address equal to the station's successor are taken as implicit acknowledgement that the successor received the RTT token and has taken control of the radio channel. Also, any token transfer on the current ring indicates that the token is circulating in the ring. This second mechanism addresses when the transmission by the successor is not heard. This monitoring is handled by L.4.5 and reflected in the variable Node.TokenTransferred.

In a two node ring, the token will be passed directly back.

At expiration of the TPST timer the token transfer is considered to have failed. The station shall resend it for a number of times until the MAX_TOKEN_PASS count has been reached. It is recommended that retransmissions are done at progressively slower speeds.

If the MAX_TOKEN_PASS count has been reached the station declares the next-hop node unreachable and reflects this in connectivity by using the breaklink() procedure. The Transmit() procedure of L.4.8 is followed again. This process continues until the token is transferred or there is no possible node to transfer to. If all other nodes are eliminated, the node transitions to SFR.

L.4.12.7.1. Monitoring State entry actions

Transmission, including token transfer, will have been completed prior to state entry. On state entry the station **shall** start the TPST timer, setting the time the station shall wait for receipt of an implicit acknowledgement that the *right-to-transmit* has been transferred.

For any entry other than a self-transition, i.e, if the preceding state is not the MON state, Node.txTokenCounter is set to 1.

Start Receive()

L.4.12.7.2. Monitoring State outbound transition table

Outbound transitions from the MON state are shown in the table below.

Transitions representing a successful pass of the *right-to-transmit* token are to HVT state (token comes back) or to IDL state.

Other transitions will cycle through the MON state to transmit to nodes in turn. When a node is joining the ring, it will only attempt to transmit back to the node that sent the token to it.

Table L-11 - MON-State Outbound-Transition Table

transition	event	Condition	Action / Comments	next state
MON-E1	State entry	RingMemberCount() < 2	If ring has been reduced to 1, move to float Set TCLT to TCLT_SILENT	FLT
MON-R1	EOT	Token		HVT
MON-R2	EOT	Node.TokenTransferred == true	Implicit Ack Set Node.txTokenCounter to 1	IDL
MON-R3	EOT	Node.TokenTransferred == false	Receive()	N/A
MON-T1	Exp: TPST	Node.txTokenCounter < MAX_TOKEN_PASS	Increment Node..txTokenCounter Return to HVT to force transmission of token again.	HVT
MON-T2	Exp: TPST	Node.txTokenCounter >= MAX_TOKEN_PASS	Set Node.txTokenCounter to 1 BreakLink(Node.Successor) Return to HVT to force transmission of token again.	HVT

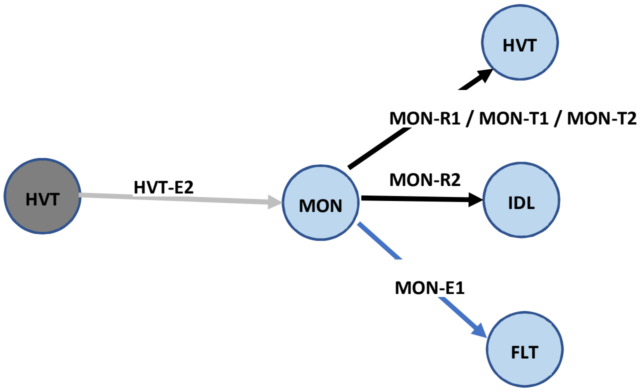

Figure L-19- MON Outbound transitions from monitoring state

L.4.12.8. Idle State (IDL)

In the *Idle State* the station waits for the *right-to-transmit token* until its TIDL timer expires. When it receives an RTT token, it transits directly to the HVT state. In the IDL state, the node listens for traffic following L.4.4, L.4.5, L.4.6 and L.4.10.

If the TIDL timer expires the station assumes an error-condition in the protocol and transits to the FLT state to recover.

L.4.12.8.1. Idle State entry Actions

The station **shall** start its TIDL timer on state entry. Receive()

L.4.12.8.2. Idle State (IDL) outbound transitions

Outbound transitions from the IDL state are shown in the table below.

Table L-12 - IDL State Outbound-Transition Table

transition	event	condition	action	next state
IDL-R1	EOT	Token		HVT
IDL-R2	EOT		Receive()	N/A
IDL-T1	Exp: TIDL		Set TCLT to TCLT_SILENT	FLT

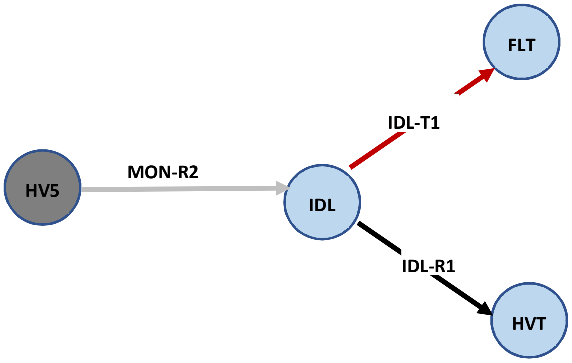

Figure L-20 - Idle State outbound transitions

L.4.12.9. Soliciting State (SLT)

The purpose of the SLT state is to allow new stations to join the ring. A station in the *Soliciting State* is a ring member and has received the *right-to-transmit* from the previous HVT state. A message.INVITE was sent on transition to SLT.

The station waits until TSLW timer times out and then returns to the HVT state. Every message.JOIN is processed, which will be used to determine the successor in HVT state.

L.4.12.9.1. Soliciting State (SLT) entry actions

On state entry the station **shall** start the TSLW timer and listen for message.JOIN.

Start Receive()

L.4.12.9.2. Soliciting State (SLT) outbound transitions

Outbound transitions from the SLT state are shown in the table below.

The normal transition (SLT-T1) to the HVT state occurs when the TSLW time expires, and is made whether or not new stations have solicited to join.

Table L-13 - SLT-State Outbound-Transition Table

transition	event	condition	action	next state
SLT-R1	EOT	Message.JOIN	Receive() Note that Message.JOIN is processed within Receive() following L.4.4	N/A
SLT-R2	EOT		Receive()	N/A
SLT-T1	Exp: TSLW			HVT

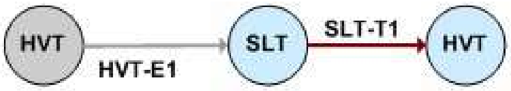

Figure L-21 - Outbound transitions from soliciting state

L.5. TIMERS AND PARAMETERS

This section reviews all of the parameters and timers. The core of this section is a table with all of the specified timers and parameters. The rationale for the parameter and considerations for optimal settings are described. Each of these parameters and timers **shall** be modifiable by an operator to change the protocol responsiveness or behaviour in response to different operational requirements or tradeoffs (e.g., increased collision probability for newly joining nodes versus reduced solicitation overhead).

L.5.1. **HF Considerations**

HF provides a difficult channel with highly variable quality where the optimal transmission speed varies significantly, with short and extended periods where no communication is possible. The nature of this channel is a primary consideration in choosing parameter and timer values.

L.5.2. **Surface Wave Considerations**

A major target for WTRP is surface wave, particularly for communication between naval vessels where surface communication is good and extends a long way beyond line of sight. WTRP is less suitable for HF Sky Wave, where it will generally be preferable to use ALE to optimize point to point links, rather than attempting to use a single channel for many nodes. For this reason, default parameters are chosen on the basis of using surface wave to communicate between a group of ships. Where WTRP is used in other scenarios, it is likely to be desirable to change defaults.

With surface wave, SNR will systematically decrease with distance. This means that SNR will change as nodes move and will be better for nodes that are closer together. For this reason, it is anticipated that the variable speed operation provided by this specification will be highly beneficial to good performance.

At the limit of communication distance, communication is expected to be very intermittent. Care needs to be taken with this, as there is possibility to spend resources trying to maintain a link between a pair of nodes where they can hear each other, but the channel is too poor for robust transfer.

L.5.3. **Ring Forming Strategy**

A node that is not in a ring will attempt to join or form a ring. A node cannot belong to more than one ring, and having two “intersecting” rings is highly undesirable and should be avoided. If a node hears an existing ring, the strategy is to wait patiently to join the ring.

If a node is not hearing anything, it is hard to distinguish between the following scenarios:

1. No ring, and potentially other nodes that could form a ring.
2. A large ring with long transit time.
3. An existing ring that is hidden by poor HF conditions

L.5.4. **Ring Maintenance Strategy**

Once a node is in a ring, efforts should be made to maintain the ring. Node failure must be allowed for, but is unlikely. The most common failure is expected to be due to poor HF conditions, and retry is the best approach here. It is also possible that surface wave distance is increasing between a pair of nodes, leading to the link between these nodes becoming not being viable.

L.5.5. **Parameter and Timer Table**

Table L-14 sets out parameters and timers. Some timers specify use of jitter to vary the time to prevent nodes deadlocking with repeat cycle.

Table L-14 – Parameters and Timers

Parameter Name	Default Value	States	Notes
TCLT	TCLT_SILENT	FLT	Claim Token Timer (TCLT) - Timer used in the floating-state (FLT). Controls the time a station waits while in the floating state to claim a token before transiting to another state; a station restarts its TCLT timer when it transits to the FLT state. A node in this state does not know if there are other rings. Primary strategy is to find another ring. If this timer is set too short, risk is that two nodes will form a ring while there is another ring. Set too long, and it will delay setting a ring. There are two values for TCLT, which are set by the state machine. TCLT is initialized to TCLT_SILENT on system start up.
TCLT_SILENT	2 minutes	FLT	This value of TCLT is used when no transmission has been heard recently.
TCLT_TRANS_HEARD	6 minutes	FLT	This value of TCLT is set when a transmission has been heard (that is not message.INVITE) and it can be inferred that a ring exists. In this situation, it makes sense to simply wait for an invitation to be issued. This is likely to take a number of ring cycles and a longer timer is appropriate. It is likely that the ring will continue to be heard before an invite is issued, so this timer will generally get reset.

Parameter Name	Default Value	States	Notes
TSLS	n/a	SFR	Solicit Successor Timer (TSLS) - This timer is used in the self-ring-state (SFR). If it times out the station shall transit to the seeking-state (SEK). The timer shall be set with a random timeout to reduce the probability of collisions by transmissions from stations attempting to establish different rings at the same time. This timer is derived from the following algorithm Random Number in range 0 to NUM_SEK_SLOTS multiplied by SEK_SLOT_TIME
NUM_SEK_SLOTS	3	SFR	Number of asynchronous seeking slots; a parameter to calculate the TSLS timer. Collisions are not expected to be common, so a fairly small number is reasonable. This variation is included to address unusual scenarios where multiple nodes reach FLT at the same time.
SEK_SLOT_TIME	5 seconds	SFR	Time of the slots for TSLS timer. There is no slot co-ordination, so the slot should be somewhat longer than the expected message.INVITE transmission time
TSRP	n/a	SRP	Solicit Reply Timer (TSRP) - Timer used in the solicit-reply-state (SRP). A station starts its solicit-reply timer when it transits to the SRP state. If it expires a station will transit to the JON state where it replies to one of the received SLS tokens, if any, with a SET token. This timer is randomized, as it is possible that multiple nodes will hear message.INVITE and send message.JOIN in response This timer is derived from the following algorithm Random Number in range 0 to NUM_SLS_SLOTS multiplied by SLS_INTERVAL
SLS_INTERVAL	5 seconds	SRP	Time of the slots for TSRP timer. There is no slot co-ordination, so the slot should be somewhat longer than the expected message.INVITE transmission time
NUM_SLS_SLOTS	3	SRP	Number of asynchronous seeking slots; a parameter to calculate the TSRP timer. Collisions are not expected to be common, so a fairly small number is reasonable. The number of slots increases the time in SLT state, which blocks operation of the ring, so this number should not be set higher than needed.

Parameter Name	Default Value	States	Notes
TCON	n/a	JON	Contention Timer (TCON) - Timer used in the joining-state (JON). It controls the time a station waits for a response from another station following an attempt to join the network, so-named because failure to receive a response is attributed to contention with other stations attempting to join the network at the same time; a station restarts its contention timer when it goes to JON state. This needs to be tied to the TSRP timer, as the time to wait is linked to the time which the inviting node will wait before it sends. Then it needs to wait the length of a transmission. The following algorithm is used: (NUM_SLS_SLOTS +1) * SLS_INTERVAL - TSRP This approximates to the time when the inviter will send out a JOIN. There is no issue if FLT is moved to prematurely, as a received token will be handled correctly.
TIDL	n/a	IDL	Idle Timer (TIDL) - Timer used in the idle-state (IDL). It controls the time a station waits for its right-to-transmit before transiting to the floating-state (FLT). The following algorithm is based on max time for token to propagate around the ring. MAX_TX_TIME * RCL() * TIDL_FACTOR
TIDL_FACTOR	1.2	IDL	Factor to adjust TIDL timer for retransmissions.
TSLW	n/a	SLT, SEK	Solicit Wait Timer (TSLW) - This timer is used in the soliciting state (SLT) and the seeking-state (SEK) by stations inviting new ring members. When it expires, the inviting station it will update the Transmit Order List. The following algorithm is used, which reflects the algorithm used is SRP state to transmit message.INVITE: (NUM_SLS_SLOTS +1) * SLS_INTERVAL

Parameter Name	Default Value	States	Notes
TPST	10 seconds	MON	Token Pass Timer (TPST) - Used in the monitoring-state (MON). It controls the time a station waits after passing an RTT (or other) token to another station and failing to hear an implicit acknowledgement before considering the right-to-transmit as lost. It is expected that the this will happen very quickly in normal conditions. If this fails, either the token was not received or the transmission from next node is not being heard. For the former condition a short value is best. It is recommended to choose a fairly conservative value as retransmitting too soon leads to problems.
TEOT	127.5 seconds	HVT	Specifies a time to move to MON state if no EOT-Transmit event occurs.
MAX_TOKEN_PASS	3	MON	Specifies the number retransmissions of a *right-to-transmit* after which a station shall stop trying the current node and to try a different node.
MAX_NET_SIZE	255	HVT	The maximum number of nodes allowed in the ring. If there is a fixed number of nodes in operation, setting this value will save the overhead of issuing invitations when the ring is “complete”. Setting a low limit will improve ring performance, but is unfair to excluded nodes. The recommended default is to use the upper bound, which in practice means no limit.
MIN_INVITE_INTERVAL	15 minutes	HVT	Minimum interval between which a node will issues invites. Setting this value high will improve performance. Setting it low will enable new nodes to join more quickly.
MIN_INVITE_CYCLES	4	HVT	Minimum number of complete stable ring cycles between a node issuing invites. Setting this value high will improve performance. Setting it low will enable new nodes to join more quickly.
EARLY_INVITE_COUNT	2	HVT	If a node has issued this number of invites or less, use EARLY_INVITE_CYCLES
EARLY_INVITE_CYCLES	1	HVT	Minimum number of complete stable ring cycles between a node issuing invites when EARLY_INVITE_COUNT is used
RECEIVE_TABLE_EXPIRY_AGE	n/a	HVT	This sets the time when entries should be removed from the receive table. A node is expected to transmit at least once per ring cycle. This timer is tied to the maximum time for a ring cycle and a factor, which allows for a number of cycles and that cycle time may be much shorter than the maximum. The following algorithm is used MAX_TX_TIME * RCL() * RTEA_FACTOR

Parameter Name	Default Value	States	Notes
RTEA_FACTOR	3	HVT	Variable to control RECEIVE_TABLE_EXPIRY_AGE
MAX_TX_TIME	127.5 seconds	HVT, MON	Maximum Transmit Time. This is a general STANAG 5066 control parameter that can be up to 127.5 seconds. By limiting this time, ring transit times will be reduced and latency reduced. However, for bulk transfers, longer times will significantly improve throughput, particularly where only a small proportion of the nodes are sending data.
NUM_OTHER_RING_HEARD	3	Active Ring	Number of transmissions that must be heard from another ring before closing current ring. Setting this too high will delay ring merging and may lead to two nodes transmitting together. Setting it too low may lead to ring tear down when it is not viable to establish a merged ring.
OTHER_RING_TIME	10 minutes	Active Ring	Period within which NUM_OTHER_RING_HEARD transmissions from another ring must be heard before closing current ring. Considerations for setting this parameter are similar.

L.6. TOKEN AND MESSAGE TRANSMISSION

L.6.1. Transmitting Tokens

Tokens are carried in EOWs. Every D_PDU carries exactly one EOW. A token can be carried in any D_PDU which is being sent to the successor, which will include all WTRP messages being sent. STANAG 5066 may require to use EOWs for other purposes. Subject to this, it is desirable to send multiple copies of the token for reliability.

EOWs **may** be sent using the ACK_PDU or the PADDING D_PDU specified in STANAG 5066 Annex C which is a compact way to transfer an EOW. This will be necessary when no other D_PDUs are being sent to the node receiving the token. It can also enable extra token copies to be sent, which can increase resilience of token transfer.

Where possible, the Tokens should be spread out over the length of the transmission, to minimize the effects of fading and data loss.

Token use of EOWs needs to be balanced with other use of EOWs, such as EOWs used by DTS for data rate selection.

L.6.2. **Transmitting Messages**

L.4.8 specifies which messages are required to be sent. Messages are small and it may be beneficial to repeat them, particularly if a higher transmission speed is chosen. Where Table messages are critical, it is particularly desirable to duplicate.

Where messages are duplicated, it is best to have them at different parts of the transmission, rather than close together.

All messages are processed after the transmission is received, so there is no particular benefit to placing them in a particular part of the transmission.

L.6.3. **Speed and Interleaver Selection**

WTRP can be operated at fixed speed or at variable speed. This specification provides a recommended maximum transmission speed for each node. When making a transmission, the speed associated with the "slowest" node needs to be considered as maximum speed for the whole transmission. Where speed is highly variable, this may impact the choice of which D_PDUs to send. Data Rate Selection is controlled by the DTS, which is recommended to take into consideration the factors noted here.

When there is a lot of data to send, it is recommended to use the maximum transmission speed with a long or very long interleaver. This is expected to maximize throughput and reliability.

Messages will often be WTRP messages only, which are generally small. Where there is a smaller amount of data a slower speed and shorter interleaver is likely to give better performance.

When sending message.INVITE to a broadcast address, it is recommended to use a conservatively slow speed. Similarly with message.JOIN. Once communication is established, both ends can use SNR to determine an appropriate operational speed.

L.6.4. **Length of Transmission**

When a node has the token it can choose how long to transmit for, to a maximum of 127.5 seconds. Long transmissions will optimize throughput, which can give significant improvements due to the time needed to transfer tokens around the ring. However this optimization gives a cost of high latency, particularly for a large ring. This is likely to need tuning with policy across the whole ring. When transmissions speeds are higher, it may be desirable to sacrifice some throughput in order to improve latency.

L.7. CHANGES IN EDITION 4

The overall model and service provided by Annex L is unchanged. The state machine has some small changes. The protocol in Edition 4 is completely different to Edition 3 and interoperability is not a goal. This change was made because of significant problems with the protocol in Edition 3.