ERTMS/ETCS – Class 1

STM FFFIS Safe Time Layer

REF : SUBSET-056

ISSUE : 2.2.0

DATE : 2003-06-19

Company	Technical Approval	Management approval
ALCATEL		
ALSTOM		
ANSALDO SIGNAL		
BOMBARDIER		
INVENSYS RAIL		
SIEMENS		

1. MODIFICATION HISTORY

Issue Number Date	Section Number	Modification / Description	Author
0.0.1 25-11-99		First draft	J. Näsström
0.0.2 01-12-99		Update after STM meeting, see ATD_MIN_991130_WGSTM	J. Näsström
0.0.3 03-01-00		New structure of the document. • Added version control • Separate messages for time sync. and time reference. • Minor improvements in the whole document. • Editorial updates.	N. Lindström
0.0.4 25-01-00		Update after STM meeting 20-01-00 • Minor improvements in the whole document. • Editorial updates. • Removed application specific items. Application specific items shall be included in the application document. • The Sync and Reference Time messages are combined into one message. • Major improvements in Section.	N. Lindström
0.0.5 07-02-00		Update after comments after comments from the STM group • General improvements of section 7. • Minor improvements in the whole document. • Version control moved to Safe Link Layer.	N. Lindström
0.0.6 11-02-00		Added a new message 'Boadcast startup'. • Improvements on clock synchronisation using redundant buses. • Added version control of broadcast messages.	N. Lindström
1.0.0 21-02-00		Update after comments from Unisig STM meeting 17-02-00 in Stuttgart. • Moved section 'Logical Addressing' to the STM FFFIS main document. • Added version control of the Sate time Layer. • Several clarification in Section. • Added appendix about provided services.	N. Lindström
1.0.1 28-03-00		Update after comments from Unisig STM meeting 23-03-00 in Paris and CENELEC meeting 09-03-00. • Harmonised several definitions in section 3.2 with definitions in Safe Link Layer. • Removed section 4.1 (It is moved to FFFIS STM) • Added definition of disconnect_reason for Disconnect Telegram , see section 4.3 • Adaptations to the new redundancy concept. • Removed Bus Time. • Tolerance of long term drift changed from 1% to 0.1%. (purposed by CENELEC) • Improved criterion for long time drift supervision (purposed by CENELEC). • Adaptations in order to let the application disconnect and reconnect a logical Connection.	N. Lindström
2.0.0 30-03-00	Ref.	3.1.1.2, 8.1.1.3, 8.1.1.5 Final Issue to ECSAG	D. Degavre (Ed)

Issue Number Date	Section Number	Modification / Description	Author
2.0.1 23-01-02		Update after comments from Unisig STM meeting 04-12-01 in Brussels. • Some definitions moved to section 3.2. • Major changes in order to add general clarifications. • Req 7.2.1.4, 7.4.1.1, 7.4.1.2 and 7.4.1.2.1, Added.	J. Näsström
2.0.2 06-02-02		Update after comments from Unisig STM meeting 22-01-02 in Sttutgart.	B. Muñoz
2.0.3 14-03-02		Update after comments from Unisig STM meeting 14-03-02 in Berlin.	B. Muñoz
2.0.4 20-03-02		Update after pending comments from Unisig STM meeting 14-03-02 in Berlin.	B. Muñoz
2.0.5 12-04-02		Update after comments from Unisig STM meeting 12-04-02 in Stuttgart	B. Muñoz
2.0.6 2002-APR-18		Update after pending comments from Unisig STM meeting 2002-APR-14 in Stuttgart.	B. Muñoz
2.0.7 2002-MAY-14		Update after comments from Unisig STM meeting 2002-MAY-14 in Brussels.	B. Muñoz
2.0.8 2002-MAY-21		Update after pending comments from Unisig STM meeting 2002-MAY-14 in Brussels.	B. Muñoz
2.0.9 2002-JUN-11		Update after comments from Unisig STM meeting 2002-06-11 in Braunschweig.	P. Luehrs
2.0.10 2002-JUN-18		Update after pending comments from Unisig STM meeting 2002-JUN-11 in Braunschweig.	B. Muñoz
2.0.11 2002-JUN-25		Update after pending comments from Unisig STM meeting 2002-JUN-25 in Madrid.	R.Ramos
2.1.0 2002-JUN-27		Editorial Changes	R. Ramos
2.2.0 2003-06-19	5.3 2	Req. about non-final disconnection for SL 0 connections (former 5.3.1.4) deleted Table of Contents updated	P. Lührs (Siemens)

Every Time the Version Number of this document is changed the Compatibility Number shall be updated, see chapter 8.2.

2. TABLE OF CONTENTS

3. GENERAL

3.1 References

3.1.1.1 /1/ SUBSET-057 STM FFFIS Safe Link Layer.

3.1.1.2 /2/ SUBSET-059 STM Performance.

3.1.1.3 /3/ CENELEC EN50170-2.

3.2 Abbreviations

3.2.1.1 SLL Safe Link Layer /1/

3.2.1.2 STL Safe Time Layer

3.2.1.3 Profibus Process Field Bus

3.2.1.4 FDL Field Data Link /3/

3.2.1.5 ASIC Application Specific Integrated Circuit

3.2.1.6 SAP Service Access Point /3/

3.3 Scope

3.3.1.1 This document specifies the Safe Time Layer.

3.3.1.2 Together with Safe Link Layer /1/ it will provide safe communication with protection against the top hazards of communication (sequence faults, undetected data corruption, data ageing faults and authenticity faults).

3.3.1.2.1 Data ageing in the sending computer before reaching the Time Layer is not part of this document.

3.3.1.3 The Safe Time Layer also provides solutions for clock synchronisation, version control and failure monitoring.

3.3.1.4 In Appendix 12: The metaphor is a non normative text which provides an overview of the specification.

3.4 Definitions

3.4.1 General

3.4.1.1 **Error**, a deviation from the intended design which could result in unintended system behaviour or failure (CENELEC prEN 50129).

3.4.1.2 **Failure**, a deviation from the specified performance of a system. A failure is the consequence of a fault or error in the system (CENELEC prEN 50129).

3.4.1.3 **Fault**, an abnormal condition that could lead to an error in a system. A fault can be random or systematic (CENELEC prEN 50129).

3.4.1.4 **Time to failure detection**, is defined as the maximum time from the time point when an error or fault occurs until the failure is detected.

3.4.1.4.1 A **node** has one physical bus address. A node may have several logical addresses, Service Access Point, SAP.

3.4.1.5 A logical connection shall consist of one Logical connection Master and one logical connection Slave.

3.4.2 Logical connection Master

3.4.2.1 **Logical connection master** is defined as the node sending the Connect Request Telegram for a point-to-point connection in the Safe Link Layer .

3.4.2.1.1 Logical connection Master is the master of a logical connection. A logical connection has one and only one master.

3.4.3 Logical connection Slave

3.4.3.1 **Logical connection slave** of point-to-point connection is defined as the node sending the Connect Confirm telegram in the Safe Link Layer.

3.4.3.1.1 Logical connection Slave is the slave of a logical connection. A point-to-point logical connection has one and only one Logical connection Slave.

3.4.4 Multicast

3.4.4.1 A Multicast connection has one **Multicast Sender**.

3.4.4.2 A Multicast connection may have zero, one or several **Multicast Receivers**.

3.4.5 Reference Clock

3.4.5.1 There shall be one and only one node with the function **Reference Clock**. This function includes a free running physical clock as a time base and a function to transmit the value of this clock to the bus.

3.4.5.1.1 This node with the function Reference Clock is defined as the **Reference Clock node**.

3.4.5.2 The **Reference_Clock_Time** is the value of the time base (free running physical clock) inside the Reference Clock function.

3.4.5.2.1 The Reference_Clock_Time is a digital number and it is not related to universal time.

3.4.5.2.2 The Reference_Clock_Time is more advanced (higher value) than the Reference Time, due to process and transmission delays.

3.4.6 Reference Time

3.4.6.1 **Reference Time** (RefTime) is defined by the Sync And Reference Time Multicast messages on the bus and the moment of their transmission on the bus.

3.4.6.1.1 The Reference Time is the time common to all nodes on the bus.

3.4.7 Local Clock

3.4.7.1 Each bus node shall have a function called **Local Clock**. The Local Clock includes a free running physical clock as a time base and a function for synchronisation.

3.4.7.1.1 The node with the function Local Clock is defined as **Local Clock Node**.

3.4.7.1.2 Exception: There may be one node without Local Clock, but it is only possible for the node including the Reference Clock.

3.4.7.2 **Local_Clock_Time** is defined as the value of the time base inside the Local Clock, i.e. the value of the free running “physical” clock. The Local_Clock_Time as such is not synchronised with any other clock.

3.4.7.3 **Local Clock Inaccuracy (LCI)** is the inaccuracy of the physical clock in a node in relation to an ideal clock due to hardware architectural constraints (includes clock resolution, clock drift, ...).

3.4.8 Local Reference Time

3.4.8.1 **Local_Reference_Time (LocalRefTime)** is defined as the estimation of Reference Time by Local Clock based on the Local_Clock_Time and an offset determined by a synchronisation function inside the Local Clock.

3.4.8.1.1 This monitored time is used for time stamping and represents a system-wide reference time.

3.4.9 STL_TIME_STAMP

3.4.9.1 STL_TIME_STAMP is the Time stamp added to the message before transmission. The value of the time stamp is the Local_Reference_Time.

3.4.10 Transfer Times

3.4.10.1 **Sender_Static_Transfer_Time** is the static error in the time stamp at transmission time. It is the minimum value of the ageing of timestamp that always occur before message is transmitted.

3.4.10.1.1 This ageing is due to the constant delay of Safe Link Layer and other computation that always occurs in conjunction with sending a message.

3.4.10.1.2 The delay caused by token rotation time shall not be included in this constant.

3.4.10.2 **Sender_Dynamic_Transfer_Time** is the additional maximum estimated delay in the value of the ageing of time stamp at transmission time.

3.4.10.2.1 This delay is not constant. The variation of this delay may be due to processing other tasks or handling interrupts.

3.4.10.2.2 The delay caused by token rotation time shall not be included in this constant.

3.4.10.2.3 Note: The minimum **Transfer Time** at the sender side is the value of Sender_Static_Transfer_Time and the maximum is Sender_Static_Transfer_Time + Sender_Dynamic_Transfer_Time.

3.4.10.3 Note: The Sender_Static_Transfer_Time and the Sender_Dynamic_Transfer_Time are used for the calculation of the maximum and minimum allowed age of a message, to ensure that a message which exceeds the time limit is rejected by the STL.

3.4.10.4 **Static_Bus_Transfer_Time** is the value of the time to transfer the minimum frame (One byte of application data).

3.4.10.4.1 Note:This value depends on the baud rate of the bus.

3.4.10.5 **Dynamic_Bus_Transfer_Time** is the value of the time to transfer twice the maximum frame (One re-transmission) plus Token Rotation Time.

3.4.10.5.1 Note:This value depends on the baud rate of the bus.

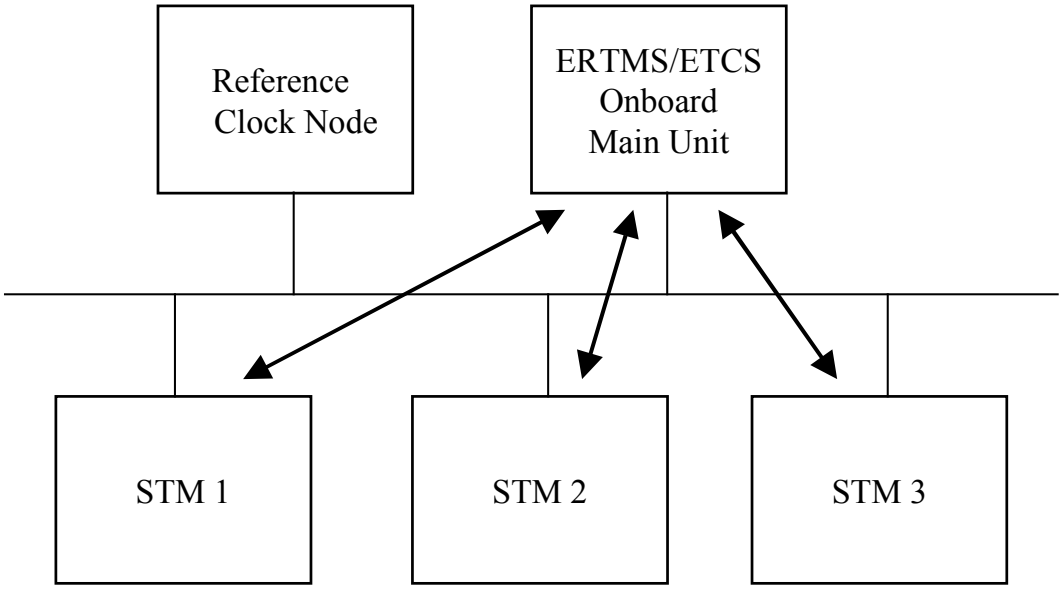
3.4.10.6 The **Safety Time Tolerance** STT is the tolerance of age at reception of safe messages.

4. SUMMARY DESCRIPTION

4.1.1.1 In the Safe Time Layer the main tasks are to monitor the local clocks of the two communicating bus nodes, and to monitor and control the age of the messages.

4.1.1.2 Through the network of nodes using the Safe Time Layer, all local clocks are monitoring each other, directly or indirectly.

4.1.1.3 Example for an STM architecture:



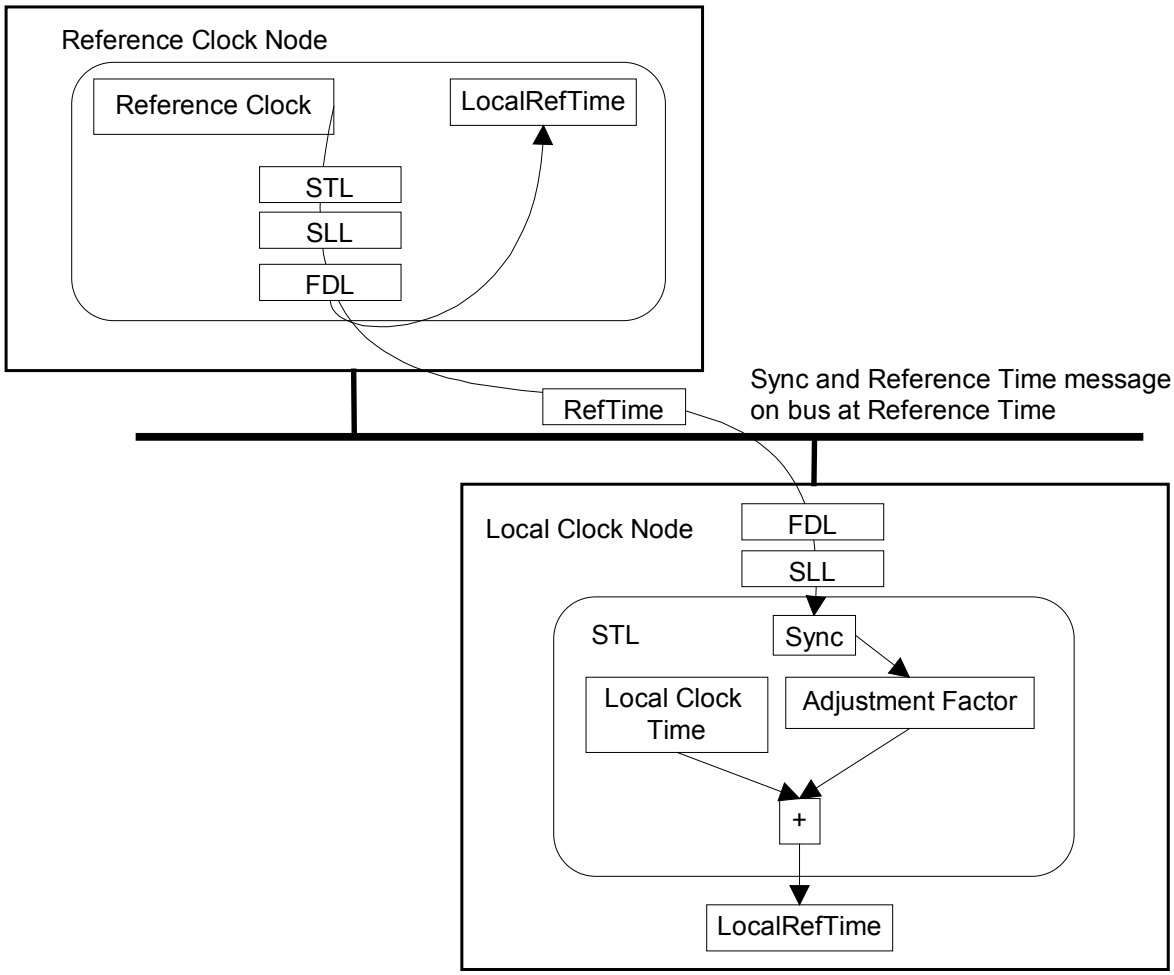
4.1.1.4 In this example the Reference Clock Node is separated,and has no safe link to any other node. (In a real system it would probably have a link to the ERTMS/ETCS On-board, but this link would be outside STM FFFIS, so this picture is valid for such a configuration.)

4.1.1.5 The STL_Time_Stamp is added to the sent application messages in order to monitor and control the ageing of the message.

4.1.1.6 The STL_Time_Stamp is added to the sent message as late as possible and has the value of Local_Reference_Time soon before transmission.

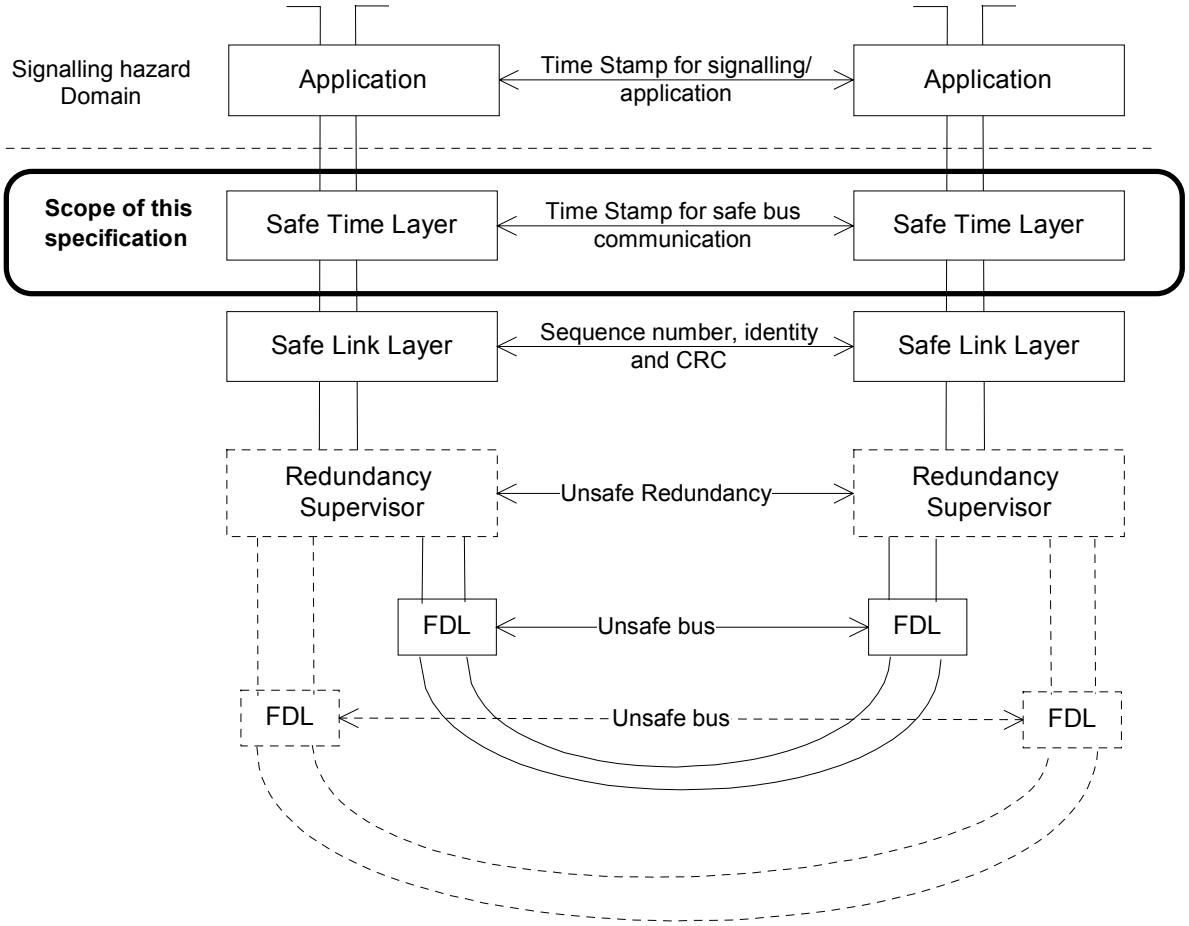
4.1.1.7 Receiving node checks the received STL_Time_Stamp. Its age shall be within a certain window Safety Time Tolerance, STT (see Section 7.8).

4.1.1.8 Reference Time Transfer



4.2 Time stamps

4.2.1.1 There are time stamps on two different levels. The time stamps on the Safe time layer (STL_Time_Stamp) keep track of the maximum time delay of data sent via the bus. The time stamps on the signalling/application layer are used to keep track on the ageing of the signalling/application data in a wider perspective.



5. SAFE TIME LAYER CONTROL

5.1 Start-up of Safe Time Layer

5.1.1.1 The Reference Clock Node starts to multicast Sync and Reference Time messages as soon as possible after the node has been powered on. This provides Reference Time (RefTime) to all other nodes.

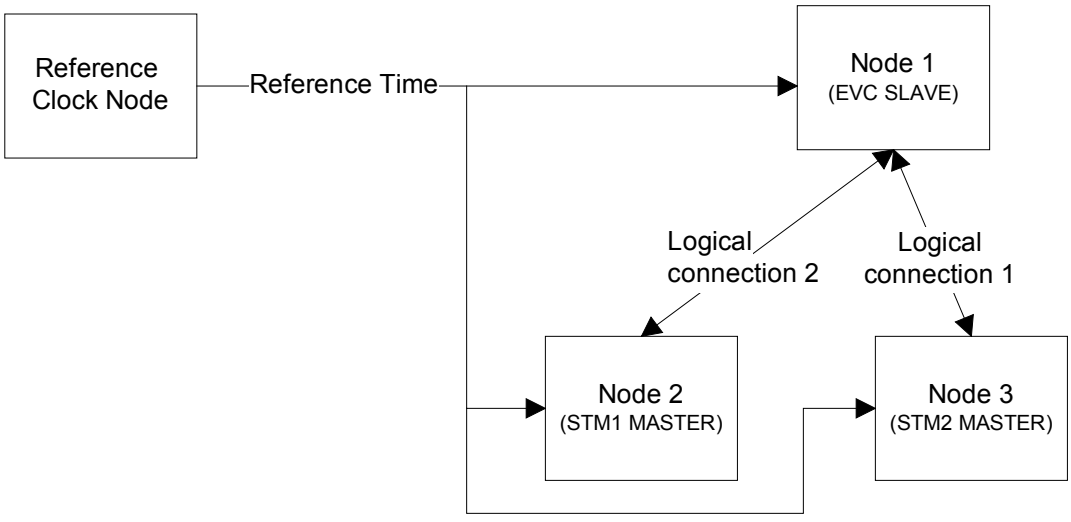
5.1.1.2 LocalRefTime is derived from the Local_Clock_Time and the received Reference Time by the Local Clock Node.

5.1.1.3 During rest of session, the Local_Reference_Time is checked and adjusted against the Reference Time.

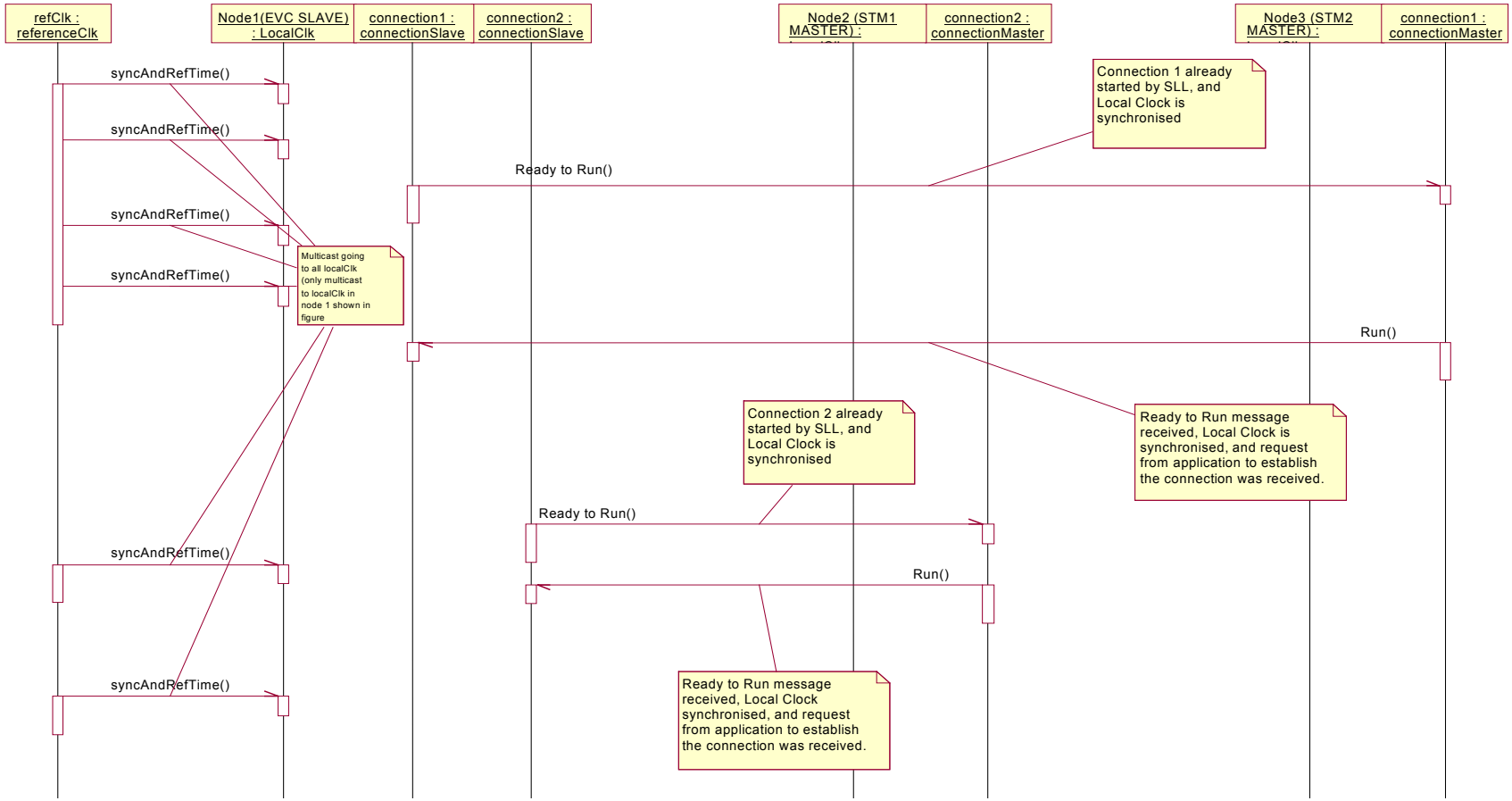
5.1.1.4 When the Local_Reference_Time is synchronised relative to the Reference Time, the Logical connection Slave and the Logical connection Master can be initiated. This is done by issuing Ready to Run message responded by a Run message (Ready to Run and Run messages shall be issued only after the coresponding connection is established by the Safe Link Layer).

5.1.1.5 An example of a successful start-up sequence of the Safe Time Layer is shown in the two figures below. The architecture is shown in 5.1.1.6 and a sequence chart for the architecture in 5.1.1.7.

5.1.1.6



5.1.1.7



5.2 Configuration data of Safe Time Layer

5.2.1.1 Connect Request Telegram and Connect Confirm Telegrams in the Safe Link Layer include fields for configuration data of the Safe Time Layer.

5.2.1.2 The Configuration Data Field is specified in the table below:

5.2.1.3

Variable	Length	Comment
Safe Time Layer version no X	8	Compatibility Number X.Y.Z implemented on the node.
Safe Time Layer version no Y	8	Compatibility Number X.Y.Z implemented on the node.
Safe Time Layer version no Z	8	Compatibility Number X.Y.Z implemented on the node.
Sender_Dynamic_Transfer_Time	32	The Sender_Dynamic_Transfer_Time of the sender (signed integer, complement to two code) [ms].
Sender_Static_Transfer_Time	32	The Sender_Static_Transfer_Time of the sender (signed integer, complement to two code) [ms].

5.3 Disconnect Reason

5.3.1.1 Detected failures in the Safe Time Layer shall result in an issue of a Disconnect request to the Safe Link Layer.

5.3.1.2 The disconnect_reason for the Disconnect request to Safe Link Layer issued by the Safe Time Layer shall be set according to the table below

5.3.1.3

#disconnect_reason		Description
20h	Final	Bad version error.
21h	Non-Final	Local Clock is not yet synchronised
22h	Final	Local Clock; lost synchronisation see chapter 7.4.1.5
23h	Non-Final	Logical Connection Master; time-out at connection start-up
24h	Non-Final	Logical Connection Slave; time-out at connection start-up
25h	Non-Final	STL_Time_Stamp of a received telegram does not fulfill the STTmin criterion
26h	Non-Final	STL_Time_Stamp of a received telegram does not fulfill the STTmax criterion
27h-3Eh		Not used (reserved for future use)
3Fh		Other (not in this list) disconnect_reason of Safe Time Layer
40h	Final or Non-Final	Disconnect on request by application

5.4 STL_Time_Stamp

5.4.1.1 The STL_Time_Stamp is a 32 bit time stamp added at the end of the message.

5.4.1.2 Resolution is 1 ms.

5.4.1.2.1 Note: The resolution and the number of bits leads to a time to wrap around of 4294967296 ms, which is every 49 days, 17 hours, 2 minutes, 47 seconds and 296 miliseconds.

5.4.1.3 All time stamps transferred over the bus shall be Local_Reference_Time or RefTime.

6. MESSAGES

6.1.1.1 **The format of the data added by the Safe Time Layer is the little endian format low-order byte first**; i.e. 16- and 32-bit values for example are transferred bytewise, and the bytes are transferred as follows:

16-bit values: lowbyte, highbyte

32-bit values: lowword.lowbyte, lowword.highbyte, highword.lowbyte, highword.highbyte

6.1.1.2 The table below specifies the command number in the Safe Time Layer for the messages.

6.1.1.3

Type	Value	Message
SL-4 messages	89h	Application Data
	A1h	Sync and Reference Time
	A2h	Ready to Run
	A3h	Run
	A4h	Safe Time Layer Startup
	A0h, A5h..BFh	Reserved for future extension
SL-2 messages	09h	Application Data
	22h	Ready to Run
	23h	Run
	20h, 21h, 24h..3Fh	Reserved for future extension
SL-0 messages	C9h	Application Data
	E2h	Ready to Run
	E3h	Run
	E0h,E1h,E4h.. FFh	Reserved for future extension

6.1.1.4 These commands numbers have to be passed to the lower layer (e.g. SLL).

6.1.1.5 Note: These command numbers are part of the Safe Link Layer commands table, see reference /1/.

6.2 Application Data

6.2.1.1 Application data shall be used by the application layer to transmit multicast data telegrams or point-to-point data telegrams.

6.2.1.2

Description	Application Data		
Transmission media	STM FFFIS		
Content	**Variable**	**Length**	**Comment**
	Application data	N*8	ERTMS language packets etc.
	STL_Time_Stamp (LocalRefTime)	32	Time stamp of the message with the Local_Reference_Time (unsigned integer) [ms].

6.3 Sync and Reference Time

6.3.1.1 The Sync and Reference messages shall be sent by the Reference Clock using Multicast.

6.3.1.2 The transmission of the Sync and Reference Time messages shall start as soon as possible after the node has been powered on.

6.3.1.3 The Reference sync number is a counter value that increments by one for every sent message of the Sync and Reference Time message.

6.3.1.4 The initial value for Reference Sync number shall be 0h.

6.3.1.5 Wrap around is not allowed for the Reference sync number.

6.3.1.5.1 Note: With a cycle time of 250ms a wrap around is reached after 24 years and 17 days.

6.3.1.6 The time of sending a Sync and Reference Time message on the bus is captured by the Reference Clock Node as accurately as possible. This captured time of sending defines the Reference Time when the message was on the bus.

6.3.1.7 The captured time is distributed to all other nodes with the following Sync and Reference Time message as the parameter Reference Time [n-1].

6.3.1.8 The Reference Time of the first message sent from the reference clock shall be set to zero. The local clocks shall not use the Reference Time value of this first message.

6.3.1.8.1 Note: This is because the Reference Time parameter does not refer to a previous Sync and Reference Time message, as it is the first one.

6.3.1.9

Description	Sync and Reference Time		
Transmission media	STM FFFIS		
Content	**Variable**	**Length**	**Comment**
	Safe Time Layer version no X	8	The node implement compatibility number **X**.Y.Z.
	Safe Time Layer version no Y	8	The node implement compatibility number X.**Y**.Z
	Safe Time Layer version no Z	8	The node implement compatibility number X.Y.**Z**.
	Reference Sync [n]	32	The messages number. (unsigned integer).
	Reference Time [n-1]	32	Captured time of previous message see 6.3.1.6 and see also 7.2 (unsigned integer) [ms].

6.4 Ready to Run

6.4.1.1 The Ready to Run message is a point-to-point message.

6.4.1.2 The Ready to Run message shall be sent by the logical connection slave and only when the Local Clock is synchronised.

6.4.1.3 The Ready to Run message shall only be sent when a new connection is established by the Safe Link Layer.

6.4.1.4

Description	Ready to Run		
Transmission media	STM FFFIS		
Content	**Variable**	**Length**	**Comment**
	STL_Time_Stamp (LocalRefTime)	32	Time stamp of the message with the Local_Reference_Time. (unsigned integer) [ms].

6.5 Run

6.5.1.1 The Run message is a point-to-point message.

6.5.1.2 The Run message shall be sent by a logical connection master when the Local Clock is synchronised and the logical connection master has got the Ready to Run message from the logical connection slave.

6.5.1.3 No Application Data messages shall be sent by the Logical Connection Master before a Run message has been sent via the logical connection.

6.5.1.4 No Application Data messages shall be sent by the Logical Connection Slave before a Run message has been received by the logical connection.

6.5.1.5

Description	Run		
Transmission media	STM FFFIS		
Content	**Variable**	**Length**	**Comment**
	STL_Time_Stamp (LocalRefTime)	32	Time stamp of the message with the Local_Reference_Time. (unsigned integer) [ms].

6.6 Safe Time Layer Startup for multicast

6.6.1.1 The Safe Time Layer Startup messages shall be sent by all multicast senders using Multicast, except for the Reference Clock Function.

6.6.1.1.1 Note: For Reference Clock Function there is a specific message for time synchronisation (see chapter 6.3).

6.6.1.2 The Safe Time Layer Start-up message shall be sent cyclically during start up time. This message is also sent after start up time to allow receivers to get the message when they are ready for it (they have started up).

6.6.1.3 The interval for the messages shall be inside SafeTimeLayerStartupInterval. This is applicable until StartupSynchronisationTimeLimit.

6.6.1.4 After expiration of StartUpSynchronisationTimeLimit the interval shall be inside SafeTimeLayerReStartInterval.

6.6.1.5

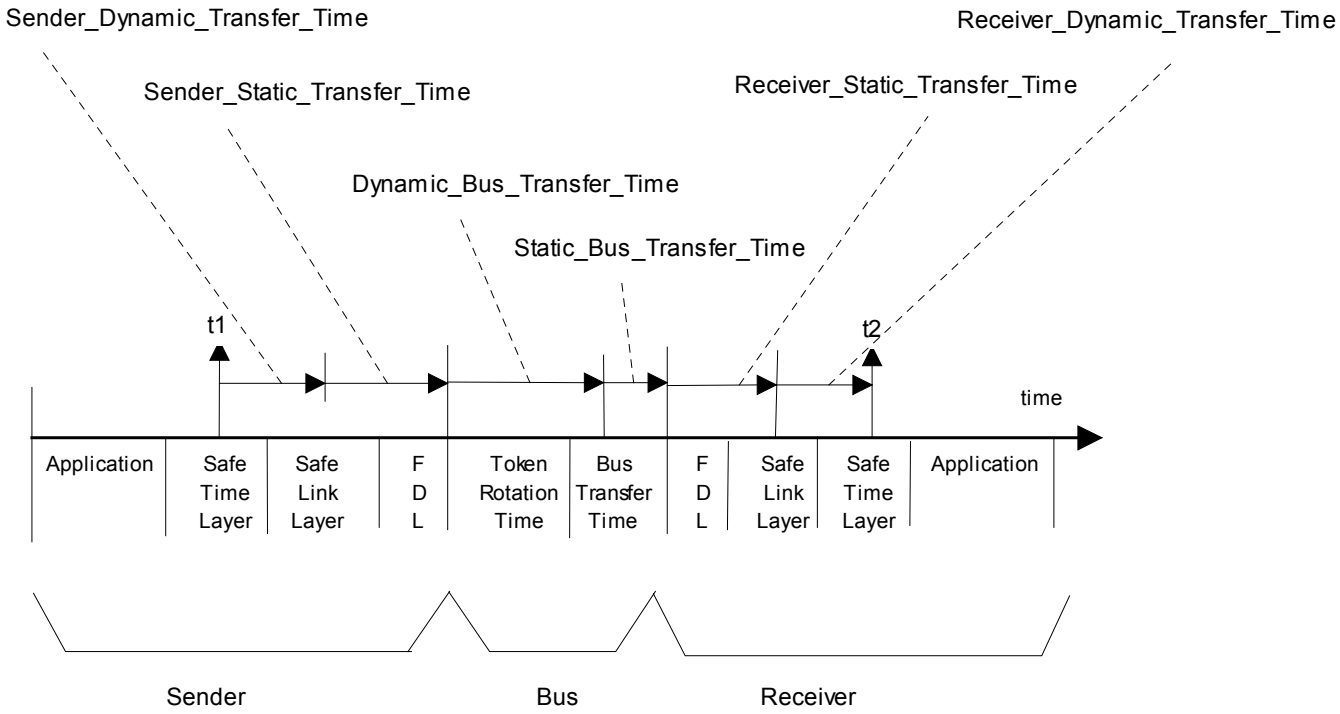
Description	Safe Time Layer Startup		
Transmission media	STM FFFIS		
Content	**Variable**	**Length**	**Comment**
	Safe Time Layer version no X	8	The node implement compatibility number **X**.Y.Z.
	Safe Time Layer version no Y	8	The node implement compatibility number X.**Y**.Z
	Safe Time Layer version no Z	8	The node implement compatibility number X.Y.**Z** .
	Sender_Dynamic_Transfer_Time	32	The Sender_Dynamic_Transfer_Time of the sender (signed integer, complement to two code) [ms].
	Sender_Static_Transfer_Time	32	The Sender_Static_Transfer_Time of the sender (signed integer, complement to two code) [ms].

7. SPECIFICATION OF FUNCTIONS

7.1 GENERAL

7.1.1.1 The figure below illustrates time stamping of a transferred message performed by the sender and receiver.

7.1.1.2



7.1.1.2.1 t1 is STL_Time_Stamp (or RefTime[n-1] in case of a Sync and Reference Time message).

7.1.1.2.2 t2 is Local_Time_Stamp (or LocalTime[n-1] in case of a Sync and Reference Time message).

7.1.1.2.3 The time stamping itself (on the senders side) is not a safety function. However the monitoring of the time stamp (on receivers side) is part of the safety process.

7.1.1.2.4 If time stamping is performed by the Safe Link Layer or interrupt driven by the bus controller , the static and dynamic transfer times may be reduced.

7.1.1.3 Wrap around is allowed for any clock value or time stamp.

7.1.1.3.1 Justification: Wrap around in time stamping can occur every 2^{32} ms, and message chronology is guaranteed by the sequence number.

7.2 Principle of Clock Synchronisation

7.2.1.1 In order to solve the hazard of 'missing time deadlines of safety critical data transferred between physical nodes' some mechanism to synchronise the clocks or translate timestamps from one clock to another clock must be introduced.

7.2.1.2 The principle shall be translation of timestamps from local time bases to a common time base.

7.2.1.3 Synchronisation shall also take care of clock drift, defined as a difference between LocalTime and Reference Clock over a period of time.

7.2.1.4 The clock drift between the clocks shall be supervised. Low rates of drift shall be compensated. High rates of drift shall be detected as failure and lead to disconnection.

7.2.1.5 The Reference Clock shall cyclically multicast the Reference Time to the bus.

7.2.1.6 All Local Clocks shall cyclically calculate the difference between the Local_Clock_Time and the received Reference Time (RefTime) by determining the Adjustment Factor. See 7.4.4 Calculation of Adjustment Factor.

7.2.1.7 All Local Clocks shall supervise the clock drift between Local_Clock_Time and the Reference Time. This shall be done according to section 7.4.3 Resynchronise.

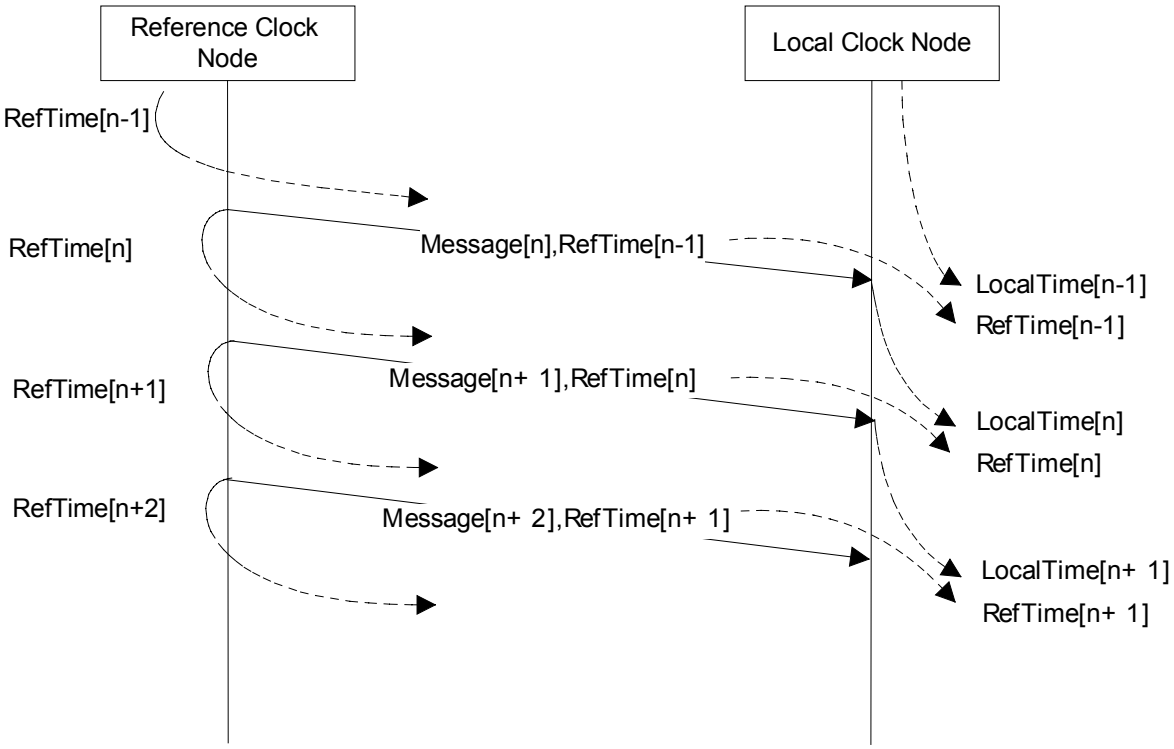
7.2.1.7.1 If the Local Clock finds the clock drift out of specified tolerance the node shall isolate itself. This shall be done according to section 7.4.1.

7.2.1.8 Functions allocated in the same node as Reference Clock may potentially benefit from increased time accuracy, if needed.

7.2.1.9 All nodes shall fulfill the specified timing requirements in reference /2/.

7.2.1.10 The figure below illustrates how the Reference Clock Node transfers the Reference Time to a Local Clock Node.

7.2.1.11



7.2.1.12 The Reference Clock shall save the Reference_Clock_Time as RefTime[n] when sending the Message[n].

7.2.1.13 The Local Clock shall save the Local_Clock_Time as LocalTime[n] at reception of Message[n].

7.2.1.14 The Reference Clock shall include RefTime[n] in Message[n+1].

7.2.1.15 The figure above shows that the Local Clock Node has got the Reference_Clock_Time value RefTime[n] and its own value LocalTime[n] for the common event, transferring of Message[n], after a delay of one message.

7.2.1.16 The Local clock is now able to calculate the time difference between the Reference_Clock_Time and the Local_Clock_Time .

7.2.1.17 Since the Reference Clock cyclically sends its time, the Local Clock is able to supervise the difference history. The clock drift supervision is one of them.

7.2.1.18 Note: The method allows to minimise the error in delay on the sender side by capturing the RefTime[n] as close as possible to the time of transmission (after the CRC is calculated).

7.2.1.19 Note: If measuring time of transmission from the bus controller, the error introduced by the token rotation time delay can be eliminated.

7.3 Reference Clock

7.3.1.1 The Reference Clock shall start to send Sync and Reference Time (multicast) messages as soon as possible after the node has been powered on.

7.3.1.1.1 Note: This is independent from the initialisation of the point to point connections by the Safe Link Layer.

7.3.1.2 The Reference Clock shall cyclically send Sync And Reference Time messages.

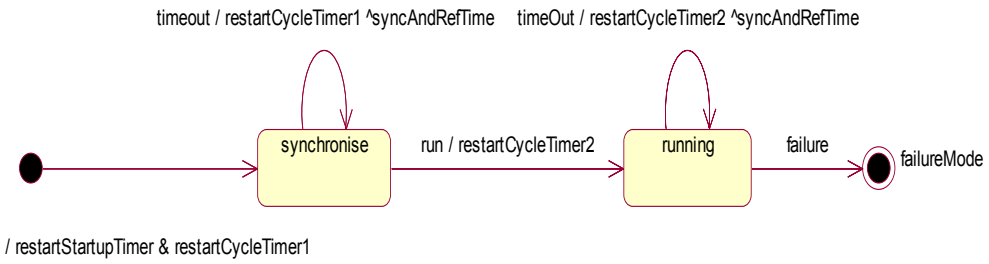
7.3.1.3 During synchronisation of the Safe Time Layer the Reference Clock sends Sync and Reference Time Messages with a short interval.

7.3.1.4 The interval for the messages shall be inside SyncAndRefTimeSyncInterval. This is applicable until SyncAndRefTimeStartupTimeLimit.

7.3.1.5 After expiration of SyncAndRefTimeStartupTimeLimit the interval becomes longer, and shall be inside SyncAndRefTimeRunInterval.

7.3.1.6 The state machine below specifies the behaviour of the Reference Clock.

7.3.1.7



7.3.1.7.1 States are:

7.3.1.7.1.1 Synchronise (state): send Sync and Reference Time messages with a short interval.

7.3.1.7.1.2 Running (state): send Sync and Reference Time messages with a long interval.

7.3.1.7.1.3 FailureMode (state): stop sending SyncAndReftime. Receivers will detect the failureMode after timeout.

7.3.1.7.2 Events are:

7.3.1.7.2.1 TimeOut (event): expiration of the Cycle Timer.

7.3.1.7.2.2 Run (event): expiration of the Startup Timer.

7.3.1.7.2.3 Failure (event): the event of failure in the RefTime node.

7.3.1.7.3 Actions are:

7.3.1.7.3.1 RestartStartupTimer (action): set a timer to SyncAndRefTimeStartupTimeLimit .

7.3.1.7.3.2 restartCycleTimer1 (action): set a timer to a value inside SyncAndRefTimeSyncInterval.

7.3.1.7.3.3 restartCycleTimer2 (action): set a timer to a value inside SyncAndRefTimeRunInterval.

7.3.1.7.4 Send-clauses are:

7.3.1.7.4.1 SyncAndRefTime (send-clause): multicast a Sync and Reference Time Message.

7.4 Local Clock

7.4.1 General

7.4.1.1 The Local Clock shall not provide Local_Reference_Time before it is synchronised.

7.4.1.2 Local Clocks shall check that its drift compared to Reference Time is not outside limits.

7.4.1.3 A Local Clock calculates the Adjustment Factor and Supervises the difference in time between Local Time and Reference Time.

7.4.1.4 If a node contains the Reference Clock, a Local Clock is not required on the node.

7.4.1.5 A Local Clock is synchronised as long as:

7.4.1.5.1 An adjustment factor can be calculated, see chapter 7.4.4.

7.4.1.5.2 Reference Time is a monotonic increasing value, see chapter 7.4.3.3.

7.4.1.5.3 Local Time is a monotonic increasing value, see chapter7.4.3.4.

7.4.1.5.4 The short term drift between the local clock and the reference clock is inside limits, see chapter 7.4.3.5.

7.4.1.5.5 A valid number of Sync And Reference Time messages have been received, see chapter 7.4.3.6.

7.4.1.5.6 The long term drift between the local clock and the reference clock is inside limits, see chapter 7.4.3.7.

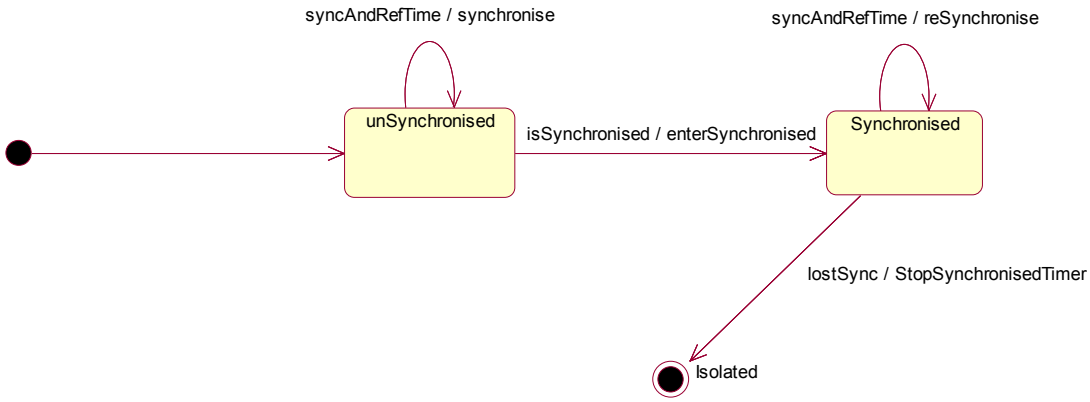
7.4.1.5.7 A resynchronisation is performed within a specified time, see chapter 7.4.1.6.

7.4.1.6 The maximum time interval between two successfully executed resynchronisations (see chapter 7.4.3) shall be limited.

7.4.1.6.1 Maximum time between two successfully executed resynchronisations shall be localClockMaxReSyncInterval.

7.4.1.7 The state machine below specifies the behaviour of a Local Clock:

7.4.1.7.1



7.4.1.7.2 States are:

7.4.1.7.2.1 unSynchronised (state): Local Clock is not synchronised.

7.4.1.7.2.2 Synchronised (state): Local Clock is synchronised.

7.4.1.7.2.3 Isolated (state): failure state, Local Clock is not synchronised.

7.4.1.7.3 Events are:

7.4.1.7.3.1 SyncAndRefTime (event): reception of a Sync and Reference Time Message from the Reference Clock.

7.4.1.7.3.2 IsSyncronised (event): this event occurs when the Local Clock is synchronised.

7.4.1.7.3.3 LostSync (event): event of failure in synchronisation that occur when the Synchronised Timer expires, or any of the conditions specified in 7.4.3 fail.

7.4.1.7.4 Actions are:

7.4.1.7.4.1 Synchronise (action): performs synchronising of the Local Clock, specified in Section 7.4.2.

7.4.1.7.4.2 EnterSynchronised (action): Reset a timer called Synchronised Timer.

7.4.1.7.4.3 ReSyncronise (action): performs resynchronising of the Local Clock, specified in Section 7.4.3 and restart the Synchronised Timer.

7.4.1.7.4.4 StopSyncronisedTimer (action): stop the Synchronised Timer and disconnect all logical connections on the node, including multicast (isolated).

7.4.2 Synchronise

7.4.2.1 Synchronise calculate the first valid Adjustment factor (see chapter 7.4.4).

7.4.3 Resynchronise

7.4.3.1 Resynchronise calculate adjustment factor (see chapter 7.4.4)

7.4.3.2 The following actions shall be performed at reception of a valid Sync and Reference Time Message (Message[n+1] in figure 7.2.1.11). If one of the actions fails the Local clock is unsynchronised and Local_Reference_Time is undefined.

7.4.3.3 Check that refTime[n] > refTime[n-1]

7.4.3.4 Check that localTime[n] > localTime[n-1]

7.4.3.5 Check that |refTime[n] – localTime[n] – adjustmentFactor| < MaxClockInaccuracyAfterAdjustFactor

7.4.3.6 The number of valid received sync and reference time messages shall be at least MinNumberofSyncAndRefMsgReceived during a TimeForLongTermDriftCheck time frame. The time frame is not floating, i.e. the start and end times of the time frame are set at one point and is not changed until the end time has been passed

7.4.3.7 The long term drift between the local clock and the reference clock shall be within 0.1%:

$$0 \leq |adjustmentFactor[n] - adjustmentFactor[n-k]|/TimeForLongTermDriftCheck \leq 0.001$$

7.4.3.7.1 The index value n is a sequence number indicating the event of Sync And Reference Time telegram reception. The index value K is a sequence number that corresponds to a Sync and Reference Time mesage received no more than TimeForLongTermDriftCheck before the time that corresponds to the index n. The time frame is not floating.

7.4.3.7.2 The check is performed over a non-floating timeslot of TimeForLongTermDriftCheck.

7.4.3.7.2.1 Note: The time to failure detection is therefore 2*TimeForLongTermDriftCheck.

7.4.4 Calculation of Adjustment Factor

7.4.4.1.1 The Adjustment Factor is calculated as the difference between the local time recorded by Local clock and the reference time from the corresponding valid telegram. The adjustment value shall be calculated as the running mean value.

7.4.4.2 The Adjustment Factor shall be calculated as:

$$AdjustmentFactor \leftarrow \frac{1}{16}\sum_{16} (refTime[N] - localTime[N])$$

7.4.4.2.1 If a Sync and Reference Time Message is lost on the bus the next one shall be used in the formula.

7.4.4.2.2 An Adjustment factor is not defined for less than 16 pairs of refTime and localClockTime values.

7.4.5 Calculation of Local_Reference_Time

7.4.5.1 The Local_Reference_Time is the adjusted Local_Clock_Time to estimate the Reference Time on the bus. The Adjustment Factor is used to make that adjustment:

LocalReferenceTime ← AdjustmentFactor + LocalClockTime

7.5 Principle of Logical Connection

7.5.1 General for Point-to-point connection

7.5.1.1 Logical connection shall be used for transferring safety related application data.

7.5.1.2 A logical connection shall be unique by its SAP and physical addresses.

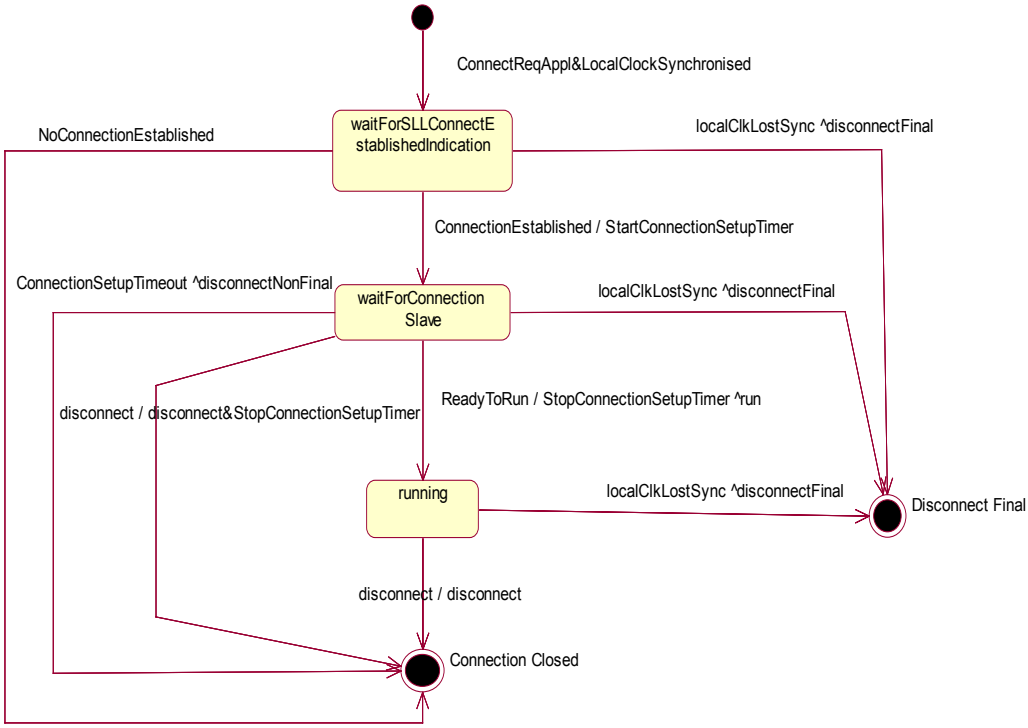
7.6 Logical Connection Master

7.6.1.1 This chapter specifies the behaviour of a Logical connection Master at the interface between the connected node:

7.6.1.1.1 Note: Every Logical connection needs a separate state machine.

7.6.1.2 If the Master Local Clock is not synchronised no Connection request shall be issued to the Safe Link Layer.

7.6.1.3



7.6.1.3.1 States are:

7.6.1.3.1.1 waitForSLLConnectEstablishedIndication(state): Waiting for the logical connection to be established by the SLL.

7.6.1.3.1.2 WaitForConnectionSlave (state): Wait for Ready to Run message from slave.

7.6.1.3.1.3 Running (state): connection opened.

7.6.1.3.1.4 Connection closed (state):The connection is closed, the connection instance is cancelled, but may be setup again.

7.6.1.3.1.5 Disconnect Final (state): connection finally closed, no re-connection possible.

7.6.1.3.2 Events are:

7.6.1.3.2.1 ConnectReqAppl&LocalClockSynchronised (event) : Local Clock is synchronised and Request from application to connect was received.

7.6.1.3.2.2 ConnectionSetupTimeout (event): Timeout of ConnectionSetupTimer.

7.6.1.3.2.3 ReadyToRun (event): reception of Ready To Run message from slave.

7.6.1.3.2.4 disconnect (event): reception of a Disconnect Telegram, or disconnect is requested by application.

7.6.1.3.2.5 ConnectionEstablished (event): Logical connection established by the SLL.

7.6.1.3.2.6 localClkLostSync (event): Local Clock in the node get unsynchronised.

7.6.1.3.2.7 NoConnectionEstablished (event): Reception of an indication from the SLL, indicating that the logical connection could not be opened.

7.6.1.3.3 Actions are:

7.6.1.3.3.1 StartConnectionSetupTimer (action): set ConnectionSetupTimer to ConnectionSetupTimeLimit.

7.6.1.3.3.2 StopConnectionSetupTimer (action): Stop ConnectionSetupTimer.

7.6.1.3.3.3 disconnect (action): A “Disconnect Telegram” is sent when requested by the application or, when a “Disconnect Message” is received, the status “Disconnected” is made visible for the application.

7.6.1.3.4 Send-clauses are:

7.6.1.3.4.1 run (send-clause): send a Run Message.

7.6.1.3.5 disconnectFinal (send-clause): send a (final) Disconnect Telegram to the Logical Connection Slave connected to the Node.

7.6.1.3.6 disconnectNonFinal (send-clause): send a (Non-final) Disconnect Telegram to the Logical Connection Slave connected to the Node.

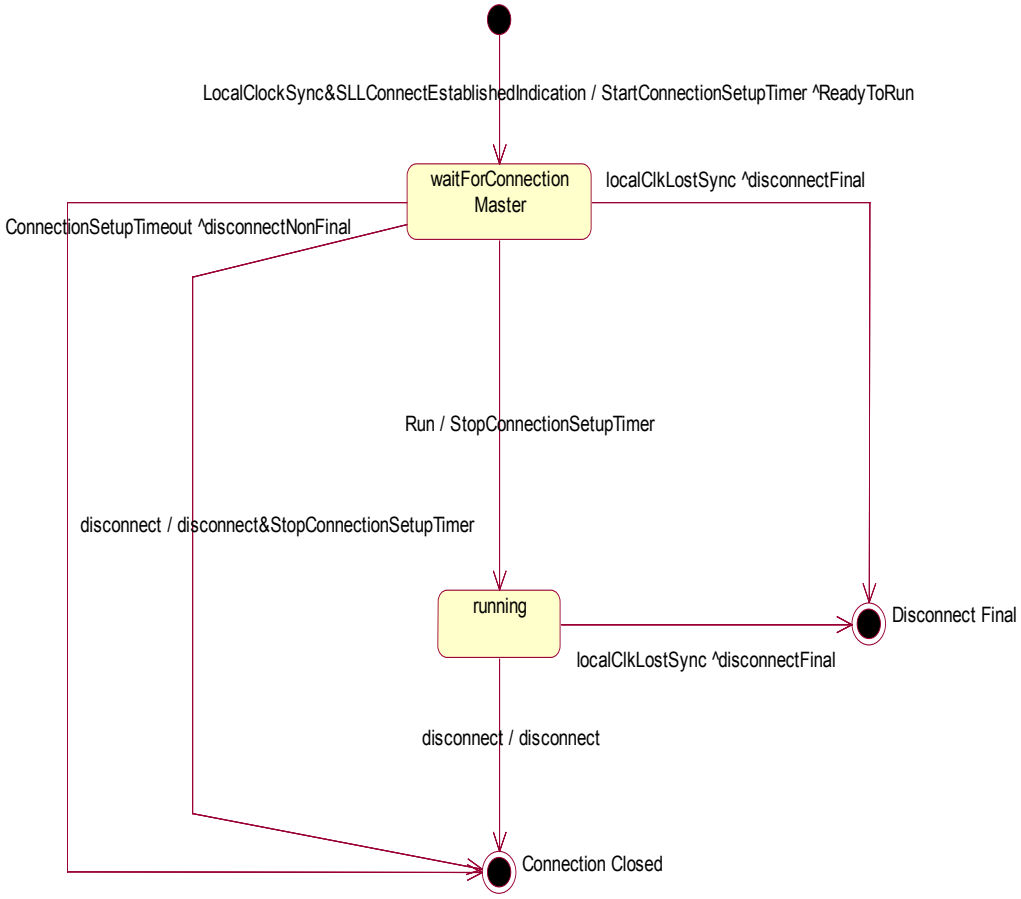
7.7 Logical Connection Slave

7.7.1.1 This chapter specifies the behaviour of a Logical connection Slave at the interface between the connected nodes.

7.7.1.1.1 Note: Every Logical connection needs a separate state machine.

7.7.1.2 If the Logical Connection Slave receives an indication that the connection has been established by the Safe Link Layer, and the Local Clock is not synchronised then a non-final Disconnect Telegram shall be sent.

7.7.1.3



7.7.1.3.1 States are:

7.7.1.3.1.1 WaitForConnectionMaster (state): Wait for Run message from Master.

7.7.1.3.1.2 Running (state): Connection opened.

7.7.1.3.1.3 Connection closed (state):The connection is closed, the connection instance is cancelled, but may be setup again.

7.7.1.3.1.4 Disconnect Final (state): connection finally closed, no re-connection possible.

7.7.1.3.2 Events are:

7.7.1.3.2.1 LocalClockSync&SLLConnectEstablishedIndication (event): Local Clock is synchronised, and logical connection established by SLL.

7.7.1.3.2.2 ConnectionSetupTimeout (event): Timeout of ConnectionSetupTimer.

7.7.1.3.2.3 run (event): Receive a Run Message.

7.7.1.3.2.4 LocalClkLostSync (event): Local Clock in the node gets unsynchronised.

7.7.1.3.2.5 disconnect (event): reception of a Disconnect Telegram from Master, or disconnect request from application.

7.7.1.3.3 Actions are:

7.7.1.3.3.1 StartConnectionSetupTimer (action): set a timer to ConnectionSetupTimeLimit.

7.7.1.3.3.2 StopConnectionSetupTimer (action): Stop ConnectionSetupTimer.

7.7.1.3.3.3 disconnect (action): A “Disconnect Telegram” is sent when requested by the application or, when a “Disconnect Message” is received, the status “Disconnected” is made visible for the application.

7.7.1.3.4 Send-clauses are:

7.7.1.3.4.1 ReadyToRun (send-clause): send a Ready to Run Message.

7.7.1.3.4.2 disconnectFinal (send-clause): send a (final) Disconnect Telegram to the Logical Connection Master connected to the Node.

7.7.1.3.4.3 disconnectNonFinal (send-clause): send a (Non-final) Disconnect Telegram to the Logical Connection Master connected to the Node.

7.8 Time Validation of Received Messages

7.8.1.1 This procedure relates both to reception of point-to-point messages and multicast messages.

7.8.1.2 The following messages shall be validated:

7.8.1.2.1 Ready To Run message (point-to-point)

7.8.1.2.2 Run message (point-to-point)

7.8.1.2.3 Application messages (point-to-point and multicast)

7.8.1.3 This procedure does not apply during clock initialisation.

7.8.1.4 During clock initialising only the processing of Sync and Reference Time Messages and Safe Time Layer Startup messages for Multicast is allowed.

7.8.1.5 In order to monitor and control the age of a received message the received STL_Time_Stamp shall be validated to the conditions in 7.8.1.14.

7.8.1.6 The value of STTmin and STTmax shall be calculated for each logical connection.

7.8.1.6.1 Justification: Senders may have different performance, or the application data has different requirements.

7.8.1.7 STTmin = Static_Transfer_Time – LCI

7.8.1.8 STTmax = Static_Transfer_Time + Dynamic_Transfer_Time + LCI

7.8.1.9 Note: LCI is regarded in the respective receiving node.

7.8.1.10 Static_Transfer_Time = Sender_Static_Transfer_Time + Receiver_Static_Transfer_Time + Static_Bus_Transfer_Time

7.8.1.11 Dynamic_Transfer_Time = Sender_Dynamic_Transfer_Time + Receiver_Dynamic_Transfer_Time + Dynamic_Bus_Transfer_Time

7.8.1.12 The transfer time delays can be positive or negative. Calculation shall be able to handle positive and negative result.

7.8.1.13 Note: The Sender_Static_Transfer_Time and Sender_Dynamic_Transfer_Time are sent to the receiver at startup of the Safe Time Layer.

7.8.1.14 Conditions are:

7.8.1.14.1 Minimum time age: Local_Time_Stamp - STL_Time_Stamp > STTmin

7.8.1.14.2 Maximum time age: Local_Time_Stamp - STL_Time_Stamp < STTmax

7.8.1.14.3 Local_Time_Stamp is the value of LocalRefTime at reception of a telegram.

7.8.1.14.4 If the conditions are not met, there is a time error, so the message will be rejected and Non-Final disconnection shall be performed for this connection.

8. CONFIGURATION MANAGEMENT

8.1 General

8.1.1 Aim and Objectives

8.1.1.1 During the life time of the Safe Time Layer there will be several versions of the specification.

8.1.1.2 The objective of the interoperability configuration is to define principles to be applied in situations where different nodes have been certified to different versions.

8.1.1.3 Note: The handling of different software versions is out of the scope of the system configuration management.

8.1.2 Evolution of the versions

8.1.2.1 The evolution of the versions of the Safe Time Layer shall be sequential, i. e. there shall only be a direct upgrade of an existing version and no branch is accepted.

8.1.2.2 The versions of the Safe Time Layer shall be identified by a compatibility number which complies with the following:

8.1.2.2.1 Each Compatibility Number will have the following format: X.Y.Z, where X, Y and Z are any number between 0 and 255 (examples: 1.12.0, 6.8.203, 65.0.15).

8.1.2.2.2 The first number (X) distinguishes not compatible versions.

8.1.2.2.3 Note: Value “0” of the first number (X) is reserved to indicate other protocol than the Safe Time Layer.

8.1.2.2.4 The second number (Y) indicates compatibility within a version X.

8.1.2.2.5 If the first number of two versions is the same, that indicates that those versions are compatible, independently of the second number (e. g. version 3.5 is compatible with 3.3, 3.14).

8.1.2.2.6 The third number (Z) is a vendor-specific (version) number that indicates the implemented version X.Y.

8.2 Compatibility Numbers

8.2.1.1 All nodes which have implemented this version of the Safe Time Layer shall transmit the following Compatibility Number see 5.2.1.3, 6.3.1.9, and 6.6.1.5.

8.2.1.2 The Compatibility Number shall be changed with every official release of this document.

8.2.1.3 Compatibility Table

Safe Time Layer		**Difference to previous version (overview)**
Version of the Document (SUBSET-056)	**Compatibilty Number**	
2.0.0	X=2, Y=0, Z=0	Initial Revision.
2.2.0	X=3, Y=0, Z	General revision of the specification. Modification of the behaviour of the Logical Connection Master and Logical Connection Slave state machines. Re-work of startup timers. Configuration management implemented. Z is vendor specific

9. LIST OF CONSTANTS

9.1.1.1 This chapter gives a list of the constants used within the Safe Time Layer together with their definition. The numeric values are specified in SUBSET-059 (Performance Requirements for STMs).

9.2 LocalClockMaxReSyncInterval

9.2.1.1 See chapter 7.4.1.6.1.

9.3 SafeTimeLayerStartupInterval

9.3.1.1 See chapter 6.6.1.3.

9.4 SafeTimeLayerReStartInterval

9.4.1.1 See chapter 6.6.1.4.

9.5 StartupSynchronisationTimeLimit

9.5.1.1 Time when the sending interval of the Safe Time Layer Startup message changes from SafeTimeLayerStartupInterval to SafeTimeLayerReStartInterval (see chapter 6.6.1.4).

9.6 MaxClockInaccuracyAfterAdjustFactor

9.6.1.1 This constant gives the maximum allowed error of LocalRefTime in relation to RefTime. See 7.4.3.5.

9.7 TimeForLongTermDriftCheck

9.7.1.1 See chapter 7.4.3.7.2.

9.8 MinNumberofSyncAndRefMsgReceived

9.8.1.1 See chapter 7.4.3.6

9.9 SyncAndRefTimeSyncInterval

9.9.1.1 See chapter 7.3.1.4.

9.10 SyncAndRefTimeRunInterval

9.10.1.1 See chapter 7.3.1.5.

9.11 SyncAndRefTimeStartupTimeLimit

9.11.1.1 Time when the sending interval of the Sync And Reference Time message changes from SyncAndRefTimeSyncInterval to SyncAndRefTimeRunInterval (see chapter 7.3.1.5).

9.12 ConnectionSetupTimeLimit

9.12.1.1 See chapters 7.6.1.3.3.1 and 7.7.1.3.3.1.

10. APPENDIX: SERVICES PROVIDED BY THE SAFE TIME LAYER

10.1.1.1 This appendix is informal and shall be read as an example.

10.1.1.2 The Safe Time Layer may provide the following services for the application:

10.1.1.2.1

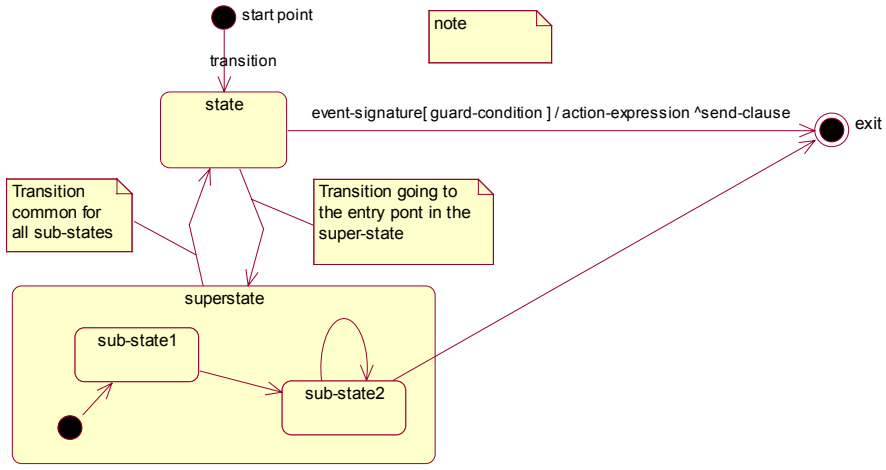
Service	Description	Parameters	Return
Connect	Initialises a logical logical connection	The parameters required by the connect service in the Safe Link Layer and; Sender_Dynamic_Transfer_Time or Receiver_Dynamic_Transfer_Time; Sender_Static_Transfer_Time or Receiver_Static_Transfer_Time	Logical connection ID or Fail
Get Data	Returns received application data	Logical connection ID	Application data, No data or Fail; Reception Time Stamp
Send Data	Send application data	Logical connection ID; Application data	OK or Fail; Send time stamp
Disconnect	Disconnect a logical connection	Logical connection ID; Reason for disconnect; Reason for disconnect text	OK or Fail
Logical connection active	Ask if a logical connection is active	Logical connection ID	Connected, Disconnected or Fail; Reason for disconnect; Reason for disconnect text
Get time	Returns the local clocks estimate of the reference time.	None	Current time, Unsynchronised or Fail

11. APPENDIX: DEFINITION OF NOTATION

11.1 State Machine

11.1.1.1 In this document the following notation for state machine view is used:

11.1.1.2



11.1.1.2.1 event-signature: Name of the event condition that shall be fulfilled to trig the transition.

11.1.1.2.2 [guard-condition]: The guard-condition must be true to enable evaluation of the event condition named by the event-signature.

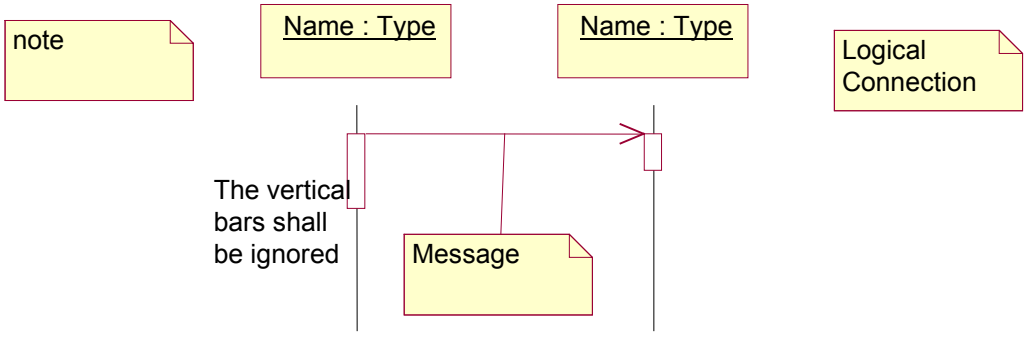
11.1.1.2.3 /action-expression: The action that is evaluated at the transition.

11.1.1.2.4 ^send-clause: The name of the message sent at the transition.

11.2 Sequence Chart

11.2.1.1 In this document the following notation for the Sequence Chart view is used:

11.2.1.2



12. APPENDIX: THE METAPHOR

12.1.1.1 This appendix is informal and non-normative

12.1.1.2 In the small town of Clockholm people are mindful of time.

12.1.1.3 Whenever the church clock strikes they take the opportunity to see that their watch is accurate.

12.1.1.4 When the church clock strikes and somebody notice a small difference with the own watch, they will consider their own clock has drifted and make a small adjustment or calibration.

12.1.1.5 If the difference with the church clock is larger, they will not know if the church clock or their own watch is faulty, but cannot rely on their watch and will become unwilling to communicate.

12.1.1.6 When people meet and talk with each other they do also check each other's watches to make sure that both are accurate.

12.1.1.7 If somebody is found to have an inaccurate clock this person is politely ignored, and should not be a discussion partner.

12.1.1.8 If somebodys watch is faulty, this person will get isolated either by his or her own discovery of mismatch with church clock or by other people who find him or her state an inaccurate time. Time matters in Clockholm!

12.1.1.9 If the church clock is faulty, all citizens of Clockholm will assume their own clock is faulty and become unwilling to communicate and all activities will stop.

12.1.1.10 When this happens, the reverend comes to help. He declares that the world ended, the First Day has come and all shall set their clocks according to the church clock.

12.1.1.11 Then the good people of Clockholm will do so. They trust their reverend, as being the one responsible of the church clock.