Council Regulation (EEC) No 3821/85 of 20 December 1985 on recording equipment in road transport

[F1[F2ANNEX I B

REQUIREMENTS FOR CONSTRUCTION, TESTING, INSTALLATION AND INSPECTION

Textual Amendments

F1 Inserted by Council Regulation (EC) No 2135/98 of 24 September 1998 amending Regulation (EEC) No 3821/85 on recording equipment in road transport and Directive 88/599/EEC concerning the application of Regulations (EEC) No 3820/85 and (EEC) No 3821/85.

F2 Substituted by Commission Regulation (EC) No 1360/2002 of 13 June 2002 adapting for the seventh time to technical progress Council Regulation (EEC) No 3821/85 on recording equipment in road transport (Text with EEA relevance).

Appendix 1

DATA DICTIONARY

2. DATA TYPE DEFINITIONS

For any of the following data types, the default value for an ‘unknown’ or a ‘not applicable’ content will consist in filling the data element with ′FF′-Bytes.

2.1. ActivityChangeInfo

This data type enables to code, within a two bytes word, a slot status at 00.00 and/or a driver status at 00.00 and/or changes of activity and/or changes of driving status and/or changes of card status for a driver or a co-driver. This data type is related to requirements 084, 109a, 199 and 219.

ActivityChangeInfo ::= OCTET STRING (SIZE(2))

Value assignment — Octet Aligned: *'scpaatttttttttt'B* (16 bits)

For Data Memory recordings (or slot status):

′s′B	Slot: ′0′B: DRIVER, ′1′B: CO-DRIVER,
′c′B	Driving status: ′0′B: SINGLE, ′1′B: CREW,
′p′B	Driver (or workshop) card status in the relevant slot: ′0′B: INSERTED, a card is inserted, 1′B: NOT INSERTED, no card is inserted (or a card is withdrawn),
′aa′B	Activity: ′00′B: BREAK/REST, ′01′B: AVAILABILITY, ′10′B: WORK, ′11′B: DRIVING,
′ttttttttttt′B	Time of the change: Number of minutes since 00h00 on the given day.

For Driver (or Workshop) card recordings (and driver status):

′s′B	Slot (not relevant when ′p′ = 1 except note below): ′0′B: DRIVER, ′1′B: 2. CO-DRIVER,
′c′B	**Driving status (case ′p′ = 0) or** / **Following activity status (case ′p′ = 1):** ′0′B: SINGLE, / ′0′B: UNKNOWN ′1′B: CREW, / ′1′B: KNOWN (= manually entered)
′p′B	Card status: ′0′B: INSERTED, the card is inserted in a recording equipment,

	′1′B: NOT INSERTED, the card is not inserted (or the card is withdrawn),
′aa′B	Activity (not relevant when ′p′ = 1 and ′c′ = 0 except note below): ′00′B: BREAK/REST, ′01′B: AVAILABILITY, ′10′B: WORK, ′11′B: DRIVING,
′ttttttttttt′B	Time of the change: Number of minutes since 00h00 on the given day.

Note for the case ‘card withdrawal’:

When the card is withdrawn:

— ′s′ is relevant and indicates the slot from which the card is withdrawn,

— ′c′ must be set to 0,

— ′p′ must be set to 1,

— ′aa′ must code the current activity selected at that time,

As a result of a manual entry, the bits ′c′ and ′aa′ of the word (stored in a card) may be overwritten later to reflect the entry.

[F3 2.2. Address

An address.

Address: = SEQUENCE {

codePage INTEGER (0..255),

address OCTET STRING (SIZE(35))

}

codePage specifies a character set defined in Chapter 4,

address is an address encoded using the specified character set.**]**

Textual Amendments

F3 Substituted by Commission Regulation (EU) No 1266/2009 of 16 December 2009 adapting for the tenth time to technical progress Council Regulation (EEC) No 3821/85 on recording equipment in road transport (Text with EEA relevance).

2.3. BCDString

BCDString is applied for Binary Code Decimal (BCD) representation. This data type is used to represent one decimal digit in one semi octet (4 bits). BCDString is based on the ISO/IEC 8824-1 ′CharacterStringType′.

BCDString ::= CHARACTER STRING (WITH COMPONENTS {

identification (WITH COMPONENTS {

fixed PRESENT }) })

BCDString uses an 'hstring' notation. The leftmost hexadecimal digit shall be the most significant semi octet of the first octet. To produce a multiple of octets, zero trailing semi octets shall be inserted, as needed, from the leftmost semi octet position in the first octet.

Permitted digits are: 0, 1, … 9.

2.4. CalibrationPurpose

Code explaining why a set of calibration parameters was recorded. This data type is related to requirements 097 and 098.

CalibrationPurpose ::= OCTET STRING (SIZE(1)).

Value assignment:

′00′H	reserved value,
′01′H	activation: recording of calibration parameters known, at the moment of the VU activation,
′02′H	first installation: first calibration of the VU after its activation,
′03′H	installation: first calibration of the VU in the current vehicle,
′04′H	periodic inspection.

2.5. CardActivityDailyRecord

Information, stored in a card, related to the driver activities for a particular calendar day. This data type is related to requirements 199 and 219.

CardActivityDailyRecord ::= SEQUENCE {

activityPreviousRecordLength INTEGER(0..CardActivityLengthRange),

[X1 `activityRecordLength INTEGER (0.. CardActivityLengthRange)`]

Editorial Information

X1 Inserted by Corrigendum to Commission Regulation (EC) No 1360/2002 of 13 June 2002 adapting for the seventh time to technical progress Council Regulation (EEC) No 3821/85 on recording equipment in road transport (Official Journal of the European Communities L 207 of 5 August 2002).

activityRecordDate TimeReal,

activityDailyPresenceCounter DailyPresenceCounter,

activityDayDistance Distance,

activityChangeInfo SET SIZE(1..1 440) OF ActivityChangeInfo

}

activityPreviousRecordLength is the total length in bytes of the previous daily record. The maximum value is given by the length of the OCTET STRING containing these records (see CardActivityLengthRange paragraph 3). When this record is the oldest daily record, the value of activityPreviousRecordLength must be set to 0.

activityRecordLength is the total length in bytes of this record. The maximum value is given by the length of the OCTET STRING containing these records.

activityRecordDate is the date of the record.

activityDailyPresenceCounter is the daily presence counter for the card this day.

activityDayDistance is the total distance travelled this day.

activityChangeInfo is the set of ActivityChangeInfo data for the driver this day. It may contain at maximum 1 440 values (one activity change per minute). This set always includes the activityChangeInfo coding the driver status at 00.00.

2.6. CardActivityLengthRange

Number of bytes in a driver or a workshop card, available to store driver activity records.

CardActivityLengthRange ::= INTEGER(0..2^{16}-1)

Value assignment: see paragraph 3.

2.7. CardApprovalNumber

Type approval number of the card.

CardApprovalNumber ::= IA5String(SIZE(8))

Value assignment: Unspecified.

2.8. CardCertificate

Certificate of the public key of a card.

CardCertificate ::= Certificate.

2.9. CardChipIdentification

Information, stored in a card, related to the identification of the card's Integrated Circuit (IC) (requirement 191).

CardChipIdentification ::= SEQUENCE {

icSerialNumber OCTET STRING (SIZE(4)),

icManufacturingReferences OCTET STRING (SIZE(4))

}

icSerialNumber is the IC serial number as defined in EN 726-3.

icManufacturingReferences is the IC manufacturer identifier and fabrication elements as defined in EN 726-3.

2.10. CardConsecutiveIndex

A card consecutive index (definition h)).

CardConsecutiveIndex ::= IA5String(SIZE(1))

Value assignment: (see this Annex Chapter VII)

Order for increase : ′0, …, 9, A, … , Z, a, … , z′.

2.11. CardControlActivityDataRecord

Information, stored in a driver or workshop card, related to the last control the driver has been subject to (requirements 210 and 225).

CardControlActivityDataRecord ::= SEQUENCE {

controlType controlType,

controlTime TimeReal,

controlCardNumber FullCardNumber,

controlVehicleRegistration VehicleRegistrationIdentification,

controlDownloadPeriodBegin TimeReal,

controlDownloadPeriodEnd TimeReal,

}

controlType is the type of the control.

controlTime is the date and time of the control.

controlCardNumber is the FullCardNumber of the control officer having performed the control.

controlVehicleRegistration is the VRN and registering Member State of the vehicle in which the control happened.

controlDownloadPeriodBegin and **controlDownloadPeriodEnd** is the period downloaded, in case of downloading.

2.12. CardCurrentUse

Information about the actual usage of the card (requirement 212).

CardCurrentUse ::= SEQUENCE {

sessionOpenTime TimeReal,

sessionOpenVehicle VehicleRegistrationIdentification

}

sessionOpenTime is the time when the card is inserted for the current usage. This element is set to zero at card removal.

sessionOpenVehicle is the identification of the currently used vehicle, set at card insertion. This element is set to zero at card removal.

2.13. CardDriverActivity

Information, stored in a driver or a workshop card, related to the activities of the driver (requirements 199 and 219).

CardDriverActivity ::= SEQUENCE {

activityPointerOldestDayRecord INTEGER(0..CardActivityLengthRange-1),

activityPointerNewestRecord INTEGER(0..CardActivityLengthRange-1),

activityDailyRecords OCTET STRING (SIZE(CardActivityLengthRange))

}

activityPointerOldestDayRecord is the specification of the begin of the storage place (number of bytes from the beginning of the string) of the oldest complete day record in the activityDailyRecords string. The maximum value is given by the length of the string.

activityPointerNewestRecord is the specification of the begin of the storage place (number of bytes from the beginning of the string) of the most recent day record in the activityDailyRecords string. The maximum value is given by the length of the string.

activityDailyRecords is the space available to store the driver activity data (data structure: CardActivityDailyRecord) for each calendar day where the card has been used.

Value assignment: this octet string is cyclically filled with records of CardActivityDailyRecord. At the first use storing is started at the first byte of the string. All new records are appended at the end of the previous one. When the string is full, storing continues at the first byte of the string independently of a break being inside a data element. Before placing new activity data in the string (enlarging current activityDailyRecord, or placing a new activityDailyRecord) that replaces older activity data, activityPointerOldestDayRecord must be updated to reflect the new location of the oldest complete day record, and activityPreviousRecordLength of this (new) oldest complete day record must be reset to 0.

2.14. CardDrivingLicenceInformation

Information, stored in a driver card, related to the card holder driver licence data (requirement 196).

CardDrivingLicenceInformation ::= SEQUENCE {

drivingLicenceIssuingAuthority Name,

drivingLicenceIssuingNation NationNumeric,

drivingLicenceNumber IA5String(SIZE(16))

}

drivingLicenceIssuingAuthority is the authority responsible for issuing the driving licence.

drivingLicenceIssuingNation is the nationality of the authority that issued the driving licence.

drivingLicenceNumber is the number of the driving licence.

2.15. CardEventData

Information, stored in a driver or workshop card, related to the events associated with the card holder (requirements 204 and 223).

CardEventData ::= SEQUENCE SIZE(6) OF {

cardEventRecords SET SIZE(NoOfEventsPerType) OF CardEventRecord

}

CardEventData is a sequence, ordered by ascending value of EventFaultType, of cardEventRecords (except security breach attempts related records which are gathered in the last set of the sequence).

cardEventRecords is a set of event records of a given event type (or category for security breach attempts events).

2.16. CardEventRecord

Information, stored in a driver or a workshop card, related to an event associated to the card holder (requirements 205 and 223).

CardEventRecord ::= SEQUENCE {

eventType EventFaultType,

eventBeginTime TimeReal,

eventEndTime TimeReal,

eventVehicleRegistration VehicleRegistrationIdentification

}

eventType is the type of the event.

eventBeginTime is the date and time of beginning of event.

eventEndTime is the date and time of end of event.

eventVehicleRegistration is the VRN and registering Member State of vehicle in which the event happened.

2.17. CardFaultData

Information, stored in a driver or a workshop card, related to the faults associated to the card holder (requirements 207 and 223).

CardFaultData ::= SEQUENCE SIZE(2) OF {

cardFaultRecords SET SIZE(NoOfFaultsPerType) OF CardFaultRecord

}

CardFaultData is a sequence of Recording Equipment faults set of records followed by card faults set of records.

cardFaultRecords is a set of fault records of a given fault category (Recording Equipment or card).

2.18. CardFaultRecord

Information, stored in a driver or a workshop card, related to a fault associated to the card holder (requirement 208 and 223).

CardFaultRecord ::= SEQUENCE {

faultType EventFaultType,

faultBeginTime TimeReal,

faultEndTime TimeReal,

faultVehicleRegistration VehicleRegistrationIdentification

}

faultType is the type of the fault.

faultBeginTime is the date and time of beginning of fault.

faultEndTime is the date and time of end of fault.

faultVehicleRegistration is the VRN and registering Member State of vehicle in which the fault happened.

2.19. CardIccIdentification

Information, stored in a card, related to the identification of the integrated circuit (IC) card (requirement 192).

CardIccIdentification ::= SEQUENCE {

clockStop OCTET STRING (SIZE(1)),

cardExtendedSerialNumber ExtendedSerialNumber,

cardApprovalNumber CardApprovalNumber

cardPersonaliserID OCTET STRING (SIZE(1)),

embedderIcAssemblerId OCTET STRING (SIZE(5)),

icIdentifier OCTET STRING (SIZE(2))

}

clockStop is the Clockstop mode as defined in EN 726-3.

cardExtendedSerialNumber is the IC card serial number and IC card manufacturing reference as defined in EN 726-3 and as further specified by the ExtendedSerialNumber data type.

cardApprovalNumber is the type approval number of the card.

cardPersonaliserID is the card personaliser ID as defined in EN 726-3.

embedderIcAssemblerId is the embedder/IC assembler identifier as defined in EN 726-3.

icIdentifier is the Identifier of the IC on the card and its IC manufacturer as defined in EN 726-3.

2.20. CardIdentification

Information, stored in a card, related to the identification of the card (requirements 194, 215, 231, 235).

CardIdentification ::= SEQUENCE

cardIssuingMemberState NationNumeric,

cardNumber CardNumber,

cardIssuingAuthorityName Name,

cardIssueDate TimeReal,

cardValidityBegin TimeReal,

cardExpiryDate TimeReal

}

cardIssuingMemberState is the code of the Member State issuing the card.

cardNumber is the card number of the card.

cardIssuingAuthorityName is the name of the authority having issued the Card.

cardIssueDate is the issue date of the Card to the current holder.

cardValidityBegin is the first date of validity of the card.

cardExpiryDate is the date when the validity of the card ends.

2.21. CardNumber

A card number as defined by definition g).

CardNumber ::= CHOICE {

SEQUENCE {

driverIdentification IA5String(SIZE(14)),

cardReplacementIndex CardReplacementIndex,

cardRenewalIndex CardRenewalIndex

}

SEQUENCE {

ownerIdentification IA5String(SIZE(13)),

cardConsecutiveIndex CardConsecutiveIndex,

cardReplacementIndex CardReplacementIndex,

cardRenewalIndex CardRenewalIndex

}

}

driverIdentification is the unique identification of a driver in a Member State.

ownerIdentification is the unique identification of a company or a workshop or a control body within a Member State.

cardConsecutiveIndex is the card consecutive index.

cardReplacementIndex is the card replacement index.

cardRenewalIndex is the card renewal index.

The first sequence of the choice is suitable to code a driver card number, the second sequence of the choice is suitable to code workshop, control, and company card numbers.

2.22. CardPlaceDailyWorkPeriod

Information, stored in a driver or a workshop card, related to the places where daily work periods begin and/or end (requirements 202 and 221).

```
CardPlaceDailyWorkPeriod ::= SEQUENCE {
   placePointerNewestRecord  INTEGER(0..NoOfCardPlaceRecords-1),
[X2 placeRecords             SET SIZE(NoOfCardPlaceRecords) OF PlaceRecord} ]
```

Editorial Information

X2 Substituted by Corrigendum to Commission Regulation (EC) No 1360/2002 of 13 June 2002 adapting for the seventh time to technical progress Council Regulation (EEC) No 3821/85 on recording equipment in road transport (Official Journal of the European Communities L 207 of 5 August 2002).

}

placePointerNewestRecord is the index of the last updated place record.

Value assignment: Number corresponding to the numerator of the place record, beginning with '0' for the first occurrence of the place records in the structure.

placeRecords is the set of records containing the information related to the places entered.

2.23. CardPrivateKey

The private key of a card.

CardPrivateKey ::= RSAKeyPrivateExponent.

2.24. CardPublicKey

The public key of a card.

CardPublicKey ::= PublicKey.

2.25. CardRenewalIndex

A card renewal index (definition i)).

CardRenewalIndex ::= IA5String(SIZE(1)).

Value assignment: (see this Annex Chapter VII).

'0'	First issue.
Order for increase:	'0, …, 9, A, …, Z'.

2.26. CardReplacementIndex

A card replacement index (definition j)).

CardReplacementIndex ::= IA5String(SIZE(1))

Value assignment: (see this Annex Chapter VII).

'0'	Original card.
Order for increase:	'0, …, 9, A, …, Z'.

2.27. CardSlotNumber

Code to distinguish between the two slots of a vehicle unit.

CardSlotNumber ::= INTEGER {

driverSlot (0),

co-driverSlot (1)

}

Value assignment: not further specified.

2.28. CardSlotsStatus

Code indicating the type of cards inserted in the two slots of the vehicle unit.

CardSlotsStatus ::= OCTET STRING (SIZE(1))

Value assignment — Octet Aligned: ′ccccdddd′B:

′cccc′B	Identification of the type of card inserted in the co-driver slot,
′dddd′B	Identification of the type of card inserted in the driver slot,

with the following identification codes:

′0000′B	no card is inserted,
′0001′B	a driver card is inserted,
′0010′B	a workshop card is inserted,
′0011′B	a control card is inserted,
′0100′B	a company card is inserted.

2.29. CardStructureVersion

Code indicating the version of the implemented structure in a tachograph card.

CardStructureVersion ::= OCTET STRING (SIZE(2))

Value assignment: ′aabb′H:

‘**[**[F4]aa’H	Index for changes of the structure, ‘00h’ for this version
‘bb’H	Index for changes concerning the use of the data elements defined for the structure given by the high byte, ‘00h’ for this version.**]**

Textual Amendments

F4 Substituted by Commission Regulation (EC) No 432/2004 of 5 March 2004 adapting for the eighth time to technical progress Council Regulation (EEC) No 3821/85 of 20 December 1985 on recording equipment in road transport (Text with EEA relevance).

2.30. CardVehicleRecord

Information, stored in a driver or workshop card, related to a period of use of a vehicle during a calendar day (requirements 197 and 217).

CardVehicleRecord ::= SEQUENCE {

vehicleOdometerBegin OdometerShort,

vehicleOdometerEnd OdometerShort,

vehicleFirstUse TimeReal,

vehicleLastUse TimeReal,

vehicleRegistration VehicleRegistrationIdentification,

vuDataBlockCounter VuDataBlockCounter

}

vehicleOdometerBegin is the vehicle odometer value at the beginning of the period of use of the vehicle.

vehicleOdometerEnd is the vehicle odometer value at the end of the period of use of the vehicle.

vehicleFirstUse is the date and time of the beginning of the period of use of the vehicle.

vehicleLastUse is the date and time of the end of the period of use of the vehicle.

vehicleRegistration is the VRN and the registering Member State of the vehicle.

vuDataBlockCounter is the value of the VuDataBlockCounter at last extraction of the period of use of the vehicle.

2.31. CardVehiclesUsed

Information, stored in a driver or workshop card, related to the vehicles used by the card holder (requirements 197 and 217).

CardVehiclesUsed := SEQUENCE {

vehiclePointerNewestRecord INTEGER(0..NoOfCardVehicleRecords-1),

cardVehicleRecords SET SIZE(NoOfCardVehicleRecords) OF CardVehicleRecord

}

vehiclePointerNewestRecord is the index of the last updated vehicle record.

Value assignment: Number corresponding to the numerator of the vehicle record, beginning with ′0′ for the first occurrence of the vehicle records in the structure.

cardVehicleRecords is the set of records containing information on vehicles used.

2.32. Certificate

The certificate of a public key issued by a Certification Authority.

Certificate ::= OCTET STRING (SIZE(194))

Value assignment: digital signature with partial recovery of a CertificateContent according to Appendix 11 ‘common security mechanisms’: Signature (128 bytes) Public Key remainder (58 Byte) Certification Authority Reference (8 bytes).

2.33. CertificateContent

The (clear) content of the certificate of a public key according to Appendix 11 common security mechanisms.

CertificateContent ::= SEQUENCE {

certificateProfileIdentifier INTEGER(0..255),

certificationAuthorityReference KeyIdentifier,

certificateHolderAuthorisation CertificateHolderAuthorisation,

certificateEndOfValidity TimeReal,

certificateHolderReference KeyIdentifier,

publicKey PublicKey

}

certificateProfileIdentifier is the version of the corresponding certificate.

Value assignment:: ′01h′ for this version.

CertificationAuthorityReference identifies the Certification Authority issuing the certificate. It also references the Public Key of this Certification Authority.

certificateHolderAuthorisation identifies the rights of the certificate holder.

certificateEndOfValidity is the date when the certificate expires administratively.

certificateHolderReference identifies the certificate holder. It also references his Public Key.

publicKey is the public key that is certified by this certificate.

2.34. CertificateHolderAuthorisation

Identification of the rights of a certificate holder.

CertificateHolderAuthorisation ::= SEQUENCE {

tachographApplicationID OCTET STRING(SIZE(6))

equipmentType EquipmentType

}

tachographApplicationID is the application identifier for the tachograph application.

Value assignment: ′FFh′ ′54h′ ′41h′ ′43h′ ′48h′ ′4Fh′. This AID is a proprietary non-registered application identifier in accordance with ISO/IEC 7816-5.

equipmentType is the identification of the type of equipment to which the certificate is intended.

Value assignment: in accordance with EquipmentType data type. 0 if certificate is the one of a Member State.

2.35. CertificateRequestID

Unique identification of a certificate request. It can also be used as a Vehicle Unit Public Key Identifier if the serial number of the vehicle Unit to which the key is intended is not known at certificate generation time.

CertificateRequestID ::= SEQUENCE {

requestSerialNumber INTEGER(0..2^{32}-1)

requestMonthYear BCDString(SIZE(2))

crIdentifier OCTET STRING(SIZE(1))

manufacturerCode ManufacturerCode

}

requestSerialNumber is a serial number for the certificate request, unique for the manufacturer and the month below.

requestMonthYear is the identification of the month and the year of the certificate request.

Value assignment: BCD coding of month (two digits) and year (two last digits).

crIdentifier: is an identifier to distinguish a certificate request from an extended serial number.

Value assignment: 'FFh'.

manufacturerCode: is the numerical code of the manufacturer requesting the certificate.

2.36. CertificationAuthorityKID

Identifier of the Public Key of a Certification Authority (a Member State or the European Certification Authority).

CertificationAuthorityKID ::= SEQUENCE {

nationNumeric NationNumeric

nationAlpha NationAlpha

keySerialNumber INTEGER(0..255)

additionalInfo OCTET STRING(SIZE(2))

caIdentifier OCTET STRING(SIZE(1))

}

nationNumeric is the numerical nation code of the Certification Authority.

nationAlpha is the alphanumerical nation code of the Certification Authority.

keySerialNumber is a serial number to distinguish the different keys of the Certification Authority in the case keys are changed.

additionalInfo is a two byte field for additional coding (Certification Authority specific).

caIdentifier is an identifier to distinguish a Certification Authority Key Identifier from other Key Identifiers.

Value assignment: '01h'.

2.37. CompanyActivityData

Information, stored in a company card, related to activities performed with the card (requirement 237).

CompanyActivityData ::= SEQUENCE {

companyPointerNewestRecord INTEGER(0..NoOfCompanyActivityRecords-1),

companyActivityRecords SET SIZE(NoOfCompanyActivityRecords) OF

companyActivityRecord SEQUENCE {

companyActivityType CompanyActivityType,

companyActivityTime TimeReal,

cardNumberInformation FullCardNumber,

vehicleRegistrationInformation VehicleRegistrationIdentification,

downloadPeriodBegin TimeReal,

downloadPeriodEnd TimeReal

}

}

companyPointerNewestRecord is the index of the last updated companyActivityRecord.

Value assignment: Number corresponding to the numerator of the company activity record, beginning with ′0′ for the first occurrence of the company activity record in the structure.

companyActivityRecords is the set of all company activity records.

companyActivityRecord is the sequence of information related to one company activity.

companyActivityType is the type of the company activity.

companyActivityTime is the date and time of the company activity.

cardNumberInformation is the card number and the card issuing Member State of the card downloaded, if any.

vehicleRegistrationInformation is the VRN and registering Member State of the vehicle downloaded or locked in or out.

downloadPeriodBegin and **downloadPeriodEnd** is the period downloaded from the VU, if any.

2.38. CompanyActivityType

Code indicating an activity carried out by a company using its company card.

CompanyActivityType ::= INTEGER {

card downloading (1),

VU downloading (2),

VU lock-in (3),

VU lock-out (4)

}

2.39. CompanyCardApplicationIdentification

Information, stored in a company card related to the identification of the application of the card (requirement 190).

CompanyCardApplicationIdentification ::= SEQUENCE {

typeOfTachographCardId EquipmentType,

cardStructureVersion CardStructureVersion,

noOfCompanyActivityRecords NoOfCompanyActivityRecords

}

typeOfTachographCardId is specifying the implemented type of card.

cardStructureVersion is specifying the version of the structure that is implemented in the card.

noOfCompanyActivityRecords is the number of company activity records the card can store.

2.40. CompanyCardHolderIdentification

Information, stored in a company card, related to the cardholder identification (requirement 236).

CompanyCardHolderIdentification ::= SEQUENCE {

companyName Name,

companyAddress Address,

cardHolderPreferredLanguage Language

}

companyName is the name of the holder company.

companyAddress is the address of the holder company.

cardHolderPreferredLanguage is the preferred language of the card holder.

2.41. ControlCardApplicationIdentification

Information, stored in a control card related to the identification of the application of the card (requirement 190).

ControlCardApplicationIdentification ::= SEQUENCE {

typeOfTachographCardId EquipmentType,

cardStructureVersion CardStructureVersion,

noOfControlActivityRecords NoOfControlActivityRecords

}

typeOfTachographCardId is specifying the implemented type of card.

cardStructureVersion is specifying the version of the structure that is implemented in the card.

noOfControlActivityRecords is the number of control activity records the card can store.

2.42. ControlCardControlActivityData

Information, stored in a control card, related to control activity performed with the card (requirement 233).

ControlCardControlActivityData ::= SEQUENCE {

controlPointerNewestRecord INTEGER(0..NoOfControlActivityRecords-1),

controlActivityRecords SET SIZE(NoOfControlActivityRecords) OF

controlActivityRecord SEQUENCE {

controlType ControlType,

controlTime TimeReal,

controlledCardNumber FullCardNumber,

controlledVehicleRegistration VehicleRegistrationIdentification,

controlDownloadPeriodBegin TimeReal,

controlDownloadPeriodEnd TimeReal

}

}

controlPointerNewestRecord is the index of the last updated control activity record.

Value assignment: Number corresponding to the numerator of the control activity record, beginning with ′0′ for the first occurrence of the control activity record in the structure.

controlActivityRecords is the set of all control activity records.

controlActivityRecord is the sequence of information related to one control.

controlType is the type of the control.

controlTime is the date and time of the control.

controlledCardNumber is the card number and the card issuing Member State of the card controlled.

controlledVehicleRegistration is the VRN and registering Member State of the vehicle in which the control happened.

controlDownloadPeriodBegin and **controlDownloadPeriodEnd** is the period eventually downloaded.

2.43. ControlCardHolderIdentification

Information, stored in a control card, related to the identification of the cardholder (requirement 232).

ControlCardHolderIdentification ::= SEQUENCE {

controlBodyName Name,

controlBodyAddress Address,

cardHolderName HolderName,

cardHolderPreferredLanguage Language

}

controlBodyName is the name of the control body of the card holder.

controlBodyAddress is the address of the control body of the card holder.

cardHolderName is the name and first name(s) of the holder of the Control Card.

cardHolderPreferredLanguage is the preferred language of the card holder.

2.44. ControlType

Code indicating the activities carried out during a control. This data type is related to requirements 102, 210 and 225.

ControlType ::= OCTET STRING (SIZE(1))

Value assignment — Octet aligned:′cvpdxxxx′B (8 bits)

′c′B	card downloading: ′0′B: card not downloaded during this control activity, ′1′B: card downloaded during this control activity
′v′B	VU downloading: ′0′B: VU not downloaded during this control activity, ′1′B: VU downloaded during this control activity
′p′B	printing: ′0′B: no printing done during this control activity, ′1′B: printing done during this control activity
′d′B	display: ′0′B: no display used during this control activity, ′1′B: display used during this control activity
′xxxx′B	Not used.

2.45. CurrentDateTime

The current date and time of the recording equipment.

CurrentDateTime ::= TimeReal

Value assignment: not further specified.

2.46. DailyPresenceCounter

Counter, stored in a driver or workshop card, increased by one for each calendar day the card has been inserted in a VU. This data type is related to requirements 199 and 219.

DailyPresenceCounter ::= BCDString(SIZE(2))

Value assignment: Consecutive number with maximum value = 9 999, starting again with 0. At the time of first issuing of the card the number is set to 0.

2.47. Datef

Date expressed in a readily printable numeric format.

Datef ::= SEQUENCE {

year BCDString(SIZE(2)),

month BCDString(SIZE(1)),

day BCDString(SIZE(1))

}

Value assignment:

yyyy	Year
mm	Month
dd	Day

′00000000′H denotes explicitly no date.

2.48. Distance

A distance travelled (result of the calculation of the difference between two vehicle's odometer value in kilometres).

Distance ::= INTEGER(0..2^{16}-1)

Value assignment: Unsigned binary. Value in km in the operational range 0 to 9 999 km.

2.49. DriverCardApplicationIdentification

Information, stored in a driver card related to the identification of the application of the card (requirement 190).

DriverCardApplicationIdentification ::= SEQUENCE {

typeOfTachographCardId EquipmentType,

cardStructureVersion CardStructureVersion,

noOfEventsPerType NoOfEventsPerType,

noOfFaultsPerType NoOfFaultsPerType,

activityStructureLength CardActivityLengthRange,

noOfCardVehicleRecords NoOfCardVehicleRecords,

noOfCardPlaceRecords NoOfCardPlaceRecords

}

typeOfTachographCardId is specifying the implemented type of card.

cardStructureVersion is specifying the version of the structure that is implemented in the card.

noOfEventsPerType is the number of events per type of event the card can record.

noOfFaultsPerType is the number of faults per type of fault the card can record.

activityStructureLength indicates the number of bytes available for storing activity records.

noOfCardVehicleRecords is the number of vehicle records the card can contain.

noOfCardPlaceRecords is the number of places the card can record.

2.50. DriverCardHolderIdentification

Information, stored in a driver card, related to the identification of the cardholder (requirement 195).

DriverCardHolderIdentification ::= SEQUENCE {

cardHolderName HolderName,

cardHolderBirthDate Datef,

cardHolderPreferredLanguage Language

}

cardHolderName is the name and first name(s) of the holder of the Driver Card.

cardHolderBirthDate is the date of birth of the holder of the Driver Card.

cardHolderPreferredLanguage is the preferred language of the card holder.

2.51. EntryTypeDailyWorkPeriod

Code to distinguish between begin and end for an entry of a daily work period place and condition of the entry.

EntryTypeDailyWorkPeriod ::= INTEGER

Begin, related time = card insertion time or time of entry (0),

End, related time = card withdrawal time or time of entry (1),

Begin, related time manually entered (start time) (2),

End, related time manually entered (end of work period) (3),

Begin, related time assumed by VU (4),

End, related time assumed by VU (5)

}

Value assignment: according to ISO/IEC8824-1.

2.52. EquipmentType

Code to distinguish different types of equipment for the tachograph application.

EquipmentType ::= INTEGER(0..255)

- - Reserved (0),

- - Driver Card (1),

- - Workshop Card (2),

- - Control Card (3),

- - Company Card (4),

- - Manufacturing Card (5),

- - Vehicle Unit (6),

- - Motion Sensor (7),

- - RFU (8..255)

Value assignment: According to ISO/IEC 8824-1.

Value 0 is reserved for the purpose of designating a Member State or Europe in the CHA field of certificates.

2.53. EuropeanPublicKey

The European public key.

EuropeanPublicKey ::= PublicKey.

2.54. EventFaultType

Code qualifying an event or a fault.

EventFaultType ::= OCTET STRING (SIZE(1)).

Value assignment:

′0x′H	General events,
′00′H	No further details,
′01′H	Insertion of a non-valid card,
′02′H	Card conflict,
′03′H	Time overlap,
′04′H	Driving without an appropriate card,
′05′H	Card insertion while driving,
′06′H	Last card session not correctly closed,
′07′H	Over speeding,
′08′H	Power supply interruption,
′09′H	Motion data error,
[[F3]′0A′H	Vehicle Motion Conflict,
′0B′H to ′0F′H	RFU,**]**
′1x′H	Vehicle unit related security breach attempt events,
′10′H	No further details,
′11′H	Motion sensor authentication failure,
′12′H	Tachograph card authentication failure,
′13′H	Unauthorised change of motion sensor,
′14′H	Card data input integrity error
′15′H	Stored user data integrity error,
′16′H	Internal data transfer error,
′17′H	Unauthorised case opening,
′18′H	Hardware sabotage,
′19′H to ′1F′H	RFU,
′2x′H	Sensor related security breach attempt events,
′20′H	No further details,
′21′H	Authentication failure,
′22′H	Stored data integrity error,
′23′H	Internal data transfer error,
′24′H	Unauthorised case opening,
′25′H	Hardware sabotage,
′26′H to ′2F′H	RFU,
′3x′H	Recording equipment faults,
′30′H	No further details,
′31′H	VU internal fault,
′32′H	Printer fault,
′33′H	Display fault,
′34′H	Downloading fault,
′35′H	Sensor fault,
′36′H to ′3F′H	RFU
′4x′H	Card faults,
′40′H	No further details,
′41′H to ′4F′H	RFU
′50′H to ′7F′H	RFU,

′80′H to ′FF′H Manufacturer specific.

2.55. EventFaultRecordPurpose

Code explaining why an event or a fault has been recorded.

EventFaultRecordPurpose ::= OCTET STRING (SIZE(1)).

Value assignment:

′00′H	one of the 10 most recent (or last) events or faults
′01′H	the longest event for one of the last 10 days of occurrence
′02′H	one of the 5 longest events over the last 365 days
′03′H	the last event for one of the last 10 days of occurrence
′04′H	the most serious event for one of the last 10 days of occurrence
′05′H	one of the 5 most serious events over the last 365 days
′06′H	the first event or fault having occurred after the last calibration
′07′H	an active/on-going event or fault
′08′H to ′7F′H	RFU
′80′H to ′FF′H	manufacturer specific.

2.56. ExtendedSerialNumber

Unique identification of an equipment. It can also be used as an equipment Public Key Identifier.

ExtendedSerialNumber ::= SEQUENCE {

serialNumber INTEGER(0..2^{32}-1)

monthYear BCDString(SIZE(2))

type OCTET STRING(SIZE(1))

manufacturerCode ManufacturerCode

}

serialNumber is a serial number for the equipment, unique for the manufacturer, the equipment's type and the month below.

monthYear is the identification of the month and the year of manufacturing (or of serial number assignment).

Value assignment: BCD coding of Month (two digits) and Year (two last digits).

type is an identifier of the type of equipment.

Value assignment: manufacturer specific, with FFh' reserved value.

manufacturerCode: is the numerical code of the manufacturer of the equipment.

2.57. FullCardNumber

Code fully identifying a tachograph card.

FullCardNumber ::= SEQUENCE {

cardType EquipmentType,

cardIssuingMemberState NationNumeric,

cardNumber CardNumber

}

cardType is the type of the tachograph card.

cardIssuingMemberState is the code of the Member State having issued the card.

cardNumber is the card number.

2.58. HighResOdometer

Odometer value of the vehicle: Accumulated distance travelled by the vehicle during its operation.

HighResOdometer ::= INTEGER(0..2^{32}-1)

Value assignment: Unsigned binary. Value in 1/200 km in the operating range 0 to 21 055 406 km.

2.59. HighResTripDistance

A distance travelled during all or part of a journey.

HighResTripDistance ::= INTEGER(0..2^{32}-1)

Value assignment: Unsigned binary. Value in 1/200 km in the operating range 0 to 21 055 406 km.

2.60. HolderName

The surname and first name(s) of a card holder.

HolderName ::= SEQUENCE {

holderSurname Name,

holderFirstNames Name

}

holderSurname is the surname (family name) of the holder. This surname does not include titles.

Value assignment: When a card is not personal, holderSurname contains the same information as companyName or workshopName or controlBodyName.

holderFirstNames is the first name(s) and initials of the holder.

2.61. K-ConstantOfRecordingEquipment

Constant of the recording equipment (definition m)).

K-ConstantOfRecordingEquipment ::= INTEGER(0..2^{16}-1)

Value assignment: Pulses per kilometre in the operating range 0 to 64 255 pulses/km.

2.62. KeyIdentifier

A unique identifier of a Public Key used to reference and select the key. It also identifies the holder of the key.

KeyIdentifier ::= CHOICE {

extendedSerialNumber ExtendedSerialNumber,

certificateRequestID CertificateRequestID,

certificationAuthorityKID CertificationAuthorityKID

}

The first choice is suitable to reference the public key of a Vehicle Unit or of a tachograph card.

The second choice is suitable to reference the public key of a Vehicle Unit (in the case the serial number of the Vehicle Unit cannot be known at certificate generation time).

The third choice is suitable to reference the public key of a Member State.

2.63. L-TyreCircumference

Effective circumference of the wheel tyres (definition u)).

L-TyreCircumference ::= INTEGER(0..2^{16}-1)

Value assignment: Unsigned binary, value in 1/8 mm in the operating range 0 to 8 031 mm.

2.64. Language

Code identifying a language.

Language ::= IA5String(SIZE(2))

Value assignment: Two-letter lower-case coding according to ISO 639.

2.65. LastCardDownload

Date and time, stored on a driver card, of last card download (for other purposes than control). This date is updateable by a VU or any card reader.

LastCardDownload ::= TimeReal

Value assignment: not further specified.

2.66. ManualInputFlag

Code identifying whether a cardholder has manually entered driver activities at card insertion or not (requirement 081).

ManualInputFlag ::= INTEGER {

noEntry (0)

manualEntries (1)

}

Value assignment: not further specified.

[F3 2.67. ManufacturerCode

Code identifying a manufacturer of type-approved equipment.

ManufacturerCode: = INTEGER (0..255)

The laboratory competent for interoperability tests maintains and publishes the list of manufacturer codes on its web site (requirement 290).

ManufacturerCodes are provisionally assigned to developers of tachograph equipment on application to the laboratory competent for interoperability tests.**]**

2.68. MemberStateCertificate

The certificate of the public key of a Member State issued by the European certification authority.

MemberStateCertificate ::= Certificate

2.69. MemberStatePublicKey

The public key of a Member State.

MemberStatePublicKey ::= PublicKey.

[F3 2.70. Name

A name.

Name: = SEQUENCE {

codePage INTEGER (0..255),

name OCTET STRING (SIZE(35))

}

codePage specifies a character set defined in Chapter 4,

name is a name encoded using the specified character set.**]**

[F3 2.71. NationAlpha

Alphabetic reference to a country shall be in accordance with the distinguishing signs used on vehicles in international traffic (United Nations Vienna Convention on Road Traffic, 1968).

NationAlpha: = IA5String (SIZE (3))

The Nation Alpha and Numeric codes shall be held on a list maintained on the website of the laboratory appointed to carry out interoperability testing, as set out in Requirement 278.**]**

[F3 2.72. NationNumeric

Numerical reference to a country.

NationNumeric: = INTEGER (0.. 255)

Value assignment: see data type 2.71 (NationAlpha)

Any amendment or updating of the Nation Alpha or Numeric specification described in the above paragraph shall only be made out after the appointed laboratory has obtained the views of type approved digital tachograph vehicle unit manufacturers.**]**

2.73. NoOfCalibrationRecords

Number of calibration records, a workshop card can store.

NoOfCalibrationRecords ::= INTEGER(0..255)

Value assignment: see paragraph 3.

2.74. NoOfCalibrationsSinceDownload

Counter indicating the number of calibrations performed with a workshop card since its last download (requirement 230).

NoOfCalibrationsSinceDownload ::= INTEGER(0..2^{16}-1),

Value assignment: Not specified further.

2.75. NoOfCardPlaceRecords

Number of place records a driver or workshop card can store.

NoOfCardPlaceRecords ::= INTEGER(0..255)

Value assignment: see paragraph 3.

2.76. NoOfCardVehicleRecords

Number of vehicles used records a driver or workshop card can store.

NoOfCardVehicleRecords ::= INTEGER(0..2^{16}-1)

Value assignment: see paragraph 3.

2.77. NoOfCompanyActivityRecords

Number of company activity records, a company card can store.

NoOfCompanyActivityRecords ::= INTEGER(0..2^{16}-1)

Value assignment: see paragraph 3.

2.78. NoOfControlActivityRecords

Number of control activity records, a control card can store.

NoOfControlActivityRecords ::= INTEGER(0..2^{16}-1)

Value assignment: see paragraph 3.

2.79. NoOfEventsPerType

Number of events per type of event a card can store.

NoOfEventsPerType ::= INTEGER(0..255)

Value assignment: see paragraph 3.

2.80. NoOfFaultsPerType

Number of faults per type of fault a card can store.

NoOfFaultsPerType ::= INTEGER(0..255)

Value assignment: see paragraph 3.

2.81. OdometerValueMidnight

The vehicle's odometer value at midnight on a given day (requirement 090).

OdometerValueMidnight ::= OdometerShort

Value assignment: not further specified.

2.82. OdometerShort

Odometer value of the vehicle in a short form.

OdometerShort ::= INTEGER(0..2^{24}-1)

Value assignment: Unsigned binary. Value in km in the operating range 0 to 9 999 999 km.

2.83. OverspeedNumber

Number of over speeding events since the last over speeding control.

OverspeedNumber ::= INTEGER(0..255)

Value assignment: 0 means that no over speeding event has occurred since the last over speeding control, 1 means that one over speeding event has occurred since the last over speeding control … 255 means that 255 or more over speeding events have occurred since the last over speeding control.

2.84. PlaceRecord

Information related to a place where a daily work period begins or ends (requirements 087, 202, 221).

PlaceRecord ::= SEQUENCE {

entryTime TimeReal,

entryTypeDailyWorkPeriod EntryTypeDailyWorkPeriod,

dailyWorkPeriodCountry NationNumeric,

dailyWorkPeriodRegion RegionNumeric,

vehicleOdometerValue OdometerShort

}

entryTime is a date and time related to the entry.

entryTypeDailyWorkPeriod is the type of entry.

dailyWorkPeriodCountry is the country entered.

dailyWorkPeriodRegion is the region entered.

vehicleOdometerValue is the odometer value at the time of place entry.

2.85. PreviousVehicleInfo

Information related to the vehicle previously used by a driver when inserting his card in a vehicle unit (requirement 081).

PreviousVehicleInfo ::= SEQUENCE {

vehicleRegistrationIdentification VehicleRegistrationIdentification,

cardWithdrawalTime TimeReal

}

vehicleRegistrationIdentification is the VRN and the registering Member State of the vehicle.

cardWithdrawalTime is the card withdrawal date and time.

2.86. PublicKey

A public RSA key.

PublicKey ::= SEQUENCE {

rsaKeyModulus RSAKeyModulus,

rsaKeyPublicExponent RSAKeyPublicExponent

}

rsaKeyModulus is the Modulus of the key pair.

rsaKeyPublicExponent is the public exponent of the key pair.

2.87. RegionAlpha

Alphabetic reference to a region within a specified country.

RegionAlpha ::= IA5STRING(SIZE(3))

Value assignment:	
' '	No information available
Spain:	
'AN '	Andalucía
'AR '	Aragón
'AST'	Asturias
'C '	Cantabria
'CAT'	Cataluña
'CL '	Castilla-León
'CM '	Castilla-La-Mancha
'CV '	Valencia
'EXT'	Extremadura
'G '	Galicia
'IB '	Baleares
'IC '	Canarias
'LR '	La Rioja
'M '	Madrid
'MU '	Murcia
'NA '	Navarra
[X2 'PV '	País Vasco]

2.88. RegionNumeric

Numerical reference to a region within a specified country.

RegionNumeric ::= OCTET STRING (SIZE(1))

Value assignment:

′00′H	No information available

Spain:

′01′H	Andalucía
′02′H	Aragón
′03′H	Asturias
′04′H	Cantabria
′05′H	Cataluña
′06′H	Castilla-León
′07′H	Castilla-La-Mancha
′08′H	Valencia
′09′H	Extremadura
′0A′H	Galicia
′0B′H	Baleares
′0C′H	Canarias
′0D′H	La Rioja
′0E′H	Madrid
′0F′H	Murcia
′10′H	Navarra
′11′H	País Vasco.

2.89. RSAKeyModulus

The modulus of a RSA key pair.

RSAKeyModulus ::= OCTET STRING (SIZE(128))

Value assignment: Unspecified.

2.90. RSAKeyPrivateExponent

The private exponent of a RSA key pair.

RSAKeyPrivateExponent ::= OCTET STRING (SIZE(128))

Value assignment: Unspecified.

2.91. RSAKeyPublicExponent

The public exponent of a RSA key pair.

RSAKeyPublicExponent ::= OCTET STRING (SIZE(8))

Value assignment: Unspecified.

2.92. SensorApprovalNumber

Type approval number of the sensor.

SensorApprovalNumber ::= IA5String(SIZE(8))

Value assignment: Unspecified.

2.93. SensorIdentification

Information, stored in a motion sensor, related to the identification of the motion sensor (requirement 077).

SensorIdentification ::= SEQUENCE {

sensorSerialNumber SensorSerialNumber,

sensorApprovalNumber SensorApprovalNumber,

sensorSCIdentifier SensorSCIdentifier,

sensorOSIdentifier SensorOSIdentifier

}

sensorSerialNumber is the extended serial number of the motion sensor (includes part number and manufacturer code).

sensorApprovalNumber is the approval number of the motion sensor.

sensorSCIdentifier is the identifier of the security component of the motion sensor.

sensorOSIdentifier is the identifier of the operating system of the motion sensor.

2.94. SensorInstallation

Information, stored in a motion sensor, related to the installation of the motion sensor (requirement 099).

SensorInstallation ::= SEQUENCE {

sensorPairingDateFirst SensorPairingDate,

firstVuApprovalNumber VuApprovalNumber,

firstVuSerialNumber VuSerialNumber,

sensorPairingDateCurrent SensorPairingDate,

currentVuApprovalNumber VuApprovalNumber,

currentVUSerialNumber VuSerialNumber

}

sensorPairingDateFirst is the date of the first pairing of the motion sensor with a vehicle unit.

firstVuApprovalNumber is the approval number of the first vehicle unit paired with the motion sensor.

firstVuSerialNumber is the serial number of the first vehicle unit paired with the motion sensor.

sensorPairingDateCurrent is the date of the current pairing of the motion sensor with the vehicle unit.

currentVuApprovalNumber is the approval number of the vehicle unit currently paired with the motion sensor.

currentVUSerialNumber is the serial number of the vehicle unit currently paired with the motion sensor.

2.95. SensorInstallationSecData

Information, stored in a workshop card, related to the security data needed for pairing motion sensors to vehicle units (requirement 214).

SensorInstallationSecData ::= TDesSessionKey

Value assignment: in accordance with ISO 16844-3.

2.96. SensorOSIdentifier

Identifier of the operating system of the motion sensor.

SensorOSIdentifier ::= IA5String(SIZE(2))

Value assignment: manufacturer specific.

2.97. SensorPaired

Information, stored in a vehicle unit, related to the identification of the motion sensor paired with the vehicle unit (requirement 079).

SensorPaired ::= SEQUENCE {

sensorSerialNumber SensorSerialNumber,

sensorApprovalNumber SensorApprovalNumber,

sensorPairingDateFirst SensorPairingDate

}

sensorSerialNumber is the serial number of the motion sensor currently paired with the vehicle unit.

sensorApprovalNumber is the approval number of the motion sensor currently paired with the vehicle unit.

sensorPairingDateFirst is the date of the first pairing with a vehicle unit of the motion sensor currently paired with the vehicle unit.

2.98. SensorPairingDate

Date of a pairing of the motion sensor with a vehicle unit.

SensorPairingDate ::= TimeReal

Value assignment: Unspecified.

2.99. SensorSerialNumber

Serial number of the motion sensor.

SensorSerialNumber ::= ExtendedSerialNumber:

2.100. SensorSCIdentifier

Identifier of the security component of the motion sensor.

SensorSCIdentifier ::= IA5String(SIZE(8))

Value assignment: component manufacturer specific.

2.101. Signature

A digital signature.

Signature ::= OCTET STRING (SIZE(128))

Value assignment: in accordance with Appendix 11, 'Common security mechanisms'.

2.102. SimilarEventsNumber

The number of similar events for one given day (requirement 094).

SimilarEventsNumber ::= INTEGER(0..255)

Value assignment: 0 is not used, 1 means that only one event of that type have occurred and have been stored on that day, 2 means that 2 events of that type have occurred on that day (one only has been stored), … 255 means that 255 or more events of that type have occurred on that day.

2.103. SpecificConditionType

Code identifying a specific condition (requirements 050b, 105a, 212a and 230a).

SpecificConditionType ::= INTEGER(0..255)

Value assignment:

′00′H	RFU
′01′H	Out of scope — Begin
′02′H	Out of scope — End
′03′H	Ferry/Train crossing
′04′H .. ′FF′H	RFU.

2.104. SpecificConditionRecord

Information, stored in a driver card, a workshop card or a vehicle unit, related to a specific condition (requirements 105a, 212a and 230a).

SpecificConditionRecord ::= SEQUENCE {

entryTime TimeReal,

specificConditionType SpecificConditionType

}

entryTime is the date and time of the entry.

specificConditionType is the code identifying the specific condition.

2.105. Speed

Speed of the vehicle (km/h).

Speed ::= INTEGER(0..255)

Value assignment: kilometre per hour in the operational range 0 to 220 km/h.

2.106. SpeedAuthorised

Maximum authorised Speed of the vehicle (definition bb)).

SpeedAuthorised ::= Speed.

2.107. SpeedAverage

Average speed in a previously defined duration (km/h).

SpeedAverage ::= Speed.

2.108. SpeedMax

Maximum speed measured in a previously defined duration.

SpeedMax ::= Speed.

2.109. TDesSessionKey

A triple DES session key.

TDesSessionKey ::= SEQUENCE {

tDesKeyA OCTET STRING (SIZE(8))

tDesKeyB OCTET STRING (SIZE(8))

}

Value assignment: not further specified.

2.110. TimeReal

Code for a combined date and time field, where the date and time are expressed as seconds past 00h.00m.00s. on 1 January 1970 GMT.

TimeReal{INTEGER:TimeRealRange} ::= INTEGER(0..TimeRealRange)

Value assignment — Octet Aligned: Number of seconds since midnight 1 January 1970 GMT.

The max possible date/time is in the year 2106.

2.111. TyreSize

Designation of tyre dimensions.

TyreSize ::= IA5String(SIZE(15))

Value assignment: in accordance with Directive 92/23 (EEC) 31.3.1992, OJ L 129, p. 95.

2.112. VehicleIdentificationNumber

Vehicle Identification Number (VIN) referring to the vehicle as a whole, normally chassis serial number or frame number.

VehicleIdentificationNumber ::= IA5String(SIZE(17))

Value assignment: As defined in ISO 3779.

2.113. VehicleRegistrationIdentification

Identification of a vehicle, unique for Europe (VRN and Member State).

VehicleRegistrationIdentification ::= SEQUENCE {

vehicleRegistrationNation NationNumeric,

vehicleRegistrationNumber VehicleRegistrationNumber

}

vehicleRegistrationNation is the nation where the vehicle is registered.

vehicleRegistrationNumber is the registration number of the vehicle (VRN).

[[F3]2.114. VehicleRegistrationNumber

Registration number of the vehicle (VRN). The registration number is assigned by the vehicle licensing authority.

VehicleRegistrationNumber: = SEQUENCE {

codePage INTEGER (0..255),

vehicleRegNumber OCTET STRING (SIZE(13))

}

codePage specifies a character set defined in Chapter 4,

vehicleRegNumber is a VRN encoded using the specified character set.

Value assignment: Country specific.**]**

2.115. VuActivityDailyData

Information, stored in a VU, related to changes of activity and/or changes of driving status and/or changes of card status for a given calendar day (requirement 084) and to slots status at 00.00 that day.

VuActivityDailyData ::= SEQUENCE {

noOfActivityChanges INTEGER SIZE(0..1 440),

activityChangeInfos SET SIZE(noOfActivityChanges) OF ActivityChangeInfo

}

noOfActivityChanges is the number of ActivityChangeInfo words in the activityChangeInfos set.

activityChangeInfos is the set of ActivityChangeInfo words stored in the VU for the day. It always includes two ActivityChangeInfo words giving the status of the two slots at 00.00 that day.

2.116. VuApprovalNumber

Type approval number of the vehicle unit.

VuApprovalNumber ::= IA5String(SIZE(8))

Value assignment: Unspecified.

2.117. VuCalibrationData

Information, stored in a vehicle unit, related to the calibrations of the recording equipment (requirement 098).

VuCalibrationData ::= SEQUENCE {

noOfVuCalibrationRecords INTEGER(0..255),

vuCalibrationRecords SET SIZE(noOfVuCalibrationRecords) OF VuCalibrationRecord

}

noOfVuCalibrationRecords is the number of records contained in the vuCalibrationRecords set.

vuCalibrationRecords is the set of calibration records.

2.118. VuCalibrationRecord

Information, stored in a vehicle unit, related a calibration of the recording equipment (requirement 098).

VuCalibrationRecord ::= SEQUENCE {

calibrationPurpose CalibrationPurpose,

workshopName Name,

workshopAddress Address,

workshopCardNumber FullCardNumber,

workshopCardExpiryDate TimeReal,

vehicleIdentificationNumber VehicleIdentificationNumber,

vehicleRegistrationIdentification VehicleRegistrationIdentification,

wVehicleCharacteristicConstant W-VehicleCharacteristicConstant,

kConstantOfRecordingEquipment K-ConstantOfRecordingEquipment,

lTyreCircumference L-TyreCircumference,

tyreSize TyreSize,

authorisedSpeed SpeedAuthorised,

oldOdometerValue OdometerShort,

newOdometerValue OdometerShort,

oldTimeValue TimeReal,

newTimeValue TimeReal,

nextCalibrationDate TimeReal

}

calibrationPurpose is the purpose of the calibration.

workshopName, workshopAddress are the workshop name and address.

workshopCardNumber identifies the workshop card used during the calibration.

workshopCardExpiryDate is the card expiry date.

vehicleIdentificationNumber is the VIN.

vehicleRegistrationIdentification contains the VRN and registering Member State.

wVehicleCharacteristicConstant is the characteristic coefficient of the vehicle.

kConstantOfRecordingEquipment is the constant of the recording equipment.

lTyreCircumference is the effective circumference of the wheel tyres.

tyreSize is the designation of the dimension of the tyres mounted on the vehicle.

authorisedSpeed is the authorised speed of the vehicle.

oldOdometerValue, newOdometerValue are the old and new values of the odometer.

oldTimeValue, newTimeValue are the old and new values of date and time.

nextCalibrationDate is the date of the next calibration of the type specified in CalibrationPurpose to be carried out by the authorised inspection authority.

2.119. VuCardIWData

Information, stored in a vehicle unit, related to insertion and withdrawal cycles of driver cards or of workshop cards in the vehicle unit (requirement 081).

```
VuCardIWData ::= SEQUENCE {

    noOfIWRecords           INTEGER(0..2^16-1),
    vuCardIWRecords         SET SIZE(noOfIWRecords) OF
[X2]                        VuCardIWRecord          ]

}
```

noOfIWRecords is the number of records in the set vuCardIWRecords.

vuCardIWRecords is a set of records related to card insertion withdrawal cycles.

2.120. VuCardIWRecord

Information, stored in a vehicle unit, related to an insertion and withdrawal cycle of a driver card or of a workshop card in the vehicle unit (requirement 081).

VuCardIWRecord ::= SEQUENCE {

cardHolderName HolderName,

fullCardNumber FullCardNumber,

cardExpiryDate TimeReal,

cardInsertionTime TimeReal,

vehicleOdometerValueAtInsertion OdometerShort,

cardSlotNumber CardSlotNumber,

cardWithdrawalTime TimeReal,

vehicleOdometerValueAtWithdrawal OdometerShort,

previousVehicleInfo PreviousVehicleInfo

manualInputFlag ManualInputFlag

}

cardHolderName is the driver or workshop card holder's surname and first names as stored in the card.

fullCardNumber is the type of card, its issuing Member State and its card number as stored in the card.

cardExpiryDate is the card's expiry date as stored in the card.

cardInsertionTime is the insertion date and time.

vehicleOdometerValueAtInsertion is the vehicle odometer value at card insertion.

cardSlotNumber is the slot in which the card is inserted.

cardWithdrawalTime is the withdrawal date and time.

vehicleOdometerValueAtWithdrawal is the vehicle odometer value at card withdrawal.

previousVehicleInfo contains information about the previous vehicle used by the driver, as stored in the card.

manualInputFlag is a flag identifying if the cardholder has manually entered driver activities at card insertion.

2.121. VuCertificate

Certificate of the public key of a vehicle unit.

VuCertificate ::= Certificate

2.122. VuCompanyLocksData

Information, stored in a vehicle unit, related to company locks (requirement 104).

VuCompanyLocksData ::= SEQUENCE {

noOfLocks INTEGER(0..20),

vuCompanyLocksRecords SET SIZE(noOfLocks) OF VuCompanyLocksRecord

}

noOfLocks is the number of locks listed in vuCompanyLocksRecords.

vuCompanyLocksRecords is the set of company locks records.

2.123. VuCompanyLocksRecord

Information, stored in a vehicle unit, related to one company lock (requirement 104).

VuCompanyLocksRecord ::= SEQUENCE {

lockInTime TimeReal,

lockOutTime TimeReal,

companyName Name,

companyAddress Address,

companyCardNumber FullCardNumber

}

lockInTime, lockOutTime are the date and time of lock-in and lock-out.

companyName, companyAddress are the company name and address related with the lock-in.

companyCardNumber identifies the card used at lock-in.

2.124. VuControlActivityData

Information, stored in a vehicle unit, related to controls performed using this VU (requirement 102).

VuControlActivityData ::= SEQUENCE {

noOfControls INTEGER(0..20),

vuControlActivityRecords SET SIZE(noOfControls) OF VuControlActivityRecord

}

noOfControls is the number of controls listed in vuControlActivityRecords.

vuControlActivityRecords is the set of control activity records.

2.125. VuControlActivityRecord

Information, stored in a vehicle unit, related to a control performed using this VU (requirement 102).

VuControlActivityRecord ::= SEQUENCE {

controlType ControlType,

controlTime TimeReal,

controlCardNumber FullCardNumber,

downloadPeriodBeginTime TimeReal,

downloadPeriodEndTime TimeReal

}

controlType is the type of the control.

controlTime is the date and time of the control.

ControlCardNumber identifies the control card used for the control.

downloadPeriodBeginTime is the begin time of the downloaded period, in case of downloading.

downloadPeriodEndTime is the end time of the downloaded period, in case of downloading.

2.126. VuDataBlockCounter

Counter, stored in a card, identifying sequentially the insertion withdrawal cycles of the card in vehicle units.

VuDataBlockCounter ::= BCDString(SIZE(2))

Value assignment: Consecutive Number with max, value 9 999, starting again with 0.

2.127. VuDetailedSpeedBlock

Information, stored in a vehicle unit, related to the vehicle's detailed speed for a minute during which the vehicle has been moving (requirement 093).

VuDetailedSpeedBlock ::= SEQUENCE {

speedBlockBeginDate TimeReal,

speedsPerSecond SEQUENCE SIZE(60) OF Speed

}

speedBlockBeginDate is the date and time of the first speed value within the block.

speedsPerSecond is the chronological sequence of measured speeds every seconds for the minute starting at speedBlockBeginDate (included).

2.128. VuDetailedSpeedData

Information, stored in a vehicle unit, related to the detailed speed of the vehicle.

VuDetailedSpeedData ::= SEQUENCE

noOfSpeedBlocks INTEGER(0.2^{16}-1),

vuDetailedSpeedBlocks SET SIZE(noOfSpeedBlocks) OF VuDetailedSpeedBlock

}

noOfSpeedBlocks is the number of speed blocks in the vuDetailedSpeedBlocks set.

vuDetailedSpeedBlocks is the set of detailed speed blocks.

2.129. VuDownloadablePeriod

Oldest and latest dates for which a vehicle unit holds data related to drivers activities (requirements 081, 084 or 087).

VuDownloadablePeriod ::= SEQUENCE {

minDownloadableTime TimeReal

maxDownloadableTime TimeReal

}

minDownloadableTime is the oldest card insertion or activity change or place entry date and time stored in the VU.

maxDownloadableTime is the latest card withdrawal or activity change or place entry date and time stored in the VU.

2.130. VuDownloadActivityData

Information, stored in a vehicle unit, related to its last download (requirement 105).

VuDownloadActivityData ::= SEQUENCE {

downloadingTime TimeReal,

fullCardNumber FullCardNumber,

companyOrWorkshopName Name

}

downloadingTime is the date and time of downloading.

fullCardNumber identifies the card used to authorise the download.

companyOrWorkshopName is the company or workshop name.

2.131. VuEventData

Information, stored in a vehicle unit, related to events (requirement 094 except over speeding event).

VuEventData ::= SEQUENCE {

noOfVuEvents INTEGER(0..255),

vuEventRecords SET SIZE(noOfVuEvents) OF VuEventRecord

}

noOfVuEvents is the number of events listed in the vuEventRecords set.

vuEventRecords is a set of events records.

2.132. VuEventRecord

Information, stored in a vehicle unit, related to an event (requirement 094 except over speeding event).

VuEventRecord ::= SEQUENCE {

eventType EventFaultType,

eventRecordPurpose EventFaultRecordPurpose,

eventBeginTime TimeReal,

eventEndTime TimeReal,

cardNumberDriverSlotBegin FullCardNumber,

cardNumberCodriverSlotBegin FullCardNumber,

cardNumberDriverSlotEnd FullCardNumber,

cardNumberCodriverSlotEnd FullCardNumber,

similarEventsNumber SimilarEventsNumber

}

eventType is the type of the event.

eventRecordPurpose is the purpose for which this event has been recorded.

eventBeginTime is the date and time of beginning of event.

eventEndTime is the date and time of end of event.

cardNumberDriverSlotBegin identifies the card inserted in the driver slot at the beginning of the event.

cardNumberCodriverSlotBegin identifies the card inserted in the co-driver slot at the beginning of the event.

cardNumberDriverSlotEnd identifies the card inserted in the driver slot at the end of the event.

cardNumberCodriverSlotEnd identifies the card inserted in the co-driver slot at the end of the event.

similarEventsNumber is the number of similar events that day.

This sequence can be used for all events other than over speeding events.

2.133. VuFaultData

Information, stored in a vehicle unit, related to faults (requirement 096).

VuFaultData ::= SEQUENCE {

noOfVuFaults INTEGER(0..255),

vuFaultRecords SET SIZE(noOfVuFaults) OF VuFaultRecord

}

noOfVuFaults is the number of faults listed in the vuFaultRecords set.

vuFaultRecords is a set of faults records.

2.134. VuFaultRecord

Information, stored in a vehicle unit, related to a fault (requirement 096).

VuFaultRecord ::= SEQUENCE {

faultType EventFaultType,

faultRecordPurpose EventFaultRecordPurpose,

faultBeginTime TimeReal,

faultEndTime TimeReal,

cardNumberDriverSlotBegin FullCardNumber,

cardNumberCodriverSlotBegin FullCardNumber,

cardNumberDriverSlotEnd FullCardNumber,

cardNumberCodriverSlotEnd FullCardNumber

}

faultType is the type of recording equipment fault.

faultRecordPurpose is the purpose for which this fault has been recorded.

faultBeginTime is the date and time of beginning of fault.

faultEndTime is the date and time of end of fault.

cardNumberDriverSlotBegin identifies the card inserted in the driver slot at the beginning of the fault.

cardNumberCodriverSlotBegin identifies the card inserted in the co-driver slot at the beginning of the fault.

cardNumberDriverSlotEnd identifies the card inserted in the driver slot at the end of the fault.

cardNumberCodriverSlotEnd identifies the card inserted in the co-driver slot at the end of the fault.

2.135. VuIdentification

Information, stored in a vehicle unit, related to the identification of the vehicle unit (requirement 075).

VuIdentification ::= SEQUENCE {

vuManufacturerName VuManufacturerName,

vuManufacturerAddress VuManufacturerAddress,

vuPartNumber VuPartNumber,

vuSerialNumber VuSerialNumber,

vuSoftwareIdentification VuSoftwareIdentification,

vuManufacturingDate VuManufacturingDate,

vuApprovalNumber VuApprovalNumber

}

vuManufacturerName is the name of the manufacturer of the vehicle unit.

vuManufacturerAddress is the address of the manufacturer of the vehicle unit.

vuPartNumber is the part number of the vehicle unit.

vuSerialNumber is the serial number of the vehicle unit.

vuSoftwareIdentification identifies the software implemented in the vehicle unit.

vuManufacturingDate is the manufacturing date of the vehicle unit.

vuApprovalNumber is the type approval number of the vehicle unit.

2.136. VuManufacturerAddress

Address of the manufacturer of the vehicle unit.

VuManufacturerAddress ::= Address

Value assignment: Unspecified.

2.137. VuManufacturerName

Name of the manufacturer of the vehicle unit.

VuManufacturerName ::= Name

Value assignment: Unspecified.

2.138. VuManufacturingDate

Date of manufacture of the vehicle unit.

VuManufacturingDate ::= TimeReal

Value assignment: Unspecified.

2.139. VuOverSpeedingControlData

Information, stored in a vehicle unit, related to over speeding events since the last over speeding control (requirement 095).

VuOverSpeedingControlData ::= SEQUENCE {

lastOverspeedControlTime TimeReal,

firstOverspeedSince TimeReal,

numberOfOverspeedSince OverspeedNumber

}

lastOverspeedControlTime is the date and time of the last over speeding control.

firstOverspeedSince is the date and time of the first over speeding following this over speeding control.

numberOfOverspeedSince is the number of over speeding events since the last over speeding control.

2.140. VuOverSpeedingEventData

Information, stored in a vehicle unit, related to over speeding events (requirement 094).

VuOverSpeedingEventData ::= SEQUENCE {

noOfVuOverSpeedingEvents INTEGER(0..255),

vuOverSpeedingEventRecords SET SIZE(noOfVuOverSpeedingEvents) OF VuOverSpeedingEventRecord

}

noOfVuOverSpeedingEvents is the number of events listed in the vuOverSpeedingEventRecords set.

vuOverSpeedingEventRecords is a set of over speeding events records.

2.141. VuOverSpeedingEventRecord

Information, stored in a vehicle unit, related to over speeding events (requirement 094).

VuOverSpeedingEventRecord ::= SEQUENCE {

eventType EventFaultType,

eventRecordPurpose EventFaultRecordPurpose,

eventBeginTime TimeReal,

eventEndTime TimeReal,

maxSpeedValue SpeedMax,

averageSpeedValue SpeedAverage,

cardNumberDriverSlotBegin FullCardNumber,

similarEventsNumber SimilarEventsNumber

}

eventType is the type of the event.

eventRecordPurpose is the purpose for which this event has been recorded.

eventBeginTime is the date and time of beginning of event.

eventEndTime is the date and time of end of event.

maxSpeedValue is the maximum speed measured during the event.

averageSpeedValue is the arithmetic average speed measured during the event.

cardNumberDriverSlotBegin identifies the card inserted in the driver slot at the beginning of the event.

similarEventsNumber is the number of similar events that day.

2.142. VuPartNumber

Part number of the vehicle unit.

VuPartNumber ::= IA5String(SIZE(16))

Value assignment: VU manufacturer specific.

2.143. VuPlaceDailyWorkPeriodData

Information, stored in a vehicle unit, related to places where drivers begin or end a daily work periods (requirement 087).

VuPlaceDailyWorkPeriodData ::= SEQUENCE {

noOfPlaceRecords INTEGER(0..255),

vuPlaceDailyWorkPeriodRecords SET SIZE(noOfPlaceRecords) OF VuPlaceDailyWorkPeriodRecord

}

noOfPlaceRecords is the number of records listed in the vuPlaceDailyWorkPeriodRecords set.

vuPlaceDailyWorkPeriodRecords is a set of place related records.

2.144. VuPlaceDailyWorkPeriodRecord

Information, stored in a vehicle unit, related to a place where a driver begins or ends a daily work period (requirement 087).

VuPlaceDailyWorkPeriodRecord ::= SEQUENCE {

fullCardNumber FullCardNumber,

placeRecord PlaceRecord

}

fullCardNumber is the driver's card type, card issuing Member State and card number.

placeRecord contains the information related to the place entered.

2.145. VuPrivateKey

The private key of a vehicle unit.

VuPrivateKey ::= RSAKeyPrivateExponent

2.146. VuPublicKey

The public key of a vehicle unit.

VuPublicKey ::= PublicKey

2.147. VuSerialNumber

Serial number of the vehicle unit (requirement 075).

VuSerialNumber ::= ExtendedSerialNumber

2.148. VuSoftInstallationDate

Date of installation of the vehicle unit software version.

VuSoftInstallationDate ::= TimeReal

Value assignment: Unspecified.

2.149. VuSoftwareIdentification

Information, stored in a vehicle unit, related to the software installed.

VuSoftwareIdentification ::= SEQUENCE {

vuSoftwareVersion VuSoftwareVersion,

vuSoftInstallationDate VuSoftInstallationDate

}

vuSoftwareVersion is the software version number of the Vehicle Unit.

vuSoftInstallationDate is the software version installation date.

2.150. VuSoftwareVersion

Software version number of the vehicle unit.

VuSoftwareVersion ::= IA5String(SIZE(4))

Value assignment: Unspecified.

2.151. VuSpecificConditionData

Information, stored in a vehicle unit, related to specific conditions.

VuSpecificConditionData ::= SEQUENCE {

noOfSpecificConditionRecords INTEGER(0..2^{16}-1)

specificConditionRecords SET SIZE (noOfSpecificConditionRecords) OF SpecificConditionRecord

}

noOfSpecificConditionRecords is the number of records listed in the specificConditionRecords set.

specificConditionRecords is a set of specific conditions related records.

2.152. VuTimeAdjustmentData

Information, stored in a vehicle unit, related to time adjustments performed outside the frame of a regular calibration (requirement 101).

VuTimeAdjustmentData ::= SEQUENCE {

noOfVuTimeAdjRecords INTEGER(0..6),

vuTimeAdjustmentRecords SET SIZE(noOfVuTimeAdjRecords) OF VuTimeAdjustmentRecord

}

noOfVuTimeAdjRecords is the number of records in vuTimeAdjustmentRecords.

vuTimeAdjustmentRecords is a set of time adjustment records.

2.153. VuTimeAdjustmentRecord

Information, stored in a vehicle unit, related to a time adjustment performed outside the frame of a regular calibration (requirement 101).

VuTimeAdjustmentRecord ::= SEQUENCE {

[[X3]. . . . . .**]**

Editorial Information

X3 Deleted by Corrigendum to Commission Regulation (EC) No 1360/2002 of 13 June 2002 adapting for the seventh time to technical progress Council Regulation (EEC) No 3821/85 on recording equipment in road transport (Official Journal of the European Communities L 207 of 5 August 2002).

oldTimeValue TimeReal,

newTimeValue TimeReal,

workshopName Name,

workshopAddress Address,

workshopCardNumber FullCardNumber

}

oldTimeValue, newTimeValue are the old and new values of date and time.

workshopName, workshopAddress are the workshop name and address.

workshopCardNumber identifies the workshop card used to perform the time adjustment.

2.154. W-VehicleCharacteristicConstant

Characteristic coefficient of the vehicle (definition k)).

W-VehicleCharacteristicConstant ::= INTEGER(0..2^{16}-1))

Value assignment: Impulses per kilometre in the operating range 0 to 64 255 pulses/km.

2.155. WorkshopCardApplicationIdentification

Information, stored in a workshop card related to the identification of the application of the card (requirement 190).

WorkshopCardApplicationIdentification ::= SEQUENCE {

typeOfTachographCardId EquipmentType,

cardStructureVersion CardStructureVersion,

noOfEventsPerType NoOfEventsPerType,

noOfFaultsPerType NoOfFaultsPerType,

activityStructureLength CardActivityLengthRange,

noOfCardVehicleRecords NoOfCardVehicleRecords,

noOfCardPlaceRecords NoOfCardPlaceRecords,

noOfCalibrationRecords NoOfCalibrationRecords

}

typeOfTachographCardId is specifying the implemented type of card.

cardStructureVersion is specifying the the version of the structure that is implemented in the card.

noOfEventsPerType is the number of events per type of event the card can record.

noOfFaultsPerType is the number of faults per type of fault the card can record.

activityStructureLength indicates the number of bytes available for storing activity records.

noOfCardVehicleRecords is the number of vehicle records the card can contain.

noOfCardPlaceRecords is the number of places the card can record.

noOfCalibrationRecords is the number of calibration records the card can store.

2.156. WorkshopCardCalibrationData

Information, stored in a workshop card, related to workshop activity performed with the card (requirements 227 and 229).

WorkshopCardCalibrationData ::= SEQUENCE {

calibrationTotalNumber INTEGER(0..2^{16}-1),

calibrationPointerNewestRecord INTEGER(0..NoOfCalibrationRecords-1),

calibrationRecords SET SIZE(NoOfCalibrationRecords) OF WorkshopCardCalibrationRecord

}

calibrationTotalNumber is the total number of calibrations performed with the card.

calibrationPointerNewestRecord is the index of the last updated calibration record.

Value assignment: Number corresponding to the numerator of the calibration record, beginning with '0' for the first occurrence of the calibration records in the structure.

calibrationRecords is the set of records containing calibration and/or time adjustment information.

2.157. WorkshopCardCalibrationRecord

Information, stored in a workshop card, related to a calibration performed with the card (requirement 227).

WorkshopCardCalibrationRecord ::= SEQUENCE {

calibrationPurpose CalibrationPurpose,

vehicleIdentificationNumber VehicleIdentificationNumber,

vehicleRegistration VehicleRegistrationIdentification,

wVehicleCharacteristicConstant W-VehicleCharacteristicConstant,

kConstantOfRecordingEquipment K-ConstantOfRecordingEquipment,

lTyreCircumference L-TyreCircumference,

tyreSize TyreSize,

authorisedSpeed SpeedAuthorised,

oldOdometerValue OdometerShort,

newOdometerValue OdometerShort,

oldTimeValue TimeReal,

newTimeValue TimeReal,

nextCalibrationDate TimeReal,

vuPartNumber VuPartNumber,

vuSerialNumber VuSerialNumber,

sensorSerialNumber SensorSerialNumber

}

calibrationPurpose is the purpose of the calibration.

vehicleIdentificationNumber is the VIN.

vehicleRegistration contains the VRN and registering Member State.

wVehicleCharacteristicConstant is the characteristic coefficient of the vehicle.

kConstantOfRecordingEquipment is the constant of the recording equipment.

lTyreCircumference is the effective circumference of the wheel tyres.

tyreSize is the designation of the dimensions of the tyres mounted on the vehicle.

authorisedSpeed is the maximum authorised speed of the vehicle.

oldOdometerValue, newOdometerValue are the old and new values of the odometer.

oldTimeValue, newTimeValue are the old and new values of date and time.

nextCalibrationDate is the date of the next calibration of the type specified in CalibrationPurpose to be carried out by the authorised inspection authority.

vuPartNumber, vuSerialNumber and **sensorSerialNumber** are the data elements for recording equipment identification.

2.158. WorkshopCardHolderIdentification

Information, stored in a workshop card, related to the identification of the cardholder (requirement 216).

WorkshopCardHolderIdentification ::= SEQUENCE {

workshopName Name,

workshopAddress Address,

cardHolderName HolderName,

cardHolderPreferredLanguage Language

}

workshopName is name of the workshop of the card holder.

workshopAddress is the address of the workshop of the card holder.

cardHolderName is the name and first name(s) of the holder (e.g. the name of the mechanic).

cardHolderPreferredLanguage is the preferred language of the card holder.

2.159. WorkshopCardPIN

Personal identification number of the Workshop Card (requirement 213).

WorkshopCardPIN ::= IA5String(SIZE(8))

Value assignment: The PIN known to the cardholder, right padded with ‘FF’ bytes up to 8 bytes.**]]**

Status:
Point in time view as at 31/01/2020.

Changes to legislation:
There are currently no known outstanding effects for the Council Regulation (EEC) No 3821/85, Division 2..