Open Communication Interface for Road Traffic Control Systems

Offene Schnittstellen für die Straßenverkehrstechnik

OCIT-O Outstations
Lichtsignalsteuergeräte

OCIT-O_Lstg_V3.0_A02

OCIT Developer Group (ODG) & Partner

OCIT-O Outstations
Lichtsignalsteuergeräte

Dokument: OCIT-O_Lstg_V3.0_A02

Herausgeber: ODG & Partner

Kontakt: www.ocit.org

Inhaltsverzeichnis

Dokumentenstand **6**

Spezifikationen **6**

1 Einführung **7**

1.1 Unterstützte Funktionen 7

1.2 Unterstützung von OCIT-C Schnittstellen 7

2 Geräte- und Systemfunktionen **9**

2.1 Lichtsignalsteuergeräte mit OCIT-O Schnittstelle 9

2.2 Fernversorgung von Lichtsignalsteuergeräten 9

2.3 Versorgungswege 10

2.3.1 Versorgungsdaten 12

2.3.2 Versionierung 15

2.3.3 Anforderungen an den Versorgungsdatenserver 15

2.3.4 Anforderungen an die Lichtsignalsteuergeräte 15

2.4 Übertragungsgeschwindigkeit 16

2.5 Synchronisierung u. Rückrechenverfahren 16

2.5.1 Berechnungsbeispiele für Rückrechenverfahren 18

2.6 Teilknoten 19

2.7 Relative Knoten 19

2.8 Zeitstempel 19

2.9 Zeitzählung 19

2.9.1 Zeitschaltwerte und Umlauf 20

3 Objektdefinitionen **21**

3.1 Übertragung von Versorgungsdaten 21

3.1.1 Objekt Transaction 21

3.1.2 Objekt SupplyTransaction (Versorgungstransaktion) 31

3.1.3 Objekt TransferParameterBlock 37

3.2 Versionierung der Versorgungsdaten 40

3.2.1 Standard-Verfahren zur Checksummenbildung 48

3.2.2 Objekt Versionierung 52

3.3 Versorgungsobjekte 59

3.3.1 Objekt VersorgbaresObjekt 60

3.3.2 Block 1: Verkehrstechnische Grunddaten / Festzeit 61

3.3.3 Block 2: Daten mit Netzbezug 81

3.3.4 Block 3: VA-Steuerverfahren 89

3.3.5 Block 4: VA-Parameter 90

3.3.6 Block 5: MAP Topologiedaten 91
3.4 Zentrale Schaltwünsche und Statuselemente 92
3.4.1 OCIT-O konforme Konstellationen der zentralen Schaltwünsche 94
3.4.2 Struktur ZEITINTERVALL....97
3.4.3 Typen und Pfade....98
3.4.4 Objekt ZSignalProgramm 98
3.4.5 Objekt ZKnotenEinAus 101
3.4.6 Objekt ZTeilKnoten....103
3.4.7 Objekt ZSondereingriff 106
3.4.8 Signalprogramm Modifikationen 108
3.4.9 Projektspezifische Modifikationen....117
3.4.10 Objekt ZentralenSchaltwunsch....120
3.4.11 Objekt ISignalProgramm 124
3.4.12 Objekt IKnotenEinAus 124
3.4.13 Objekt ITeilknoten 124
3.4.14 Objekt ISondereingriff....125
3.4.15 Objekt IVAEinAus....125
3.4.16 Objekt IVAIndividualverkehrEinAus 126
3.4.17 Objekt IOepnvEinAus 126
3.4.18 Objekt IBetriebsart....127
3.4.19 Objekt IstVektor....129
3.4.20 Objekt Gerätestatus 131
3.4.21 Objekt DeviceState (Gerätestatus mit Zeitstempel) 132
3.5 Meldungen und Messwerte 133
3.5.1 Objekttypen und Klassenübersicht 135
3.5.2 Messwertaufträge für Lichtsignalanlagen 137
3.5.3 Auftragselemente 149
3.5.4 AP-Werte 167
3.5.5 Detektoren und Signale....175
3.5.6 Archive der Lichtsignalsteuergeräte 178
4 Car-2-X Kommunikation....192
4.1 Statusinformationen....192
4.1.1 Auftrag MWAuftragSingleCarExt 193
4.1.2 Auftragselement für aggregierte spur - bezogenen Werte 195
4.1.3 Auftragselement für aggregierte Verbindungs - bezogenen Werte 198
4.2 OEV – Priorisierung 199
4.2.1 Auftrag für CAM-Prio-Telegramme....199
4.2.2 Objekt Oepnv 200

4.3 Gefahrenmeldungen 201
4.4 Abfrage von Signalplandaten 201
4.5 Objekt Spat 202
4.5.1 Auftragselement AeSpat 206
4.6 Objekt Map 208
4.6.1 Enum VDArt 209
4.6.2 Digitale Topologie des Verkehrsknoten 210
Referenzen 211
Abbildungen 211
Glossar 212

Dokumentenstand

Version Zustand	Datum	Verteiler	Kommentar
V3.0 A01	15.03.2018	PUBLIC	OCIT-O V3.0
V3.0 A02	19.04.2024	PUBLIC	3.3.6 / 4.6 Referenz auf neues OCIT-C MAP-Datenschema MAPEM-PDU-Descriptions.xsd angepasst. 3.3.6 / 4.6 Verweise auf ODG_MAP_ADD_ON wurden entfernt 3.4.21 / 3.5.6.3 DeviceState um RSUState ergänzt 3.5.4 Anpassung der AP-Werte Beschreibung 3.5.4.5 / 3.5.4.6 Ergänzung der „41.X“ Bezeichnung 3.5.6 Anpassung der Beschreibung der Archive 37, 38 und 39 4.2.2 Oepnv.SetAnforderung um Timestamp ergänzt

Spezifikationen

Das **OCIT-Outstations Konfigurationsdokument OCIT-O KD V3.0** enthält eine Übersicht über alle von der ODG urheberrechtlich verwalteten Spezifikationen und ordnet Versionen und Ausgabestände nach:

- zusammengehörenden Spezifikationen der Schnittstelle „OCIT-Outstations für Lichtsignalsteuergeräte“ mit Referenz auf die dazugehörigen OCIT-C Spezifikationen (siehe dazu Hinweis in 1.2),
- gibt Hinweise zum Einsatz der Übertragungsprofile und
- enthält eine Übersicht über Pakete von Spezifikationen für Schnittstellen, für deren Nutzung von der ODG eine Schutzgebühr verlangt wird

Der jeweils aktuelle Stand ist auf www.ocit.org veröffentlicht.

1 Einführung

In diesem Dokument werden alle für die OCIT-Schnittstelle zwischen einer Zentrale und den Lichtsignalsteuergeräten relevanten Funktionen festgelegt. Die herausragend neue Funktion in OCIT-O Lstg V3.0 ist die Möglichkeit C2X Daten von einer Road-Side-Unit (RSU) für die Lichtsignalsteuerung zu nutzen.

1.1 Unterstützte Funktionen

Die Schnittstelle OCIT-Outstations für Lichtsignalsteuergeräte in der vorliegenden Version basiert auf den vorne aufgeführten Referenzspezifikationen.

Eine OCIT-Outstations-Schnittstelle kann unterschiedliche Übertragungsprofile benutzen, die in den optionalen Definitionen festgelegt sind.

Es ist nicht verpflichtend, dass an OCIT-Outstations betriebene Gerätschaften alle in den Referenzspezifikationen festgelegten Funktionen unterstützen. Sie unterstützen nur diejenigen Funktionen, die für den jeweiligen Zweck und Ausbau notwendig sind. So wird z. B. ein Lichtsignalsteuergerät für Fußgängerüberwege weniger Funktionen unterstützen als ein Gerät mit verkehrsabhängiger ÖPNV-Bevorzugung. Die Nichtverfügbarkeit eines von der Zentrale aufgerufenen Leistungsmerkmals muss zu einer erkennbaren Reaktion (Returncode) des Lichtsignalsteuergeräts führen.

Die Spezifikationen der Schnittstelle OCIT-Outstations Version 3.0 für Lichtsignalsteuergeräte sind rückwärtskompatibel zu Zentralen mit OCIT-O Version 1.x. und Version 2.0

Hinweis: In der Version 2.x wurden die zentralen Schaltkombinationen eindeutig definiert, in der Version 1.x war dies jedoch nicht der Fall. Wenn Lichtsignalsteuergeräte mit Version 2.x an Zentralen mit Version 1.x betrieben werden, kann daher bei Geräten unterschiedlicher Hersteller weiterhin ein abweichendes Schaltverhalten auftreten. Abhilfe ist durch Anpassung der zentralen Schaltwünsche an die Festlegungen nach Version 2.x möglich.

Neue oder erweiterte Funktionen in OCIT-O Lstg V3.0:

3.4.21	Objekt DeviceState (Gerätestatus mit Zeitstempel)
4	Car-2-X Kommunikation
4.1	Statusinformationen
4.2	OEV – Priorisierung
4.3	Gefahrenmeldungen
4.4	Abfrage von Signalplandaten
4.5	Objekt Spat
4.6	Objekt Map

1.2 Unterstützung von OCIT-C Schnittstellen

Die Lichtsignalsteuergeräte mit OCIT-O Lstg Version 3 stellen standardisierte Daten und Funktionen bereit, die auf Festlegungen in den OCIT-C Spezifikationen für LSA beruhen.

Folgende OCIT-C Schnittstellen werden unterstützt:

- **OCIT-C Versorgungsdaten**

  Das herausragende Merkmal von OCIT-O Lstg V2.0 ist die standardisierte Fernversorgung der Lichtsignalsteuergeräte von einem Planungsplatz aus. Dazu wurden Versorgungsdaten, die aus verkehrstechnischen Gründen häufig geändert werden müssen, in OCIT-O standardisiert. Diese in OCIT-O standardisierten Versorgungsdaten werden ab der OCIT-O Version 2.0 als "Anwenderversorgung" bezeichnet. Sie sind ein Teil der in OCIT-C VD festgelegten, wesentlich umfangreicheren Daten, die von den Planungsplätzen genutzt werden. Die Festlegungen dazu finden sich als Standard sowohl in der OCIT-C Dokumentation (OCIT-C-Daten) als auch korrespondierend dazu in diesem Dokument. Die Formatumsetzung von OCIT-O auf OCIT-C übernimmt der OCIT Systemkomponente VD Server. Die Realisierung des VD Servers erfolgt gemäß den Festlegungen in OCIT-C-Daten und OCIT-O Lstg V3.0 Spezifikation.

- **OCIT-C Prozessdatenerfassung**

  Prozessdaten sind Daten und Messwerte, die vom Lichtsignalsteuergerät erfasst werden. Die Erfassung und Bereitstellung der Prozessdaten im Format OCIT-O wurde bereits mit der ersten OCIT-O Version realisiert. Ab OCIT-O Lstg V2.0 kommen die erweiterten Detektorwerte hinzu (siehe Pkt. 3.5.2.6). Ab OCIT-O Lstg V3.0 kommen Daten für C2X-Anwendungsfälle hinzu. Weitere Festlegungen dazu werden in diesem Dokument nicht getroffen. Die Formatumsetzung von OCIT-O auf OCIT-C übernimmt die OCIT-C Komponente PD Server. Die Realisierung des PD Servers erfolgt gemäß den Festlegungen in OCIT-C-Daten und OCIT-O Lstg V3.0 Spezifikation.

- **OCIT-C Verkehrsmanagementdatenerfassung**

  Die Verkehrsmanagementdaten gehen über LSA-bezogene Daten hinaus und werden hier nicht behandelt.

Hinweis: OCIT-C löst die bisher gebräuchlichen Funktionen von OCIT-I-VD-DM-LSA und OCIT-I-PD-DM-LSA ab. Die entsprechenden Schemata sind auf http://www.ocit.org/downloadOCIT-C.htm veröffentlicht. Formatumsetzungen erfolgen wie bei OCIT-I über entsprechende Server. Die OCIT-C Version 1 wurde auch als Vornorm veröffentlicht (DIN V VDE V 0832 - Straßenverkehrs-Signalanlagen - Teil 601 und Teil 602: Schnittstelle zwischen zentralen Einrichtungen zum Austausch verkehrsbezogener Daten).

2 Geräte- und Systemfunktionen

In diesem Kapitel finden sich Festlegungen, die für den Betrieb der definitionsgemäßen Lichtsignalsteuergeräte notwendig sind und die keine Festlegung als OCIT-Outstations-Objekte verlangen.

2.1 Lichtsignalsteuergeräte mit OCIT-O Schnittstelle

Auf Grund des Zeitverhaltens des OCIT-Outstations-Protokolls sind OCIT-Lichtsignalsteuergeräte speziell für Einsatz in dezentral aufgebauten Systemen gebaut. Sie beherrschen komplexe lokale Verkehrsabhängigkeiten und können Verkehrsmesswerte erfassen und verarbeiten („intelligente Steuergeräte“). Sie besitzen folgende charakteristischen Eigenschaften:

- Sie verfügen über leistungsfähige Mikroprozessoren, die komplexe Verkehrsabhängigkeiten lokal beherrschen und eine Verarbeitung von Messwerten durchführen.
- Sie verfügen über genaue Uhren, die Synchronisiervorgänge steuern, und deren Zeit zur Kennzeichnung von Ereignissen dient.
- Schaltvorgänge werden durch Signalprogramme gesteuert, wobei folgende Vorgaben gemacht werden:
    - Vordefinierte Signalpläne, die entweder im Gerät gespeichert sind, und/oder über die Zentrale im laufenden Betrieb versorgt werden können, werden über Schaltbefehle der Zentrale oder über interne Schalttabellen ausgewählt.
    - Wird eine lokale verkehrsabhängige Logik verwendet, werden ausgewählte Signalprogramme entsprechend der Verkehrssituation variiert.
    - Die verkehrsabhängige Logik ihrerseits ist durch Parameter auf verschiedene Situationen einzustellen.
- Nicht für den Betrieb an der Schnittstelle OCIT-O Lstg ausgelegt sind folgende Lichtsignalsteuergeräte:
    - So genannte Schaltgeräte, deren Kennzeichen es ist, dass Schaltvorgänge vorzugsweise von der Zentrale veranlasst werden und innerhalb von einer Sekunde ausgeführt werden und die bei Ausfall der Zentrale nur einen Notbetrieb gewährleisten.
    - Gruppensteuergeräte, daher Geräte, die zentrale Steueraufgaben übernehmen, und kleinere Gerätegruppen über eigene Schnittstellen steuern.

2.2 Fernversorgung von Lichtsignalsteuergeräten

In OCIT-C werden die von Planungsplätzen erzeugten Versorgungsdaten für Lichtsignalsteuergeräte spezifiziert. Das Planungsergebnis für eine Geräteversorgung ist eine OCIT-C -LSA konforme XML-Datei (in diesem Papier kurz als XML-Versorgungsdatei bezeichnet), die die in Blöcke gegliederten Versorgungs- und Planungsdaten enthält.

Eine standardisierte Untermenge der Versorgungsdaten (die sog. Anwenderversorgung) kann von der Zentrale oder vom VIAP aus, in standardisierter Weise in Lichtsignalsteuergeräte die die Festlegungen ab OCIT-O Lstg V2.0 unterstützen, übertragen werden. Diese Anwenderversorgung ist in jedem störungsfreien Gerätezustand möglich. Die nicht standardisierten Versorgungsteile werden mit proprietären Mitteln versorgt.

2.3 Versorgungswege

Das als XML-Versorgungsdatei vorliegende Planungsergebnis (als Gesamtversorgung) kann nicht ohne weiteres in das Lichtsignalsteuergerät übertragen werden:

- Das Planungsergebnis für eine Geräteversorgung ist eine OCIT-C_LSA_Versorgungsdaten konforme XML-Datei, die die gesamten in OCIT-C standardisierten Versorgungsdaten enthalten kann. Lichtsignalsteuergeräte mit OCIT-O Lstg V2.0 unterstützen davon das Subset „Anwenderversorgung“, das sind verkehrstechnisch häufig zu ändernden Versorgungsdaten. Die Anwenderversorgung ist in OCIT-C und OCIT-O korrespondierend spezifiziert.
- Die in der XML-Versorgungsdatei enthaltenen Daten der OCIT-C_LSA_Versorgungsdaten „Gerätetechnik“ und „Sicherheitstechnik“ können nur herstellerspezifisch und nicht über den VD Server versorgt werden.
- In den Spezifikationen zu OCIT-C werden andere Datenformate und Protokolle als in OCIT-Outstations verwendet. Es ist daher notwendig einen Umsetzer, den so genannten VD Server in die Versorgungskette einzubinden. Dieser extrahiert die in OCIT-C enthaltenen und von OCIT-O Lstg ab V2.0 unterstützten Versorgungsdaten und setzt sie in das Format OCIT-O Lstg um. Die Realisierung des VD Servers erfolgt gemäß den Festlegungen in OCIT-C_LSA_Versorgungsdaten (siehe Referenzspezifikationen) und OCIT-O Lstg V3.0.

Der VD Server kann integriert werden (*Abbildung 1*):

- in die zentrale Systemrealisierung
  Die Versorgungsdaten gelangen hier vom Planungswerkzeug über die Schnittstelle OCIT-C_LSA_Versorgungsdaten zum VD-Server, werden dort in das OCIT-O Format gewandelt und über die zentrale Kommunikationskomponente zum Lichtsignalsteuergerät übertragen, und /oder
- in ein Planungs- oder Versorgungswerkzeug
  Die Versorgungsdaten gelangen hier vom einem, im Werkzeug integrierten VD-Server über einen Systemzugang mittels OCIT-O Protokoll zum Lichtsignalsteuergerät.

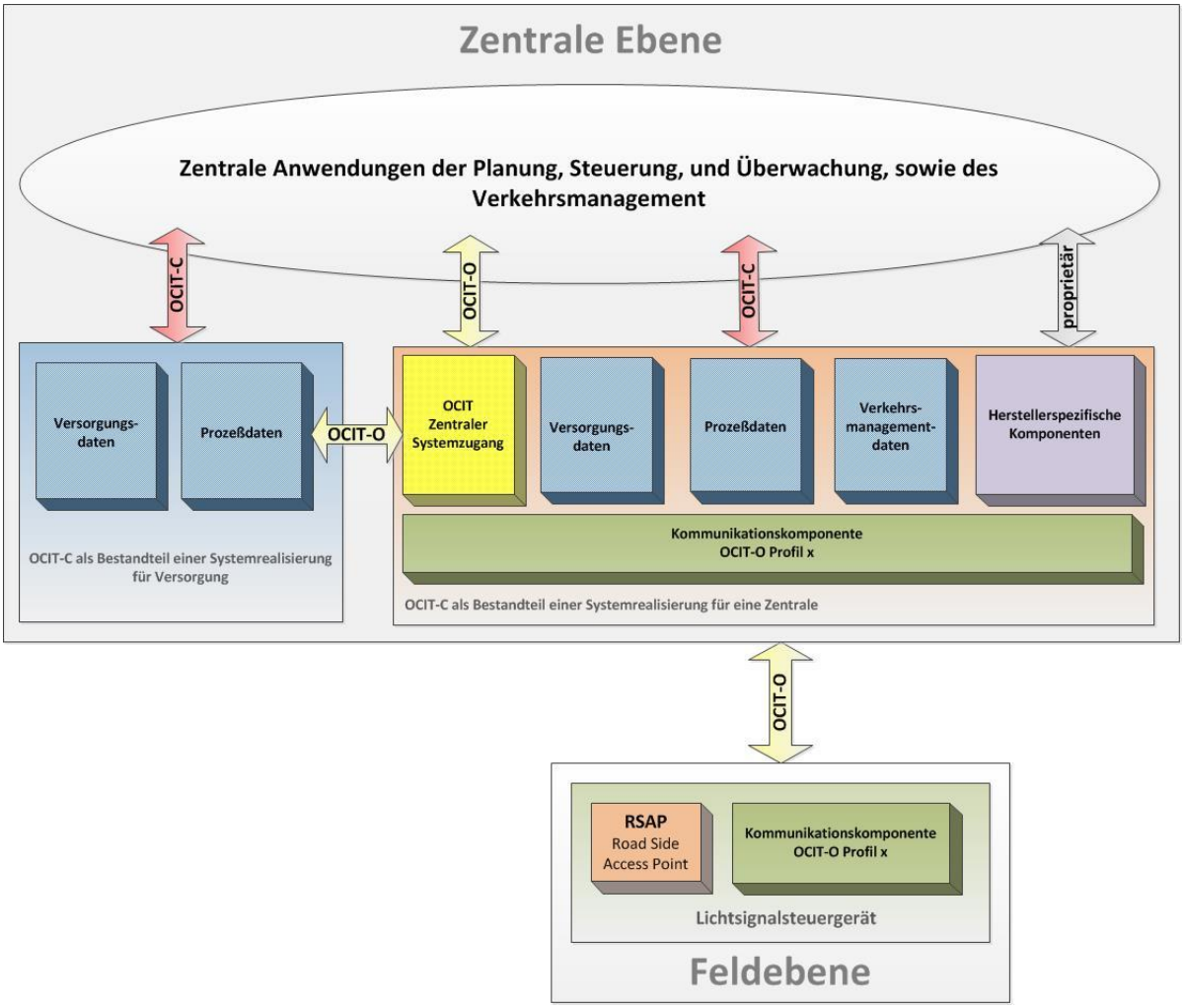


Abbildung 1: Lichtsignalsteuerungssystem mit OCIT-C Schnittstellen [1]

Der Versorgungsvorgang für die von OCIT-O Lstg ab V2.0 unterstützten Daten kann herstellerübergreifend direkt vom Standard-Planungs- und/oder Versorgungswerkzeug aus gestartet werden. Der VD Server steuert dann den geräteseitigen Versorgungsablauf.

Hinweis: Das Protokoll OCIT-C sieht „blockierende Funktionsaufrufe“ vor. Diese Einschränkung kann dazu führen, dass beim Aufruf von Funktionen am VD-Server, so lange gewartet werden muss, bis das Ergebnis des Aufrufs gemeldet wird. Dabei können im Dialog zwischen VIAP und Lichtsignalsteuergeräten lange Antwortzeiten – im Bereich von mehreren Minuten – auftreten.

Die Antwortzeiten werden beeinflusst durch:

- Bearbeitungszeit im VD-Server
- Übertragungszeit zur LSA (Geschwindigkeit und Dateigröße)
- Zeit für die Überprüfung der Daten in den Lichtsignalsteuergeräten

[1] Der „OCIT-O Lokaler Systemzugang“ wird in OCIT-O Lstg nicht angeboten.

- Je nach VIAP-Bedienkonzept Zeit für die Aktivierung (bis zu 2 Signalplanumläufe)
- Bei Profil 2 mit GSM ist wegen der gegenüber Festverbindungen geringeren Übertragungskapazität und der für den Verbindungsaufbau benötigten Zeit mit weiteren, im Dokument OCIT-O-Profil_2 im Anhang beschriebenen Einschränkungen zu rechnen.

2.3.1 Versorgungsdaten

Die Versorgung des Lichtsignalsteuergeräts teilt sich von der Handhabung her auf, in

- die in OCIT-C LSA Versorgungsdaten standardisierte Anwenderversorgung der Verkehrstechnik die von einem Standard-Versorgungswerkzeug aus erfolgen kann, und
- die Herstellerversorgung ist in OCIT-C LSA Versorgungsdaten standardisierten Daten der Sicherheits- und Gerätetechnik, sowie von proprietären Daten. Die Herstellerversorgung wird mit Mitteln der jeweiligen Gerätehersteller vorgenommen.

Hinweis: Die Sorgfaltspflicht auf Stimmigkeit der Versorgung liegt beim Versorger. In den Geräten ist vor der Aktivierung der Versorgung nur eine grobe Prüfung möglich. Es ist daher nicht ausgeschlossen, dass das Gerät als Folge einer fehlerhaften Versorgung eine Sicherheitsabschaltung durchführt.

Die Versorgungdaten sind entsprechend ihrer Aufgabe in Blöcke gegliedert. Entsprechend den Festlegungen erfolgt die Anwenderversorgung immer blockweise, das heißt, auch bei einer Änderung nur eines Wertes werden immer alle Daten eines Blocks versorgt.

Folgendes Schema zeigt die Versorgungsdatenblöcke:

In OCIT-LSA Versorgungsdaten standardisierte Versorgungsdaten, herstellerübergreifend versorgbar und auslesbar **Anwenderversorgung**				
Verkehrstechnik				
Verkehrs-technische Grunddaten / Festzeit	Daten mit Netz-bezug	VA-Steuerverfahren	VA-Parameter	Map Versor-gung
Einschalt-programme	Kopfdaten	Anwendungs-spezifische Da-teien (Binär)	Anwendungs-spezifische Da-teien (Binär)	Anwendungs-spezifische Dateien
Ausschaltpro-gramme	JAUT: Tagesplan			
Signalprogram-me	JAUT: Wochenplan			
Versatzzeiten-matrizen	JAUT: Sondertag jähr-lich			
VT-Zwischenzeiten-matrizen	JAUT: Sondertag Aufzählung			
VT-Mindestfreigabe-zeiten	JAUT: Zeitbereich			
VT-Mindestsperrzei-ten				

Abbildung 2: Schema des Versorgungsdatenblock Anwenderversorgung

In OCIT-C LSA Versorgungsdatenstandardisierte Versorgungsdaten, nur proprietär versorgbar und auslesbar **Herstellerversorgung**			
Gerätetechnik		**Sicherheitstechnik**	
OCIT-C Versor-gungsdaten	Proprietäre Daten	OCIT-C Sicherheits-daten	Proprietäre Daten
Detektoren bzw. digi-tale Eingänge	Gerätetechnische Versorgungen	Sicherheitsrelevante Unverträglichkeits-matrix	Übergänge, Farb-feindlichkeiten..
Signalgruppen bzw. digitale Ausgänge		Sicherheitsrelevante Zwischenzeiten-matrix	
Zuordnung zum Teilknoten		Sicherheitsrelevante Mindest-freigabe-zeiten	
Übergangszeiten		Sicherheitsrelevante Mindestsperrzeiten	
ÖV-Meldepunkte und Meldeketten			

Abbildung 3: Schema des Versorgungsdatenblock Herstellerversorgung

Inhalt der Versorgungsdatenblöcke der **Anwenderversorgung:**

- **Block 1: Verkehrstechnische Grunddaten / Festzeit**

  n Einschaltprogramme (EProgramm)

  n Ausschaltprogramme (AProgramm)

  n Signalprogramme (SignalprogrammV)

  n Versatzzeitenmatrizen (Versatzzeitenmatrix)

  n Verkehrstechnische Zwischenzeitenmatrizen (VTZwischenzeitenmatrix)

  n Verkehrstechnische Mindestfreigabezeiten (VTMinFreigabe)

  n Verkehrstechnische Mindestsperrzeiten (VTMinGesperrt)

- **Block 2: Daten mit Netzbezug**

  Kopfdaten

  Jahresautomatik / JAUT/ Schaltuhr: Tagesplan, Wochenplan, Sondertag Jährlich, Sondertag Aufzählung, Zeitbereich

- **Block 3: VA-Steuerverfahren**

  n Anwendungsspezifische Dateien (Binär VASteuerverfahren)

- **Block 4: VA-Parameter**

  n Anwendungsspezifische Dateien (Binär VAParameter)

- **Block 5: Map Versorgung**

  die Topologiedaten der Kreuzung

2.3.2 Versionierung

Die Versionierung von Versorgungsständen hat den Zweck, jederzeit nachzuvollziehen zu können, dass Versorgungsänderungen passiert sind, und zwar unabhängig davon ob sie zentral oder lokal vorgenommen wurden. Das Versionierungsverfahren ist zwischen Lichtsignalsteuergerät, VD-Server und Versorgungswerkzeug durchgängig und umfasst sowohl die Anwender- als auch Teile der Herstellerversorgung.

Spezifikation siehe Pkt.3.2.

2.3.3 Anforderungen an den Versorgungsdatenserver

- Der VD- Server muss sicherstellen, dass nur ein Versorgungsvorgang – der aus mehreren Blöcken bestehen kann – zu einer Zeit erfolgt und dass dieser Vorgang abgeschlossen wird.
- Der VD-Server überträgt immer alle Versorgungsdaten eines Blocks und die dazugehörigen Versionierungsdaten. Der funktionelle Inhalt der Versorgungsblöcke ist im Standard festgelegt und darf nicht verändert werden. Eine spätere Änderung der Blöcke hätte Auswirkungen auf die Software der Lichtsignalsteuergeräte. Durch die starren Inhalte ergeben sich Vorteile für die Nachvollziehbarkeit und Prüfbarkeit.
- Die Versorgungsdaten werden als XML und Trace-Datei im Dateisystem des VD Servers gespeichert (mandatory).

2.3.4 Anforderungen an die Lichtsignalsteuergeräte

- TCP-Telegrammgröße: siehe OCIT-O_Protokoll_V2.0 „Verwendung der OSI-Schicht 4 Protokolle (UDP, TCP)“

  Hinweis: Bei sehr großen Versorgungen inkl. VA-Parametern kann es zu einer Überschreitung der max. möglichen Telegrammgröße kommen. Der VD-Server kann in solchen Fällen die Versorgung blockweise senden bzw. lesen. Sollte in einem Feldgerät durch ein Hersteller-Tool eine zu große Versorgung eingespielt worden sein, muss das FG bei einem BTPPL-Leseversuch mit dem RetCode TOO_MANY antworten.

Es ist beabsichtigt ab der nächsten Version eine Fragmentierung einzuführen, durch welche die Größenbeschränkung der Telegramme aufgehoben wird.

- Die stand. Versorgungsdaten der Anwenderversorgung werden als OCIT-O Objekte abgebildet.
- Es werden die bestehenden OCIT-O Methoden, Fehlermeldungen etc. verwendet. Versorgungsobjekte werden mit btppl als Nachricht mit niedriger Priorität übertragen.
- Bei einer Änderung der Versorgung eines Blocks, muss die gesamte, dem Block zugeordnete alte Versorgung gelöscht werden um nicht Reste davon weiter aktiv zu halten.
- Die standardisierten Daten der Anwenderversorgung können so wie sie im Lichtsignalsteuergerät wirksam versorgt sind ausgelesen werden.
- Falls das Lichtsignalsteuergerät bestimmte Werte der Versorgungsdaten nur in abweichender Form unterstützt, muss die Versorgung abgewiesen werden. Eine automatische Korrektur dieser Werte durch das Lichtsignalsteuergerät ist nicht zulässig.
- Die Versorgungsdaten der Herstellerversorgung werden in OCIT-O nicht spezifiziert und können daher nur mit herstellerspezifischen Mitteln versorgt und gelesen werden. Mit den Mittel der Schnittstelle OCIT-O Lstg ab V2.0 sind jedoch über
  InstanceInfo oder ExtendedInstanceInfo die Outstations-Nummern der Signalgruppen- und Detektoren auslesbar. Die Namen der Signalgruppen- und Detektoren findet man über die entsprechenden Objekte.

2.4 Übertragungsgeschwindigkeit

Mit einer Übertragungsgeschwindigkeit von 19200 bis 28800 bit/sec ist nach den bisherigen Einsatzerfahrungen die intensive Nutzung des OCIT-Signalisierungsarchivs (Darstellung der aktuellen Signalisierung in der Zentrale / Visualisierung) und anderer Archive möglich. Bei Übertragungsgeschwindigkeiten unter 9600 bit/s sind diese Möglichkeiten stark eingeschränkt. Sehr schlechte Übertragungswege erlauben oft nur mehr den Betrieb mit 2400 bit/s. Dies ist das absolute Minimum, mit dem nur noch Bedienen und Melden möglich ist.

2.5 Synchronisierung u. Rückrechenverfahren

Die Synchronisierung der Signalisierung von Lichtsignalsteuergeräten im Straßennetz erfolgt uhrengesteuert. Mit Hilfe von Rückrechenverfahren werden die Umlaufzählers der einzelnen Geräte auf einen bestimmten Bezugszeitpunkt synchronisiert. Ein Steuergerät gilt hierbei als synchron, wenn die Umlaufsekunde des Umlaufzählers TX identisch mit dem „modulo TU-Wert“ der seit dem jeweiligen Bezugszeitpunkt abgelaufenen Sekunden ist. Als TU (Umlaufzeit eines Signalplans) wird die Umlaufzeit des aktuell laufenden Signalplans herangezogen. Bei Signalplanwechsel wechselt der TU-Wert in Abhängigkeit vom verwendeten Umschaltverfahren mit dem Vollzug des Signalplanwechsels. Sinnvollerweise wird die Synchronisierung bei Anstoß

eines Signalplanwechsels ausgesetzt und erst nach vollzogenem Wechsel im neuen Signalplan fortgesetzt.

Synchronitätsbedingung: TX = (RRS + SignalzeitenVersatz) % TU

RSS: Rückrechensekunde, d. h. die aktuell abgelaufenen Sekunden seit Bezugszeitpunkt

TU: Umlaufzeit des aktuellen Signalplans

SignalzeitenVersatz: siehe Pkt. 3.3.2.1.3

Für das Rückrechnen haben sich vier verschiedene Bezugszeitpunkte und davon abgeleitet vier verschiedene Rückrechenverfahren eingebürgert. Das Rückrechenverfahren (RVV) ist daher projektspezifisch festzulegen, da es im System (Bestand + OCIT-O) gleich sein muss. Lichtsignalsteuergeräte mit OCIT-O Schnittstelle müssen mindesten folgende Rückrechenverfahren beherrschen:

1. **RVV UTC (Bezugszeitpunkt 1.1.1970 0:00:00 Uhr, Universal Time Coordinated)**
   Bei diesem einfachen aber wenig verbreiteten Rückrechenverfahren wird die aktuelle UTC – Zeit als Bezugszeitpunkt genommen. Für die Ermittlung der aktuellen Referenzsekunde muss die aktuelle UTC – Zeitsekunde modulo der Umlaufzeit des aktuell laufenden Signalplans genommen werden.
2. **RVV 1.1 (Bezugszeitpunkt 1.1. 0:00:00 Uhr Lokalzeit aktuelles Jahr)**
   Dieses weit verbreitete Rückrechenverfahren benutzt den 1.1. 0:00:00 Uhr Lokalzeit des aktuellen Jahres als Bezugszeitpunkt. Die Rückrechensekunde (RRS) ergibt sich aus den seit dem Bezugszeitpunkt gemäß der aktuellen lokalen Uhrzeit abgelaufenen Sekunden. Der Sommerzeitsprung von einer Stunde wird bei diesem Verfahren als abgelaufene Sekunden berücksichtigt, d. h. die RRS springt beim Sommerzeitwechsel um 3600 Sekunden, so als wäre die übersprungene Zeit tatsächlich abgelaufen. Beim Rücksprung gilt dies analog.
3. **RVV 1.1.1980 (Bezugszeitpunkt 1.1.1980 0:00:00 Uhr Lokalzeit)**
   Dieses weniger weit verbreitete Rückrechenverfahren benutzt den 1.1.1980 0:00:00 Uhr als Bezugszeitpunkt. Die Rückrechensekunde (RRS) ergibt sich aus den seit dem Bezugszeitpunkt tatsächlich abgelaufenen Sekunden. Der Sommerzeitsprung von einer Stunde wird bei diesem Verfahren nicht berücksichtigt, d. h. die RRS springt beim Sommerzeitwechsel nicht, sondern läuft kontinuierlich weiter.
4. **RVV Mitternacht (Bezugszeitpunkt 0:00:00 Uhr Lokalzeit des aktuellen Tages)**
   Dieses weniger weit verbreitete Rückrechenverfahren benutzt Mitternacht (0:00:00 Uhr) als Bezugszeitpunkt. Die Rückrechensekunde (RRS) ergibt sich aus den seit dem Bezugszeitpunkt gemäß der aktuellen Uhrzeit abgelaufenen Sekunden. Der Sommerzeitsprung von einer Stunde wird bei diesem Verfahren als abgelaufene Sekunden berücksichtigt, d.h. die RRS springt beim Sommerzeitwechsel um 3600 Sekunden, so als wäre die übersprungene Zeit tatsächlich abgelaufen. Beim Rücksprung gilt dies analog. Dies ist jedoch nur an den beiden Wechseltagen relevant und hat an allen anderen Tagen keine Auswirkung

2.5.1 Berechnungsbeispiele für Rückrechenverfahren

Sekundenwerte für die Rückrechenverfahren:

UTC_1_1_1980OFFSET = 315529200 (Offset der UTC-Zeit vom 1.1.1970 auf den 1.1.1980)

JAHRESSEKUNDEN = 31536000

TAGESSEKUNDEN = 86400

STUNDENSEKUNDEN = 3600

MINUTENSEKUNDEN = 60

RRV UTC:

20.3.07 16:30:00 Uhr	-> RRS: 1174404600	RefZeit bei TU = 70: 40
25.3.07 03:10:00 Uhr	-> RRS: 1174785000	RefZeit bei TU = 70: 60
20.4.07 16:50:22 Uhr	-> RRS: 1177080622	RefZeit bei TU = 70: 32

Berechnungsvorschrift: RefZeit = akt. UTC-Zeit % TU

RRV 1.1.:

20.3.07 16:30:00 Uhr	-> RRS: 6798600	RefZeit bei TU = 70: 60
25.3.07 03:10:00 Uhr	-> RRS: 7182600	RefZeit bei TU = 70: 40
20.4.07 16:50:22 Uhr	-> RRS: 9478222	RefZeit bei TU = 70: 12

Berechnungsvorschrift: RefZeit = ((Tag des Jahres * TAGESSEKUNDEN) + (akt. Stunde * STUNDENSEKUNDEN) + (akt. Minute * MINUTENSEKUNDEN) + akt. Sekunde) % TU

RRV 1.1.1980:

20.3.07 16:30:00 Uhr	-> RRS: 858875400	RefZeit bei TU = 70: 40
25.3.07 03:10:00 Uhr	-> RRS: 859255800	RefZeit bei TU = 70: 60
20.4.07 16:50:22 Uhr	-> RRS: 861551422	RefZeit bei TU = 70: 32

Berechnungsvorschrift: RefZeit = (akt. UTC-Zeit – UTC_1_1_1980OFFSET) % TU

RRV Mitternacht:

20.3.07 16:30:00 Uhr	-> RRS: 59400	RefZeit bei TU = 70: 40
25.3.07 03:10:00 Uhr	-> RRS: 11400	RefZeit bei TU = 70: 60
20.4.07 16:50:22 Uhr	-> RRS: 60622	RefZeit bei TU = 70: 2

Berechnungsvorschrift: RefZeit = ((akt. Stunde * STUNDENSEKUNDEN) + (akt. Minute * MINUTENSEKUNDEN) + akt. Sekunde) % TU

2.6 Teilknoten

Teilknoten sind zu einzelnen Signalisierungsbereichen zusammengefasste Signalgruppen eines Gesamtknotens (relativer Knoten), die zueinander nicht feindlich sind. Alle Teilknoten arbeiten zu einer bestimmten Zeit mit dem gleichen Signalprogramm. Teilknoten können von der Zentrale ein- und ausgeschaltet werden.

Ein OCIT-O Lichtsignalsteuergerät ab Version 2.0 besitzt immer mindesten einen Teilknoten. Der Gesamtknoten (oder auch relative Knoten) kann aus 1 bis max. 4 Teilknoten bestehen. Die Nummerierung beginnt bei 0.

Meldung der Teilknoten im IstVektor:

	Anzahl der gemeldeten Teilknoten			
Gesamtknoten (Rel. Knoten)	1 TK	2 TK	3 TK	4 TK
Index / Nummer im IstVektor	0	0 1	0 1 2	0 1 2 3

In der Nummerierung der Teilknoten darf keine Lücke sein.

Beim Schalten eines nicht vorhandenen Teilknotens liefert das Lichtsignalsteuergerät einen Fehler als Returncode.

Im Auslieferungszustand oder unbestimmten Zuständen folgen die Lichtsignalsteuergeräte dem Zustand des Gesamtknotens (TKZustand = 1).

2.7 Relative Knoten

Relative Knoten sind mehrere Knoten innerhalb eines Lichtsignalsteuergerätes.

Das Adressierungsschema von OCIT-Outstations sieht vor, dass mit einem Gerät (theoretisch) bis zu 255 logisch voneinander unabhängige Knotenpunkte (relative Knoten) realisiert werden können. Die Nummerierung der Relative Knoten beginnt bei 0. Der Default-Wert bei einem Relativen Knoten ist 0. Jeder relative Knoten kann Teilknoten enthalten.

Hinweis: Nicht alle Hersteller können derartige (aufwändige) Geräte anbieten.

2.8 Zeitstempel

Meldungen vom Lichtsignalsteuergerät zur Zentrale, Archiv- oder Listeneinträge sind mit einem Zeitstempel mit einer Auflösung von einer Sekunde zu versehen. Der Zeitstempel gibt an, wann ein Ereignis aufgetreten ist. Format: UTC (siehe OCIT-O Protokoll).

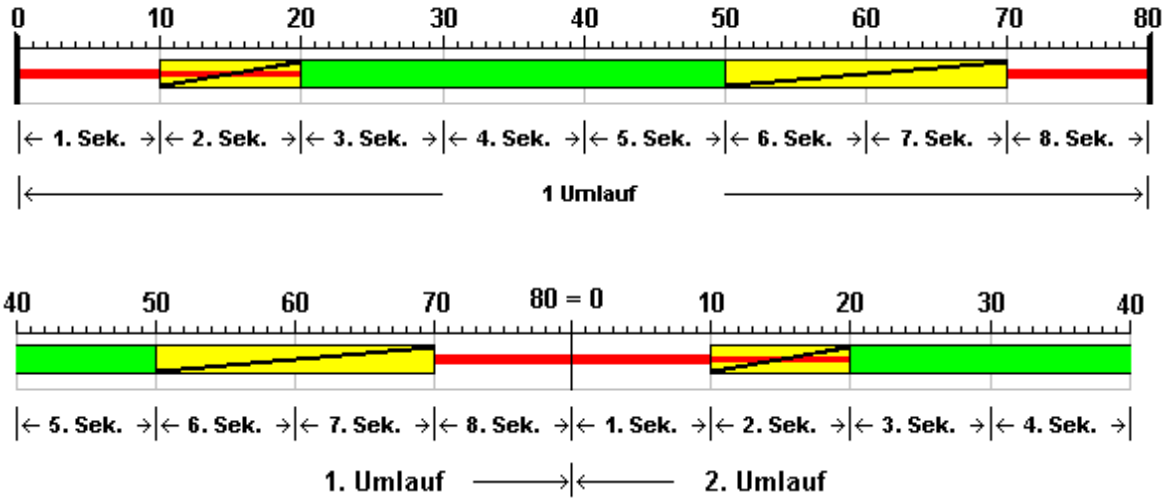
2.9 Zeitzählung

Die Zählung von Zeiten, z. B. der Umlaufzeit eines Signalplans, beginnt mit Sekunde 0. Sekunde 0 bezeichnet den Zeitabschnitt von Beginn bis zum Ende der ersten Sekunde.

Standardkonform sind Schaltzeitpunkte 0 bis TU-1, die Referenzzeit zeigt ebenfalls von 0 bis TU-1. Beispiel: Bei einem 60 Sekunden Umlauf muss das TX=0 auf den Beginn der vollen Minute liegen.

2.9.1 Zeitschaltwerte und Umlauf

- Der Start eines Umlaufs erfolgt immer bei TX=0.
- Der Umlaufzähler TX beginnt beim Schaltwert 0 (= Beginn der 1. Sekunde des Umlaufs) und endet beim letzten Schaltwert des Umlaufs (< TU).
- Die in den Versorgungsdaten angegebenen Zeiten haben eine Auflösung von 0,1 Sekunden.
- Die 1. Sekunde umfasst die Schaltwerte 0 bis 9.
- Wert 0 und Wert TU bezeichnen demnach einen identischen Zeitpunkt. Zugelassen ist der Schaltwert 0, nicht zugelassen ist der Schaltwert TX=TU.



Hinweis: Darstellung in 0.1 Sekunden-Auflösung!

Abbildung 4: Beispiel eines Programms mit einer Umlaufzeit TU = 8 Sekunden

Im Beispiel (Abbildung 4) bewirkt ein Zeitschaltwert 10 den Beginn einer Aktion zum Beginn der 2. Sekunde des Umlaufs.

Versorgungen, die Zeitschaltwerte enthalten, die ein Lichtsignalsteuergerät nicht unterstützt (z.B. Schalten in 0,1 Sekunden- Auflösung) werden bei der Aktivierung abgewiesen. Eine automatische Korrektur der zu versorgenden Zeitschaltwerte durch das Lichtsignalsteuergerät ist nicht zulässig!

Hinweis: Standardisiert ist das Verhalten auf der Schnittstelle. OCIT-O macht keine Aussage zur Visualisierung!

3 Objektdefinitionen

In der zu diesem Dokument gehörenden Datei OCIT-O-Lstg.xml finden sich detaillierte Beschreibungen der Daten und Methoden der Objekte. Details, die aus Gründen der Übersichtlichkeit bei den folgenden Objektbeschreibungen nicht aufgeführt wurden, sind nur dort zu finden.

Hinweis: Es gilt die Festlegung zu den nummerierten Elementen aus OCIT-O Basis: Die Adressierung nummerierter Elemente wie Signalgruppen und Detektoren etc. beginnt mit dem Indexwert 1. Der Index wir nicht gemappt: Index 1 adressiert Element 1 usw. Damit ist sichergestellt, dass der Indexwert mit der von den Anwendern verwendeten Nummer eines nummerierten Elements übereinstimmt.

3.1 Übertragung von Versorgungsdaten

Versorgungsvorgänge werden durch Transaktionen geklammert. Eine Transaktion ermöglicht es den VD-Server zu versorgende Daten schrittweise ins Lichtsignalsteuergerät zu übertragen, zu überprüfen ob die übertragenen Daten konsistent sind diese dann zu einem definierten Zeitpunkt zu aktivieren.

Die Transaktion wird in zwei verschiedenen Zusammenhängen genutzt:

1. Es gibt die Möglichkeit Daten der Anwenderversorgung zu aktivieren. Dies ist umgesetzt durch das Objekt Versorgungstransaktion. Dieses Objekt stellt Funktionalitäten bereit, die neben der Grundfunktionalität der Transaktion den Versorgungsvorgang durch geeignete Meldungen dokumentieren.
2. Mit dem Objekt TransferParameterBlock können Änderungen von AP-Werten durch eine Transaktion geklammert werden.

Die Gemeinsamkeiten dieser beiden Arten von Transaktionen sind im Objekt Transaction zusammengefasst.

3.1.1 Objekt Transaction

Das **Objekt Transaction** ist die Basedomain für

- die Transaktion für die Übertragung der Daten der Anwenderversorgung (SupplyTransaction, Pkt. 3.1.2) und
- die Transaktion für das blockweise Schreiben von AP-Werten (TranferParameterBlock, Pkt. 3.1.3).

Die Übertragung von Daten für die Anwenderversorgung oder für Blöcke von AP-Werten wird vom VIAP oder der Zentrale aus initiiert.

Grundsätzlich benötigt das Lichtsignalsteuergerät für die zu versorgenden Daten Zwischenspeicher (Datenpuffer) und zwar getrennt für die

- Versorgungstransaktion (Versorgungsdatenpuffer) und
- die AP-Wert Blöcke (AP-Blockpuffer).

Zu einem Zeitpunkt, können immer **nur je eine** Versorgungstransaktion und **eine** Übertragung von AP-Werten laufen.

Im den Datenpuffern werden die Daten der Transaktionen zwischengespeichert, wobei sich jeweils nur ein Stand dieser Daten im den entsprechenden Puffern befinden kann. Die Daten im Puffer können bereits aktiviert (d.h. von Anwendungen im Lichtsignalsteuergerät übernommen) sein oder auf ihre Aktivierung warten. Der Puffer kann gelöscht oder mit neuen Daten gefüllt werden. Festlegungen zur Größe siehe Punkte 3.1.2.

Zu Beginn einer Transaktion muss initialisiert werden. Dadurch werden alle bereits im Versorgungsdatenpuffer des Lichtsignalsteuergeräts enthaltenen Daten gelöscht. Danach wird der Versorgungsdatenpuffer im Lichtsignalsteuergerät mit den zu versorgenden Daten mit der Methode AddChangeSet gefüllt. Das Lichtsignalsteuergerät prüft, ob die im Puffer angelangten Versorgungsdaten von den Anwendungen im Lichtsignalsteuergerät angenommen werden können.

Ist die Übertragung vollständig erfolgt und sind alle Daten geprüft, so werden die Änderungen mit der Methode Activate in die entsprechenden Anwendungen des Lichtsignalsteuergeräts übernommen. Eine eventuell laufende Transaktion kann durch die Methode Abort abgebrochen werden.

Die Transaktion kann sich in einem der folgenden Zustände befinden:

none	Die Transaktion ist nicht initialisiert.
empty	Die Transaktion ist initialisiert und leer.
receiving	Die Transaktion enthält bereits Änderungen, ist jedoch noch nicht geprüft und nicht aktiviert.
checkFailed	Die Änderungen sind geprüft und nicht aktivierbar.
checked	Die Änderungen sind geprüft und als aktivierbar eingestuft worden.
complete	Die Änderungen sind komplett übertragen. Sie sind jetzt nicht mehr änderbar, jedoch aktivierbar.
activationSet	Ein Aktivierungstermin wurde gesetzt.
activating	Die Anwenderversorgung wird im Lichtsignalsteuergeräte aktiviert (Dauer voraussichtlich 1 bis 2 Umläufe).

Es ist nicht garantiert, dass die Daten der Transaktion vor ihrer Aktivierung persistent gehalten werden können und z.B. einen Netzausfall überstehen. In diesem Fall wäre die Transaktion nach der Störung wieder im Zustand „none“. Dieser Zustandsübergang wird wie unten beschrieben durch eine Meldung „VersorgungEnde“ mit dem Nebenmeldungsteil „TransactionFailed“ dokumentiert.

Transaction (1:710)

Transaction		
METHOD	**Name**	**Beschreibung**
100	**Init**	Die Transaktion wird zurückgesetzt: alle Daten werden gelöscht.
	Eingabeparameter	
	Vorgang: SYSJOBID	Die Vorgangskennung der Versorgung.
	Ausgabeparameter	
	RetCode	OK ILLEGAL_STATE falls die Transaktion sich nicht im Zustand none befindet. NOT_CONFIGURED falls das Lichtsignalsteuergerät die Versorgung nicht unterstützt. Dies ist insbesondere für die Implementierung von SupplyTransactions relevant, da diese auch blockweise begrenzt sein können. EXISTS_ALREADY falls die angegebene SYSJOBID mit der SYSJOBID der zuletzt initialisierten Transaction übereinstimmt.
101	**AddChangeSet**	Mit dieser Methode können Änderungen zur Transaktion hinzugefügt werden können.
	Eingabeparameter	
	Vorgang: SYSJOBID	Die Vorgangskennung der Versorgung.
	Objects: BaseObjType[ ]	Ein Array von Objekten bestehend aus: - relativer Path - Daten (<REFPATH_DATA>3</REFPATH_DATA>)
	Ausgabeparameter	

Transaction		
METHOD	**Name**	**Beschreibung**
	RetCode	OK ILLEGAL_STATE falls die Änderungen gerade aktiviert werden oder die Transaktion sich im Zustand none oder complete, activationSet oder activating befindet. TOO_MANY falls der Speicher nicht für alle Elemente der Änderungen ausreicht. PARAM_INVALID falls versucht wurde ein Objekt mehrfach hinzuzufügen, oder falls versucht wurde ein Objekt hinzuzufügen das für diese Transaction nicht zulässig ist, z.B. falscher Versorgungsblock. ACCESS_DENIED falls die angegebene SYSJOBID nicht mit der SYSJOBID dieser Transaction übereinstimmt. Hinweis: Bei RetCode != OK werden KEINE Objekte der Transaktion hinzugefügt.
	Flaws: TransactionFlaw[ ]	Bei RetCode PARAM_INVALID gibt der Array der Schwächen alle Fehler zurück, die beim Versuch die Objekte des ChangeSets der Transaktion hinzuzufügen aufgetreten sind. Die Fehler sind die aufgeführten und entsprechend beschriebenen Messageparts.
102	**ReleaseObjects**	Mit dieser Methode können Objekte, die bereits aktiviert wurden und nicht mehr genutzt werden, gelöscht werden. Die Methode wird für Teilversorgungen[2] benötigt und ist daher optional.
	Eingabeparameter	
	Vorgang: SYSJOBID	Die Vorgangskennung der Versorgung.
	Refs: BaseObjType[ ]	Ein Array mit Referenzen auf die zu löschenden Objekte.
	Ausgabeparameter	

[2] Teilversorgungen sind in OCIT-O Lstg V2.0 nicht vorgesehen, die Objektmodellierung berücksichtigt jedoch diese Option auf die Zukunft.

Transaction		
METHOD	**Name**	**Beschreibung**
	RetCode	OK PARAM_INVALID falls eine Objektreferenz unbekannt ist, wird dieser RetCode zurückgegeben. ACCESS_DENIED falls die angegebene SYSJOBID nicht mit der SYSJOBID dieser Transaction übereinstimmt. Hinweis: Bei RetCode != OK werden KEINE Objekte gelöscht.
	Refs: BaseObjType[ ]	Dieser Rückgabewert beinhaltet eine Liste mit allen unbekannten Objekten.
103	**Completed**	Diese Methode markiert die Transaction als vollständig. Hinterher kann sich nicht mehr verändert werden.
	Eingabeparameter	
	Vorgang: SYSJOBID	Vorgang: SYSJOBID
	Ausgabeparameter	
	RET_CODE	OK ILLEGAL_STATE falls die Transaction sich nicht im Zustand checked befindet. ACCESS_DENIED falls die angegebene SYSJOBID nicht mit der SYSJOBID dieser Transaction übereinstimmt.
104	**Activate**	Die Anforderung der Aktivierung einer Transaction kann nur in den Zuständen checked oder complete, d.h. nach dem Aufruf der Methode Check bzw. Complete erfolgen. Ein gesetzter Aktivierungszeitpunkt (Zustand im Zustand activationSet) kann durch einen nochmaligen Aufruf der Methode Activate geändert werden. Die Transaktion wird zu einem Zeitpunkt aktiviert. Ist der angegebene Zeitpunkt in der Vergangenheit, werden die zur Transaktion gehörenden Änderungen sofort aktiviert.
	Eingabeparameter	
	Vorgang: SYSJOBID	Vorgang: SYSJOBID
	Zeit: UTC	Der Zeitpunkt der Aktivierung der Änderungen
	Ausgabeparameter	

Transaction		
METHOD	**Name**	**Beschreibung**
	RetCode	OK ILLEGAL_STATE falls die Transaction sich nicht im Zustand checked, complete, activationSet befindet. ACCESS_DENIED falls die angegebene SYSJOBID nicht mit der SYSJOBID dieser Transaction übereinstimmt.
105	**Abort**	OK ILLEGAL_STATE falls die Transaktion sich im Zustand none oder activating befindet.
106	**Check**	Mit der Methode Check können die Änderungen einer Transaktion daraufhin geprüft werden, ob es möglich ist, sie in das Lichtsignalsteuergeräts zu übernehmen. Hierbei werden die in der Anwenderversorgung standardisierten Daten überprüft.
	Eingabeparameter	
	Vorgang: SYSJOBID	Vorgang: SYSJOBID
	Ausgabeparameter	
	RetCode	OK falls die Änderungen gültig sind und in das Lichtsignalsteuergeräts übernommen werden können. PARAM_INVALID falls der Check fehlgeschlagen ist, mit den Fehlern in der Versorgung als Rückgabewert. Dazu werden die unten definierten Messageparts UndefinedReferenceInObject, DeletedObjectInUse, MissingMandatoryElement, ustAPWertNotWritable und UnspecifiedSupplyError verwendet. ILLEGAL_STATE falls sich die Transaktion nicht im Zustand Receiving befindet. ACCESS_DENIED falls die angegebene SYSJOBID nicht mit der SYSJOBID dieser Transaction übereinstimmt.
	Flaws: TransactionFlaw[ ]	Bei den RetCodes PARAM_INVALID gibt der Array der Schwächen alle Fehler bzw. notwendigen Änderungen zurück, die bei einer Übernahme der Versorgung notwendig wären. Die Schwächen/Fehler sind die unten aufgeführten und entsprechend beschriebenen Messageparts.

Transaction		
METHOD	**Name**	**Beschreibung**
109	**SetEventDestination**	Setzt das Ziel des OnTransactionStateChanged-Events dieser Transaktion.
	Eingabeparameter	
	Vorgang: SYSJOBID	Die Vorgangskennung der Versorgung.
	EventDst: ZNR_FNR	Zentralen- und Lichtsignalsteuergerätenummer des Gerätes, welches den Event empfangen soll.
	Ausgabeparameter	
	RetCode	OK ACCESS_DENIED falls die angegebene SYSJOBID nicht mit der SYSJOBID dieser Transaction übereinstimmt.

Da beim Aufruf der Methoden AddChangeSet und ReleaseObjects potentiell große Datenmengen transportiert werden, wird für den Aufruf ausschließlich der **TCP Port PNP** verwendet.

Das folgende UML Diagramm illustriert die Korrelation zwischen den Methodenaufrufen und den Zustandsübergängen.

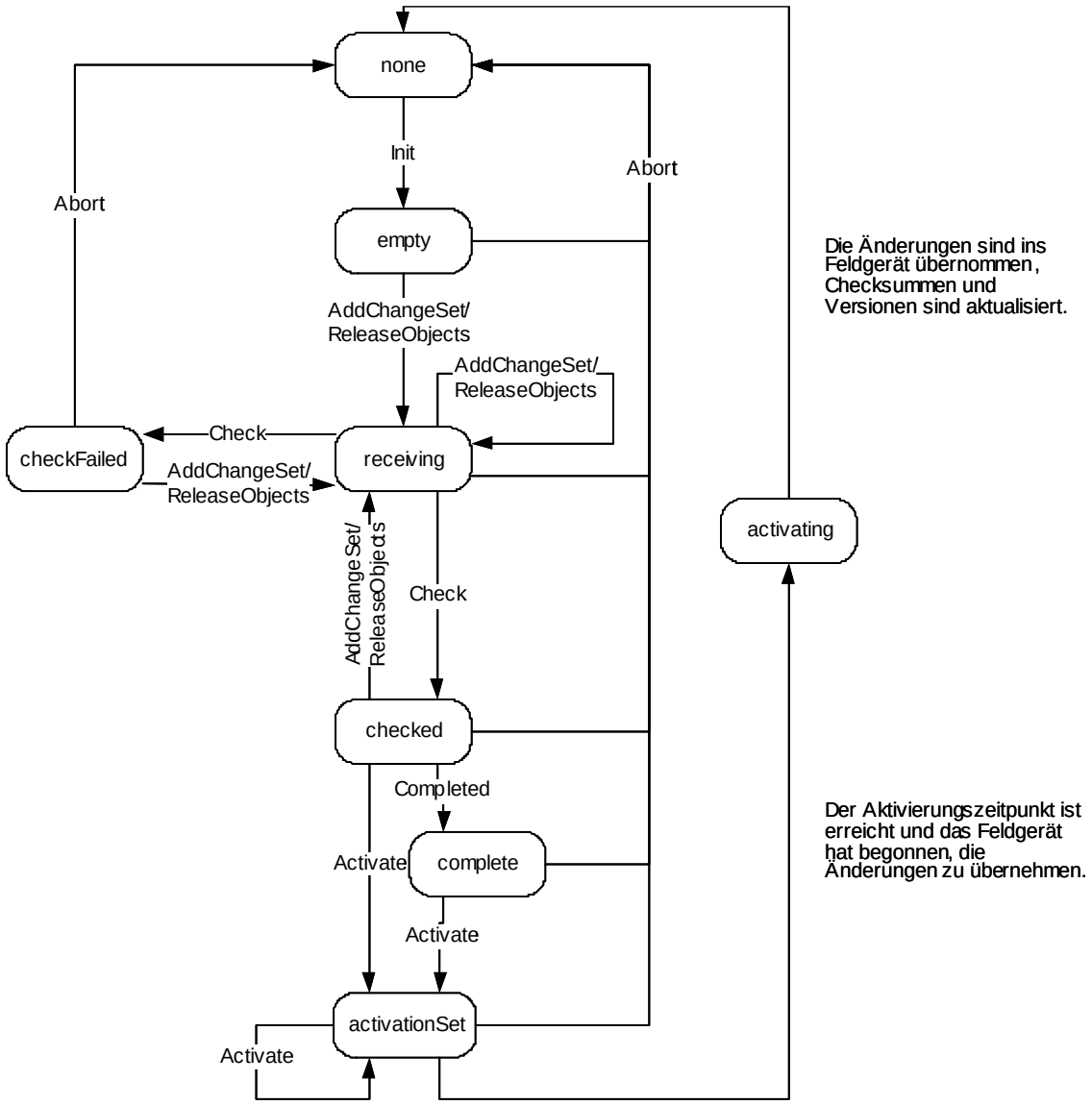


Abbildung 5: Zustandsdiagramm bei Transaktionen

Für jeden Zustand gilt, dass der Aufruf einer unzulässigen Methode mit ILLEGAL_STATE quittiert wird und die Transaktion ihren Zustand nicht verändert.

Folgende Ereignisse werden in Meldungen definiert:

- Beim Aufruf von init wird die Meldung VersorgungBeginn erzeugt mit dem unten definierten Nebenmeldungsteil TransactionInitialized.
- Ein Übergang in den Zustand none wird durch eine Meldung VersorgungEnde dokumentiert. Der Grund wird ebenfalls durch einen unten spezifizierten Nebenmeldungsgteil detailliert (TransactionAborted oder TransactionFailed). Insbesondere TransactionFailed kann in jedem Zustand auftreten, z.B. durch einen Netzausfall.
- Der erfolgreiche Aufruf der Methode completed (bzw. activate im Zustand checked) wird durch die Meldung TransactionDefined dokumentiert.

Im Detail:

(MeldungsDegree **I**: Information, **W**: Warnung, **F**: Fehler, **S**: Schwerer Fehler)

OType	Kurzname	MeldungsDegree	Beschreibung
60300	TransactionInitialized	I	Diese Meldung wird als Nebenmeldungsteil der Meldung VersorgungsBeginn erzeugt, wenn die Methode init aufgerufen wird. Parameter sind der Fully Qualified Domain Name (FQDN) bzw. die IP-Adresse (wenn FQDN nicht aufgelöst werden kann) des Lichtsignalsteuergeräts des Aufrufers (RemoteDevice)
60301	TransactionDefined	I	Diese Meldung wird erzeugt, wenn die Transaction sich im Zustand checked befindet und die Methode Completed aufgerufen wird.
60318	TransactionActivationRequest	I	Diese Meldung wird erzeugt, wenn die Aktivierung einer Transaktion angefordert wurde, d. h. die Methode Activate aufgerufen wurde. Parameter ist der Zeitpunkt der angeforderten Aktivierung.
60302	TransactionAborted	I	Diese Meldung wird als Nebenmeldungsteil der Meldung VersorgungsEnde erzeugt, falls die Methode abort aufgerufen wird.
60303	TransactionFailed	F	Diese Meldung wird als Nebenmeldungsteil der Meldung VersorgungsEnde erzeugt, falls aus irgendeinem undefinierten Grund die Transaktion fehlgeschlagen ist.

Die Zuordnung der Meldungen erfolgt über die SYSJOBID.

Folgende Meldungsteile werden als Nebenmeldungsteil(e) verwendet und dienen zusätzlich als Rückgabewerte der Funktion Transaction.Check():

OType	Kurzname	MeldungsDegree	Beschreibung
60309	UnspecifiedFlaw	W	Falls eine Versorgung eine Schwäche enthält, so wird diese Meldung als Nebenmeldungsteil der Meldung VersorgungsEnde erzeugt. Parameter sind das Objekt, das die Schwäche enthält und ein Text mit einer Beschreibung.
60310	UnpecifiedSupplyError	F	Falls eine Versorgung einen Fehler enthält, so wird diese Meldung als Nebenmeldungsteil der Meldung VersorgungsEnde erzeugt. Parameter sind das Objekt, das die Schwäche enthält und ein Text mit einer Beschreibung.

Folgende Meldungsteile werden nicht in Archive eingetragen, sondern dienen nur als Rückgabewerte der Funktion Transaction.Check():

OType	Kurzname	MeldungsDegree	Beschreibung
60304	UndefinedReferenceInObject	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Es gibt 2 Parameter: eine Referenz auf das Objekt in dem der Fehler aufgetreten ist und die undefinierte Objektreferenz.
60305	DeletedObjectInUse	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er zeigt an, dass versucht wurde, ein Objekt zu löschen, das noch in Benutzung ist. Parameter ist eine Referenz auf das Objekt, das gelöscht werden sollte.
60306	MissingMandatoryElement	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er zeigt an, dass ein notwendiges Element nicht versorgt wurde. Dies können im Falle einer SupplyTransaction (siehe unten) die OCIT-C Versionen der geänderten Blocks oder Standardelemente (z.B.

			Standardtages- oder Wochenplan) sein. Parameter ist eine Referenz auf das fehlende Element.
60308	ObjectNotInBlock	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden und wird erzeugt, falls versucht wurde ein Objekt zu erzeugen, das nicht im zu versorgenden Block liegt. Parameter ist eine Referenz auf dieses Objekt.
60317	APWertWriteNotPossible	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er wird erzeugt, falls versucht wird, einen AP-Wert zu ändern, der nicht schreibbar ist. Parameter ist eine Referenz auf den APWert.
60320	DuplicateObject	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er wird erzeugt, falls versucht wird, ein Objekt mehrmals zu einer Transaktion hinzuzufügen. Parameter ist eine Referenz auf dieses Objekt.
60322	OffsetTimeViolation	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er wird erzeugt, falls ein Festzeitenprogramm eine Versatzzeit verletzt.
60323	IntergreenTimeViolation	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er wird erzeugt, falls ein Festzeitenprogramm eine Zwischenzeit verletzt.
60324	MinGreenTimeViolation	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er wird erzeugt, falls ein Festzeitenprogramm eine Mindestfreigabezeit verletzt.
60325	MinRedTimeViolation	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu melden. Er wird erzeugt, falls ein Festzeitenprogramm eine Mindestgesperrtzeit verletzt.

3.1.2 Objekt SupplyTransaction (Versorgungstransaktion)

Zur Übertragung und Aktivierung von Anwenderversorgungsdaten wird im Lichtsignalsteuergerät ein Objekt implementiert, die SupplyTransaction, die diese Funktionalität zur Verfügung stellt. Grundsätzlich muss dazu ein Versorgungsdatenpuffer vorgesehen werden, in dem die mit der SupplyTransaction übertragenen Daten zwischengespeichert werden. Der Puffer muss ausreichend dimensioniert sein, um eine

Neuversorgung aller Blöcke des Lichtsignalsteuergeräts zwischenspeichern zu können.

Die Basedomain der SupplyTransaction ist die Transaction. Die Abbildung 5: Zustandsdiagramm bei Transaktionen gilt auch für die Versorgungstransaktion.

Die SupplyTransaction erweitert die Transaction um Funktionalität zur Versorgung der Blöcke der Anwenderversorgung.

Hinweis: In OCIT-O Lstg V2.0 erfolgt die Anwenderversorgung immer blockweise, d. h. sie ist mit allen Objekten eines der 4 Versorgungsdatenblöcke durchzuführen (siehe Pkt. 2.3.1 und 2.3.3).

Für die Aktivierung einer Versorgungstransaktion gelten besondere Bedingungen:

- Die OCIT-I Versionsinformationen (Version und Checksumme, siehe Pkt. 3.2.2) eines Blocks müssen immer geändert werden, wenn sich ein Datum innerhalb des Blocks ändert.
- Im Block „Daten mit Netzbezug“ müssen Standardelemente (Standardtagesplan und Standardwochenplan) immer vorhanden sein, sonst wird die Versorgung abgelehnt.
- Falls alle Daten eines Blocks der Anwenderversorgung versorgt werden (Standardmethode) werden alle zum Block gehörenden Objekte vor der Aktivierung gelöscht. Die SupplyTransaction erkennt diese Objekte an der Blockkennung (VDArt) der zu versorgenden Objekte.

SupplyTransaction (1:711)

SupplyTransaction		
METHOD	**Name**	**Beschreibung**
120	**InitSupplyTransaction**	Die Transaktion wird zurückgesetzt: alle Daten werden gelöscht. Die Methode InitSupplyTransaction enthält im Eingabeparameter die Information darüber, von welcher Art die Versorgung ist. Für jeden zu versorgenden Block wird dessen VDArt im Eingabeparameter übergeben. Soll auf diese Art blockweise versorgt werden, so wird er komplett versorgt. Dazu werden vor der Aktivierung alle Objekte des Blocks gelöscht und mit dem Inhalt der Transaktion überschrieben. Sind der Array von VDArten leer, so soll eine Teilversorgung[3] stattfinden. Steuergeräte, die diese Versorgungsart nicht unterstützen, beantworten den Methodenaufruf mit einer Fehlermeldung. Bei einer Teilversorgung müssen überschüssige Objekte explizit durch Aufruf der Methode ReleaseObjects gelöscht werden. Hinweis: In diesem Fall entspricht der Aufruf dem Aufruf von Init (Methode 100) der Basedomain Transaction 1:710.
	Eingabeparameter	
	Vorgang: SYSJOBID	Die Vorgangskennung der Versorgung.
	Blocks.Anzahl	Anzahl folgender Blocks
	Blocks: VDArt[0...3]	Ein Array mit allen zu versorgenden Blocks. Falls Blocks die Länge 0 hat, wird eine Teilversorgung durchgeführt. Jede VDArt darf nur einmal enthalten sein!
	Ausgabeparameter	

[3] Teilversorgungen sind in OCIT-O Lstg V2.0 nicht vorgesehen, die Objektmodellierung berücksichtigt jedoch diese Option auf die Zukunft.

SupplyTransaction		
METHOD	**Name**	**Beschreibung**
	RetCode	OK ILLEGAL_STATE falls die Transaktion sich nicht im Zustand none befindet. NOT_CONFIGURED falls Teilversorgung angefordert wurde und das Lichtsignalsteuergerät diese nicht unterstützt. EXISTS_ALREADY falls die angegebene SYSJOBID mit der SYSJOBID der zuletzt initialisierten Transaction übereinstimmt.
121	**ReadVD**	Mit der Methode ReadVD können die Versorgungsdaten eines Blockes auf einmal gelesen werden. Die Methode benötigt keine Transaktionsinstanz und liefert die aktuell aktiven Versorgungsobjekte.
	Eingabeparameter	
	VDArtFilter.Anzahl	Anzahl folgender VDArt Elemente
	VDArt:VDArtFilter[0...3]	Kennung der Blöcke deren Versorgungsdaten geliefert werden sollen. Falls VDArtFilter die Länge 0 hat, sind alle Blöcke gemeint. Jede VDArt darf nur einmal enthalten sein!
	Ausgabeparameter	
	RetCode	OK PARAM_INVALID falls unbekannter VDArtFilter
	VD: VersorgbaresObjekt[ ]	Liste versorgbarer Objekte. Diese werden Lichtsignalsteuergeräterelativ mit Pfad und Daten übertragen (s.a. Transaction: **AddChangeSet).** Der Inhalt der Versorgungsobjekte ist der mit welchem das Lichtsignalsteuergerät zum Zeitpunkt des Aufrufs arbeitet.
122	**ReadVDExt**	Mit der Methode ReadVDExt können die Versorgungsdaten eines Blockes, die LSAVersionen sowie die Gesamtversion auf einmal gelesen werden. Die Methode benötigt keine Transaktionsinstanz und liefert die aktuell aktiven Versorgungsobjekte.
	Eingabeparameter	
	VDArtFilter.Anzahl	Anzahl folgender VDArt Elemente

SupplyTransaction		
METHOD	**Name**	**Beschreibung**
	VDArt:VDArtFilter[0...3]	Kennung der Blöcke deren Versorgungsdaten geliefert werden sollen. Falls VDArtFilter die Länge 0 hat, sind alle Blöcke gemeint. Jede VDArt darf nur einmal enthalten sein!
	Ausgabeparameter	
	RetCode	OK PARAM_INVALID falls unbekannter VDArtFilter
	VD: VersorgbaresObjekt[ ]	Liste versorgbarer Objekte . Diese werden Lichtsignalsteuergeräterelativ mit Pfad und Daten übertragen (s.a. Transaction: **AddChangeSet).** Der Inhalt der Versorgungsobjekte ist der mit welchem das Lichtsignalsteuergerät zum Zeitpunkt des Aufrufs arbeitet.
	GesamtVersion	Gesamtversion
	LsaVersion[ ]	Liste der LSA-Versionsobjekte
123	**ReadVDExtPlus**	Mit der Methode ReadVDExt können die Versorgungsdaten eines Blockes, die LSAVersionen sowie die Gesamtversion auf einmal gelesen werden. Die Methode benötigt keine Transaktionsinstanz und liefert die aktuell aktiven Versorgungsobjekte.
	Eingabeparameter	
	VDArtFilter.Anzahl	Anzahl folgender VDArt Elemente
	VDArt:VDArtFilter[0...3]	Kennung der Blöcke deren Versorgungsdaten geliefert werden sollen. Falls VDArtFilter die Länge 0 hat, sind alle Blöcke gemeint. Jede VDArt darf nur einmal enthalten sein!
	Ausgabeparameter	
	RetCode	OK PARAM_INVALID falls unbekannter VDArtFilter
	VD: VersorgbaresObjekt[ ]	Liste versorgbarer Objekte . Diese werden Lichtsignalsteuergeräterelativ mit Pfad und Daten übertragen (s.a. Transaction: **AddChangeSet).** Der Inhalt der Versorgungsobjekte ist der mit welchem das Lichtsignalsteuergerät zum Zeitpunkt des Aufrufs arbeitet.

SupplyTransaction		
METHOD	**Name**	**Beschreibung**
	GesamtVersion	Gesamtversion
	LsaVersionPlus[ ]	Liste der LSA-Versionsobjekte mit SYSJOBID.
0	**Standardmethode Get**	
	Ausgabeparameter	
	RetCode	OK
	Vorgang: SYSJOBID	Die Versorgungskennung der laufenden Transaction. NULLVALUE falls sich die Versorgung im Zustand none befindet.
	EventDst: ZNR_FNR	Zentralen- und Lichtsignalsteuergerätenummer des Gerätes an welches diese Transaktion des OnTransactionStateChanged-Event sendet.
	CompletionTime: UTC	Der Zeitpunkt, zu dem die Transaction als complete markiert wurde. Sonst NULLVALUE.
	ActivationTime: UTC	Der aktuelle Aktivierungszeitpunkt, falls gesetzt. Sonst NULLVALUE.
	State	Der aktuelle Zustand der Transaktion.
	Blocks: VDArt[ ]	Der Parameter Blocks, der bei Get zurückgegeben wird, ist derselbe wie derjenige, der mit InitSupplyTransaction gesetzt wird.

Die Versorgungsänderungen werden mit einem speziellen Nebenmeldungsteil, der den Hauptmeldungsteil genauer spezifiziert eingetragen:

(MeldungsDegree **I**: Information, **W**: Warnung, **F**: Fehler, **S**: Schwerer Fehler)

OType	Kurzname	MeldungsDegree	Beschreibung
60311	SupplyDefined	I	Diese Meldung wird als Nebenmeldungsteil der Meldung TransactionDefined erzeugt. Parameter ist ein VDArt-Array der Blöcke für die eine neue Versorgung definiert wurde.
60312	SupplyActivated	I	Diese Meldung wird als Nebenmeldungsteil der Meldung VersorgungsEnde erzeugt, wenn die Änderungen einer Versorgungstransaktion in die Geräteversorgung über-

			nommen wurden.
60314	CurrentBlockVersion	I	Diese Meldung wird als Nebenmeldungsteil der Meldung SupplyVersionChanged für jeden geänderten Block erzeugt. Parameter sind die VDVersion und die LsaVersion des Blocks. Anmerkung: Die SYSJOBID des Vorgangs wird in der SYSJOBID der Meldung bereits dokumentiert. Daher wird in in diesem Fall nicht die LsaVersionPlus genutzt.
60313	CurrentFeldgeraete-Version	I	Diese Meldung wird als Nebenmeldungsteil der Meldung SupplyVersionChanged erzeugt. Parameter ist die GesamtVersion der Versorgung.
60319	SupplyVersionChanged	I	Diese Meldung wird erzeugt wenn sich VDVersion und/oder LsaVersion mindestens eines Versorgungsblocks geändert haben (bei Anwender- oder Herstellerversorgung). Die Meldung wird nur in das Versorgungsarchiv, nicht in das Standardmeldearchiv, eingetragen. Als Nebenmeldungsteile von SupplyVersionChanged werden CurrentBlockVersion (für jeden geänderten Block) und CurrentFeldgeraeteVersion eingetragen.

3.1.3 Objekt TransferParameterBlock

Das Objekt TransferParameterblock dient zum blockweisen Schreiben von AP-Werten. Es ist als optional deklariert.

Um mehrere AP-Werte oder komplex strukturierte AP-Werte im Lichtsignalsteuergerät zu setzen (so weit sie setzbar sind), wird das Objekt TransferParameterBlock definiert. Dieses Objekt stellt analog wie die Versorgungstransaktion für versorgbare Objekte diese Funktionalität für AP-Werte bereit.

Grundsätzlich muss dazu ein AP-Blockpuffer vorgesehen werden, in dem die mit TransferParameterBlock übertragenen Daten zwischengespeichert werden. Der Puffer muss entsprechend den vorgesehenen Anwendungen dimensioniert sein.

Die Basedomain des TransferParameterBlock ist die Transaction.

Zur effizienten Übertragung der Änderungen bietet TransferParameterBlock eine zusätzliche Methode AddPrefixedChangeSet an, die einen zusätzlichen Pfad Parameter besitzt, der allen nachfolgenden AP-Werten vorangestellt wird (getrennt durch einen Punkt).

Für jedes Schreiben der AP-Werte wird eine entsprechende Kurzmeldung im Standardmeldearchiv eingetragen und ausführliche Informationen in das Versorgungsarchiv.

TransferParameterBlock ist abgeleitet von Transaction. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

TransferParameterBlock (1:712)

TransferParameterBlock		
METHOD	**Name**	**Beschreibung**
150	**AddPrefixedChangeSet**	Mit dieser Methode können Änderungen zur Transaktion hinzugefügt werden können.
	Eingabeparameter	
	Vorgang: SYSJOBID	Vorgang: SYSJOBID
	Prefix: ANYPATH	Ein ANYPATH, der den Namen der nachfolgend referenzierten APWerte vorangestellt wird.
	Objects: BaseObjType[ ]	Ein Array von Objekten bestehend aus: - relativer Path - Daten (<REFPATH_DATA>3</REFPATH_DATA>)
	Ausgabeparameter	
	RetCode	OK ILLEGAL_STATE falls die Änderungen gerade aktiviert werden oder die Transaktion sich im Zustand none befindet. TOO_MANY falls der Speicher nicht für alle Elemente der Änderungen ausreicht. ACCESS_DENIED falls die angegebene SYSJOBID nicht mit der SYSJOBID dieser Transaction übereinstimmt. PARAM_INVALID falls versucht wurde ein Objekt hinzuzufügen das für diese Transaction nicht zulässig ist, z.B. Objekt ist kein APWert. Hinweis: Bei RetCode != OK werden KEINE Objekte der Transaktion hinzugefügt.

TransferParameterBlock		
METHOD	**Name**	**Beschreibung**
	Flaws: TransactionFlaw[ ]	Bei RetCode PARAM_INVALID gibt der Array der Schwächen alle Fehler zurück, die beim Versuch die Objekte des ChangeSets der Transaktion hinzuzufügen aufgetreten sind. Die Fehler sind die aufgeführten und entsprechend beschriebenen Messageparts.
106	**Check**	Die Methode der BASEDOMAIN wird hier übernommen. Zusätzlich jedoch schlägt Check fehl, wenn versucht wurde, einen AP-Wert zu ändern, der nicht schreibbar ist.
	Eingabeparameter	
	Vorgang: SYSJOBID	Vorgang: SYSJOBID
	Ausgabeparameter	
	RetCode	Siehe RetCode der BASEDOMAIN Transaction.
0	**Standardmethode Get**	
	Ausgabeparameter	
	RetCode	OK
	Vorgang: SYSJOBID	Die Versorgungskennung der laufenden Transaction. NULLVALUE falls sich die Versorgung im Zustand none befindet.
	EventDst: ZNR_FNR	Zentralen- und Lichtsignalsteuergerätenummer des Gerätes an welches diese Transaktion des OnTransactionStateChanged-Event sendet.
	CompletionTime: UTC	Der Zeitpunkt, zu dem die Transaction als complete markiert wurde. Sonst NULLVALUE.
	ActivationTime: UTC	Der aktuelle Aktivierungszeitpunkt, falls gesetzt. Sonst NULLVALUE.
	State	Der aktuelle Zustand der Transaktion.

Zur Dokumentation der Änderung der AP-Werte werden bereits bei der Transaktion definierten Meldungen in das Standardmeldearchiv eingetragen.

Zusätzlich stehen folgende Meldungen zur Verfügung, die nur in das Versorgungsarchiv, nicht in das Standardmeldearchiv eingetragen werden.

OType	Kurzname	MeldungsDegree	Beschreibung
60315	APWertChangeRequested	I	Diese Meldung wird erzeugt, wenn das Activate des TransferParameterBlocks aufgerufen wird. Parameter sind • Referenz auf den APWert • Alter Wert • Neuer Wert
60316	APWertChangeCommitted	I	Diese Meldung wird erzeugt, wenn die Änderungen aus der Meldung APWertChangeRequested übernommen wurden. Parameter sind Programmnummer, Programmzeit (TX) und Phasennummer (PH) vom Zeitpunkt der APWert-Übernahme.
60317	APWertWriteNotPossible	F	Dieser Meldungsteil dient dazu, Fehler in der Versorgung zu dokumentieren. Er wird erzeugt, falls versucht wird, einen AP-Wert zu ändern, der nicht schreibbar ist. Parameter ist eine Referenz auf den APWert.

3.2 Versionierung der Versorgungsdaten

Mittels des in OCIT-C und OCIT-O vorgesehenen Versionierungsverfahrens können Versorgungsänderungen durch Kontrolle von Versionsnummern und Checksummen nachvollzogen werden (auch als Rückdokumentation bezeichnet). Einem Versorgungswerkzeug werden mit dem Versionierungsverfahren alle Möglichkeiten gegeben, sowohl Veränderungen als auch den Erhalt der eigentlichen verkehrstechnischen Daten zu analysieren. Zusätzlich gibt es eine Möglichkeit zum Lesen von Versorgungsdaten des Lichtsignalsteuergeräts.

Die Dokumentation der Versorgung (was gehört wie zusammen, z. B. Anwenderversorgung und Herstellerversorgung) ist nicht Bestandteil der Spezifikation von OCIT-O Lstg V3.0, sondern eine Eigenschaft des Versorgungswerkzeugs. Die Registratur der verschiedenen Versionsstände, ihrer Kombinationen, lokalen Änderungen und der technischen Freigaben muss der Betreiber organisieren.

Entsprechend der blockweisen Versorgung wird auch blockweise versioniert. Versionierungsdaten werden sowohl im VD Server als auch im Lichtsignalsteuergerät gespeichert. Im Lichtsignalsteuergerät gespeicherte Daten können vom Server gelesen werden.

Versionierungsdaten, die vom VD Server erzeugt und im Lichtsignalsteuergerät gespeichert werden sollen, werden zusammen mit den Versorgungsdaten mittels der SupplyTransaction übertragen. Versionierungsdaten, die im Lichtsignalsteuergerät erzeugt werden, werden mit jeder Versorgungsänderung (lokal oder zentral) aktualisiert.

Das Lichtsignalsteuergerät bietet mit den gespeicherten, erzeugten und auslesbaren Versionierungsdaten die Grundlage für die Verwaltung und Kontrolle der Versionen und Daten von Geräteversorgungen.

Schema der Versorgungsdatenblöcke und Versionierungsdaten:

In OCIT-C LSA Versorgungsdaten standardisierte Versorgungsdaten, herstellerübergreifend versorgbar und auslesbar **Anwenderversorgung**				
Verkehrstechnik			**Topologie**	
Verkehrs-technische Grund-daten / Festzeit	Daten mit Netzbe-zug	VA-Steuerverfahren	VA-Parameter	MAP
.....	.....	.....	.....	.....
.....	.....	.....	.....	.....
.....	.....	.....	.....	.....
OCIT-O Check-summen Server [4])	OCIT-O Check-summen Server [4])	OCIT-O Check-summen Server [4])	OCIT-O Check-summen Server [4])	OCIT-O Check-summen Server [4])
OCIT-C Version [1])	OCIT-C Version [1])	OCIT-C Version [1])	OCIT-C Version [1])	OCIT-C Version [1])
OCIT-C Checksumme [1])	OCIT-C Checksumme [1])	OCIT-C Checksumme [1])	OCIT-C Checksumme [1])	OCIT-C Checksumme [1])
Auftraggeber [1])	Auftraggeber [1])	Auftraggeber [1])	Auftraggeber [1])	Auftraggeber [1])
OCIT-O Check-summe Gerät [3])	OCIT-O Check-summe Gerät [3])	OCIT-O Check-summen Gerät [3])	OCIT-O Check-summe Gerät [3])	OCIT-O Check-summ Gerät [3])
Build-Nr. [3])	Build-Nr. [3])	Build-Nr. [3])	Build-Nr. [3])	Build-Nr. [3])
Session-ID [5])	Session-ID [5])	Session-ID [5])	Session-ID [5])	Session-ID [5])
Zeitstempel Über-tragung beendet [3])	Zeitstempel Über-tragung beendet [3])	Zeitstempel Über-tragung beendet [3])	Zeitstempel Über-tragung beendet [3])	Zeitstempel Über-tragung beendet [3])
Zeitstempel Akti-vierung [3])	Zeitstempel Aktivie-rung [3])	Zeitstempel Akti-vierung [3])	Zeitstempel Akti-vierung [3])	Zeitstempel Akti-vierung [3])
Fully Qualified Domain Name [3])	Fully Qualified Domain Name [3])	Fully Qualified Domain Name [3])	Fully Qualified Domain Name [3])	Fully Qualified Domain Name [3])
OCIT-O Checksumme Gerät Gesamt [3])				
Build-Nr. Gesamt [3])				

Abbildung 6: Schema der Versorgungsdatenblöcke und Versionierungsdaten Anwenderversorgung

1) VomVIAP gebildet und im Lichtsignalsteuergerät gespeichert.
2) Vom Herstellerwerkzeug gebildet und im Lichtsignalsteuergerät gespeichert.
3) Im Lichtsignalsteuergerät erzeugt und gespeichert.
4) Im VD Server erzeugt und gespeichert.
5) Im VD Server erzeugt und im Lichtsignalsteuergerät gespeichert

Blau umrandet: Versionierungsdaten des OCIT-C VD Servers, die in OCIT-C Komponenten verwaltet werden.

In OCIT-C LSA Versorgungsdaten teilweise stand. Versorgungsdaten, nur proprietär versorgbar und auslesbar **Herstellerversorgung**			
Gerätetechnik		**Sicherheitstechnik**	
OCIT-C Versorungsdaten	Proprietäre Daten	OCIT-C Sicherheitsdaten	Proprietäre Daten
.....	.....	.....	.....
.....	.....	.....	.....
.....	.....	.....	.....
Hersteller-Version [2])		Hersteller-Version [2])	
Hersteller-Checksumme [2])		Hersteller-Checksumme [2])	
Hersteller-Checksumme Gerät [3])		Hersteller-Checksumme Gerät [3])	
Build-Nr. [3])		Build-Nr. [3])	
OCIT-O Checksumme Gerät Gesamt [3])			
Build-Nr. Gesamt [3])			

Abbildung 7: Schema der Versorgungsdatenblöcke und Versionierungsdaten Herstellerversorgung

[2]) Vom Herstellerwerkzeug gebildet und im Lichtsignalsteuergerät gespeichert.
[3]) Im Lichtsignalsteuergerät erzeugt und gespeichert.

Anwenderversorgung		**Herstellerversorgung**	
Verkehrstechnik	Topologie	Gerätetechnik	Sicherheitstechnik
OCIT-O Checksumme Gerät Gesamt [3])			
Build-Nr. Gesamt [3])			

Abbildung 8: Schema Versionierungsdaten

Schwarz umrandet: Versionierungsdaten, die im Lichtsignalsteuergerät verwaltet werden (OCIT-O Objekt Versionierung)

Tabelle 1: Beschreibung der Versionierungsdaten

Versionierungsdaten	Quelle	Zeitpunkt der Entstehung	Speicherorte	Beschreibung
Versionierungsdaten des OCIT VD Servers, die in OCIT-Komponenten verwaltet werden:				
OCIT-O Checksumme Server Block 1 bis Block 5	VD Server	Während einer Versorgungstransaktion.	VD Server VIAP	*OCIT-O Checksummen Server* werden im Server über die ins Lichtsignalsteuergerät zu übertragenen Versorgungsdaten mittels eines **standardisierten Verfahrens** [4]gebildet. Option: Bildung der OCIT-O Checksummen Server im VD-Server nach Abschluss der Planung auf Anforderung des Planungstools. Damit kann bereits zum Planungszeitpunkt, ohne die Versorgung in das Lichtsignalsteuergerät zu übertragen, jene Checksumme erzeugt werden, die nach einer Versorgung auch im Lichtsignalsteuergerät erzeugt werden soll.
Versionierungsdaten, die im Lichtsignalsteuergerät verwaltet werden (OCIT-O Objekt Versionierung)				
OCIT-C Version Block 1 bis Block 5	Planer	Planungsvorgang	VIAP OCIT-O Objekt Versionierung	Für jeden Block der Anwenderversorgung werden vom Planer am VIAP die OCIT-C Versionsbezeichnungen vergeben, beim Versorgungsvorgang ins Lichtsignalsteuergerät übertragen und dort gespeichert. Diese Informationen werden im Lichtsignalsteuergerät nicht verändert und können über OCIT-O ausgelesen werden.
OCIT-C Checksumme Block 1 bis Block 5	VIAP	Planungsvorgang	VIAP OCIT-O Objekt Versionierung	Für jeden Block der Anwenderversorgung werden vomVIAP die OCIT-C Checksummen berechnet, beim Versorgungsvorgang ins Lichtsignalsteuergerät übertragen und dort gespeichert. Diese Informationen werden im Lichtsignalsteuergerät nicht verändert und können über OCIT-O ausgelesen werden.

[4] Ziel ist es, die OCIT-O Checksumme Server und OCIT-O Checksumme Gerät so zu bilden, dass bei gleichen Versorgungsdaten die gleiche Checksumme entsteht.

Versionierungsdaten	Quelle	Zeitpunkt der Entstehung	Speicherorte	Beschreibung
Auftraggeber Block 1 bis Block 5	Planer	Planungsvorgang	VIAP OCIT-O Objekt Versionierung	Für jeden Block der Anwenderversorgung werden vom Planer am VIAP die Namen der Auftraggeber (String 256 Zeichen) eingetragen, ins Lichtsignalsteuergerät übertragen und dort gespeichert. Diese Informationen werden im Lichtsignalsteuergerät nicht verändert und können über OCIT-O ausgelesen werden.
OCIT-O Checksumme Gerät Block 1 bis Block 5	Lichtsignalsteuergerät	Nach Aktivierung der Versorgung eines Blocks oder bei lokalen Versorgungsänderungen.	OCIT-O Objekt Versionierung	*OCIT-O Checksummen Gerät* werden über die im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten mittels des **standardisierten Verfahrens** gebildet. Die Checksummen können über OCIT-O ausgelesen werden.
Build-Nummer Block 1 bis Block 5, Block Gerätetechnik, Block Sicherheitstechnik	Lichtsignalsteuergerät	Nach Aktivierung der Versorgung eines Blocks oder bei lokalen Versorgungsänderungen.	OCIT-O Objekt Versionierung	Im Steuergerät wird bei jeder Aktivierung eines Versorgungsblocks eine neue, den Blöcken zugeordnete *Build-Nummer* erzeugt, indem ein Zähler hochgezählt wird. Eine Änderung der Build-Nummer signalisiert dem Versorger eine Änderung der Versorgungsdaten im Gerät. Ein Rücksetzen ist nicht vorgesehen, der Zähler läuft einfach über. Build-Nummern können von den Versorgungswerkzeugen über OCIT-O ausgelesen werden.
Session-ID Block 1 bis Block 5	Lichtsignalsteuergerät	Während eines Versorgungsvorgangs	OCIT-O Objekt Versionierung	Die während einer Versorgung eines Blocks verwendete *Session-ID* (entspricht der im OCIT VD Server erzeugten OCIT-O Job-ID) wird im Lichtsignalsteuergerät gespeichert und kann von den Versorgungswerkzeugen über OCIT-O ausgelesen werden.
Zeitstempel Übertragung beendet Block 1 bis Block 5	Lichtsignalsteuergerät	Nach Abschluss der Übertragung	OCIT-O Objekt Versionierung	Der *Zeitstempel Übertragung beendet* enthält die Uhrzeit beim Ende der Übertragung der Versorgungsdaten eines Blocks. Er wird im Lichtsignalsteuergerät gespeichert und kann über OCIT-O ausgelesen werden.
Zeitstempel Aktivierung	Lichtsignal-	Nach Aktivierung der	OCIT-O Objekt	Der *Zeitstempel Aktivierung* enthält die Uhrzeit zu der die aktuell

Versionierungsdaten	Quelle	Zeitpunkt der Entstehung	Speicherorte	Beschreibung
Block 1 bis Block 5	steuergerät	Versorgung eines Blocks	Versionierung	gültige Versorgung eines Blocks von einem VD Server aus aktiviert wurde. Er wird im Lichtsignalsteuergerät gespeichert und kann über OCIT-O ausgelesen werden.
Fully Qualified Domain Name Block 1 bis Block 5	Versorgungswerkzeug	Nach Aktivierung der Versorgung eines Blocks	OCIT-O Objekt Versionierung	Fully Qualified Domain Name (FQDN) oder falls der Reverse Lookup fehlschlägt die IP-Adresse (in dotted dezimal notation) von der aus die letzte erfolgreiche Versorgung dieser Art übernommen wurde. Im Fall der direkten lokalen Versorgung mittels nicht IP- basiertem Herstellertool ist der Name des Geräts selbst einzutragen. Der FQDN wird im Lichtsignalsteuergerät gespeichert und kann über OCIT-O ausgelesen werden.
Hersteller-Version Block Gerätetechnik Block Sicherheitstechnik	Spezialist des Geräteherstellers	Planungsvorgang	Herstellerwerkzeug OCIT-O Objekt Versionierung	Für jeden dieser Blöcke werden vom Spezialisten am Herstellerwerkzeug die Versionsbezeichnungen vergeben, ins Lichtsignalsteuergerät übertragen und dort gespeichert. Diese Informationen werden im Lichtsignalsteuergerät nicht verändert und können über OCIT-O ausgelesen werden.
Hersteller-Checksumme Block Gerätetechnik, Block Sicherheitstechnik	Herstellerwerkzeug	Nach Aktivierung der Versorgung eines Blocks oder bei lokalen Versorgungsänderungen.	Herstellerwerkzeug OCIT-O Objekt Versionierung	Für jeden dieser Blöcke werden vom Herstellerwerkzeug die *Hersteller-Checksummen* mittels des **standardisierten** oder eines **proprietären Verfahrens** gebildet, ins Lichtsignalsteuergerät übertragen und dort gespeichert. Diese Informationen werden im Lichtsignalsteuergerät nicht verändert und können über OCIT-O ausgelesen werden.
Hersteller-Checksumme Gerät Block Gerätetechnik, Block Sicherheitstechnik	Lichtsignalsteuergerät	Nach Aktivierung der Versorgung eines Blocks oder bei lokalen Versorgungsänderungen.	OCIT-O Objekt Versionierung	*Hersteller-Checksummen Gerät* werden über die im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten mittels des **standardisierten** oder eines **proprietären Verfahrens** gebildet. Die Checksummen können über OCIT-O ausgelesen werden.

Versionierungsdaten	Quelle	Zeitpunkt der Entstehung	Speicherorte	Beschreibung
OCIT-O Checksumme Gerät gesamt	Lichtsignalsteuergerät	Nach Aktivierung der Versorgung eines Blocks oder bei lokalen Versorgungsänderungen.	OCIT-O Objekt Versionierung	Die *OCIT-O Checksumme gesamt* wird über alle im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten mittels des **standardisierten** oder eines **proprietären Verfahrens** gebildet. Die Checksumme kann über OCIT-O ausgelesen werden.
Build-Nummer gesamt	Lichtsignalsteuergerät	Nach Aktivierung der Versorgung eines Blocks oder bei lokalen Versorgungsänderungen.	OCIT-O Objekt Versionierung	Im Steuergerät wird bei jeder Aktivierung eines der Versorgungsblöcke eine neue *Build-Nummer gesamt* erzeugt, in dem ein Zähler hochgezählt wird. Eine Änderung der Build-Nummer signalisiert dem Versorger eine Änderung der Versorgungsdaten im Gerät. Ein Rücksetzen ist nicht vorgesehen, der Zähler läuft einfach über. Die Build-Nummer gesamt kann über OCIT-O ausgelesen werden.

Hinweis: Es wird empfohlen, bei der Darstellung von Checksummen an Oberflächen die im OCIT-O Protokoll für Meldungen festgelegte Darstellungsform zu verwenden.

Die Versionierungsdaten geben Auskunft darüber:

- welche Version versorgt ist (Versionen),
- ob die zur Version zugehörigen und im Versorgungswerkzeug gespeicherten Daten jene sind, die ins Lichtsignalsteuergerät übertragen wurden (OCIT-C, OCIT-O und Hersteller-Checksummen),
- das geändert wurde (Build-Nr.),
- welcher Block lokal oder auf anderen Wegen verändert wurde (OCIT-O Checksummen),
- wann übertragen wurde (Zeitstempel Übertragung beendet)
- wann geändert wurde (Zeitstempel Aktivierung) und
- von wo aus geändert wurde (Fully Qualified Domain Name, Session-ID).

Hinweis: Es wird empfohlen bei jeder Aktivierung einer Versorgung alle Versionierungsdaten aus dem Lichtsignalsteuergerät und dem VD Server zu lesen und in der Versionsverwaltung des VIAPs abzulegen. Was geändert wurde, kann durch blockweises Auslesen der im Lichtsignalsteuergerät tatsächlich vorhandenen Anwenderversorgung gefunden werden.

3.2.1 Standard-Verfahren zur Checksummenbildung

Die „OCIT-O Checksummen Gerät“ sollen in allen 5 Versorgungsblöcken so gebildet werden, dass bei gleichen Versorgungsdaten in allen OCIT konformen Lichtsignalsteuergeräten die gleichen Checksummen entstehen. Um dies zu erreichen, wird für die Checksummenbildung ein standardisiertes Verfahren verwendet.

Es wird empfohlen auch im VD-Server dieses Verfahren anzuwenden, um nach Abschluss der Planung die „OCIT-O Checksumme Server“ zu bilden. Damit kann bereits zum Planungszeitpunkt, ohne die Versorgung in das Lichtsignalsteuergerät zu übertragen, jene Checksumme erzeugt werden, die nach einer Versorgung auch im Lichtsignalsteuergerät erzeugt werden soll. Man erreicht dadurch eine sehr hohe Sicherheit, dass die vom Planer gewollte Versorgung auch tatsächlich im Lichtsignalsteuergerät umgesetzt wird.

Die Vorschrift zur Checksummenberechnung besteht aus drei Komponenten:

- Festlegung, in welcher Reihenfolge die Versorgungsdaten der Versorgungsobjekte zur Bildung der Checksummen gelesen werden (alle Elemente der Versorgungsobjekte in aufsteigender Reihenfolge, siehe dazu die Festlegungen in Pkt. 3.3 zu „sortiert und gelesen“).
- dass die Versorgungsdaten vom VD-Server in dieser Reihenfolge übertragen werden, und
- Vorschrift zur Berechnung der Checksumme (mit Standardalgorithmus SHA-1, siehe Dokument OCIT-O Basis).

Hinweis: Die „OCIT-O Checksummen Gerät“ wird nicht aus den Daten im Versorgungsdatenpuffer gebildet, sondern aus den „echten“ Versorgungsdaten, die entweder nach der Aktivierung einer Versorgung oder bei lokalen Änderungen, abgespeichert werden. Der Hersteller des Lichtsignalsteuergeräte muss sicherstellen, dass die geladenen Daten auch noch zu einem späteren Zeitpunkt so zur Verfügung stehen, so dass die Checksumme korrekt gebildet werden kann und die Daten zurückgelesen werden können.

Vorschrift zur Checksummenbildung:

Die Vorschrift gilt sinngemäß für VD-Server und Lichtsignalsteuergerät, die Sortierung wird jedoch zwingend im VD-Server und bei Bedarf im Lichtsignalsteuergerät vorgenommen.

Für die Bildung der Checksumme über alle Objekte eines Blockes werden alle Objekte sortiert und in eine Folge von Bytes serialisiert. Auch bei einer optional möglichen Teilversorgung[5] müssen die Checksummen über den gesamten Block gebildet werden. Über diese Folge wird dann ein SHA1-Digest gebildet:

- Durch die Verkettung von Member, OType und Pfad eines Objektes wird zunächst für jedes Objekt ein dem Objekt zugeordneter Bytestring, der Sortierschlüssel, gebildet. Member und OType werden dabei in Network Byte Order geschrieben.

[5] Teilversorgungen sind in OCIT-O Lstg V2.0 nicht vorgesehen, die Objektmodellierung berücksichtigt jedoch diese Option auf die Zukunft.

Member	OType	Pfad
2 Byte	2 Byte	0..n Byte

Abbildung 9: Aufbau des für den Vergleich zweier Objekte genutzten Sortierschlüssel

- Mittels eines paarweisen Vergleiches dieser Sortierschlüssel beginnend beim höherwertigen Byte des Members und endend beim letzten Byte des kürzeren der Pfade beider Objekte werden die Objekte verglichen und sortiert. Die einzelnen Bytes der Schlüssel werden dabei paarweise und strikt numerisch verglichen. Sind Member, OType und der Pfad beider Objekte bis zum letzten Byte des kürzeren Pfades gleich, so ist das Objekt mit dem längeren Pfad als größer zu betrachten. In der sortierten Folge der Objekte steht nun das Objekt mit dem kleinsten Member und innerhalb der Objekte mit gleichem Member das Objekt mit dem numerisch kleinsten OType am Anfang.
- Die entstandene Folge von Objekten wird nun als Liste versorgbarer Objekte, wie für SupplyTransaction: ReadVD bzw. Transaction: AddChangeSet im Folgenden beschrieben, im BTPPL-Format serialisiert. Die in den beiden Methoden ebenfalls übertragene „Anzahl folgender VersorgbaresObjekt Elemente“ wird jedoch nicht mit in die Berechnung der Checksumme einbezogen.

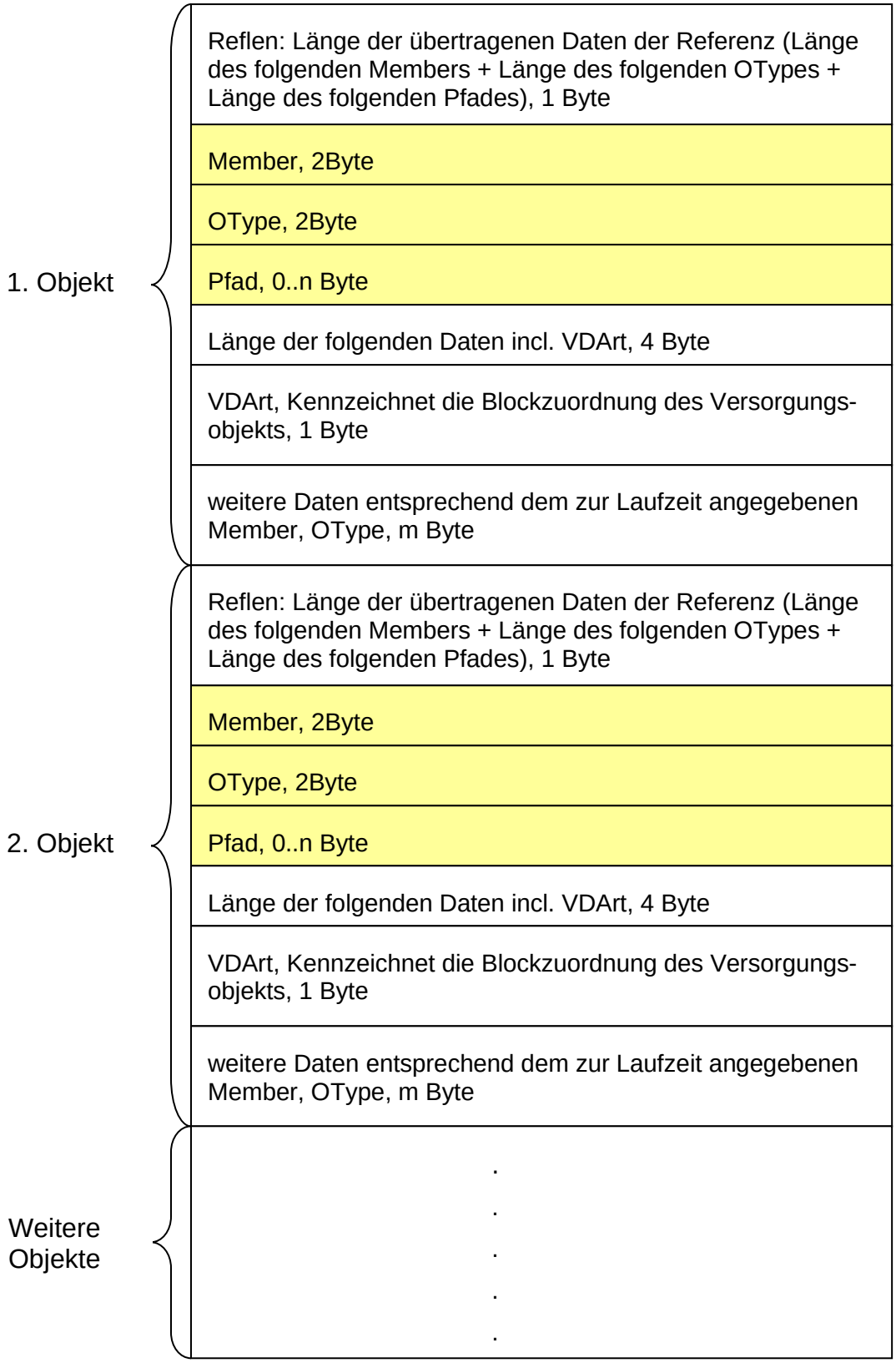


Abbildung 10: Über diese serialisierte Folge von Bytes wird ein SHA1-Digest gebildet und als Checksumme dieses Blockes verwendet.

Die Checksumme eines Blocks wird durch Aufruf der Funktion sha1_chksum(vdart) berechnet. Die Vorschrift zur Bildung der Checksumme wird hier als C-Pseudocode dargestellt:

```
/** Die Funktion compare (cmp) wird bei der Sortierung benutzt um
    die Reihenfolge der Objekte zu definieren.
        a <  b ->  -1
        a == b ->  0
        a >  b ->  1 */

int cmp(int a, int b) {
        if (a < b) return -1;
        else if (a == b) return 0;
        else return 1;
}

/** Funktion zum Vergleich von 2 Pfade, um die Pfade zu sortieren. */

int path_cmp(path a, path b) {
        int member_cmp = cmp(a.member, b.member);
        if (member_cmp != 0) return member_cmp;

        int otype_cmp = cmp(a.otype, b.otype);
        if (otype_cmp != 0) return otype_cmp;

        int relknoten_cmp = cmp(a.relknoten, b.relknoten);
        if (relknoten_cmp != 0) return relknoten_cmp;

        return cmp(a.nr, b.nr);
}

/** Berechne die SHA1 Checksumme der BTPPL Daten. */
sha1_chksum sha1(btpplized_object p);

/** Hänge 2 BTPPL Pakete a und b aneinander */
btpplized_object concat(btpplized_object a, btpplized_coding b);

/** Berechne die Checksumme eines Versorgungsblocks vdart */
sha1_chksum chkblock(int vdart) {

        // Hole alle Objekte des Blocks
        object[] o = all_objects_in_block(vdart);

        // Berechne den Pfad für alle Objekte
        path[] p = path(o);

        // Sortiere die Pfade mit der Funktion path_cmp (s.o.)
        sort(p, path_cmp);

        // coding soll die Codierung aller Pfade + Objekte
        // des Blocks enthalten
        btpplized_object coding;
        for (int i = 0; i < o.length; i++) {
             / Füge den Pfad eines Objekts zur Codierung zu
             coding = concat(coding, btppl(p[i]));
             // Füge den Aufruf von get des zum Pfad gehörenden
             // Objekts der Codierung zu.
             // Dazu sind die separat beschriebenen Bedingungen
             // für die Versorgungsobjekte zu erfüllen
             coding = concat(coding, btppl(call_method(object(p[i]), get)));
        }

        // Die eindeutige Checksumme der Versorgungsobjekte
        // eines Blocks wird hier zurückgegeben.
        return sha1_chksum(coding);
}
```

3.2.2 Objekt Versionierung

Dieses Objekt verwaltet die Versionierungsdaten des Lichtsignalsteuergeräts (siehe Pkt.3.23.2.2, Tabelle 1)

3.2.2.1.1 Enum VDArt

Definition der im Lichtsignalsteuergerät einzeln versionierten Versorgungsdatenarten. Diese Enumeration wird als Pfad der Versorgungsdaten Versionsinformationen (s. 3.2.2.1.2) verwendet.

Enum VDArt (1: 680)

Name	Description	Value
Grunddaten	Diese Versorgungsdatenart kennzeichnet die Grundversorgungsdaten. (Signalprogramme, verkehrstechnische Zwischen- und Versatzzeiten)	0
Netz	Diese Versorgungsdatenart für Versorgungsdaten mit Netzbezug. Diese sind derzeit nur die Kopfdaten und die Schaltuhr (Kalender und Tagespläne).	1
VA_Steuerverfahren	Versorgungsdatenart der Steuerverfahren (XML/Binär)	2
VA_Parameter	Versorgungsdatenart der verkehrsabhängigen Programme. (Rahmenpläne, ...)	3
MAP	Topologie Daten der Kreuzung im MAP Format	4
Gerätetechnik	Standardisierte sonstige Versorgungsdaten (Detektoren, Signalgruppen, Zuordnung zum Teilknoten, OEPNV Meldepunkte, -ketten). Proprietäre Daten Hardware Zuordnung u.a.	5
Sicherheitstechnik	Standardisierte Sicherheitsdaten, proprietäre Daten.	6

Basetypename=UBYTE maximal reservierter Wertebereich MAX=8

3.2.2.1.2 Objekt VDVersion

Die Meta-Daten der Versorgungsdaten, Version und Datensignatur (Checksumme und Auftraggeber) werden über den VD-Server an das Steuergerät versandt und dort abgelegt. Dazu muss pro versorgten Block eine Instanz VDVersion in jeder Versorgungstransaktion mittels Methode AddChangeSet übertragen werden.

Diese Daten werden vom Steuergerät nur verändert, wenn eine Änderung eines Blocks der Anwenderversorgung ohne Planungstool, d.h. mit herstellerspezifischen Mitteln erfolgt. In diesem Fall müssen im Objekt VDVersion des veränderten Blocks folgende Werte gesetzt werden:

Version: herstellerspezifisch
Checksum: herstellerspezifisch
Auftraggeber: herstellerspezifisch

Das Steuergerät aktualisiert bei jeder Versorgung automatisch die zugehörig(en) LsaVersion und LsaVersionPlus Instanz(en).

Ein Get liefert wie bei allen anderen Versorgungsdatenobjekten die Werte der aktiven Versorgung

Pfad (ab Lichtsignalsteuergerät): RelKnoten(OBJECT_ID_UBYTE=UBYTE)/Art (VDArt=UBYTE)

VDVersion (1:681)

VDVersion		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock
	Version: string	Im Fall der Versorgung über OCIT-C ist dies die **OCIT-C Version**, die der Planer für jeden Block der Anwenderversorgung am VIAP vergibt. Im Falle der Herstellerversorgung ist das die **Hersteller-Version** die der Versorger für die Blöcke Geräte-und Sicherheitstechnik vergibt. Das Lichtsignalsteuergerät speichert die Version ohne sie zu verändern.
	Checksum[0...19]: ui1	**OCIT-C Checksumme:** Für jeden Block der Anwenderversorgung werden vom Standard-Planungs- o. Versorgungswerkzeug die Checksummen erzeugt, ins Lichtsignalsteuergerät übertragen und dort gespeichert. **Hersteller-Checksumme:** Für jeden Block der Herstellerversorgung werden vom Versorgungswerkzeug die Checksummen erzeugt, ins Lichtsignalsteuergerät übertragen und dort gespeichert. Die Checksummen werden im Lichtsignalsteuergerät nicht verändert und können über OCIT-O ausgelesen werden.
	Auftraggeber: string	Für jeden Block der Anwenderversorgung werden vom Planer am VIAP die Namen der **Auftraggeber** (String 256 Zeichen) eingetragen, ins Lichtsignalsteuergerät übertragen und dort gespeichert. Diese Informationen werden im Lichtsignalsteuergerät nicht verändert und können über OCIT-O ausgelesen werden.

	OCIT_I_Session_ID	Diese Session ID wird von VD-Servern nicht genutzt. Hier muss der VD-Server immer den NULLVALUE (ULONG 0xFFFFFFFF) verwenden. Anmerkung: Das Feldgerät kann sich hierauf nicht verlassen, da alte VD Server ggf. dieses Feld noch verwenden.

3.2.2.1.3 LsaVersion

Diese Versionsobjektinstanz aktualisiert das Lichtsignalsteuergerät selbstständig im Verlauf einer erfolgreichen Versorgungsdatenaktivierung.

Hinweis: Neue VD-Server sollen das Objekt LsaVersionPlus nutzen, um die Informationen abzufragen. Feldgeräte müssen die LsaVersion dennoch zur Verfügung stellen (zur Nutzung in Meldungen und aus Rückwärtskompatibilitätsgründen).

Pfad (ab Lichtsignalsteuergerät): RelKnoten(OBJECT_ID_UBYTE=UBYTE)/Art (VDArt=UBYTE)

LsaVersion (1:682)

LsaVersion		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Checksum[0...19]: ui1	**OCIT-O Checksummen Gerät** werden über die im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten der Anwenderversorgung mittels des standardisierten Verfahrens gebildet. **Hersteller-Checksummen Gerät** werden über die im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten der Blöcke Geräte-und Sicherheitstechnik mittels des standardisierten oder eines proprietären Verfahrens gebildet. Die Checksummen können über OCIT-O ausgelesen werden. Gleiche Versorgungsdaten müssen gleiche Checksumme ergeben. Ist die Checksumme verschieden ist sicher, dass die zugehörigen Versorgungsdaten auch verschieden sind

LsaVersion		
METHOD	**Name**	**Beschreibung**
	BuildNr	Im Steuergerät wird bei jeder erfolgreichen Aktivierung eines Versorgungsblocks eine neue, den Blöcken zugeordnete **Build-Nummer** erzeugt. Es handelt sich dabei um einen Zähler, den das Gerät automatisch mit jeder (auch mittels lokalen herstellerspezifischem Tool vorgenommenen) Versorgung hochzählt. Eine Änderung der Build-Nummer signalisiert dem Versorger eine Änderung der Versorgungsdaten im Gerät. Ein Rücksetzen ist nicht vorgesehen, der Zähler läuft einfach über. Build-Nummern können von den Versorgungswerkzeugen über OCIT-O ausgelesen werden.
	Aktivierungszeitpunkt	**Zeitstempel Aktivierung:** Der Zeitpunkt zu dem die aktuelle gültige Versorgung von einem VD Server aus aktiviert wurde, wird im Lichtsignalsteuergerät gespeichert und ist auslesbar
	UebertragungsEndezeitpunkt	Der **Zeitstempel Übertragung beendet** enthält die Uhrzeit beim Ende der Übertragung der Versorgungsdaten eines Blocks (=Zustandswechsel der Versorgungstransaktion nach completed). Er wird im Lichtsignalsteuergerät gebildet und gespeichert, er kann über OCIT-O ausgelesen werden. Im Fall einer Versorgung ohne Versorgungstransaktion (z.B. Herstellerspezifischen Versorgungsdatenänderung bei VDArt=Gerätetechnik und Sicherheitstechnik) ist der ÜbertragungsEndezeitpunkt gleich dem Aktivierungszeitpunkt zu setzen.
	Origin	**Fully Qualified Domain Name** (oder falls reverse lookup failed die IP-Adresse in dotted dezimal notation) von der aus die letzte erfolgreiche Versorgung dieser Art übernommen wurde. Im Fall der direkten lokalen Versorgung mittels nicht IP basiertem Herstellertool ist der **Fully Qualified Domain Name** des Geräts selbst einzutragen.

3.2.2.1.4 LsaVersionPlus

Diese Versionsobjektinstanz aktualisiert das Lichtsignalsteuergerät selbstständig im Verlauf einer erfolgreichen Versorgungsdatenaktivierung. Insbesondere enthält dieses Objekt die OCIT-O-SYSJOBID der Transaction, die identisch zur OCIT_I_Session_ID ist.

Pfad (ab Lichtsignalsteuergerät): RelKnoten (OBJECT_ID_UBYTE=UBYTE)/Art (VDArt=UBYTE)

LsaVersionPlus (1:684)

LsaVersionPlus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Checksum[0...19] : ui1	**OCIT-O Checksummen Gerät** werden über die im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten der Anwenderversorgung mittels des standardisierten Verfahrens gebildet. **Hersteller-Checksummen Gerät** werden über die im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten der Blöcke Geräte-und Sicherheitstechnik mittels des standardisierten oder eines proprietären Verfahrens gebildet. Die Checksummen können über OCIT-O ausgelesen werden. Gleiche Versorgungsdaten müssen gleiche Checksumme ergeben. Ist die Checksumme verschieden ist sicher, dass die zugehörigen Versorgungsdaten auch verschieden sind

LsaVersionPlus		
METHOD	**Name**	**Beschreibung**
	BuildNr	Im Steuergerät wird bei jeder erfolgreichen Aktivierung eines Versorgungsblocks eine neue, den Blöcken zugeordnete **Build-Nummer** erzeugt. Es handelt sich dabei um einen Zähler, den das Gerät automatisch mit jeder (auch mittels lokalen herstellerspezifischem Tool vorgenommenen) Versorgung hochzählt. Eine Änderung der Build-Nummer signalisiert dem Versorger eine Änderung der Versorgungsdaten im Gerät. Ein Rücksetzen ist nicht vorgesehen, der Zähler läuft einfach über. Build-Nummern können von den Versorgungswerkzeugen über OCIT-O ausgelesen werden.
	Aktivierungszeitpunkt	**Zeitstempel Aktivierung:** Der Zeitpunkt zu dem die aktuelle gültige Versorgung von einem VD Server aus aktiviert wurde, wird im Lichtsignalsteuergerät gespeichert und ist auslesbar
	UebertragungsEndezeitpunkt	Der **Zeitstempel Übertragung beendet** enthält die Uhrzeit beim Ende der Übertragung der Versorgungsdaten eines Blocks (=Zustandswechsel der Versorgungstransaktion nach completed). Er wird im Lichtsignalsteuergerät gebildet und gespeichert, er kann über OCIT-O ausgelesen werden. Im Fall einer Versorgung ohne Versorgungstransaktion (z.B. Herstellerspezifischen Versorgungsdatenänderung bei VDArt=Gerätetechnik und Sicherheitstechnik) ist der Übertragungs-Endezeitpunkt gleich dem Aktivierungszeitpunkt zu setzen.
	Origin	**Fully Qualified Domain Name** (oder falls reverse lookup failed die IP-Adresse in dotted dezimal notation) von der aus die letzte erfolgreiche Versorgung dieser Art übernommen wurde. Im Fall der direkten lokalen Versorgung mittels nicht IP basiertem Herstellertool ist der **Fully Qualified Domain Name** des Geräts selbst einzutragen.

LsaVersionPlus		
METHOD	**Name**	**Beschreibung**
	SYSJOBID	Die während einer Versorgung eines Blocks verwendete **SYSJOBID** der Transaction (= die OCIT_I_Session_ID die vom VD-Server vergeben wird) wird im Lichtsignalsteuergerät gespeichert und kann von den Versorgungswerkzeugen über OCIT-O ausgelesen werden. Bei lokalen Versorgungen wird hier die lokal erzeugte SYSJOBID eingetragen.

Hinweis: Mit diesem Objekt wird ein Fehler in der Checksummenberechnung der Versorgungsdaten behoben, der bei dem optional möglichen Vergleich OCIT-O Checksumme Server mit OCIT-O Checksumme Gerät auftritt. Dabei ändert sich die Session ID mit jedem Versorgungsvorgang. Daher ändert sich auch bei gleichbleibenden Daten bei jedem Versorgungsvorgang die Checksumme. Das neue Objekt LsaVersionPlus wurde eingeführt, um diesen Fehler zu beheben und um rückwärtskompatibel zu bleiben, das heißt, im Feld befindliche Versorgungsdaten -Server funktionieren auch mit Steuergeräten ab Ausgabe V2.0 A04.

Geräte ab Ausgabe V2.0 A04 müssen das Objekt LsaVersionPlus zusätzlich zum Objekt LsaVersion unterstützen. Das Objekt LSAVersionPlus enthält die LsaVersion und die SYSJOBID der Transaktion. Um den Fehler im System zu beseitigen, muss die SessionID von (neuem) Versorgungsdaten-Server auf NULL gesetzt werden. Die

3.2.2.1.5 GesamtVersion

Gesamt Buildnummer und Prüfsumme. Das Lichtsignalsteuergerät aktualisiert diese selbstständig im Verlauf einer erfolgreichen Versorgungsdatenaktivierung.

Die GesamtVersion zeigt jede Versorgungsänderung an.

Pfad (ab Lichtsignalsteuergerät): keiner, d.h. es gibt genau eine Instanz des Objekts GesamtVersion pro Lichtsignalsteuergerät.

GesamtVersion (1:683)

GesamtVersion		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen

	Checksum[0...19]: ui1	Die **OCIT-O Checksumme Gerät gesamt** wird über alle im Lichtsignalsteuergerät abgespeicherten Versorgungsdaten mittels des standardisierten oder eines proprietären Verfahrens gebildet. Diese Checksumme, folgt den üblichen technischen Regeln zur Bildung von Checksummen. Die Checksumme kann nur gelesen werden. Gleiche Versorgungsdaten müssen gleiche Checksumme ergeben. Ist die checksum verschieden, ist sicher, dass die zugehörigen Versorgungsdaten auch verschieden sind.
	BuildNr	Im Steuergerät wird bei jeder erfolgreichen Aktivierung eines Versorgungsblocks eine neue, allen Blöcken zugeordnete **Build-Nummer gesamt** erzeugt. Es handelt sich dabei um einen Zähler, den das Gerät automatisch mit jeder (auch mittels lokalen herstellerspezifischem Tool vorgenommenen) Versorgung hochzählt. Eine Änderung der Build-Nummer gesamt signalisiert dem Versorger eine Änderung der Versorgungsdaten im Gerät. Ein Rücksetzen ist nicht vorgesehen, der Zähler läuft einfach über. Build-Nummern können von den Versorgungswerkzeugen über OCIT-O ausgelesen werden.

3.3 Versorgungsobjekte

Folgende Versorgungsobjekte sind in OCIT-O Lstg V3.0 als „Anwenderversorgung“ standardisiert:

OType Nummern der Versorgungsobjekte, Member=1 (ODG):

OType	**Name**	**Pfad (ab Lichtsignalsteuergerät)**	**VDArt**
669	EProgramm	RelKnoten(UBYTE)/ Nr(UBYTE)	Verkehrstechnische Grunddaten / Festzeit
670	AProgramm	RelKnoten(UBYTE)/ Nr(UBYTE)	Verkehrstechnische Grunddaten / Festzeit
666	SignalprogrammV	RelKnoten(UBYTE)/ Nr(UBYTE)	Verkehrstechnische Grunddaten / Festzeit

667	Versatzzeitenmatrix	RelKnoten(UBYTE)/ Nr(UBYTE)	Verkehrstechnische Grunddaten / Festzeit
668	VTZwischenzeitenmatrix	RelKnoten(UBYTE)/ Nr(UBYTE)	Verkehrstechnische Grunddaten / Festzeit
673	VTMinFreigabe	RelKnoten(UBYTE)/ Nr(UBYTE)	Verkehrstechnische Grunddaten / Festzeit
675	VTMinGesperrt	RelKnoten(UBYTE)/ Nr(UBYTE)	Verkehrstechnische Grunddaten / Festzeit
650	Kopfdaten	RelKnoten(UBYTE)	Daten mit Netzbezug
660	Tagesplan	RelKnoten(UBYTE)/ Nr(UBYTE)	Daten mit Netzbezug
661	Wochenplan	RelKnoten(UBYTE) /Nr(UBYTE)	Daten mit Netzbezug
662	SondertagJaehrlich	RelKnoten(UBYTE)/ Nr(UBYTE)	Daten mit Netzbezug
663	SondertagAufzaehlung	RelKnoten(UBYTE)/ Nr(UBYTE)	Daten mit Netzbezug
664	Zeitbereich	RelKnoten(UBYTE)/ Nr(UBYTE)	Daten mit Netzbezug
672	BinaerVASteuerverfahren	RelKnoten(UBYTE))/ Kennung (STRING)	VA-Steuerverfahren
676	BinaerVAParameter	RelKnoten(UBYTE))/ Kennung(STRING)	VA-Parameter
677	MAP	RelKnoten(UBYTE)	Topologiedaten

3.3.1 Objekt VersorgbaresObjekt

VersorgbaresObjekt ist die BASEDOMAIN für alle versorgbaren Objekte.

VersorgbaresObjekt (1:648)

VersorgbaresObjekt		
METHOD	**Name**	**Beschreibung**
16	**GetVDArt**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Kennzeichnet die Blockzuordnung des Versorgungsobjekts.

Folgende Objekte sind von „VersorgbaresObjekt“ abgeleitet:

- EProgramm
- AProgramm
- SignalprogrammV
- Versatzzeitenmatrix
- VTZwischenzeitenmatrix
- VTMinFreigabe
- VTMinGesperrt
- Kopfdaten
- Tagesplan
- Wochenplan
- Sondertag
- Zeitbereich
- BinaerVASteuerverfahren
- BinaerVAParameter
- VDVersion

3.3.2 Block 1: Verkehrstechnische Grunddaten / Festzeit

3.3.2.1.1 Objekt Eprogramm

In diesem Objekt werden Einschaltprogramme gespeichert.

Während des Einschaltprogramms gelten die Zwischenzeiten des Zielprogramms.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

- EAZeile Elemente sortiert aufsteigend nach Signalgruppennummer.
- Schaltzeit Elemente sortiert aufsteigend nach Schaltzeitpunkt.

Eprogramm (1:669)

Eprogramm		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.

Eprogramm			
METHOD	**Name**		**Beschreibung**
	Bezeichnung: bezeichnungType		Signalprogrammname, (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	Dauer		Dauer des Einschaltprogramms
	Signalsicherungszeitpunkt		Zeitpunkt, an dem die Signalsicherung eingeschaltet wird.
	Anzahl		Anzahl folgender EAZeile Elemente
	EAZeile[1…254]		
		Signalgruppe.Nr	Nummer der zu schaltenden Signalgruppe
		Anzahl	Anzahl folgender SchaltzeitTyp Elemente
		SchaltzeitTyp[1...254]	
		Schaltzeit	Schaltung der Signalgruppe in ein neues Signalbild. Schaltzeit einer Signalgruppe. Zum gleichen Schaltzeitpunkt darf nur eine Zielfarbe angegeben werden.
		Signalbild	Zu schaltendes Signalbild

Hinweis: Der Zustand der Signalgruppen vor dem Einschalten wird bis zum ersten Schaltbefehl der jeweiligen Signalgruppe des Einschaltprogramms beibehalten!

3.3.2.1.2 Objekt AProgramm

In diesem Objekt werden Ausschaltprogramme gespeichert.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

- EAZeile Elemente sortiert aufsteigend nach Signalgruppennummer.
- Schaltzeit Elemente sortiert aufsteigend nach Schaltzeitpunkt.

AProgramm (1:670)

AProgramm		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.

AProgramm				
METHOD	**Name**			**Beschreibung**
	Bezeichnung: bezeichnungType			Signalprogrammname, (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	Dauer			Dauer des Ausschaltprogramms
	Signalsicherungszeitpunkt			Zeitpunkt, an dem die Signalsicherung ausgeschaltet wird.
	Anzahl			Anzahl folgender EAZeile Elemente
	EAZeile[1...254]			
		Signalgruppe.Nr		Nummer der zu schaltenden Signalgruppe
		Anzahl		Anzahl folgender SchaltzeitTyp Elemente
		SchaltzeitTyp[1...254]		
			Schaltzeit	Schaltung der Signalgruppe in ein neues Signalbild. Schaltzeit einer Signalgruppe. Zum gleichen Schaltzeitpunkt darf nur eine Zielfarbe angegeben werden.
			Signalbild	Zu schaltendes Signalbild

Hinweis: Der Zustand der Signalgruppen im Ausschaltzeitpunkt wird bis zum ersten Schaltbefehl der jeweiligen Signalgruppe des Ausschaltprogramms beibehalten!

3.3.2.1.3 Objekt SignalprogrammV

In diesem Objekt werden Signalprogrammdaten gespeichert.

Ein- und Ausschaltprogramme werden in gesonderten Objekten verwaltet.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

- SPZeile Elemente sortiert aufsteigend nach Signalgruppennummer.
- ReferenzUebergang Elemente sortiert nach aufsteigendem Startbild, für die gleichen Starbilder sortiert aufsteigend nach Zielbild.
- Schaltzeit Elemente sortiert aufsteigend nach Schaltzeitpunkt.

Hinweis: Bei Blockversorgung müssen immer alle im Lichtsignalsteuergerät vorhandenen Signalprogramme versorgt werden. Es können keine neuen Signalprogramme hinzugefügt oder bestehende gelöscht werden.

SignalprogrammV (1:666)

SignalprogrammV		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock
	Bezeichnung: BezeichnungType	Signalprogrammname (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	ZWZMatrix.Nr	Nummer der verwendeten Zwischenzeitmatrix. Wenn die Sicherheits-ZWZ-Matrix verwendet wird, steht an der Stelle 0
	VZMatrix.Nr[0 ... 2]: ui1	Nummer der verwendeten Versatzzeitenmatrix: 0 = BB 1 = EE 2 = BE Ist eine Matrix nicht definiert, steht an der entsprechenden Position ein NULLVALUE (0xFF). Die Versorgung von nicht unterstützten VZMatrix-Arten darf vom Gerät zurückgewiesen werden.
	VTMinFreigabe.Nr	Nummer der verwendeten Mindestfreigabezeitenliste
	VTMinGesperrt.Nr	Nummer der verwendeten Mindestgesperrtzeitenliste
	TU	Umlaufzeit des Programms
	EP	Einschaltpunkt, sofern definiert, sonst NULLVALUE
	AP	Ausschaltpunkt, sofern definiert, sonst NULLVALUE
	UP	Kopfdaten Umschaltpunkt (in Siemens-Terminologie GSP), sonst NULLVALUE.
	SY_Vor	Vorrastpunkt, falls vorhanden, sonst NULLVALUE
	SY_Haupt	Hauptrastpunkt, falls vorhanden, sonst NULLVALUE.
	SY_MaxDauer	Maximale Dauer, für die synchronisiert wird falls vorhanden, sonst NULLVALUE.

SignalprogrammV			
METHOD	Name		Beschreibung
	SignalzeitenVersatz		Versatzzeit bezogen auf die Versatzzeit-Berechnungsvorschrift für die Synchronisierung des Gerätes. Der Wert kann auf Geräten mit unterschiedlichen Versatz-Berechnungen zu verschiedenen Zeiten führen. Anmerkung: Wenn kein Versatz gewünscht ist, soll der Wert 0 verwendet werden.
	EProgramm.Nr		Zugehöriges Einschaltprogramm Der VD Server muss prüfen, ob das Einschaltprogramm vorhanden ist.
	AProgramm.Nr		Zugehöriges Ausschaltprogramm DerVD Server muss prüfen, ob das Ausschaltprogramm vorhanden ist.
	Anzahl		Anzahl folgender SPZeile Elemente
		Signalgruppe.Nr	Nummer der zu schaltenden Signalgruppe
		Anzahl	Anzahl folgender Referenzen auf Zusatzübergänge [0 ... 2]
		ReferenzUebergang	Verweis auf Zusatzübergänge die in den Signalgruppendaten definiert sind.
		Anzahl	Anzahl folgender SchaltzeitType Elemente
		Schaltungen oder Dauersignalbild [1 ... 10]	
		Schaltzeitpunkt	Schaltzeit der Signalgruppe (NULLVALUE = Dauersignalbild)
		Signalbild	Zu schaltendes Signalbild

Hinweis: SY_Vor und SY_Haupt sind Synchronisierpunkte zum Dehnen und Stauchen eines Signalplans. Bei Steuerverfahren, die diese Methode verwenden, läuft der TX bis zum SY_Vor, springt direkt auf SY_Haupt und wartet dort auf die Synchronisierung. Achtung: Schaltbefehle zwischen SY_Vor und SY_Haupt sind unzulässig und werden nicht ausgeführt.

3.3.2.1.4 OCIT-O Referenzen auf Zusatzübergänge

In OCIT-O werden Zusatzübergänge über die im OCIT-C Format vorliegenden Namen der Zusatzübergänge referenziert.

In OCIT-C sind die Zusatzübergänge in den Signalgruppendaten der „intersection_config_data.xsd“ definiert. Die im Signalprogramm gültigen Zusatzübergänge werden in den Signalplanzeilen für jede Signalgruppe referenziert. Der Name ist ein String beliebiger Länge, wobei die Namensgebung standardisiert ist, um das Format

leichter lesbar zu machen. Der Name setzt sich aus dem Bitcode des Start-Signalbildes, den Namen aller Übergangs-Signalbilder inkl. Dauer und dem End-Signalbild zusammen.

Um in OCIT-O auf eine kompakte Darstellung der Referenz auf einen Zusatzübergang zu kommen wird der Name wie in Pkt. 3.3.2.1.4.1 beschrieben kodiert. Aufgrund der Darstellung der Namen in der intersection_config_data .xsd, die über Pattern vorgegeben ist, ist eine einfache Umwandlung möglich. Auch eine Zuordnung der Referenzen auf die im Gerät vorhanden hinterlegten Zusatzübergänge ist möglich.

Es wird festgelegt, dass bei OCIT-O maximal 3 Übergangselemente pro Übergang möglich sind.

Hinweise: Die Versorgung kann abgelehnt werden, wenn die übergebenen Zusatzübergänge nicht in der Signalsicherung vorhanden sind. Bei der Verwendung von Zusatzübergängen sind gerätespezifische Grenzen besonders kritisch, da die resultierenden Kombinationen die Möglichkeiten der Geräte überschreiten können.

3.3.2.1.4.1 Kodierung der Referenz auf einen Zusatzübergang

Kodierung als ui8:

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Startbild: ui1	Länge 1. Übergangselement: ui1	Signalbild 1. Übergangselement: ui1	Länge 2. Übergangselement: ui1	Signalbild 2. Übergangselement: ui1	Länge 3. Übergangselement: ui1	Signalbild 3. Übergangselement: ui1	Zielbild

Nicht verwendete Übergangselemente werden mit der Länge (0x0) und Signalbild dunkel (0x0) kodiert.

Beispiel: Übergang „rot-1srotgelb-gruen“

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
0x03 (rot)	0x0a (10 x 100 ms)	0x0f (rtge)	0x0	0x0	0x0	0x0	0x30 (gruen)

Achtung: Bei dieser Kodierung handelt es sich ausschließlich um eine Referenz (mittels Namen) auf einen Zusatzübergang. Es **handelt sich nicht um die realen Daten** des Zusatzübergangs.

3.3.2.1.5 Objekt Versatzzeitenmatrix

In diesem Objekt werden die Versatzzeitenmatrizen gespeichert.

Im Steuergerät sind bis zu neun verschiedene Versatzzeitenmatrizen vorhanden, von denen bis zu drei in der Signalprogrammversorgung ausgewählt werden.

Es darf gleichzeitig nur eine Versatzzeitenmatrix gleicher Art angewählt werden.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

- Zeilen aufsteigend sortiert nach SGrBasis, dann nach SGrAbhaengig, dann nach Operator.

Hinweis: Die Versatzzeiten können vom VA-Verfahren verwendet werden. Der Umgang mit den Versatzzeiten ist nicht standardisiert. Die maximale Anzahl und Nummerierung der Versatzzeitenmatrizen kann gerätespezifisch Restriktionen unterliegen. Die Nummerierung der Versatzzeitenmatrix ist frei und kann Lücken aufweisen.

Versatzzeitenmatrix (1:667)

Versatzzeitenmatrix			
METHOD	**Name**		**Beschreibung**
0	**Get**		
	Ausgabeparameter		
	RetCode		OK: folgende Parameter richtig gelesen
	VDArt		Zuordnung zum Versorgungsblock.
	Bezeichnung: bezeichnungType		Name der Versatzzeitenmatrix, (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	Art		Versatzbedingungen zwischen- Beginn → Beginn- Ende → Ende- Beginn → Ende.
	Anzahl		Anzahl folgender VersatzZeitEintrag Elemente
	VersatzZeitEintrag [0...9999]		Eintrag in der Versatzzeitmatrix. Der Eintrag bezieht sich immer auf das Verhältnis der Abhängigen Signalgruppe zur Basissignalgruppe. Beispiel: Bei Art = "BeginnBeginn" SGrBasis = "Bas", SGrAbhaengig = "Abh", Wert = 12 und Operator = "gleich" muss der Beginn der Freigabezeit der Signalgruppe "Abh" 12 Sekunden hinter der Signalgruppe "Bas" liegen.
		SGrBasis.Nr	Nummer der Basis-Signalgruppe
		SgrAbhaengig.Nr	Nummer der abhängigen Signalgruppe
		Wert	Zeitwert (kann negativ sein)
		Operator	

Art (VersatzzeitArt):

Wert	Bedeutung	Name
1	diese Art kennzeichnet Versatzzeiten von Beginn zu Beginn	BeginnBeginn
2	diese Art kennzeichnet Versatzzeiten von Ende zu Ende	EndeEnde
3	diese Art kennzeichnet Versatzzeiten von Beginn zu Ende	BeginnEnde

Hinweis: Es ist nicht verbindlich festgelegt, dass ein OCIT-O Lichtsignalsteuergerät alle aufgeführten Versatzzeitenarten beherrschen muss.

Operator (VersatzzeitOperator):

Wert	Bedeutung	Name
1	größergleich (der Zeitwert der abhängigen Signalgruppe muss größer oder gleich dem Zeitwert der maßgebenden Signalgruppe (Basissignalgruppe) sein)	ge
2	kleinergleich (der Zeitwert der abhängigen Signalgruppe muss kleiner oder gleich dem Zeitwert der maßgebenden Signalgruppe (Basissignalgruppe) sein)	le
3	gleich (der Zeitwert der abhängigen Signalgruppe muss gleich dem Zeitwert der maßgebenden Signalgruppe (Basissignalgruppe) sein)	eq

Es gilt die mathematische Sichtweise z. B. -6 le -5.

3.3.2.1.6 Beschreibung der Versatzvarianten

Im Folgenden sind an Beispielen die möglichen Versatzvarianten dargestellt. OCIT definiert bei Versätzen die Begriffe Basis-Signalgruppe und abhängige Signalgruppe, wobei die Basis-Signalgruppe der maßgebenden Signalgruppe entspricht.

Anmerkung: Bei der Verwendung von Versätzen sind immer die Fähigkeiten des Zielgeräts zu berücksichtigen, d.h. es kann nicht davon ausgegangen werden, dass jedes OCIT-Gerät die verschiedenen Versatzvarianten auch vollständig unterstützt. Insbesondere die in der Theorie möglichen LE – Versätze sind zwar überwachbar, es kann jedoch nicht garantiert werden, dass der Versatz wie gewünscht geschalten wird, da hier die abhängige Signalgruppe bereits zu einem Zeitpunkt schaltet, wo noch nicht bekannt ist ob die maßgebende überhaupt passend einen Schaltwunsch in der Zukunft erhalten wird. Insofern wird diese Versatzart üblicherweise nur zur Generierung der Signalpläne während der Planungsphase und nicht zur Überwachung im Gerät benutzt.

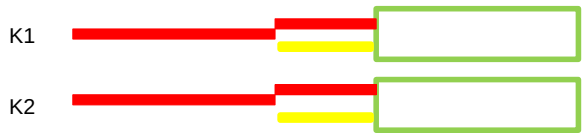
Auch sind die beschriebenen Reaktionen bei Versatzverletzungen nur als Beispiele zu verstehen, da auch hier das Geräteverhalten stark unterschiedlich sein kann. Aus OCIT-Sicht wird deshalb nur festgelegt, dass ein Versatzverstoß über eine entsprechende Meldung dokumentiert werden soll.

3.3.2.1.7 Feste Versätze

Feste Versätze werden bei OCIT mit dem ‚eq' - Operator (gleich) versorgt. Hier machen nur Versatzwerte >= 0 Sinn, da im Falle eines negativen Versatzes durch Austausch der maßgebenden und abhängigen Signalgruppe der Wert in den entsprechenden positiven Wert verwandelt werden kann. Außerdem muss die maßgebende Signalgruppe immer als erstes Schalten, damit die Abhängigkeit überwachbar und ggf. beeinflussbar wird.

Fester Beginn-Versatz (Operator ‚eq')

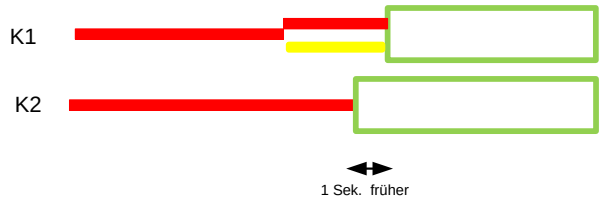
Freigabebeginn gleichzeitig



K1 und K2 schalten zum gleichen Zeitpunkt auf Freigabe.

Praktischer Anwendungsfall: An einem Fußgängerübergang ohne Mittelinsel müssen die beiden feindlichen Fahrzeugströme gleichzeitig schalten.

Freigabebeginn mit festem Versatz



K2 schaltet genau 1 Sekunde vor K1 auf Freigabe.

Bei festen Versätzen sind beide Signalgruppen gegenseitig voneinander abhängig, d.h. es ist im Prinzip keine Zuordnung zwischen maßgebender und abhängiger Signalgruppe möglich. Die Signalgruppen müssen gemäß dem festgelegten Versatz schalten. Um dies überwachen und ggf. beeinflussen zu können, muss die maßgebende Signalgruppe zuerst schalten.

Praktischer Anwendungsfall: Hessenerlass, der Fußgänger muss bei bedingt feindlichen Verkehrsströmen die Konfliktfläche 1 Sekunde vor dem Fahrzeugstrom erreichen.

Festlegung für die eindeutige Umsetzung der Kodierung:

Bei Equal-Versätzen (Operator ‚Eq') wird aus OCIT Sicht immer nur ein Wertepaar generiert, da sich das umgekehrte Wertepaar implizit ergibt. Hierbei wird festgelegt, dass in diesen Fällen immer mit positiven Versatzwerten gearbeitet wird und demzufolge immer die Signalgruppe, welche zuerst den Freigabezustand annimmt, als maßgebende (Basis-Signalgruppe) festgelegt wird. Im Beispiel würde das die Signalgruppe K2 sein (OCIT-Kodierung: K2, K1, 1, eq).

Reaktion bei Zeitverletzung:

- Ansteuerung Freigabebeginn von K1 vor K2
  -> Freigabebeginn von K2 wird unterdrückt, Freigabe von K1 erfolgt wie angesteuert
- Ansteuerung Freigabebeginn von K1 gleichzeitig mit K2
  -> Freigabebeginn von K1 wird geschoben, bis der Versatz von 1 Sek zum Freigabebeginn von K2 eingehalten ist.
- Ansteuerung Freigabebeginn von K1 später als 1 Sek. nach K2
  -> Freigabe von K1 wird unterdrückt.

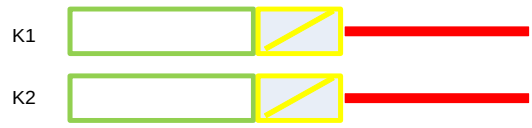
Sonderfall, Signalgruppen kommen gleichzeitig' (0-Versatz)
Für diesen Fall wird festgelegt, dass immer die Signalgruppe mit dem niedrigeren Index die maßgebende (Basis-Signalgruppe) ist, in diesem Fall also K1. (OCIT-Kodierung: K1, K2, 0, eq)

Reaktion bei Zeitverletzung:

Unterschiedliche Ansteuerung des Freigabebeginn
-> die Signalgruppe, mit dem späteren Beginn erhält keine Freigabe.

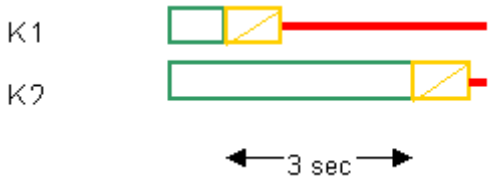
Fester Ende-Versatz (Operator ‚eq')

Freigabeende gleichzeitig



K1 und K2 schalten zum gleichen Zeitpunkt auf Freigabeende.

Freigabeende mit festem Versatz



K2 schaltet genau 3 Sekunden nach K1 auf Freigabeende.

Auch hier sind beide Signalgruppen gegenseitig voneinander abhängig, keine Zuordnung maßgebende/abhängige ist möglich.

Festlegung für die eindeutige Umsetzung der Kodierung:

Bei Equal-Versätzen (Operator ‚Eq') wird aus OCIT Sicht immer nur ein Wertepaar generiert, da sich das umgekehrte Wertepaar implizit ergibt. Hierbei wird festgelegt, dass in diesen Fällen immer mit positiven Versatzwerten gearbeitet wird und demzufolge immer die Signalgruppe, welche zuerst den Sperrzustand annimmt, als maßge-

bende (Basis-Signalgruppe) festgelegt wird. Im Beispiel würde das die Signalgruppe K1 sein (OCIT-Kodierung: K1, K2, 3, eq).

Reaktion bei Zeitverletzung:

Ansteuerung des Freigabeende von K2 erfolgt zu früh oder zu spät gegenüber dem Ende von K1
-> das Freigabeende von K2 wird geschoben bzw. verkürzt, damit die Versatzzeit eingehalten ist.

Sonderfall, Signalgruppen sperren gleichzeitig' (0-Versatz)
Für diesen Fall wird festgelegt, das immer die Signalgruppe mit dem niedrigeren Index die maßgebende (Basis-Signalgruppe) ist, in diesem Fall also K1. (OCIT-Kodierung: K1, K2, 0, eq)

Reaktion bei Zeitverletzung:

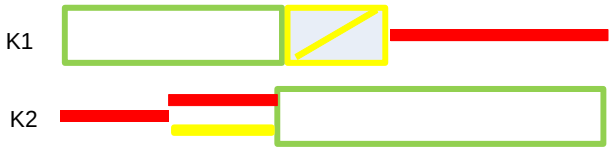
unterschiedlich angesteuertes Freigabeende
-> In diesem Fall gibt es zwei zulässige Varianten.

a) Die Freigabe der Signalgruppe mit dem früher angesteuerten Freigabeende wird verlängert bis zum Freigabeende der anderen Signalgruppe.

b) Die zweite Signalgruppe wird ebenfalls sofort mit abgebrochen, sofern deren Mindestzeit bereits abgelaufen ist.

Fester Beginn-Ende-Versatz (Operator ‚eq')

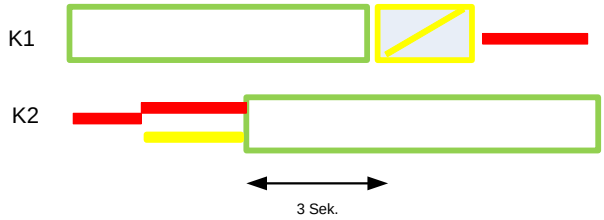
Freigabebeginn und Freigabeende gleichzeitig



K1 endet und K2 beginnt zum gleichen Zeitpunkt.

Fester Ende-Versatz

Freigabebeginn und Freigabeende mit festem Versatz



K2 beginnt genau 3 Sekunden bevor K1 endet.

Festlegung für die eindeutige Umsetzung der Kodierung:

Bei festen Beginn-Ende-Versätzen (Operator ‚eq') wird aus OCIT Sicht immer nur ein Wertepaar generiert. Hierbei wird festgelegt, dass die erste Signalgruppe immer die

beginnende (Freigabebeginn) und damit die Maßgebende ist und die zweite die endende Signalgruppe (Freigabeende) und damit die abhängige ist. Im Beispiel würde das die Signalgruppe K2 sein (OCIT-Kodierung: K2, K1, -3, eq).

Reaktion bei Zeitverletzung:

Fall 1: K1 will bereits früher abbrechen, d.h. K2 hat noch keine 3 Sekunden Freigabebeginn.
-> das Freigabeende von K1 wird geschoben damit die Versatzzeit eingehalten ist.

Fall 2: K1 will noch nicht abbrechen, obwohl K2 schon 3 Sekunden Freigabebeginn hat.
-> das Freigabeende von K1 wird eingeleitet damit die Versatzzeit eingehalten ist.

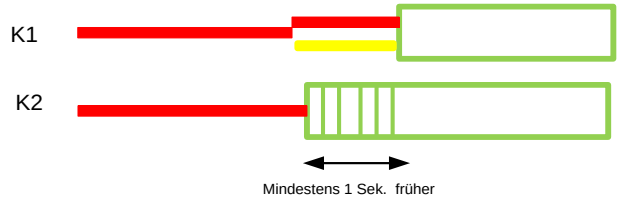
3.3.2.1.8 Variable Versätze

Variable Versätze werden bei OCIT mit den ‚ge' - (größer gleich) oder ‚le' - Operator (kleiner gleich) versorgt.

Variabler Beginn-Versatz (Operator ‚ge')

‚Greater equal' – Beginn Versätze bedeuten, dass die abhängige Signalgruppe frühestens nach dem angegebenen Versatzwert starten darf.

variable Freigabe von abhängiger Sg nach Freigabe der maßgebenden Sg

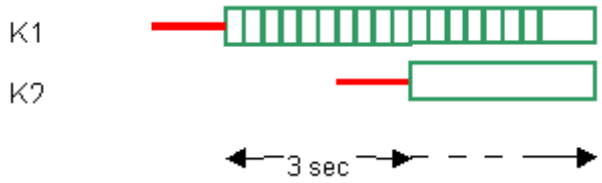


K2 schaltet mindestens 1 Sekunde oder mehr vor K1.

Hier ist K2 maßgebend, K1 abhängig. Dies wird bei OCIT mit dem ‚ge' - Operator (größer gleich) versorgt.

Praktischer Anwendungsfall: Hessenerlass, der Fußgänger muss bei bedingt feindlichen Verkehrsströmen die Konfliktfläche mindestens 1 Sekunde vor dem Fahrzeugstrom erreichen.

variable Freigabe von abhängiger Sg vor maßgebender Sg (negative Werte)



Hier ist K2 maßgebend obwohl sie erst später startet, K1 abhängig, d.h. K1 darf maximal 3 Sekunden vor K2 kommen.

Festlegung für die eindeutige Umsetzung der Kodierung:

Bei größer- gleich-Versätzen (Opertator ‚ge') wird aus OCIT Sicht immer nur ein Wertepaar generiert. Hierbei wird festgelegt, dass bei positiven Werten immer die Signalgruppe, welche zuerst den Freigabezustand annimmt als maßgebende (Basis-Signalgruppe) festgelegt wird und bei negativen Werten die maßgebende erst nach der abhängigen startet.

Reaktion bei Zeitverletzung:

Positive Werte:

- Ansteuerung der Freigabe von K2 vor K1
  -> K2 geht auf Freigabe, K1 wird unterdrückt.
- Ansteuerung der Freigabe von K2 mit K1 gleichzeitig oder innerhalb des Versatz von 3 Sek.
  -> K2 wird geschoben, bis der Versatz von 3 Sek eingehalten werden kann.

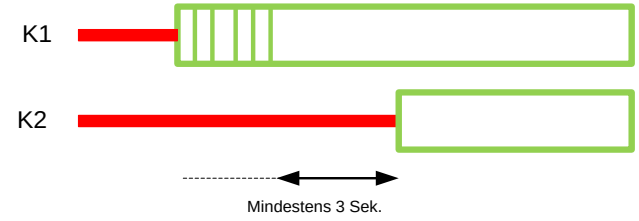
Nagative Werte:

- Ansteuerung der Freigabe von K2 später als 3 Sekunden nach K1
  -> K2 wird unterdrückt. Wenn K2 bereits Freigabe hat, dann kann K1 jederzeit dazu kommen.

Variabler Beginn-Versatz (Operator ‚le')

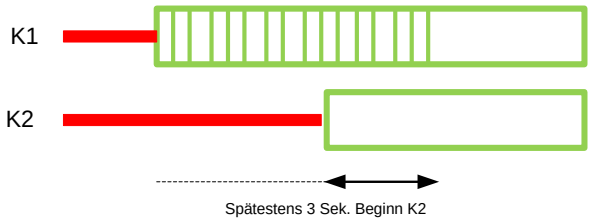
‚Less equal' – Versätze sind wenig praxisrelevant und werden normalerweise nur zur automatischen Signalplangenerierung in der Planungsphase benutzt. Deshalb wird diese Versatzart auch meist von den Steuergeräten nicht unterstützt. Die folgenden Beispiele sollen deshalb nur den grundsätzlichen Gedanken der OCIT-Definition dokumentieren. LE-Versätze besagen, dass die abhängige Signalgruppe spätestens bis zur angegebenen Versatzzeit gestartet sein muss, ansonsten muss sie unterdrückt werden. Bei negativen Werten tritt hier eine Umkehrung der Abhängigkeit ein, d.h. wenn die maßgebende Signalgruppe nicht spätestens bis zum angegeben Wert gestartet wird, dann wird sie unterdrückt.

variable Freigabe von abhängiger Sg vor maßgebender Sg (negative Werte)



Hier ist K2 maßgebend, K1 abhängig, d.h. K1 muss mindestens um 3 Sekunden vor K2 kommen. Dies wird bei OCIT mit dem ‚le' - Operator (kleiner gleich) versorgt.

variable Freigabe von abhängiger Sg vor maßgebender Sg (positive Werte)



Hier ist K2 maßgebend, K1 abhängig, d.h. K1 darf spätestens bis 3 Sekunden nach K2 kommen. Dies wird bei OCIT mit dem ‚le' - Operator (kleiner gleich) versorgt.

Festlegung für die eindeutige Umsetzung der Kodierung:

Bei kleiner-gleich-Versätzen (Opertator ‚le') wird aus OCIT Sicht immer nur ein Wertepaar generiert. Im Beispiel würde die Signalgruppe K2 die maßgebende (Basissignalgruppe) sein und K1 muss den angegebenen Versatzwert einhalten. (OCIT-Kodierung: K2, K1, 3, le).

Reaktion bei Zeitverletzung:

Positive Werte:

- Ansteuerung der Freigabe von K1 um mehr als 3 Sek. nach K2
  -> K1 wird unterdrückt.

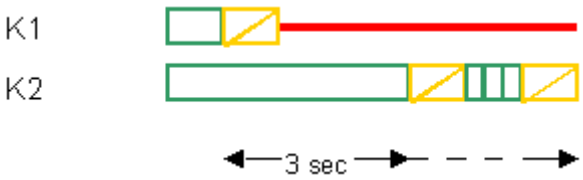
Negative Werte:

- Ansteuerung der Freigabe von K2 um weniger als 3 Sek. nach K1
  -> K2 wird verzögert bis die 3 Sekunden eingehalten sind.

Variabler Ende-Versatz (Operator ‚ge')

Bei ‚greater equal' – Ende Versätzen machen nur Versatzwerte >= 0 Sinn, da ja die abhängige Signalgruppe frühestens nach dem angegebenen Versatzwert enden darf.

variables Freigabeende der abhängigen Sg nach Freigabeende der maßgebenden Sg



K2 schaltet (endet) 3 oder mehr Sekunden nach K1.

Hier ist K1 maßgebend, K2 abhängig. Dies wird bei OCIT mit dem ‚ge' - Operator (größer gleich) versorgt.

Festlegung für die eindeutige Umsetzung der Kodierung:

Bei größer-gleich-Versätzen (Opertator ‚ge') wird aus OCIT Sicht immer nur ein Wertepaar generiert. Hierbei wird festgelegt, dass in diesen Fällen immer die Signalgruppe, welche zuerst den Sperrzustand annimmt als maßgebende (Basis-Signalgruppe) festgelegt wird und der angegebene Versatzwert größer gleich 0 sein muss. Im Bei-

spiel würde das die Signalgruppe K1 sein (OCIT-Kodierung: K1, K2, 3, ge). K2 dürfte dann frühestens 3 Sekunden nach dem Ende von K1 in Sperren wechseln.

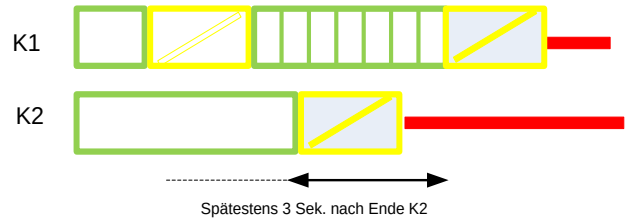
Reaktion bei Zeitverletzung:

- Ansteuerung des Freigabeende von K2 vor dem parametrierten Versatz zu K1
  -> K2 wird geschoben bis der Versatz des Freigabeende zu K1 eingehalten wird.

Variabler Ende-Versatz (Operator ‚le')

Bei ‚less equal' – Versätzen endet die abhängige Signalgruppe spätestens um den angegeben Versatzwert nach der maßgebenden, sie darf aber auch schon früher enden. D.h. abhängige Signalgruppe muss dann spätestens nach dem angegebenen Versatzwert zwangsweise beendet werden. Auch diese Versatzart ist wenig praxisrelevant und wird, wenn überhaupt, nur zur automatischen Signalplangenerierung während der Planungsphase verwendet.

variables Freigabeende der abhängigen Signalgruppe



K1 schaltet weniger als 3 Sekunden nach K2, d.h. K1 müsste spätestens 3 Sekunden nach dem Sperrbefehl für K2 abgebrochen werden.

Hier ist K2 maßgebend, K1 abhängig. Dies wird bei OCIT mit dem ‚le' - Operator (kleiner gleich) versorgt.

Festlegung für die eindeutige Umsetzung der Kodierung:

Bei Kleiner-gleich-Versätzen (Opertator ‚le') wird aus OCIT Sicht immer nur ein Wertepaar generiert. Im Beispiel würde die Signalgruppe K2 die maßgebende (Basissignalgruppe) sein. (OCIT-Kodierung: K2, K1, 3, le).

Reaktion bei Zeitverletzung:

Positive Werte:

- Ansteuerung des Freigabeende von K1 um mehr als 3 Sek. nach K2
  -> K1 wird zwangsweise abgebrochen.

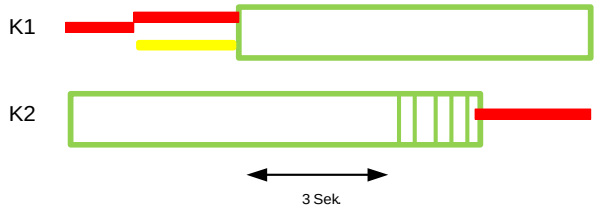
Negative Werte:

- Ansteuerung des Freigabeende von K2 erfolgt vor dem parametrierten Versatz zu K1
  -> Freigabeende von K2 wird verzögert um den Versatz zum Freigabeende zu K1 einzuhalten.

Variabler Beginn - Ende-Versatz (Operator ‚ge')

Variable Beginn – Ende Versätze werden nur in seltenen Ausnahmefällen während der Planung zur Unterstützung der Signalplangenerierung benutzt. Deshalb haben sie in der Praxis für eine Überwachung im Gerät wenig Bedeutung und werden von den meisten Geräten nicht unterstützt.

variables Freigabeende der abhängigen Sg nach Freigabebeginn der maßgebenden Sg



K2 schaltet 3 oder mehr Sekunden nach dem Freigabebeginn von K1 auf Freigabeende.

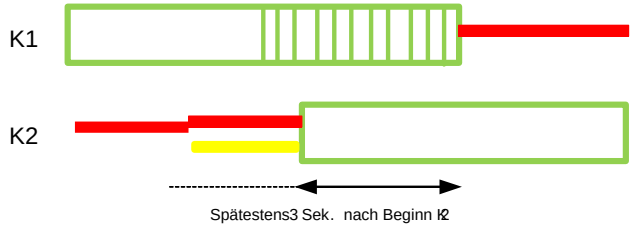
Hier ist K1 mit seinem Freigabebeginn maßgebend, K2 abhängig. Dies wird bei OCIT mit dem ‚ge' - Operator (größer gleich) versorgt.

<u>Reaktion bei Zeitverletzung:</u>

- Ansteuerung des Freigabeende von K2 vor dem parametrierten Versatz zu K1
  -> K2 wird geschoben bis der Versatz des Freigabeende zu K1 eingehalten wird.

Variabler Beginn - Ende-Versatz (Operator ‚le')

variables Freigabeende der abhängigen Sg vor Beginn der maßgebenden Sg



K1 schaltet sein Freigabeende spätestens 3 Sekunden nach dem Freigabebeginn K2, d.h. K1 müsste spätestens 3 Sekunden nach dem Freigabebefehl für K2 abgebrochen werden.

Hier ist K2 als beginnende Sg maßgebend, K1 abhängig. Dies wird bei OCIT mit dem ‚le' - Operator (kleiner gleich) versorgt. (OCIT-Kodierung: K2, K1, 3, le)

Reaktion bei Zeitverletzung:

- Ansteuerung des Freigabeende von K1 erfolgt nach dem parametrierten Versatz zu K2
  -> Freigabeende von K1 erfolgt vorzeitig um den Versatz zum Freigabebeginn zu K2 einzuhalten.
- Bei negativen Werten:
  Ansteuerung des Freigabeendes von K1 muss dann bereits vor Freigabebeginn K2 erfolgt sein. Ist dies nicht der Fall, muss K1 zwangsweise beendet werden und der Freigabebeginn von K2 muss ggf. verzögert werden, um den Versatz einzuhalten.

Allgemeine Anmerkungen:

Stellt sich bei Betrachtung der Versätze heraus, dass Versatzzeiten nicht eingehalten werden können, wird der Schaltvorgang entsprechend verschoben bzw. ein Freigabeschaltwunsch wird ggf. nicht mehr bedient. Die Reaktionen der Steuergeräte sind hier sehr unterschiedlich und deshalb wird hier auch kein Sollverhalten explizit festgelegt, d.h. die angegebenen Lösungen dienen nur als Beispiel, wie ein Verhalten aussehen könnte.

Die Einhaltung von Versatzbedingungen ist gegenüber der Einhaltung von Zwischen- und Mindestzeiten nachrangig.

Die maßgebende Signalgruppe muss in der Regel vor der abhängigen schalten, denn nur so kann sichergestellt werden, dass die maßgebende nicht beeinflusst (z.B. unterdrückt) werden muss.

Bei Beginn – Beginn Versätzen kommen die Versatzbedingungen erst ab dem Zeitpunkt zum Tragen, wo beide beteiligten Signalgruppen den Sperrzustand angenommen haben.

Bei Ende – Ende Versätzen kommen die Versatzbedingungen erst ab dem Zeitpunkt zum Tragen, wo beide beteiligten Signalgruppen den Freigabezustand angenommen haben.

Beginn – Ende Versätze wurden nur der Vollständigkeit halber aufgenommen, da sie strukturell in OCIT-O definiert sind. Es gibt jedoch keine bekannten praxisrelevanten Anwendungen und demzufolge werden sie derzeit von keinem Steuergerät unterstützt.

3.3.2.1.9 Objekt VTZwischenzeitenmatrix

In diesem Objekt werden die verkehrstechnischen Zwischenzeitenmatrizen für besondere Signalprogramme (z.B. Schlechtwetter) gespeichert. Verkehrstechnische Zwischenzeitenmatrizen haben die Nummern 1...3. Alle Zeitwerte der VTZwischenzeitenmatrix müssen grösser oder gleich den Zeitwerten der sicherheitsrelevanten Zwischenzeitenmatrix sein. Es kann jeweils nur eine VTZwischenzeitenmatrix aktiv sein.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

- ZwischenZeitEintrag aufsteigend sortiert nach Räumer, dann nach Einfahrer.

VTZwischenzeitmatrix (1:668)

VTZwischenzeitmatrix			
METHOD	Name		Beschreibung
0	**Get**		
	Ausgabeparameter		
	RetCode		OK: folgende Parameter richtig gelesen
	VDArt		Zuordnung zum Versorgungsblock.
	Bezeichnung: bezeich-nungType		Signalprogrammname (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	Anzahl		Anzahl folgender Zwischenzeiten
	ZwischenZeitEintrag [0...9999]		Eintrag in der VTZwischenzeitenmatrix
		Raeumer.Nr	Nummer der Räumenden Signalgruppe
		Einfahrer.Nr	Nummer der Einfahrenden Signalgruppe
		Wert	Zeitwert

3.3.2.1.9.1 Objekt VTMinFreigabe

In diesem Objekt werden die verkehrstechnischen Mindestfreigabezeiten für besondere Signalprogramme (z.B. Schlechtwetter) gespeichert. Die Listen der verkehrstechnischen Mindestfreigabezeiten haben die Nummern 1...3. Alle Zeitwerte der VTMinFreigabe müssen grösser oder gleich den Zeitwerten der sicherheitsrelevanten Mindestfreigabezeiten sein. Es kann jeweils nur eine Mindestfreigabezeitenliste aktiv sein.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

- Zeiten aufsteigend sortiert nach Signalgruppennummer.

VTMinFreigabe (1:673)

VTMinFreigabe		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.

VTMinFreigabe			
METHOD	**Name**		**Beschreibung**
	Bezeichnung: bezeichnungType		Name der Mindesfreigabezeitliste, (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	Anzahl		Anzahl folgender Mindestzeiten
	Zeit [0...254]		
		SG.Nr	Nummer der Signalgruppe
		Wert	Zeitwert

3.3.2.1.9.2 Objekt VTMinGesperrt

In diesem Objekt werden die verkehrstechnischen Mindestsperrzeiten für besondere Signalprogramme (z.B. Schlechtwetter) gespeichert. Die Listen der verkehrstechnischen Mindestsperrzeiten haben die Nummern 1...3. Alle Zeitwerte der VTMinGesperrt müssen grösser oder gleich den Zeitwerten der sicherheitsrelevanten Mindestsperrzeiten sein. Es kann jeweils nur eine Mindestsperrzeitenliste aktiv sein.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

- Zeiten aufsteigend sortiert nach Signalgruppennummer.

VTMinGesperrt (1:675)

VTMinGesperrt			
METHOD	**Name**		**Beschreibung**
0	**Get**		
	Ausgabeparameter		
	RetCode		OK: folgende Parameter richtig gelesen
	VDArt		Zuordnung zum Versorgungsblock.
	Bezeichnung: bezeichnungType		Name der Mindesgesperrtzeitliste, (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	Anzahl		Anzahl folgender Mindestzeiten
	Zeit [0..254]		
		SG.Nr	Nummer der Signalgruppe
		Wert	Zeitwert

3.3.3 Block 2: Daten mit Netzbezug

3.3.3.1 Objekt Kopfdaten

Dieses Objekt speichert die Grunddaten des Lichtsignalsteuergeräts. Diese Daten haben nur informativen Charakter für Bediener und werden nicht in die Ablaufsteuerung des Lichtsignalsteuergeräts übernommen.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

- Wie in der folgenden Methodenbeschreibung eingetragen.

Kopfdaten (1:650)

Kopfdaten		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.
	Kurzbezeichnung: kurzStringType	Kurzbezeichnung der Anlage z.B. K123
	Name: langStringType	Langname des Knotens z.B. Meierstraße / Müllerstraße
	UnitID	Logische Adressierung des Knotenpunktes aus der Sicht OCIT-C
	SystemNr	Amts- bzw. Bezirkskennung
	SubSystemNr.	Stadt, falls nicht gesetzt, mit NULLVALUE belegt
	UnitNr	Eindeutige Nummer des Knotenpunktes im Steuerungsgebiet (1 bis 4294967295)
	Bemerkung: langStringType	kundenspezifische Bemerkungen

3.3.3.2 Schaltuhr

Die Objekte Tagesplan, Wochenplan, SondertagJaehrlich, SondertagAufzaehlung und Zeitbereich beinhalten die Daten der Schaltuhr (Jahresuhr, Jahresautomatik, JAUT) des Lichtsignalsteuergeräts.

3.3.3.2.1 Projektspezifische Modifikationen der Schaltuhr

Insgesamt sind 16 Modifikationen verfügbar. 3 Modifikationen (ModVA, ModOepnv, ModVAIndividualverkehrEinAus) sind bereits belegt, 13 Modifikationen sind für die

projektspezifische Nutzung über die Schaltuhr frei. Diese projektspezifischen Modifikationen haben jeweils eine Nummer, die von 0 bis 254 laufen kann. Für 8 davon werden Empfehlungen zur Belegung gegeben. Sie umfassen die Nummer und den Namen der Modifikation. Jede projektspezifische Modifikation kann Ein oder Aus geschaltet werden. Der Name der projektspezifischen Modifikation ist versorgbar (Herstellerversorgung) und auslesbar.

Da die Modifikationen nicht nur von der Schaltuhr, sondern auch von der Zentrale aus geschaltet werden können, wird festgelegt, dass die Zentrale Priorität hat.

Hinweis: Die mit den Modifikationen geschalteten Anwendungen sind nicht standardisiert und müssen projektspezifisch vereinbart werden!

Projektspezifische Modifikationen der Schaltuhr (Empfehlungen zur Belegung)		
Nr.	**Name**	**Bemerkung**
0	Blindenakustik Orientierungston	
1	Blindenakustik Freigabeton	
2	Detektorüberwachung	Falls mehrere Überwachungszeiten, z.B. Morgenspitze, Nachmittagsspitze, Normalverkehr, Schwachverkehr notwendig sind, müssen dazu noch freie projektspezifische Modifikationen benutzt werden.
3	Digitaler Ausgang A	Angabe der Kanalnummern und DigAusgang: Get wird derzeit nicht unterstützt.
4	Digitaler Ausgang B	
5	Digitaler Ausgang C	
6	Digitaler Ausgang D	
7	LSA-Standby	Zum Beispiel Anlage Dunkel oder alles Rot.

3.3.3.2.2 Objekt Tagesplan

In diesem Objekt werden die Daten der Tagespläne gespeichert.

Es ist mindestens ein Standard-Tagesplan vorhanden, der immer dann ausgeführt wird, wenn keine andere Regelung greift. Der Standard-Tagesplan hat die Nr 1.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden. Wie in der folgenden Methodenbeschreibung eingetragen:

- Befehle mit aufsteigender Uhrzeit.
- Teilknoten Elemente sortiert aufsteigend nach Teilknotennummer

- Projektspezifische Modifikation Elemente sortiert nach Nummer der Modifikation

Tagesplan (1:660)

Tagesplan			
METHOD	**Name**		**Beschreibung**
0	**Get**		
	Ausgabeparameter		
	RetCode		OK: folgende Parameter richtig gelesen
	VDArt		Zuordnung zum Versorgungsblock.
	BezeichnungKurz: bezeichnungType		Bezeichnung des Tagesplans. Die Bezeichnung innerhalb der Tagespläne muss eindeutig sein
	Anzahl		Anzahl folgender Jaut-Befehl Elemente
	Befehl[0 … 254] Die Befehle müssen mit aufsteigender Uhrzeit sortiert sein		
		Uhrzeit	Uhrzeit, an dem der Befehl begonnen wird. Zu einem Zeitpunkt kann nur ein Befehl begonnen werden, da keine Nullvalues zugelassen sind. Als Uhrzeit werden die Sekunden seit Mitternacht lokaler Zeitangegeben. Bei Winter-Sommerzeitumstellung wird der letzte übersprungene Schaltzeitpunkt nachgeholt.
		Programmwunsch	Programmwunsch
		KnotenEinAus	Gesamtknoten Ein/Aus
		ModVA	Verkehrsabhängigkeit Ein/Aus
		ModOepnv	OEPNV Bevorzugung
		ModVAIndividualverkehrEinAus	Dieser Teilbefehl schaltet die Verkehrsabhängige Beeinflussung des Signalprogramms durch Individualverkehr Ein/Aus.
		Anzahl	Anzahl folgender Teilknotenzustände. Die Anzahl der TK muss mit den tatsächlich vorhandenen übereinstimmen!
		TkZustand[0…3]	
			TeilKnotenNr. Teilknotennummer
			SollZustand Teilknoten Ein/Aus
		Modifikation[0 … 12]	Projektspezifische Modifikationen (alle 13)
			Nr.ui1 Nummer der Modifikation
			Wert Wert der Modifikation. Nicht genutzte Modifikationen werden mit Wert „Aus“ belegt!

Hinweis: Jeder Tagesplan muss mindestens einen Eintrag haben. Der Schaltwunsch muss vollständig sein, NULLVALUES und Zustand „keiner“ sind generell nicht erlaubt!

3.3.3.2.3 Objekt Wochenplan

In diesem Objekt werden die Daten der Wochenpläne gespeichert.

Es ist mindestens ein Standard-Wochenplan vorhanden, der immer dann ausgeführt wird, wenn keine andere Regelung greift. Der Standard Wochenplan hat die Nr 1.

Festlegung zur Reihenfolge in der die Versorgungsdaten werden sortiert und gelesen werden:

- Wie in der folgenden Methodenbeschreibung eingetragen.

Wochenplan (1:661)

Wochenplan		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.
	BezeichnungKurz: bezeichnungType	Bezeichnung des Wochenplans. Die Bezeichnung innerhalb der Wochenpläne muss eindeutig sein. (ist nur für Rücklesen der Versorgung in OTEC Format nötig)
	Mo	Nummer des Tagesplans, der am Montag ausgeführt wird
	Di	Nummer des Tagesplans, der am Dienstag ausgeführt wird
	Mi	Nummer des Tagesplans, der am Mittwoch ausgeführt wird
	Do	Nummer des Tagesplans, der am Donnerstag ausgeführt wird
	Fr	Nummer des Tagesplans, der am Freitag ausgeführt wird
	Sa	Nummer des Tagesplans, der am Samstag ausgeführt wird
	So	Nummer des Tagesplans, der am Sonntag ausgeführt wird

Hinweis: NULLVALUES sind nicht erlaubt!

3.3.3.2.4 Objekt SondertagJaehrlich

In diesem Objekt werden die Daten aller jährlich wiederkehrenden Feier- oder Sondertage gespeichert.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

- Wie in der folgenden Methodenbeschreibung eingetragen.

Festlegung zur Priorität:

Größerer Zahlen bedeuten höhere Gewichtung (Priorität):

0 Standardgewichtung von normalen Tagen
1 Standardgewichtung von Zeitbereichstagen (Ferien)
2 Standardgewichtung von SondertagJaehrlich (Feiertagen)
3 Standardgewichtung von SondertagAufzaehlung
4…9 Hochpriore Sondertage für Spezialfälle

Bei gleichen Prioritäten gilt die Reihenfolge Zeitbereich, SondertagJaehrlich und Sondertag Aufzaehlung (höchste Priorität).

SondertagJaehrlich (1:662)

SondertagJaehrlich		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.
	Name: bezeichnungType	Name des Feiertags (ist nur für Rücklesen der Versorgung in OTEC Format nötig).
	Tagesplan	Referenz auf den Tagesplan, der an diesem Tag verwendet wird.
	Prioritaet	Wertebereich 1-9, da 0 für normale Tage reserviert (siehe oben)
	Datum	Tagescodierung siehe Feiertag. Achtung! Im Fall von Sondertagen werden hier auch nicht im Enum definierte Werte eingetragen.

Die beweglichen und festen Feiertage werden im Tagescode wie folgt codiert:

Feiertage mit festem Datum:

Feiertage mit festem Datum werden über den Zahlenbereich 0-365 abgedeckt.

Das Datum des Feiertags wird als Tag im Jahr abgelegt (Beginnend ab 0 für 1.1.).

Bei der Bestimmung des Tags wird immer ein Schaltjahr angenommen (Februar=29 Tage, Jahr = 366 Tage).

Osterabhängige Feiertage:

Bei Feiertagen, die vom Ostersonntag abhängig sind, wird zu der Differenz zum Ostersonntag der Wert 500 addiert.

Wenig schwankende Feiertage am gleichen Wochentag:

Wenig schwankende Feiertage, die nicht von Ostern abhängig sind, sind Feiertage, die immer am selben Wochentag stattfinden.

Es wird der erste Tag angegeben, an dem der Feiertag stattfinden kann (beginnend ab 0 für 1.1.). Danach wird je nach Wochentag ein Offset aufaddiert:

Mo = 1000
Di = 2000
Mi = 3000
Do = 4000
Fr = 5000
Sa = 6000
So = 7000

Bei der Bestimmung des Tags wird immer ein Schaltjahr angenommen (Februar = 29 Tage, Jahr = 366 Tage).

Beispiele für Feiertage:

Name	**Berechnung**	**Wert**
Neujahr	fest 1.1.	0
Epiphanias	fest 6.1.	5
Tag der Arbeit	fest 1.5.	121
Tag der deutschen Einheit	fest 3.10.	276
Reformationsfest	fest 31.10.	304
Allerheiligen	fest 1.11.	305
Weihnachtsfeiertag1	fest 25.12.	359
Weihnachtsfeiertag2	fest 26.12.	360
Fastnacht	OSTERN-47	453
Karfreitag	OSTERN-2	498
Ostersonntag	OSTERN	500
Ostermontag	OSTERN+1	501
Muttertag	Sonntag im Bereich 8.-14.Mai	7128
Christi Himmelfahrt	OSTERN+39	539
Pfingstsonntag	OSTERN+49	549

Pfingstmontag	OSTERN+50	550
Fronleichnam	OSTERN+60	560
Maria Himmelfahrt	fest am 15.8.	227
Buß und Bettag	Mittwoch im Bereich 16.-22. November	3320
Maria Empfängnis	fest 8.12.	342

3.3.3.2.5 Objekt SondertagAufzaehlung

In diesem Objekt werden die einmaligen Feier- oder Sondertage mit vorgegebenem Datum gespeichert.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

Wie in der folgenden Methodenbeschreibung eingetragen.

Festlegung zur Priorität:

Größerer Zahlen bedeuten höhere Gewichtung (Priorität):

0 Standardgewichtung von normalen Tagen
1 Standardgewichtung von Zeitbereichstagen (Ferien)
2 Standardgewichtung von Feiertagen
3 Standardgewichtung von Sondertagen
4..9 Hochpriore Sondertage für Spezialfälle

Bei gleichen Prioritäten gilt die Reihenfolge Zeitbereich, SondertagJaehrlich und Sondertag Aufzaehlung (höchste Priorität).

SondertagAufzaehlung (1:663)

SondertagAufzaehlung		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.
	Name: bezeichnungType	Name des Feiertags. (ist nur für Rücklesen der Versorgung in OTEC Format nötig)
	Tagesplan	Referenz auf den Tagesplan, der an diesem Tag verwendet wird.
	Prioritaet	Wertebereich 1 - 9, da 0 für normale Tage reserviert (siehe oben)
	Tag	Tag des Sondertags

SondertagAufzaehlung		
METHOD	**Name**	**Beschreibung**
	Monat	Monat des Sondertags
	Jahr	Jahr des Sondertags

3.3.3.2.6 Objekt Zeitbereich

In diesem Objekt werden die Schaltuhr-Zeitbereiche gespeichert.

Schaltuhr Zeitbereiche haben anders als die Feiertage immer einen Beginn und ein Ende.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

- Wie in der folgenden Methodenbeschreibung eingetragen.

Festlegung zur Priorität:

- Größerer Zahlen bedeuten höhere Gewichtung (Priorität):

  0 Standardgewichtung von normalen Tagen
  1 Standardgewichtung von Zeitbereichstagen (Ferien)
  2 Standardgewichtung von Feiertagen
  3 Standardgewichtung von Sondertagen
  4..9 Hochpriore Sondertage für Spezialfälle

Bei gleichen Prioritäten gilt die Reihenfolge Zeitbereich, SondertagJaehrlich und Sondertag Aufzaehlung (höchste Priorität).

Zeitbereich (1:664)

Zeitbereich		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungsblock.
	Name: bezeichnungType	Name, z.B. Sommerferien. Der Name muss nicht eindeutig sein. (Ist nur für Rücklesen der Versorgung in OTEC Format nötig). Falls in Instations nicht angegeben, hier Leerstring eintragen.
	Wochenplan	Referenz auf verwendeten Wochenplan
	Prioritaet	Wertebereich 1 - 9, Defaultwert Ferien (Zeitbereiche) = 1
	Start	

Zeitbereich			
METHOD	Name		Beschreibung
		Tag: ui1	Starttag des Zeitbereichs
		Monat: ui1	Startmonat des Zeitbereichs
		Jahr	Startjahr des Zeitbereichs. NULLVALUE (0xffff) bedeutet jährlicher Bereich
	Ende		
		Tag: ui1	Endtag des Zeitbereichs
		Monat: ui1	Endmonat des Zeitbereichs
		Jahr	Endjahr des Zeitbereichs. NULLVALUE (0xffff) bedeutet jährlicher Bereich

3.3.4 Block 3: VA-Steuerverfahren

3.3.4.1 Objekt BinaerVASteuerverfahren

In diesem Objekt werden die nicht in OCIT-O standardisierten Daten von VA-Steuerverfahren übertragen.

Das Steuergerät erkennt anhand der Kennungen „Member“, „Kennung“ und „Daten-Binaer.Typ“ (definiert vom Hersteller des Steuerverfahrens) den Typ der Daten und verarbeitet diese weiter. Die Kennungen müssen vom Lieferanten des VA-Verfahrens mitgeteilt werden. Dem Steuergerät nicht bekannte Daten werden abgelehnt und führen zu einem Versorgungsfehler.

Festlegung zur Reihenfolge in der die Daten sortiert und gelesen werden:

- DatenBinaer Element sortiert aufsteigend nach dem ASCII-Wert der Zeichen von Begin bis Ende des DatenBinaer.Typ Strings.

BinaerVASteuerverfahren (1:672)

BinaerVASteuerverfahren		
METHOD	Name	Beschreibung
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungs-block.
	Member	OCIT Memberkennung Kennung des Herstellers des Steuerverfahrens

BinaerVASteuerverfahren			
METHOD	Name		Beschreibung
	VerfahrensVersion: kurzStringType		OCIT-C VerfahrensVersion Text: Hersteller des Steuerverfahrens
	Anzahl		Anzahl folgender DatenBinaer Elemente [1 ... 254].
		DatenBinaer.Typ:bezeichnungType	Bezeichnung des herstellerspezifischen Typs. Text: Hersteller des Steuerverfahrens
		DatenBiner.Daten:WERT_BLOB	Daten Daten in binärer Darstellung. Sollte base64 Encoding erforderlich sein, ist das vom VD-Server zu erledigen!

3.3.5 Block 4: VA-Parameter

3.3.5.1 Objekt BinaerVAParameter

In diesem Objekt werden die nicht in OCIT-O standardisierten Daten von VA-Parametern übertragen.

Das Steuergerät erkennt anhand der Kennungen „Member“, „Kennung“ und „DatenBinaer.Typ“ (definiert vom Hersteller des Steuerverfahrens) den Typ der Daten und verarbeitet diese weiter. Die Kennungen müssen vom Lieferanten des VA-Verfahrens mitgeteilt werden. Dem Steuergerät nicht bekannte Daten werden abgelehnt und führen zu einem Versorgungsfehler.

Festlegung zur Reihenfolge in der die Versorgungsdaten sortiert und gelesen werden:

- DatenBinaer Element sortiert aufsteigend nach dem ASCII-Wert der Zeichen von Begin bis Ende des DatenBinaer.Typ Strings.

BinaerVAParameter (1:676)

BinaerVAParameter		
METHOD	Name	Beschreibung
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen

BinaerVAParameter			
METHOD	Name		Beschreibung
	VDArt		Zuordnung zum Versorgungs-block.
	Member		OCIT Memberkennung Kennung des Herstellers des Steuerverfahrens
	VerfahrensVersion: kurzStringType		OCIT-I VerfahrensVersion Text: Hersteller des Steuerverfahre
	Anzahl		Anzahl folgender DatenBinaer Elemente [1 ... 254].
		DatenBinaer.Typ:bezeichnungType	Bezeichnung des herstellerspezifischen Typs.
		DatenBinaer.Daten:WERT_BLOB	Daten. Daten in binärer Darstellung. Sollte base64 Encoding erforderlich sein, ist das vom VD-Server zu erledigen!

3.3.6 Block 5: MAP Topologiedaten

Der Block 5 enthält die Topologie für den Knoten. Die MAP Daten sind auf die Nutzung in Kraftfahrzeugen, für dessen Anforderungen zugeschnitten. Der Hauptanwendungsfall ist das weiterreichen der Daten an RSU's, welche diese Daten für C2X Anwendungsfälle nutzen und auch an die Fahrzeuge weiterreichen.

Map (1:677)

Map		
METHOD	Name	Beschreibung
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	VDArt	Zuordnung zum Versorgungs-block.
	Anzahl	Anzahl folgender DatenBinaer Elemente [1 ... 254].

Map			
METHOD	Name		Beschreibung
		DatenBinaer.Typ:bezeichnungType	Bezeichnung des Datenblocks MAP (für Map-Daten)
		DatenBinaer.Daten:WERT_BLOB	Daten.Xml-Daten gemäß OCIT-C Map Definition MAPEM-PDU-Descriptions.xsd

3.4 Zentrale Schaltwünsche und Statuselemente

Ein zentraler Bediener kann folgende Schaltvorgänge automatisch oder manuell veranlassen:

- Gesamtknoten ein- ausschalten
- Lokale Gesamtknoten Ein-/Ausschaltung freigeben. Der Auszustand kann sein: Aus-Dunkel oder Aus-Blinken (RiLSA und Sonderblinken).
- Zentrales Signalprogramm wählen (max. 255); lokale Signalprogrammwahl freigeben
- Teilknoten wie Gesamtknoten schalten oder ausschalten (in Auszustand).
- Lokale Schaltung der Teilknoten freigeben. Der Ist-Zustand eines Teilknotens kann ein oder aus (in Auszustand) sein. Der Hauptknoten lässt sich nicht über diesen Mechanismus schalten. Der Auszustand kann sein: Aus-Dunkel oder Aus-Blinken (RiLSA und Sonderblinken).
- Verkehrsabhängigkeit ein-, ausschalten, lokale Schaltung der Verkehrsabhängigkeit freigeben.
- Sondereingriff x ein-, ausschalten, lokalen Sondereingriff freigeben.

Für jedes Schaltobjekt ist neben dem eigentlichen Schaltwert und der Vorgangskennung ein Gültigkeitszeitraum vorgesehen, der in Form einer Start- und einer Endzeit vorgegeben wird. (Auflösung eine Sekunde).

Die **Startzeit** ermöglicht den Ausgleich unterschiedlicher Übertragungszeiten für das synchrone Schalten mehrerer Lichtsignalsteuergeräte. Schaltwünsche gelten erst mit Erreichen der Startzeit; bis dahin verbleibt der neue Wunsch in Warteposition und der alte aktuell. Ein in der Zukunft liegender Schaltwunsch überschreibt immer dem in Warteposition. Pro Schalteinheit gibt es zwei Schaltvorgänge zur Speicherung der Zentralenschaltwünsche, einen für den aktuellen und einen für den nächsten Schaltvorgang. Ein in der Zukunft liegender Schaltwunsch überschreibt immer den nächsten Schaltvorgang.

Auf Grund der möglichen Zeitdifferenzen (zulässig ist +- 500 ms) zwischen Gerät und Zentrale sollte bei Schaltwünschen, die sofort gültig werden sollen, die Startzeit ausreichend in der Vergangenheit liegen.

Die **Endzeit** ermöglicht grundsätzlich das Zurückschalten auf lokalen Betrieb zur definierten Uhrzeit ohne weitere Verbindung zur Zentrale. Jeder vom Gerät akzeptierte

Zentralenschaltwunsch bleibt im Gerät, unabhängig von etwaigen Störungen des Übertragungswegs, bis zu seiner Endzeit gültig.

Folgende Bedienvorgänge werden im Gerät als **Schaltwünsche** der Zentrale gespeichert:

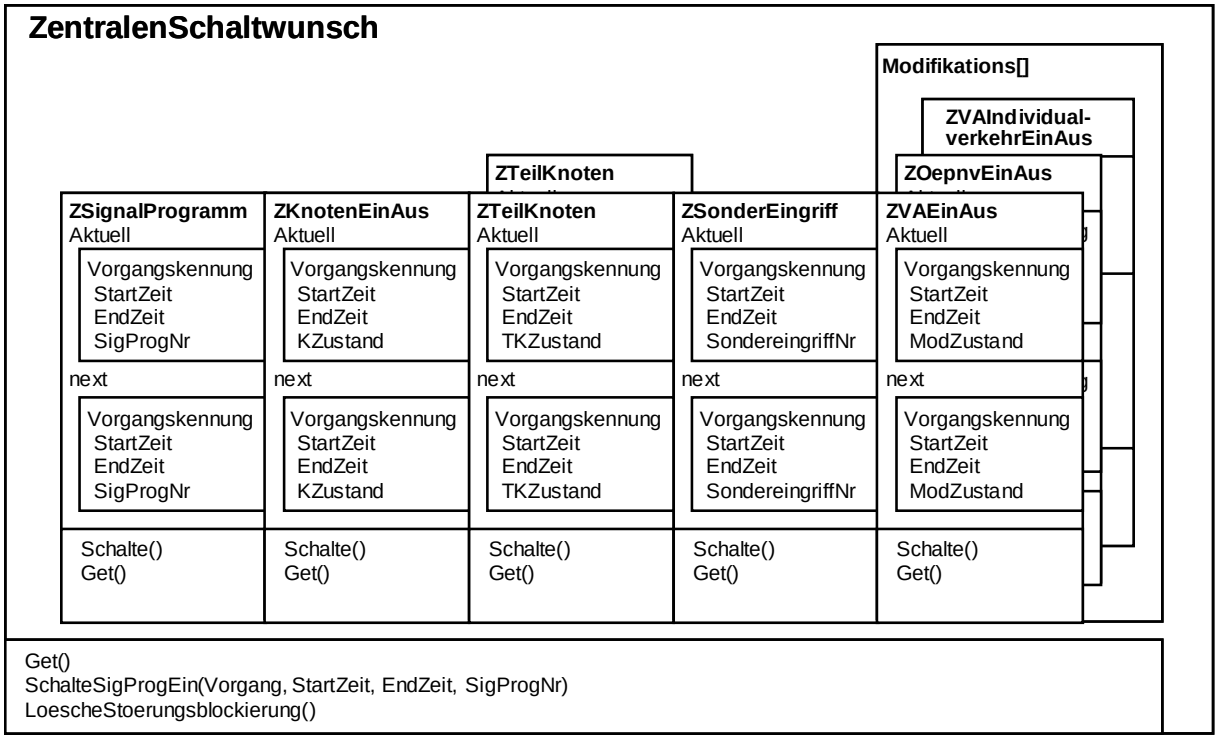


Abbildung 11: Zentralen Schaltwunsch

Obiges Modell geht davon aus, dass jedes Steuerelement des Zentralenschaltwunsches unabhängig von den anderen Steuerelementen einstellbar ist. Schaltet z.B. die Zentrale ein neues Signalprogramm, so bleiben die anderen Schaltwünsche, also auch ZKnotenEinAus und die Modifikationen unverändert erhalten. Die Schaltwünsche haben die der Zentrale zugeordnete Priorität gegenüber lokalen Schaltwünschen. Gibt die Zentrale z.B. die lokale Signalprogrammwahl frei, wählt das Lichtsignalsteuergerät das Signalprogramm aufgrund anderer lokaler Kriterien (gemäß Schaltuhr oder fest eingestellter Ortsplan). OCIT-konforme Konstellationen der Schaltwünsche siehe Pkt. 3.4.1.

Die Vorgangskennung dient der Zentrale zur Zuordnung der Schaltung zu Bedienern und Gruppenschaltungen. Bei Schaltungen in den Zentralenschaltwunsch überträgt das Gerät mit dem Erreichen des gewünschten Zustandes, diesen und dessen Vorgangskennung in das zugehörige IstVektorelement. Zu jedem Steuerelement gibt es ein Statuselement im IstVektor. Der IstVektor und seine Statuselemente sind einzeln oder gesamt nur lesbar (nicht beschreibbar). Er dient der Beobachtung des am Gerät eingestellten Zustands.

Schema des IstVektors:

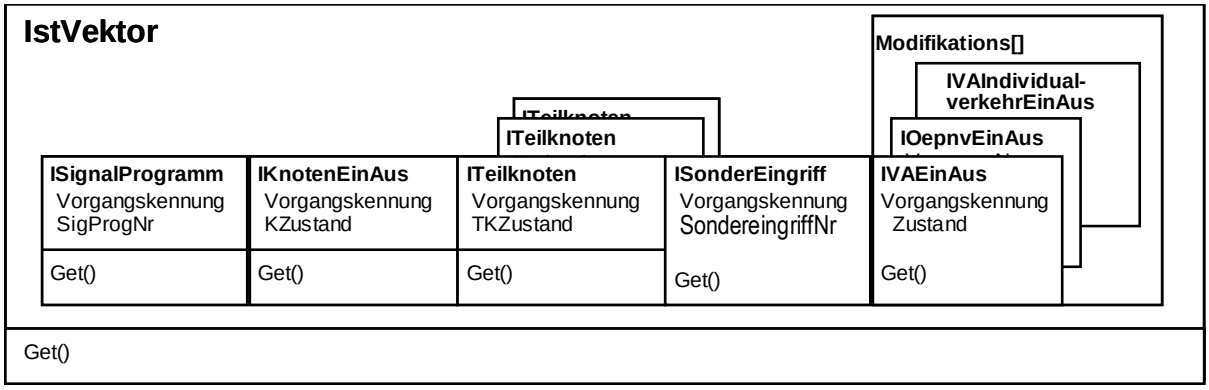


Abbildung 12: Ist Vektor

3.4.1 OCIT-O konforme Konstellationen der zentralen Schaltwünsche

In der OCIT-Definition sind sowohl der Signalplan, der Gesamtknotenzustand und auch der Zustand der einzelnen Teilknoten mit eigenen Methoden und damit verbunden auch unterschiedlichen Gültigkeitsdauern sehr flexibel einstellbar. Um ein eindeutiges Geräteverhalten zu erreichen, werden hier OCIT-konforme Konstellationen angegeben.

Folgende Zustände sind für die oben genannten Objekte definiert:

Objekt	**Zulässige Werte**	**Bedeutung**
Signalprogramm	0 1 - 255	Von der Zentrale kein Signalprogramm gewählt, daraus folgt lokale Signalprogrammwahl. Auswahl des entsprechenden Signalprogramms. Falls dieses im Gerät nicht definiert ist, erfolgt eine Fehlerquittung und es wird kein neuer Schaltwunsch eingetragen, d.h. der alte Zustand bleibt bestehen.
KZustand	0 - 5	**0** Keiner = Kein Zentralenschaltwunsch bezüglich Ein oder Auszustand d. h. Freigabe der lokalen KZustandswahl bzw. unbekannter Ein-/Auszustand des Knotens. **1** Gesamtknoten Ein = Knoten ist in das durch Objekt ZSignalProgramm angegebene Signalprogramm einzuschalten bzw. für alle eingeschalteten Teilknoten gilt, das das durch ISignalProgramm angegebene Signalprogramm bearbeitet wird. **2** Gesamtknoten AusDefault = Knoten ist nach Aus Default zu schalten bzw. für alle eingeschalteten

Objekt	Zulässige Werte	Bedeutung
		Teilknoten gilt: durch Versorgung ausgewählte Signalgruppen, normalerweise die Fahrzeugsignalgruppen der Nebenrichtung, blinken, die restlichen Signalgruppen sind dunkel. **3** Gesamtknoten AusBlinkenNebenrichtung = Knoten ist nach Aus-Blinken-Nebenrichtung zu schalten bzw. für alle eingeschalteten Teilknoten gilt: Die Fahrzeugsignalgruppen der Nebenrichtung, blinken, die restlichen Signalgruppen sind dunkel. **4** Gesamtknoten AusDunkel = Knoten ist nach Aus-Dunkel zu schalten bzw. alle Signalgruppen der eingeschalteten Teilknoten sind dunkel. **5** Gesamtknoten AusBlinkenAlle = Knoten ist nach Aus Blinken alle zu schalten bzw. alle Signalgruppen der eingeschalteten Teilknoten blinken. **6 – 255** nicht zulässig
TKZustand	0 - 5	**0** keiner = kein Zustand eingestellt oder unbestimmter Zustand **1** Ein = bedeutet • bei Verwendung als Schaltbefehl: Teilknoten ist in den KZustand des Gesamtknotens zu schalten • bei Verwendung als Ist-Zustand: Teilknoten ist eingeschaltet. **2** AusDefault = Durch Versorgung ausgewählte Signalgruppen, normalerweise die Fahrzeugsignalgruppen der Nebenrichtung, blinken, die restlichen Signalgruppen sind dunkel. **3** AusBlinkenNebenrichtung = Alle Signalgruppen der Nebenrichtung des Teilknotens blinken. **4** AusDunkel = Alle Signalgruppen des Teilknotens sind dunkel. **5** AusBlinkenAlle = Alle Signalgruppen des Teilknotens blinken. **6 – 255** = nicht zulässig

Wenn die Gültigkeitsdauer eines einzelnen Schaltwunsches ausläuft und kein weiterer gültiger entsprechender Schaltwunsch vorliegt, dann fällt der jeweilige Schaltwunsch in den Zustand 0 mit einer unbegrenzten Gültigkeitsdauer zurück.

KZustand = 0 (oder TKZustand = 0) bedeutet, dass der Gesamtknoten (oder Teilknoten) so sein soll wie aktuell in der lokalen Zeitsteuerung hinterlegt, das Gerät aber trotzdem im Zentralenbetrieb mit dem gewünschten Signalplan läuft.

Auf Grund der Schaltsystematik von OCIT-O können Schaltzustände erzeugt werden, die mehrdeutig interpretiert werden können. Ab der Version OCIT-O V2.0 werden diese eindeutig festgelegt.

Signal-programm	**KZu-stand**	**TKZu stand**	**Reaktion**	**IstVektor *)**
0	0	0	Betrieb gemäß lokaler Steuerung	Lokal 0 bis 4
		1	Betrieb gemäß lokaler Steuerung	Lokal 0 bis 4
		2 - 5	Betrieb gemäß lokaler Steuerung, aber TK-Zustand entsprechend zentralem TK-Zustand.	Zentrale
	1	0	Signalplanauswahl gemäß lokaler Steuerung. Gesamtknoten zentral. Achtung Sonderfall: Es kann sein dass alle Teilknoten durch die lokale JAUT ausgeschaltet werden obwohl der Gesamtknoten eingeschaltet ist.	Zentrale
		1	Signalplanauswahl gemäß lokaler Steuerung. Knotenzustände zentral.	Zentrale
		2 - 5	Betrieb gemäß lokaler Steuerung, aber TK-Zustand entsprechend zentralem TK-Zustand.	Zentrale
	2 - 5	0	Auszustand gemäß zentralen Knoten-Zustand	Zentrale
		1	Auszustand gemäß zentralen Knoten-Zustand	Zentrale
		2 - 5	Auszustand der einzelnen TK’s wie in zentralem TKZustand hinterlegt	Zentrale
1-255	0	0	Zentrale Signalprogrammwahl, aber lokaler Knotenzustand.	Zentrale
		1	Zentrale Signalprogrammwahl, aber lokaler Knotenzustand.	Zentrale

Signal-programm	KZu-stand	TKZu stand	Reaktion	IstVektor *)
		2 - 5	Auszustand der einzelnen TK's wie in zentralen TKZustand hinterlegt	Zentrale
	1	0	Zentralenbetrieb Signalprogramm wie hinterlegt	Zentrale
		1	Zentralenbetrieb Signalprogramm wie hinterlegt	Zentrale
		2 - 5	Zentralenbetrieb Teilknoten im hinterlegten Aus-Zustand	Zentrale
	2 - 5	0	Zentralenbetrieb Aus - Teilknoten im hinterlegten Aus-Zustand (Kzustand)	Zentrale
		1	Zentralenbetrieb Aus - Teilknoten im hinterlegten Aus-Zustand (Kzustand)	Zentrale
		2 - 5	Zentralenbetrieb Aus - Teilknoten im hinterlegten Aus-Zustand (TKzustand)	Zentrale

*****) Der Zustand der Modifikationen hat keinen Einfluss auf die zurückgemeldete Betriebsart!

Die Vorgangskennung der Betriebsart ist durch den Standard nicht festgelegt, da sie durch unterschiedliche Teilzustände beeinflusst werden kann und dem zufolge gerätespezifisch gesetzt werden kann.

Verhalten der Steuergeräte bei Anforderungen von nicht versorgten Signalprogrammen durch eine Zentrale:

- Wird von der Zentrale ein nicht versorgtes Signalprogramm angefordert, so wird der Schaltwunsch mit einer Fehlermeldung (Returncode) zurückgewiesen und nicht vom Steuergerät übernommen. Dies führt dazu, dass nach Ablauf des letzten gültigen Schaltwunsches das Gerät auf lokale Betriebsart zurückfällt.
- Sollte aus dem Gerätezustand Aus der Versuch gemacht werden in ein nicht versorgtes Programm einzuschalten, so muss das Gerät Aus bleiben. Fehlermeldung des Btppl-Protokolls: Param_invalid.

3.4.2 Struktur ZEITINTERVALL

ZEITINTERVALL besteht aus StartZeit und EndZeit. Es ist gültig falls:

0 <= StartZeit < EndZeit und die Endzeit nicht in der Vergangenheit liegt.

Ein Zeitintervall ist aktiv falls es die aktuelle Zeit enthält:

0 <= StartZeit <= aktuelle Zeit <= EndZeit

3.4.3 Typen und Pfade

Alle Knotenbezogenen Strukturen erhalten als ersten Pfadparameter die relative Knotennummer. Damit ist es möglich mehrere Knoten innerhalb eines Lichtsignalsteuergerätes zu adressieren.

Mit dieser relativen Knotennummer (RelKnotenNr) sehen die Pfade wie folgt aus:

ZentralenSchaltwunsch

OType	**Pfad** (Betreiber()/Vsr()/Lichtsignalsteuergerät() immer im BTPPL Header)
220	ZentralenSchaltwunsch/RelKnotenNr()
222	ZentralenSchaltwunsch/ZSignalProgramm/RelKnotenNr()
224	ZentralenSchaltwunsch/ZKnotenEinAus/RelKnotenNr()
226	ZentralenSchalt-wunsch/ZTeilknoten/RelKnotenNr()/TeilknotenNummer()
228	ZentralenSchaltwunsch/ZSonderEingriff/RelKnotenNr()
230	ZentralenSchaltwunsch/ZVAEinAus/RelKnotenNr()
232	ZentralenSchaltwunsch/ZOepnvEinAus/RelKnotenNr()
234	ZentralenSchaltwunsch/ZProjEinAus/RelKnotenNr()/ProjModNr()
238	ZentralenSchaltwunsch/ZVAIndividualverkehrEinAus/RelKnotenNr()

IstVektor

OType	**Pfad** (Betreiber()/Vsr()/Lichtsignalsteuergerät() immer im BTPPL Header)
221	IstVektor/RelKnotenNr()
223	IstVektor/ISignalProgramm/RelKnotenNr()
225	IstVektor/IKnotenEinAus/RelKnotenNr()
227	IstVektor/ITeilknoten/RelKnotenNr()/TeilknotenNummer()
229	IstVektor/ISonderEingriff/RelKnotenNr()
231	IstVektor/IVAEinAus/RelKnotenNr()
233	IstVektor/IOepnvEinAus/RelKnotenNr()
235	IstVektor/IProjEinAus/RelKnotenNr()/ProjModNr()
239	IstVektor/IVAIndividualverkehrEinAus /RelKnotenNr()

3.4.4 Objekt ZSignalProgramm

ZSignalProgramm speichert den Signalprogramm Schaltwunsch der Zentrale.

ZSignalProgramm (1:222)

ZSignalProgramm		
METHOD	**Name**	**Beschreibung**

ZSignalProgramm		
METHOD	**Name**	**Beschreibung**
16	**Schalte**	Nächsten Signalprogrammschaltwunsch der Zentrale entgegennehmen. Falls das durch Start- und EndZeit angegebene Intervall ungültig oder in der Vergangenheit liegt, kehrt diese Methode mit RetCode = INTERVALL_INVALID zurück. Die Serverfunktion in F prüft ob zur angegebenen SigProgNr ein Signalprogramm versorgt ist oder SigProgNr gleich 0 ist (Freigabe der lokalen Signalprogrammwahl). Falls nein kehrt sie mit RetCode=PARAM_INVALID zurück. Falls das übergebene Intervall die aktuelle Zeit enthält (StartZeit <= aktuelle Zeit <EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur Aktuell, stößt den Signalprogrammwechsel an und kehrt mit RETCODE=OK zurück (wartet nicht bis SigProgNr tatsächlich läuft). Ist der Knoten zur Zeit über ' ZKnotenEinAus ' ausgeschaltet, so schaltet er nicht ein bzw. um. Dadurch kann die Zentrale ein Signalprogramm vorgeben und die Ein Ausschaltungen von der lokalen ZAUT vornehmen lassen. Falls das übergebene Intervall in der Zukunft liegt (aktuelle Zeit < StartZeit < EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur next. Ein eventuell in next stehender Schaltauftrag wird storniert und überschrieben. Wird die StartZeit eines in next abgelegten Signalprogrammschaltwunsches erreicht, überträgt F diesen in die Unterstruktur Aktuell, ermittelt und schaltet das dann wirksame Signalprogramm. Wird die EndZeit eines in der Unterstruktur Aktuell abgelegten Schaltwunsches erreicht, beendet F diesen Schaltwunsch. D.h. F ermittelt und schaltet das dann wirksame Signalprogramm. Enthält ZSignalProgramm kein zur Zeit aktives Intervall, liegt kein Zentralen Signalprogramm-Schaltwunsch vor. Die Behandlung entspricht dann einem Schaltwunsch mit SigProgNr=0, d.h. es liegt kein aktueller Signalprogrammschaltwunsch der Zentrale vor.
	Eingabeparameter	

ZSignalProgramm		
METHOD	**Name**	**Beschreibung**
	Vorgang: SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Signalprogrammschaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Signalprogrammschaltwunsch
	EndZeit: utc	Bis „EndZeit“ gilt dieser Signalprogrammschaltwunsch
	SigProgNr: ui1	0 Freigabe der lokalen Signalprogrammwahl 1 - 255 Signalprogramme.
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstrichenes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültige Signalprogrammnummer, Auftrag abgelehnt.
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	Vorgangskennung des Vorgangs welcher diese Signalprogrammnummer eingestellt hat.
	Aktuell.StartZeit	Seit diesem Zeitpunkt ist/war dieser Schaltwunsch aktiv.
	Aktuell.EndZeit	Bis zu diesem Zeitpunkt ist/war dieser Schaltwunsch aktiv.
	Aktuell.SigProgNr	Von der Zentrale für obiges Zeitintervall gewünschte Signalprogrammnummer.
	next.Vorgang	Nächster Zentralenschaltwunsch.
	next.StartZeit	
	next.EndZeit	
	next.SigProgNr	

3.4.5 Objekt ZKnotenEinAus

Dieses Objekt speichert den Schaltwunsch der Zentrale für den Gesamtknotenzustand:

KZustand	
Name	**Bedeutung im Schaltwunsch bzw. IstStand**
Keiner =0	Freigabe der lokalen KZustandswahl, kein Zentralenschaltwunsch bezüglich Ein oder Auszustand.
Ein =1	Knoten ist in das durch das angegebene Signalprogramm einzuschalten bzw. für alle eingeschalteten Teilknoten gilt: das durch ISignal-Programm angegebene Signalprogramm wird bearbeitet.
AusDefault =2	Knoten ist nach AusDefault zu schalten. Für alle eingeschalteten Teilknoten gilt: durch Versorgung ausgewählte Signalgruppen, normalerweise die Fahrzeugsignalgruppen der Nebenrichtung, blinken, die restlichen Signalgruppen sind dunkel.
AusBlinkenNebenrichtung =3	Knoten ist nach AusBlinkenNebenrichtung zu schalten bzw. für alle eingeschalteten Teilknoten gilt: Die Fahrzeugsignalgruppen der Nebenrichtung, blinken, die restlichen Signalgruppen sind dunkel.
AusDunkel =4	Knoten ist nach AusDunkel zu schalten bzw. alle Signalgruppen der eingeschalteten Teilknoten sind dunkel.
AusBlinkenAlle =5	Knoten ist nach AusBlinken alle zu schalten bzw. alle Signalgruppen der eingeschalteten Teilknoten blinken.
6..255	Reserviert

ZKnotenEinAus steuert die Ein- Ausschaltung des Gesamtknotens. Wird z.B. mittels ZKnotenEinAus der gesamte Knoten eingeschaltet, schalten alle Teilknoten deren TKZustand gleich Ein =1 ist ein.

Prinzipiell könnte die Gesamtknoten Ein/Ausschaltung auch über die ZTeilknoten erfolgen. Da aber die Kreuzungsgeräte verschiedene Abläufe für zu und Abschalten von Teilknoten oder Gesamt ein Ausschalten ausführen, erscheint die direkte Vorgabe von Gesamtknoten Ein/Aus transparenter.

ZKnotenEinAus (1:224)

ZKnotenEinAus		
METHOD	**Name**	**Beschreibung**
16	**Schalte**	Nächsten Ein- / Ausschaltwunsch der Zentrale entgegennehmen. Falls das durch Start- und EndZeit angegebene Intervall ungültig oder in der Vergangenheit liegt, kehrt diese Methode mit RetCode = INTERVALL_INVALID zurück. Die Serverfunktion in F prüft der angegebene KZustand gültig ist. Falls nein kehrt sie mit RetCode=PARAM_INVALID zurück. Falls das übergebene Intervall die aktuelle Zeit enthält (StartZeit <= aktuelle Zeit <EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur Aktuell, stößt die Ein- Ausschaltung an und kehrt mit RETCODE=OK zurück (wartet nicht bis tatsächlich geschaltet ist). Falls das übergebene Intervall in der Zukunft liegt (aktuelle Zeit < StartZeit < EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur next. Ein eventuell in next stehender Schaltauftrag wird storniert und überschrieben. Wird die StartZeit eines in next abgelegten Schaltwunsches erreicht, überträgt F diesen in die Unterstruktur Aktuell, ermittelt und schaltet den dann wirksamen KZustand. Wird die EndZeit eines in der Unterstruktur Aktuell abgelegten Schaltwunsches erreicht, beendet F diesen Schaltwunsch. D.h. F ermittelt und schaltet den dann wirksamen KZustand. Enthält ZKnotenEinAus kein zurzeit aktives Intervall, liegt kein Zentralen EinAus-Schaltwunsch vor. Die Behandlung entspricht dann einem Schaltwunsch mit KZustand=0, d.h. es liegt kein aktueller Ein- Ausschaltwunsch der Zentrale vor. Jedes Gerät muss in den Knotenzustand AusDefault schalten können. Kann ein Gerät nicht in AusBlinkenNebenrichtung, AusDunkel oder AusBlinkenAlle schalten, so schaltet es stattdessen in AusDefault. In diesem Fall meldet das Gerät im IKnotenEinAus AusDefault, in ZKnotenEinAus wird jedoch der übergebene KZustand eingetragen.
	Eingabeparameter	

ZKnotenEinAus		
METHOD	**Name**	**Beschreibung**
	Vorgang : SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Signalprogrammschaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Ein- Ausschaltwunsch
	EndZeit : utc	Bis „EndZeit“ gilt dieser Ein- Ausschaltwunsch
	KZustand : ui1	Siehe KZustand
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstrichenes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültiger KZustand, Auftrag abgelehnt.
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	
	Aktuell.StartZeit	
	Aktuell.EndZeit	
	Aktuell.KZustand	
	next.Vorgang	
	next.StartZeit	
	next.EndZeit	
	next.KZustand	

3.4.6 Objekt ZTeilKnoten

ZTeilKnoten speichert den Schaltwunsch der Zentrale für einen Teilknoten:

TKZustand	
Name	**Bedeutung im Schaltwunsch bzw. IstStand**
Keiner =0	Freigabe der lokalen TKZustandswahl, kein Zentralenschaltwunsch bezüglich TKEin-/ Auszustand bzw. unbekannter Zustand.
Ein =1	Verwendung als Schaltbefehl: Teilknoten ist in den KZustand des Gesamtknotens zu schalten Verwendung als Ist-Zustand: Teilknoten ist einge-

TKZustand	
Name	**Bedeutung im Schaltwunsch bzw. IstStand**
	schaltet.
AusDefault =2	Teilknoten ist Aus default, unabhängig vom KZustand des Gesamtknotens. D. h. durch Versorgung ausgewählte Signalgruppen, normalerweise die Fahrzeugsignalgruppen der Nebenrichtung, blinken, die restlichen Signalgruppen sind dunkel.
AusBlinkenNebenrichtung =3	Die Fahrzeugsignalgruppen der Nebenrichtung, blinken, die restlichen Signalgruppen sind dunkel, unabhängig vom KZustand des Gesamtknotens.
AusDunkel =4	Alle Signalgruppen des Teilknotens sind dunkel, unabhängig vom KZustand des Gesamtknotens.
AusBlinkenAlle =5	Alle Signalgruppen des Teilknotens blinken, unabhängig vom KZustand des Gesamtknotens.
6 ... 255	Reserviert

Um Methoden einer Instanz ZTeilKnoten aufzurufen sind im Feld Path (siehe OCIT-O Protokoll) die gewünschte relative Knoten- und Teilknotennummer jeweils als ui1 anzugeben.

ZTeilknoten (1:226)

ZTeilknoten		
METHOD	**Name**	**Beschreibung**
16	**Schalte**	Nächsten Teilknoten Ein- Ausschaltwunsch der Zentrale entgegennehmen. Falls das durch Start- und EndZeit angegebene Intervall ungültig oder in der Vergangenheit liegt, kehrt diese Methode mit RetCode = INTERVALL_INVALID zurück. Die Serverfunktion in F prüft der angegebene TKZustand gültig ist. Falls nein kehrt sie mit RetCode=PARAM_INVALID zurück. Falls das übergebene Intervall die aktuelle Zeit enthält (StartZeit <= aktuelle Zeit <EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur Aktuell, stößt die Teilknoten Ein- Ausschaltung an und kehrt mit RETCODE=OK zurück (wartet nicht bis tatsächlich geschaltet ist). Falls das übergebene Intervall in der Zukunft liegt (aktuelle Zeit < StartZeit < EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur next. Ein eventuell in next stehender Schaltauftrag wird storniert und überschrieben. Wird die StartZeit eines in next abgelegten Schaltwunsches erreicht, überträgt F diesen in die Unterstruktur Aktuell, ermittelt und schaltet den dann wirksamen TKZustand. Wird die EndZeit eines in der Unterstruktur Aktuell abgelegten Schaltwunsches erreicht, beendet F diesen Schaltwunsch. D.h. F ermittelt und schaltet den dann wirksamen TKZustand. Enthält ZTeilKnoten kein zurzeit aktives Intervall, liegt kein Zentralen Teilknoten-Schaltwunsch vor. Die Behandlung entspricht dann einem Schaltwunsch mit TKZustand=0, d.h. es liegt kein aktueller Teilknoten Ein- Ausschaltwunsch der Zentrale vor.
	Eingabeparameter	
	Vorgang: SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Signalprogrammschaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Teilknoten Ein- Ausschaltwunsch
	EndZeit: utc	Bis „EndZeit“ gilt dieser Teilknoten Ein- Ausschaltwunsch

ZTeilknoten		
METHOD	**Name**	**Beschreibung**
	TKZustand: ui1	Siehe TKZustand
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstrichenes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültiger TKZustand, Auftrag abgelehnt. PATH_INVALID: keine oder ungültige Teilknotennummer im Path Feld angegeben, Auftrag abgelehnt.
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	
	Aktuell.StartZeit	
	Aktuell.EndZeit	
	Aktuell.TKZustand	
	next.Vorgang	
	next.StartZeit	
	next.EndZeit	
	next.TKZustand	

3.4.7 Objekt ZSondereingriff

ZSondereingriff speichert den Schaltwunsch der Zentrale für Sondereingriffe ab. Die darin enthaltene SonderEingriffNr hat folgende Bedeutungen:

SonderEingriffNr	
Wert	**Bedeutung**
0	Freigabe lokaler Sondereingriffe, kein Sondereingriff.
1 … 254	Temporär gültiges Signalprogramm, z.B. Feuerwehrplan Route 1 ... n.
255	Sondereingriff Aus, Blockierung lokaler Sondereingriffe.

Liegt ein zentraler Signalprogrammschaltwunsch (ZSignalProgramm) und ein Sondereingriff Schaltwunsch für die gleiche Zeit an, so schaltet das Gerät den Sondereingriff, jedoch nur falls der Knoten eingeschaltet ist (ZKnotenEinAus).

Das Objekt ZSondereingriff ist nötig, damit das Gerät nach Ablauf des Sondereingriffs ohne weitere Zentralenkommunikation wieder in das normale zentrale Signalprogramm schalten kann.

ZSondereingriff (1:228)

ZSondereingriff		
METHOD	**Name**	**Beschreibung**
16	**Schalte**	Nächsten Sonder-Signalprogrammschaltwunsch der Zentrale entgegennehmen. Falls das durch Start- und EndZeit angegebene Intervall ungültig oder in der Vergangenheit liegt, kehrt diese Methode mit RetCode = INTERVALL_INVALID zurück. Die Serverfunktion in F prüft ob zur angegebenen SonderEingriffNr ein Signalprogramm versorgt ist. Falls nein kehrt sie mit Ret-Code=PARAM_INVALID zurück. Falls das übergebene Intervall die aktuelle Zeit enthält (StartZeit <= aktuelle Zeit <EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur Aktuell, stößt den Signalprogrammwechsel an und kehrt mit RETCODE=OK zurück (wartet nicht bis SonderEingriffNr tatsächlich läuft). Falls das übergebene Intervall in der Zukunft liegt (aktuelle Zeit < StartZeit < EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur next. Ein eventuell in next stehender Schaltauftrag wird storniert und überschrieben. Wird die StartZeit eines in next abgelegten Sonder- Signalprogrammschaltwunsches erreicht, überträgt F diesen in die Unterstruktur Aktuell, ermittelt und schaltet das dann wirksame Signalprogramm. Wird die EndZeit eines in der Unterstruktur Aktuell abgelegten Schaltwunsches erreicht, beendet F diesen Schaltwunsch. D.h. F ermittelt und schaltet das dann wirksame Signalprogramm. Enthält ZSondereingriff kein zurzeit aktives Intervall, liegt kein Zentralen Sonder-Signalprogramm-Schaltwunsch vor. Die Behandlung entspricht dann einem Schaltwunsch mit SonderEingriffNr =0, d.h. es liegt kein aktueller Sonder-Signalprogrammschaltwunsch der Zentrale vor.
	Eingabeparameter	

ZSondereingriff		
METHOD	**Name**	**Beschreibung**
	Vorgang : SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Signalprogrammschaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Sonder-Signalprogrammschaltwunsch
	EndZeit: utc	Bis „EndZeit“ gilt dieser Sonder-Signalprogrammschaltwunsch
	SonderEingriffNr: ui1	Siehe SonderEingriffNr.
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstrichenes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültige Sonder-Signalprogrammnummer, Auftrag abgelehnt.
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	
	Aktuell.StartZeit	
	Aktuell.EndZeit	
	Aktu-ell.SonderEingriffNr	
	next.Vorgang	
	next.StartZeit	
	next.EndZeit	
	next.SonderEingriffNr	

3.4.8 Signalprogramm Modifikationen

Es gibt einige ein-/ ausschaltbare Signalprogramm Parameter. ModEinAusZustand gibt die Codierung des von der Zentrale gewünschten Zustands einer Modifikation an:

ModEinAusZustand	
Wert	**Bedeutung**
Keiner = 0	Kein Zustand eingestellt, unbestimmter Zustand oder lokale Zustandswahl freigegeben

ModEinAusZustand	
Wert	**Bedeutung**
Aus = 1	Modifikation ist ausgeschaltet.
Ein = 2	Modifikation ist eingeschaltet

Die Semantik von ModEinAusZustand ist abhängig vom Objekt in welchem es verwendet wird.

Modifikationen werden wirksam in Abhängigkeit von der Priorität der Schaltquelle:

1. Höchste Priorität Manueller Eingriff
2. Zentrale
3. Niedrigste Priorität: Schaltuhr

3.4.8.1 Objekt ZModEinAus

Das Objekt ZModEinAus dient als Basisklasse für alle Modifikationen.

ZModEinAus (1:206)

ZModEinAus		
METHOD	**Name**	**Beschreibung**
16	**Schalte**	Nächsten Modifikations Ein- Ausschaltwunsch der Zentrale entgegennehmen. Falls das durch Start- und EndZeit angegebene Intervall ungültig oder in der Vergangenheit liegt, kehrt diese Methode mit RetCode = INTERVALL_INVALID zurück. Die Serverfunktion in F prüft der angegebene VAZustand gültig ist. Falls nein kehrt sie mit Ret-Code=PARAM_INVALID zurück. Falls das übergebene Intervall die aktuelle Zeit enthält (StartZeit <= aktuelle Zeit <EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur Aktuell, stößt die Ein- Ausschaltung an und kehrt mit RETCODE=OK zurück (wartet nicht bis tatsächlich geschaltet ist). Falls das übergebene Intervall in der Zukunft liegt (aktuelle Zeit < StartZeit < EndZeit) überträgt diese Methode die angegebenen Parameter in die Unterstruktur next. Ein eventuell in next stehender Schaltauftrag wird storniert und überschrieben. Wird die StartZeit eines in next a gelegten Schaltwunsches erreicht, überträgt F diesen in die Unterstruktur Aktuell, ermittelt und schaltet den dann wirksamen Zustand. Wird die EndZeit eines in der Unterstruktur Aktuell abgelegten Schaltwunsches erreicht, beendet F diesen Schaltwunsch. D.h. F ermittelt und schaltet den dann wirksamen Zustand. Enthält ModEinAus kein zurzeit aktives Intervall, liegt kein Zentralen EinAus-Schaltwunsch vor. Die Behandlung entspricht dann einem Schaltwunsch mit Zustand=0, d.h. es liegt kein aktueller Ein-/ Ausschaltwunsch der Zentrale vor.
	Eingabeparameter	
	Vorgang: SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Signalprogrammschaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Ein-/ Ausschaltwunsch

ZModEinAus		
METHOD	**Name**	**Beschreibung**
	EndZeit: utc	Bis „EndZeit“ gilt dieser Ein-/ Ausschaltwunsch
	Zustand: ModeEin-AusZustand	Einzustellender Zustand siehe ModeEinAusZu-stand.
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstriche-nes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültiger VAZustand, Auftrag abgelehnt. NOT_CONFIGURED: Angegebene Modifikation ist nicht versorgt, Auftrag abgelehnt.
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	Bedienvorgangskennung des Aufrufers für diese Modifikationsschaltung
	Aktuell.StartZeit	Ab diesem Zeitpunkt war dieser Schaltwunsch gül-tig
	Aktuell.EndZeit	Bis zu diesem Zeitpunkt ist oder war dieser Schalt-wunsch gültig
	Aktuell.Zustand	Eingestellter Zustand s. ModEinAusZustand
	Next.Vorgang	Zeitlich nächster Zentralenschaltwunsch
	Next.StartZeit	
	Next.EndZeit	
	Next.Zustand	

3.4.8.2 Objekt IModEinAus

Das Objekt IModEinAus zeigt den aktuell eingestellter Modifikations- Ein / Aus Zustand und zugehörige Vorgangskennung. Bedeutung je nach abgeleiteter Klasse VA, ÖPNV, Orientierungstakt.

IModEinAus (1:207)

IModEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden Ein-/ Ausschaltung der Modifikation geführt hat.
	Zustand	Ein- Auszustand dieser Modifikation s. ModeEin-Auszustand.

Für alle Modifikationen wird je eine Spezialisierung von ZModEinAus und ImodEin-Aus definiert. Das hat den Vorteil, dass (wenn später nötig) auch Modifikationen mit Parametern machbar sind. Für Alle Modifikationen gilt: Solange im Gerät eine Modifikation nicht implementiert ist,

- liefern die Methoden ZModEinAus.Schalte(...), ZModEinAus.Get(), IModEin-Aus.Get() einen Fehler zurück (ERR_TYPE oder NOT_CONFIGURED),
- wird IModEinAus weder im IstVektor.Modifikations[] noch ZModEinAus im ZentralenSchaltwunsch.Get(Modifikations[ ]) übertragen.

3.4.8.3 Objekt ZVAEinAus

ZVAEinAus ist eine Spezialisierung von ZModeEinAus und speichert den Schaltwunsch der Zentrale für den übergeordneten Zustand der lokalen Verkehrsabhängigkeit.

ZVAEinAus (1:230)

ZVAEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	Bedienvorgangskennung des Aufrufers für diese Modifikationsschaltung

	Aktuell.StartZeit	Ab diesem Zeitpunkt war dieser Scha
	Aktuell.EndZeit	Bis zu diesem Zeitpunkt ist oder war dieser Schaltwunsch gültig
	Aktuell.Zustand	Eingestellter Zustand siehe Tabelle VAZustand
	Next.Vorgang	Zeitlich nächster Zentralenschaltwunsch
	Next.StartZeit	
	Next.EndZeit	
	Next.Zustand	

Der aktuelle Zustand hat im Objekt ZVAEinAus folgende Bedeutung:

VAZustand	
Wert	**Bedeutung**
Keiner = 0	Freigabe der lokalen VA-Zustandswahl.
Aus = 1	Aus: Die lokale verkehrsabhängige Logik arbeitet nicht, d.h. Festzeitbetrieb
Ein = 2	Ein: Die lokale verkehrsabhängige Logik arbeitet.

Bei aktiviertem Zustand der Verkehrsabhängigkeit kann durch die Objekte ZVAIndividualverkehrEinAus und ZOepnvEinAus das Verhalten der verkehrsabhängigen Logik detailliert werden.

3.4.8.4 Objekt ZOepnvEinAus

Falls ein Gerät bzw. Knoten keine ein- ausschaltbare ÖPNV Bevorzugung unterstützt, liefert die Methode Schalte einen Fehler zurück (ERR_TYPE, NOT_CONFIGURED).

siehe oben (allgemeine Modifikation)

ZOepnvEinAus ist eine Spezialisierung von **ZModEinAus** und speichert den von der Zentrale eingestellten übergeordneten Zustand der lokalen OEPNV Bevorzugung.

ZOepnvEinAus (1:232)

ZOepnvEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	Bedienvorgangskennung des Aufrufers für diese Modifikationsschaltung
	Aktuell.StartZeit	Ab diesem Zeitpunkt war dieser Schaltwunsch gültig
	Aktuell.EndZeit	Bis zu diesem Zeitpunkt ist oder war dieser Schaltwunsch gültig
	Aktuell.Zustand	Eingestellter Zustand siehe Tabelle OepnvEinAus
	Next.Vorgang	Zeitlich nächster Zentralenschaltwunsch
	Next.StartZeit	
	Next.EndZeit	
	Next.Zustand	

Der aktuelle Zustand hat im Objekt OepnvEinAus folgende Bedeutung:

OepnvEinAus	
Wert	**Bedeutung**
Keiner = 0	Freigabe der lokalen OEPNV Bevorzugung.
Aus = 1	Aus: Die lokale OEPNV Bevorzugung arbeitet nicht
Ein = 2	Ein: Die lokale OEPNV Bevorzugung arbeitet.

Bei abgeschalteter OEPNV-Beschleunigung führen Anforderungen des OEPNV nicht zur Beeinflussung der Signalisierung, d.h. der OEPNV wird nicht beschleunigt.

3.4.8.4.1 Objekt ZVAIndividualverkehrEinAus

Falls ein Gerät bzw. Knoten keine separate Steuerung der VA unterstützt, liefert die Methode Schalte einen Fehler zurück (ERR_TYPE, NOT_CONFIGURED).
siehe oben (allgemeine Modifikation).

ZVAIndividualverkehrEinAus ist eine Spezialisierung von **ZmodEinAus** und speichert den von der Zentrale eingestellten Zustand der Beeinflussung der lokalen verkehrsabhängigen Logik des Knotens durch den Individualverkehr.

ZVAIndividualverkehrEinAus (1:238)

ZVAIndividualverkehrEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	Bedienvorgangskennung des Aufrufers für diese Modifikationsschaltung
	Aktuell.StartZeit	Ab diesem Zeitpunkt war dieser Schaltwunsch gültig
	Aktuell.EndZeit	Bis zu diesem Zeitpunkt ist oder war dieser Schaltwunsch gültig
	Aktuell.Zustand	Eingestellter Zustand siehe Tabelle ZVAIndividualverkehrEinAus
	Next.Vorgang	Zeitlich nächster Zentralenschaltwunsch
	Next.StartZeit	
	Next.EndZeit	
	Next.Zustand	

Der aktuelle Zustand hat im Objekt ZVAIndividualverkehrEinAus folgende Bedeutung:

ZVAIndividualverkehrEinAus	
Wert	**Bedeutung**
Keiner = 0	Freigabe der VA-Beeinflussung durch Individualverkehr / unbekannter Zustand.
Aus = 1	Aus: Individualverkehr beeinflusst VA nicht (VA-Reduzierung)
Ein = 2	Ein: Individualverkehr beeinflusst VA

Befindet sich die Modifikation im Zustand „AUS“, so beeinflussen Ereignisse des Individualverkehrs, wie z.B. Detektoren, nicht die verkehrstechnische Logik. Dieser Zustand wird als Reduzierung der VA bezeichnet.

3.4.8.5 Kombination von Modifikationen

Aus der Kombination der Modifikationen lässt sich die Verarbeitung von verkehrstechnischen Anforderungen beeinflussen. Die mögliche Bedeutung der Kombinationen bei Verwendung aller drei Kombinationen zeigt folgende Tabelle. Es wird die mögliche Beeinflussung der verkehrstechnischen Logik durch den Individualverkehr und OEPNV abgebildet.

VA EinAus	**VAIndividualverkehr EinAus**	**ÖPNV EinAus**	**Geräteverhalten**
Aus	Aus	Aus	Übergeordneter VA-Zustand ist aus, d.h. Gerät läuft in Festzeit, keine VA aktiv und damit keine Beeinflussung möglich
Aus	Aus	Ein	Übergeordneter VA-Zustand ist aus, d.h. Gerät läuft in Festzeit, keine VA aktiv und damit keine Beeinflussung möglich
Aus	Ein	Aus	Übergeordneter VA-Zustand ist aus, d.h. Gerät läuft in Festzeit, keine VA aktiv und damit keine Beeinflussung möglich
Aus	Ein	Ein	Übergeordneter VA-Zustand ist aus, d.h. Gerät läuft in Festzeit, keine VA aktiv und damit keine Beeinflussung möglich
Ein	Aus	Aus	VA läuft im Festzeitbetrieb ohne Beschleunigung des IV / OEPNV, d.h. nur Hintergrundfunktionen der VA (z.B. Archive schreiben, sonstige Funktionen) werden ausgeführt.
Ein	Aus	Ein	VA läuft verkehrsabhängig ohne Beschleunigung des IV, d.h. nur der OEPNV beeinflusst die Signalisierung.
Ein	Ein	Aus	VA läuft verkehrsabhängig ohne Beschleunigung des OEPNV, d.h. nur der Individualverkehr beeinflusst die Signalisierung.
Ein	Ein	Ein	VA läuft voll verkehrsabhängig, d.h. sowohl der Individualverkehr als auch der OEPNV beeinflusst die Signalisierung.

VA EinAus	VAIndividu-alverkehr EinAus	ÖPNV EinAus	Geräteverhalten
keiner	Aus	Aus	undefinierter Zustand
keiner	Aus	Ein	undefinierter Zustand
keiner	Ein	Aus	undefinierter Zustand
keiner	Ein	Ein	undefinierter Zustand

3.4.9 Projektspezifische Modifikationen

Für die Bedienung und Anzeige von projektspezifischen Erweiterungen die der Gerätehersteller im Gerät implementiert, wie beispielsweise das Schalten von digitalen Ausgängen, sind die Objekte ZProjEinAus, IProjEinAus vorgesehen. Damit ist eine OCIT-Zentrale in der Lage diese zu schalten (ZProjEinAus) und anzuzeigen (IProjEinAus). Damit die Zentrale die Bedeutung einer Modifikation dem Bediener anzeigen kann liefert das Gerät einen Bedeutungstext. Solange im Gerät eine projektspezifische Erweiterung nicht implementiert ist:

- liefern die Methoden ZProjEinAus.Schalte(), ZProjEinAus.Get(), IProjEinAus.Get() einen Fehler zurück (ERR_TYPE oder NOT_CONFIGURED),
- wird ZProjEinAus weder im IstVektor.Modifikations[ ] noch im Zentralen-Schaltwunsch.Get(Modifikations[ ]) übertragen.

Damit auch mehrere gleichartige projektspezifische Modifikationen machbar sind erweitern die Objekte ZProjEinAus, IProjEinAus den Path ihre Basisklassen ZModEinAus, IModEinAus um eine Nummer (ProjModNr) zur Unterscheidung.

3.4.9.1 Objekt ZProjEinAus

Von der Zentrale einstellbare Ein-/ Ausschaltzustände der projektspezifischen Modifikationen.

ZProjEinAus (1:234)

Objekt ZProjEinAus		
Path	RelKnotenNr	Knotennummer innerhalb eines Lichtsignalsteuergerätes, (von Basisklasse ZModEinAus geerbt).
	ProjModNr	Nummer zur Unterscheidung mehrerer projektspezifischer Modifikationen innerhalb eines Knotens.
METHOD	**Name**	**Beschreibung**
16	**Schalte**	Nächsten Modifikations Ein-/ Ausschaltwunsch der Zentrale entgegennehmen. s. ZModEinAus

Objekt ZProjEinAus		
Path	RelKnotenNr	Knotennummer innerhalb eines Lichtsignalsteuergerätes, (von Basisklasse ZModEinAus geerbt).
	ProjModNr	Nummer zur Unterscheidung mehrerer projektspezifischer Modifikationen innerhalb eines Knotens.
METHOD	**Name**	**Beschreibung**
	Eingabeparameter	
	Vorgang: SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Signalprogrammschaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Ein-/ Ausschaltwunsch
	EndZeit: utc	Bis „EndZeit“ gilt dieser Ein-/ Ausschaltwunsch
	Zustand: ModeEinAus-Zustand	Einzustellender Zustand siehe ModeEinAusZustand
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstrichenes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültiger VAZustand, Auftrag abgelehnt. NOT_CONFIGURED: Angegebene Modifikation ist nicht versorgt, Auftrag abgelehnt.
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Aktuell.Vorgang	Bedienvorgangskennung des Aufrufers für diese Modifikationsschaltung
	Aktuell.StartZeit	Ab diesem Zeitpunkt war dieser Schaltwunsch gültig
	Aktuell.EndZeit	Bis zu diesem Zeitpunkt ist oder war dieser Schaltwunsch gültig
	Aktuell.Zustand	Eingestellter Zustand s. ModeEinAusZustand
	Next.Vorgang	
	Next.StartZeit	
	Next.EndZeit	
	Next.Zustand	

3.4.9.2 Objekt IProjEinAus

Aktuell eingestellter Zustand der projektspezifischen Modifikation ProjNr und zugehörige Vorgangskennung.

IProjEinAus (1:235)

IProjEinAus		
Path	RelKnotenNr	Knotennummer innerhalb eines Lichtsignalsteuergerätes, (von Basisklasse ZModEinAus geerbt).
	ProjModNr	Nummer zur Unterscheidung mehrerer projektspezifischer Modifikationen innerhalb eines Knotens.
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden Ein-/ Ausschaltung der Modifikation geführt hat.
	Zustand	Ein-/ Auszustand dieser Modifikation s. ModeEinAusZustand
33	**Bedeutung**	Zur Abfrage eines Bedeutungstextes zur Anzeige und Unterscheidung in der Zentrale.
	Ausgabeparameter	
	RetCode	OK, NOT_CONFIGURED Modifikation ist nicht versorgt.
	Text: STRING	Bedeutungstext zur Anzeige und Unterscheidung der projektspezifischen Modifikation in der Zentrale.

3.4.10 Objekt ZentralenSchaltwunsch

Dieses Objekt enthält Methoden die mehrere Unterobjekte des Zentralenschaltwunschs betreffen. Es dient auch dazu alle Zentralenschaltwünsche mit einem Get Aufruf zu holen.

ZentralenSchaltwunsch (1:220)

ZentralenSchaltwunsch		
METHOD	**Name**	**Beschreibung**
16	**SchalteSigProgEin**	Diese Methode ist eine Abkürzung um das Signalprogramm und Knoten Ein mit nur einem Aufruf zu schalten. Falls das durch Start- und EndZeit angegebene Intervall ungültig oder in der Vergangenheit liegt, kehrt diese Methode mit RetCode = INTERVALL_INVALID zurück. Die Serverfunktion in F prüft ob zur angegebenen SigProgNr ein Signalprogramm versorgt ist. Falls nein kehrt sie mit RetCode=PARAM_INVALID zurück. Nun führt diese Methode sinngemäß die folgenden Operationen aus: RetCode = ZSignalProgramm.Schalte(Vorgang, StartZeit, EndZeit, SigProgNr); if(RetCode == OK) RetCode = ZKnotenEinAus.Schalte(Vorgang, StartZeit, EndZeit, Ein); return RetCode;
	Eingabeparameter	
	Vorgang: SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Signalprogrammschaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Signalprogrammschaltwunsch
	EndZeit: utc	Bis „EndZeit“ gilt dieser Signalprogrammschaltwunsch
	SigProgNr: ui1	0 Freigabe der lokalen Signalprogrammwahl 1 - 255 Signalprogramme.
	Ausgabeparameter	

ZentralenSchaltwunsch		
METHOD	**Name**	**Beschreibung**
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstrichenes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültige Signalprogrammnummer, Auftrag abgelehnt.
17	**LoescheStoerungsblockierung**	Falls das Gerät durch eine Störung abgeschaltet hat, ermöglicht diese Methode einen erneuten Einschaltversuch. Die Methode kehrt sofort zurück (wartet nicht bis erneut eingeschalten ist).
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen PARAM_INVALID: es liegt keine Störungsblockierung vor, Auftrag abgelehnt.
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	SignalProgramm: ZSignalProgramm	Siehe Objekt ZSignalProgramm
	KnotenEinAus: ZKnotenEinAus	Siehe Objekt ZKnotenEinAus
	Teilknoten[0...3]: ZTeilknoten	Übertragung als Array mit festem Typ: - zuerst ein UBYTE Anzahl folgender ZTeilknoten Daten - Daten der Teilknoten (siehe Objekt ZTeilKnoten) Die Daten der Teilknoten werden in aufsteigender Reihenfolge übertragen (keine Übertragung der Teilknotennummer).
	SonderEingriff: ZSonderEingriff	Siehe Objekt ZSondereingriff

ZentralenSchaltwunsch		
METHOD	**Name**	**Beschreibung**
	Modifikations[0...15]: ZModEinAus	Modifikationen des Signalprogramms. Hier kann jede von ZmodEinAus abgeleitete Klasse stehen, dies sind z. Zt. ZVAEinAus, ZOepnvEinAus, ZProjEinAus. Übertragung als Array mit variablen Typen: - zuerst ein UBYTE Anzahl folgender ZModEinAus Daten - RefLen, Länge der Referenz - ID der Daten, OdgMember OType - relativer Path (im Fall von ZVAEinAus, ZOepnvEinAus nichts, bei ZProjEinAus ist es ProjModNr) - Datenlänge - Daten einer von ZModEinAus abgeleiteten Klasse
18	**SchalteKnoten**	Diese Methode ist eine Abkürzung um das Signalprogramm, Knoten, Teilknoten, Sondereingriff und Modifikationen mit nur einem Aufruf zu schalten. Falls das durch Start- und EndZeit angegebene Intervall ungültig oder in der Vergangenheit liegt, kehrt diese Methode mit RetCode = INTERVALL_INVALID zurück.
	Eingabeparameter	
	Vorgang: SYSJOBID	Bedienvorgangskennung des Aufrufers für diese Schaltung.
	StartZeit: utc	Ab „StartZeit“ gilt dieser Schaltwunsch
	EndZeit: utc	Bis „EndZeit“ gilt dieser Schaltwunsch
	SigProgNr: ui1	0 Freigabe der lokalen Signalprogrammwahl 1 - 255 Signalprogramme.
	KZustand: ui1	Siehe KZustand
	Teilknoten[0...3]: TKZustand	Übertragung als Array mit festem Typ: - zuerst ein UBYTE Anzahl folgender TKZustand Daten - Daten der Teilknoten (siehe Objekt TKZustand) Die Daten der Teilknoten werden in aufsteigender Reihenfolge übertragen (keine Übertragung der Teilknotennummer).
	SonderEingriff: ui1	Nummer eines Sondereingriffs.

ZentralenSchaltwunsch		
METHOD	**Name**	**Beschreibung**
	Modifikationen[0...15]: GModZustand	Modifikationen des Signalprogramms. Hier kann jede von GModZustand abgeleitete Klasse stehen, dies sind z.Zt. GVAZustand, GOepnvZustand, GVAIndividualverkehrZustand, GProjZustand. Übertragung als Array mit variablen Typen: - zuerst ein UBYTE Anzahl folgender GModZustand Daten - RefLen, Länge der Referenz - ID der Daten, OdgMember OType - relativer Path (im Fall von GVAZustand, GOepnvZustand, GVAIndividualverkehr nichts, bei GProjZustand, ist es ProjModNr) - Datenlänge: ui2 - Modifikationszustand und weitere Daten entsprechend dem zur Laufzeit angegebenen Member, Otype
	Ausgabeparameter	
	RetCode	OK: Auftrag angenommen INTERVALL_INVALID: ungültiges oder verstrichenes Zeitintervall angegeben, Auftrag abgelehnt. PARAM_INVALID: ungültige Signalprogrammnummer, ungültige TK - Zustände, ungültige Sonder-Signalprogrammnummer, ungültige Modifikationszustände angegeben, Auftrag abgelehnt. PATH_INVALID: keine oder ungültige Teilknotennummer angegeben, Auftrag abgelehnt. NOT_CONFIGURED: Angegebene Modifikationen sind nicht versorgt, Auftrag abgelehnt.

Hinweis: Es wird empfohlen ab OCIT-O Lstg Version 2.0, Ausgabe 02 die Methode **SchalteKnoten** (Methode 18) zu verwenden, weil damit ein Einfluss des zeitlichen Verhaltens der Übertragungstrecke auf die Schaltwünsche ausgeschlossen wird.

3.4.11 Objekt ISignalProgramm

Das Objekt ISignalProgramm liefert das zum Zeitpunkt der Abfrage vom Gerät bearbeitete Signalprogramm und Vorgangskennung des zugehörigen Auftrags.

Hinweis: Im Falle von vollständigen Störabschaltungen werden die Zustände zum Zeitpunkt der Abschaltung im IstVektor beibehalten.

ISignalProgramm (1:223)

ISignalProgramm		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden SigProgNr Schaltung geführt hat.
	SigProgNr	Nummer des zur Zeit abgearbeiteten Signalprogramms.

3.4.12 Objekt IKnotenEinAus

IKnotenEinAus liefert den zum Zeitpunkt der Abfrage eingestellten Ein- Auszustand des Knotens mit der Vorgangskennung des zugehörigen Auftrags.

Hinweis: Im Falle von vollständigen Störabschaltungen wird der anliegende Auszustand in IstVektor eigetragen.

IKnotenEinAus (1:225)

IKnotenEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden Knoten Ein-/ Ausschaltung geführt hat.
	KZustand	Ein-/ Auszustand des Knotens.

3.4.13 Objekt ITeilknoten

Vom Typ ITeilknoten gibt es pro Teilknoten eine Instanz.

ITeilknoten enthält den aktuell eingestellten Zustand des adressierten Teilknotens.

Hinweis: Im Falle von vollständigen Störabschaltungen wird der anliegende Auszustand in IstVektor eigetragen.

ITeilknoten (1:227)

ITeilknoten		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden TeilKnoten Ein-/ Ausschaltung geführt hat.
	Zustand	Ein-/ Auszustand dieses TeilKnotens.

3.4.14 Objekt ISondereingriff

Das Objekt ISondereingriff liefert das zum Zeitpunkt der Abfrage eingestellte Sonder-Signalprogramm und Vorgangskennung des zugehörigen Auftrags.

Hinweis: Im Falle von vollständigen Störabschaltungen werden die Zustände zum Zeitpunkt der Abschaltung im IstVektor beibehalten.

ISondereingriff (1:229)

ISondereingriff		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden Sondereingriff Schaltung geführt hat.
	SonderEingriffNr	Nummer des Sondereingriffs. Der Wert 0 bedeutet hier: aktuell kein Sondereingriff.

3.4.15 Objekt IVAEinAus

IVAEinAus ist eine Spezialisierung von IModEinAus und liefert den zum Zeitpunkt der Abfrage eingestellten übergeordneten Zustand der lokalen verkehrsabhängigen Logik des Knotens mit der Vorgangskennung des zugehörigen Auftrags.

Hinweis: Im Falle von vollständigen Störabschaltungen werden die Zustände zum Zeitpunkt der Abschaltung im IstVektor beibehalten.

IVAEinAus (1:231)

IVAEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden Ein-/ Ausschaltung der Verkehrsabhängigkeit geführt hat.
	VAZustand	Ein-/ Auszustand der Verkehrsabhängigkeit.

3.4.16 Objekt IVAIndividualverkehrEinAus

IVAIndividualverkehrEinAus ist eine Spezialisierung von IModEinAus. Das Objekt liefert den zum Zeitpunkt der Abfrage aktiven Zustand der Beeinflussung der lokalen verkehrsabhängigen Logik des Knotens durch den Individualverkehr mit der Vorgangskennung des zugehörigen Auftrags.

Hinweis: Im Falle von vollständigen Störabschaltungen werden die Zustände zum Zeitpunkt der Abschaltung im IstVektor beibehalten.

IVAIndividualverkehrEinAus (1:239)

IVAIndividualverkehrEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden Ein-/ Ausschaltung der Verkehrsabhängigkeit geführt hat.
	VAModifikationZustand	Ein-/ Auszustand des Einflusses des Individualverkehrs auf die VA (Steuerung der VA-Reduzierung)

3.4.17 Objekt IOepnvEinAus

IOepnvEinAus ist eine Spezialisierung von IModEinAus und liefert den zum Zeitpunkt der Abfrage eingestellten übergeordneten Zustand der lokalen OEPNV Bevorzugung des Knotens mit der Vorgangskennung des zugehörigen Auftrags.

Hinweis: Im Falle von vollständigen Störabschaltungen werden die Zustände zum Zeitpunkt der Abschaltung im IstVektor beibehalten.

IOepnvEinAus (1:233)

IOepnvEinAus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur folgenden Ein-/ Ausschaltung der lokalen OEPNV Bevorzugung geführt hat.
	VAZustand	Ein-/ Auszustand der lokalen OEPNV Bevorzugung.

3.4.18 Objekt IBetriebsart

Das Objekt IBetriebsart gibt Auskunft über die aktuell laufende Betriebsart (mit Vorgangsnummer) eines relativen Knotens.

Hinweis: Es gibt keine OCIT-Outstations Funktion um die Betriebsart von der Zentrale aus einzustellen. Eine Betriebsart ist gültig, wenn sie mindestens über einen Zustand die Kontrolle ausübt. Die Betriebsart ergibt sich aus dem IstVektor und kann in wenigen Sonderfällen (z. B. Teilknotenblockierung durch Bedienteil oder Schalter) geräteabhängig verschieden sein.

Im Falle von vollständigen Störabschaltungen werden die Zustände zum Zeitpunkt der Abschaltung im IstVektor beibehalten.

IBetriebsart (1:209)

IBetriebsart		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Vorgang	Vorgangskennung des Vorgangs welcher zur Einstellung der Betriebsart geführt hat

IBetriebsart		
METHOD	**Name**	**Beschreibung**
	Betriebsart	Eingestellte Betriebsart: - Sonderbetrieb - Eigensteuerung - Handstoppbetrieb - LokalFix - LokalZeitsteuerung - Zentrale

3.4.19 Objekt IstVektor

Das Objekt IstVektor liefert den aktuellen Betriebszustand und eine Sammelstörungskennung. Wenn sich die Sammelstörung ändert, wird dazu ein Event-Telegramm abgesetzt. Daraufhin kann die Zentrale den IstVektor lesen.

Objekt IstVektor (1:221)

IstVektor			
METHOD	**Name**		**Beschreibung**
0	G**et**		
	Ausgabeparameter		
	RetCode		OK: folgende Parameter richtig gelesen
	Zeitstempel: utc		Zeitstempel des Gerätes: wann der Zustand erreicht oder zuletzt geändert wurde.
	Sammelstörung: ui1		0 = keine Störung 1 = Störung ohne Abschaltung 2 = Störung mit Abschaltung 3 = Störung mit Teilabschaltung 4 = Interne Störung ohne Abschaltung Weitere Festlegungen dazu am Ende der Tabelle!
	IBetriebsart		Zeigt an, welche Betriebsart aufgrund welchen Vorgangs eingestellt ist.
		Vorgangskennung	Vorgangskennung des Systemteils welcher die Kontrolle über die aktuell eingestellte Betriebsart ausübt.
		Betriebsart	Zu dieser Zeit eingestellte Betriebsart.
	ISignalProgramm		Zeigt an welches Signalprogramm aufgrund welches Vorgangs eingestellt ist.
		Vorgangskennung	Vorgangskennung des Systemteils welcher die Kontrolle über das aktuell eingestellte Signalprogramm ausübt.
		SigProgNr	Zu dieser Zeit eingestellte Signalprogrammnummer
	IKnotenEinAus		Zeigt an ob Gesamtknoten ein oder ausgeschaltet ist.
		Vorgangskennung	Kennung des Vorgangs, welcher zu folgendem KZustand führte.
		KZustand	Siehe KZustand.
	ITeilknoten[ ]		Übertragung aller vorhandenen ITeilknoten als Array mit festen Typen: - Anzahl:UBYTE folgender ITeilknotenstrukturen - relativer Path = Teilknotennummer

IstVektor			
METHOD	**Name**		**Beschreibung**
		Vorgangskennung	Vorgangskennung des Systemteils welcher die Kontrolle über den aktuell eingestellten TKZustand ausübt.
		TKZustand	Zu dieser Zeit eingestellter Teilknotenzustand. Siehe TKZustand.
	ISondereingriff		Zeigt die aktuell eingestellte Einsatzfahrzeugroute.
		Vorgangskennung	Vorgangskennung des Systemteils welcher die Kontrolle über den aktuell eingestellten Sondereingriff ausübt.
		SondereingriffNr	Siehe SonderEingriffNr.
	Modifikations[0...15]		Modifikationen des Signalprogramms. Hier kann jede von IModEinAus abgeleitete Klasse stehen, dies sind z. Zt. Objekt IVAEinAus, IOepnvEinAus, IProjEinAus. Übertragung als Array mit variablen Typen: - zuerst ein UBYTE Anzahl folgender IModEinAus Daten - RefLen, Länge der Referenz - OdgMember Otype - relativer Path = ProjModNr nur falls IProjEinAus folgt - Datenlänge - Daten einer von IModEinAus abgeleiteten Klasse
		Vorgangskennung	Vorgangskennung des Systemteils welcher die Kontrolle über den aktuell eingestellten Zustand der Modifikation ausübt.
		Zustand	Zustand der Modifikation

Festlegung zur Kennung Sammelstörung:

Das Sammelstörungsbyte hat fest definierte Werte für die verschiedenen Fehlerkategorien, so dass bei mehreren gleichzeitig vorliegenden Fehlerarten eine Priorisierung vorgenommen werden muss, d.h. ein Fehler mit einer höheren Priorität übersteuert einen Fehler niedriger Priorität. Die Sammelstörung wird gesetzt wie folgt:

- Störung ohne Abschaltung (1) (Priorität 2):
  - sekundäre Lampenfehler (ohne Abschaltung)
  - sonstige Signalsicherungs-Alarme (Sisi-Alarme) ohne Abschaltung
- Störung mit Abschaltung der gesamten Anlage (2) (Priorität 4):
  - Netzausfall (nur möglich bei USV)
  - alle Sisi-Störabschaltungen (z.B. primäre Lampenfehler)
  - Abschaltung wg. Umlaufkontrolle, gravierende interne Fehler (z.B. nicht auflösbare Feindlichkeit im Signalplan) welche zu einer Abschaltung führen.
- Störung mit Teilabschaltung der Anlage (3) (Priorität 3):
  - Abschaltung von Teilknoten durch Sisi, aber mindestens 1 Teilknoten läuft noch
  - Abschaltung eines Teilknotens wegen einem internen Fehler
- Interne Störung ohne Abschaltung (4) (Priorität 1):
  - Kommunikationsstörungen
  - Detektorstörung
  - Störung OEV – Empfang
  - Ortsbetrieb / Festzeitbetrieb als Rückfallebene (z.B. VA abgeschaltet wg. Fehler, Umlaufkontrolle)
  - ein wichtiger Systemprozess hat einen Fehler (z.B. (Teil-)Prozess reagiert nicht mehr)
  - Zeitquelle gestört

3.4.20 Objekt Gerätestatus

Neben dem Istvektor gibt es einen Gerätestatus pro Lichtsignalsteuergerät. Dieser ist abfragbar, wird aber nicht in das Betriebszustandsarchiv geschrieben, da er relativ umfangreich ist, und das sich oft ändernde Betriebszustandsarchiv unnötig vergrößern würde. Beim Auftreten von Störungen generiert das Lichtsignalsteuergerät entsprechende Meldungen im Meldungsarchiv.

Geraetestatus (1:236)

Geraetestatus		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	Zeitquelle: ui1	Gibt die aktuelle Quelle der Gerätezeit an
	NotAus: bool	Gibt an, ob ein ggf. vorhandener NOTAUS Schalter betätigt ist
	TuerAuf: bool	TuerAuf = true bedeutet: Der Türschließkontakt meldet: Mindestens eine Tür des Gerätes ist offen. Wenn kein Türschließkontakt vorhanden ist, ist TuerAuf = false
	NetzSpannungOk: bool	Gibt an, ob die für den vollen Gerätebetrieb nötige Netzspannung anliegt
	Gestörte Detektoren	Liste der gestörten Detektoren
	Gestörte Lampen	Liste der gestörten Lampen
	PersistenzSpeicherOk: bool	Gibt an, ob der gesamte Persistenzspeicher konsistent ist. Dieses Flag wird nach Netz Ein oder häufiger vom Gerät gesetzt

3.4.21 Objekt DeviceState (Gerätestatus mit Zeitstempel)

Das Objekt DeviceState liefert den aktuellen Gerätestatus mit Zeitstempel und ist eine Erweiterung des Objektes Gerätestatus (1:236).

Dieses Objekt ist direkt abfragbar und ist auch als die Meldung DeviceStateMsg (1:60321) in das Status-Archiv geschrieben.

DeviceState (1:237)

DeviceState		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	RetCode	OK: folgende Parameter richtig gelesen
	When	Zeitstempel des Gerätes, wann es diesen Zustand erreicht (zuletzt geändert) hat.
	TimeSource	Gibt die aktuelle Quelle der Gerätezeit an

DeviceState		
METHOD	**Name**	**Beschreibung**
	EmerOff	Gibt an, ob ein ggf. vorhandener NOTAUS Schalter betätigt ist.
	DoorOpen	TuerAuf = true bedeutet: Der Türschließkontakt meldet: Mindestens eine Tür des Gerätes ist offen. Wenn kein Türschließkontakt vorhanden ist, ist TuerAuf = false.
	PowerSupplyVoltageOk	Gibt an, ob die für den vollen Gerätebetrieb nötige Netzspannung anliegt.
	MalfunctioningDetectors	Liste der gestörten Detektoren.
	FaultLamps	Liste der gestörten Lampen.
	PersistenceStorageOk	Gibt an, ob der gesamte Persistenzspeicher konsistent ist. Dieses Flag wird nach Netz Ein oder häufiger vom Gerät gesetzt.
	PTReceiver	Gibt an, ob der ÖV-Empfänger funktioniert.
	Synchron	Gibt an, ob das Gerät synchron zur Referenzzeit ist.
	CentralCommandAccepted	Gibt an, ob das Gerät aktuell auf Zentralen-Schaltbefehle reagieren würde.
	UPS	Gibt an, ob die USV im Gerät installiert ist und funktioniert.
	SyncTimeSource	Gibt an ob mindestens eine der konfigurierten Zeitquellen zur Zeitsynchronisation verfügbar ist.
	RSUState	Gibt den Zustand der RSU an.
	Maintenance	Gibt an, ob sich das Gerät im Wartungszustand befindet.

3.5 Meldungen und Messwerte

In Archiven der Lichtsignalsteuergeräte werden ausgewählte Betriebsdaten gesammelt. In jedem Gerät existieren mehrere Archive. Welche Daten in welchem Archiv gespeichert werden wird durch Aufträge der Zentrale festgelegt. Pro Archiv sind bis zu 256 verschiedene Aufträge möglich. OCIT-Outstations vereinigt die bisher getrennten Messwert- und Meldungsarchive unter einer gemeinsamen Schnittstelle. Die Datenstrukturen und die definierten Funktionen der Schnittstelle sind für Meldungen und Messwerte strukturell gleich.

Die Daten aus den Archiven können von der Zentrale oder über Tools am Systemzugang ausgelesen werden. Dazu kann die Zentrale von Gerät archivierte Daten die an bestimmten Positionen stehen oder Daten die zu bestimmten Zeiten erfasst wurden anfordern. Im Normalbetrieb werden die archivierten Daten von der Zentrale beim Eintreten bestimmter Ereignisse abgeholt. Beim Eintritt eines solchen Ereignisses sendet das Gerät ein Event-Telegramm (enthält nicht die Daten) an die Zentrale, die daraufhin einzelne oder mehrerer Daten aus den Archiven anfordern kann. Event-Telegramme können ausgelöst werden:

- bei Erreichen eines eingestellten Füllgrads des Archivs,
- beim Eintragen bestimmter variabler Werte,
- bei Änderung der Zieladresse für die Event-Telegramme.

Die Archive der Geräte können während des Betriebs über die Zentrale parametriert werden. Festgelegt werden können: Größe, Art der Aufträge, Ereignisse die zu Event-Telegrammen führen, Erfassung von Daten Anhalten und Freigeben, Reset.

Eine ausführliche Beschreibung der Handhabung von Meldungen und Messwerten findet sich im Dokument OCIT-O-Basis.

Die für Lichtsignalsteuergeräte definierten Archive sind in Pkt 3.5.6 beschrieben.

3.5.1 Objekttypen und Klassenübersicht

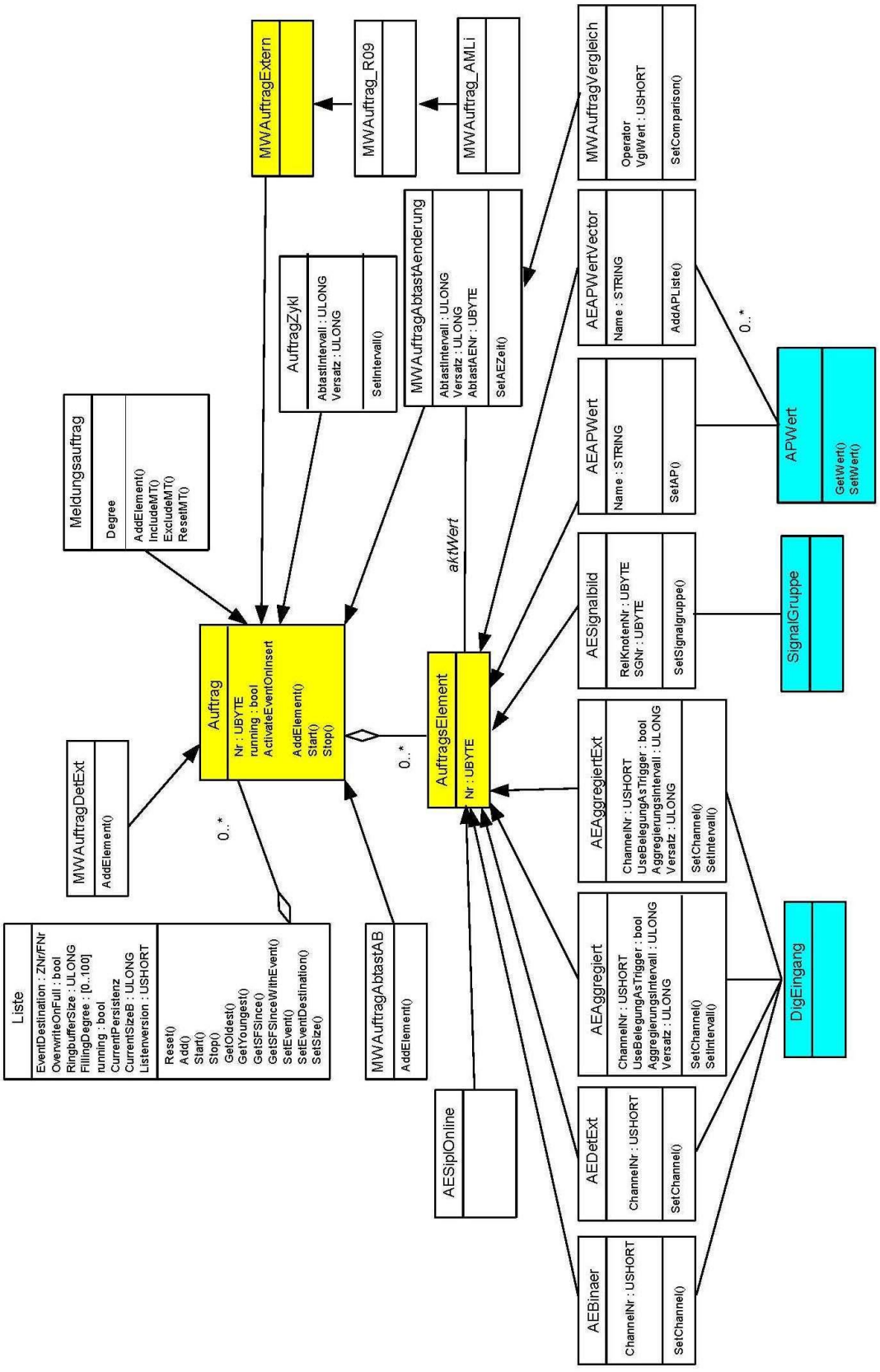


Abbildung 13: Meldungen und Messwerte: Schema der Objekttypen und Klassen (Auswahl)

Member = 1: OType

OType	Name	Pfad (ab Lichtsignalsteuergerät)
403	AuftragZykl	Liste(UBYTE)/Auftragsnummer(UBYTE)
406	MWAuftragAbtastAB	Liste(UBYTE)/Auftragsnummer(UBYTE)
407	MWAuftragAbtastAenderung	Liste(UBYTE)/Auftragsnummer(UBYTE)
408	MWAuftragVergleich	Liste(UBYTE)/Auftragsnummer(UBYTE)
409	MWAuftragExtern	Liste(UBYTE)/Auftragsnummer(UBYTE)
410	MWAuftragR09	Liste(UBYTE)/Auftragsnummer(UBYTE)
411	MWAuftragAMLi	Liste(UBYTE)/Auftragsnummer(UBYTE)
412	MWAuftragDetExt	Liste(UBYTE)/Auftragsnummer(UBYTE)
431	AEBinaer	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
432	AEAggregiert	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
433	AESignalBild	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
434	AEAPWert	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
435	AEDetExt	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
436	AEAggregiertExt	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
437	AEAPWertVektor	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
438	AESiplOnline	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
439	AEDigAusgang	Liste(UBYTE)/Auftragsnummer(UBYTE)/ AENr(UBYTE)
500	DigEingang	ChannelNr(USHORT)
501	SignalGruppe	RelKnotenNr(UBYTE)/SignalgruppenNr (UBYTE)
502	SignalGeber	RelKnotenNr(UBYTE)/DigAusgangNr (USHORT)
503	SignalKammer	RelKnotenNr(UBYTE)/SignalKammerNr (USHORT)
504	DigAusgang	RelKnotenNr(UBYTE)/DigAusgangNr (USHORT)
505	APWert	Name(STRING)
506	APWertUShort	Name(STRING)

OType	Name	Pfad (ab Lichtsignalsteuergerät)
507	APWertLong	Name(STRING)
508	APWertBlock	Name(STRING)
510	APWertRk	Name(STRING)/ RelKnotenNr(UBYTE)
511	APWertRkUShort	Name(STRING)/ RelKnotenNr(UBYTE)
512	APWertRkLong	Name(STRING)/ RelKnotenNr(UBYTE)
513	APWertRkBlock	Name(STRING)/ RelKnotenNr(UBYTE)
515	APWertGroup	Name(STRING)
516	APWertGroupRk	Name(STRING)/ RelKnotenNr(UBYTE)

Alle Objekte mit Ausnahme des Events unterstützen die Standardfunktion ‚Get'. Sie unterstützen nicht die Funktion ‚Set'. Die zurückgelieferten Parameter werden in der XML-Datei genauer beschrieben.

3.5.2 Messwertaufträge für Lichtsignalanlagen

Siehe auch Kapitel 5. „Abläufe Meldung und Messwerte" im Dokument OCIT-O-Basis.

3.5.2.1 Zyklisch abgefragter Auftrag

Der zyklisch abgefragte Auftrag (AuftragZykl) trägt die Auftragselemente zyklisch ein. Die Zeitpunkte TZykl und Versatz werden im Sekunden-Maßstab eingetragen, sie beziehen sich auf das eingestellte Rückrechenverfahren (siehe Pkt. 2.5)

AuftragZykl ist abgeleitet von Auftrag. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

AuftragZykl (1:403)

AuftragZykl		
METHOD	**Name**	**Beschreibung**
120, 121, 122	**AddElement, Start, Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte" Dokument OCIT-O-Basis.
130	**SetZyklus**	Setzt die Zykluszeit und den Versatz im Sekunden-raster.
	Eingabeparameter	
	Abtastintervall: ULONG	Zykluszeit in 10 Millisekunden-Einheiten.

AuftragZykl		
METHOD	**Name**	**Beschreibung**
	Versatz : ULONG	Signalzeiten Versatz gegenüber den Standard „OCIT-Outstations-Rückrechenverfahren“ (Pkt. 2.5) in 10 Millisekunden-Einheiten. Der Versatz wird MOD Abtastintervall gerechnet.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Zyklus erfolgreich gesetzt wurde. CYCLE_TOO_SHORT: Die Zykluszeit ist zu kurz.
	MinIntervall: ULONG	Zykluszeit >= übergebene Abtastintervall, in dem das Gerät abtasten kann.
	ListenversionAlt, ListenversionNeu	Listenversion vor SetZyklus Listenversion nach SetZyklus

3.5.2.2 Auftrag bei Abtaständerungen

Auftrag bei Abtaständerungen. Dieser Auftrag betrachtet den Wert eines Auftragselements (eine Prozessvariable) im angegebenen Intervall. Ändert sich dieser Wert, so schreibt der Auftrag einen Sekundenframe.

MWAuftragAbtastAenderung ist abgeleitet von Auftrag. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

MWAuftragAbtastAenderung (1:407)

MWAuftragAbtastAenderung		
METHOD	**Name**	**Beschreibung**
119, 120, 121, 122	**ActivateEvent,** **AddElement,** **Start,** **Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O-Basis.
130	**SetAEZeit**	Setzt die Zykluszeit und den Versatz im Sekundenraster.
	Eingabeparameter	
	AbtastAENr: UBYTE	Nummer des Auftragselements, das abgetastet wird.
	AbtastIntervall: ULONG	Zeitraster in 10ms, in denen abgetastet wird.
	Versatz: ULONG	Versatz gegenüber den Standard-OCIT-Outstations-Rückrechenverfahren in 10 Millisekunden-Einheiten. Der Versatz wird MOD Abtastintervall gerechnet.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Zyklus erfolgreich gesetzt wurde. CYCLE_TOO_SHORT: Die Zykluszeit ist zu kurz.
	MinIntervall: ULONG	Zykluszeit >= übergebene AbtastIntervall, in dem das Gerät abtasten kann.
	ListenversionAlt, ListenversionNeu	Listenversion vor SetAEZeit Listenversion nach SetAEZeit
	ListenversionAlt, ListenversionNeu	Listenversion vor ActivateEvent Listenversion nach ActivateEvent

Hinweis: Wenn eine Liste mit einem Auftrag MWAuftragAbtastAenderung ohne Konfiguration des Intervalls (SetAEZeit) gestartet wird, dann hat der Auftrag das Intervall 0 und liefert damit keine Einträge in die Liste.

3.5.2.3 Auftrag Abtaständerungen mit Wertevergleich

Dieses Verfahren ist eine Spezialisierung der Prüfung auf Werteänderung. Das Verfahren überträgt nur wenn a) eine Werteänderung stattgefunden hat und b) die eingegebene Bedingung erfüllt ist.

MWAuftragVergleich ist abgeleitet von MWAuftragAbtastAenderung. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

MWAuftragVergleich (1:408)

MWAuftragVergleich		
METHOD	**Name**	**Beschreibung**
119, 120, 121, 122	**ActivateEvent,** **AddElement,** **Start,** **Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O-Basis.
130	**SetAEZeit**	Setzt die Zykluszeit und den Versatz im Sekundenraster.
	Eingabeparameter	
	AbtastAENr: UBYTE	Nummer des Auftragselements, das abgetastet wird.
	AbtastIntervall: ULONG	Zeitraster in 10ms, in denen abgetastet wird.
	Versatz: ULONG	Versatz gegenüber den Standard-OCIT-Outstations-Rückrechenverfahren in 10 Millisekunden-Einheiten. Der Versatz wird MOD Abtastintervall gerechnet.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Zyklus erfolgreich gesetzt wurde. CYCLE_TOO_SHORT: Die Zykluszeit ist zu kurz.
	MinIntervall: ULONG	Zykluszeit >= übergebene AbtastIntervall, in dem das Gerät abtasten kann.
	ListenversionAlt, ListenversionNeu	Listenversion vor SetAEZeit Listenversion nach SetAEZeit
150	**SetComparison**	Setzt die Zykluszeit und den Versatz im Sekundenraster.
	Eingabeparameter	

MWAuftragVergleich		
METHOD	**Name**	**Beschreibung**
	Operator: CHAR	‚>' der aktuelle Wert ist größer als der Vergleichswert ‚<' der aktuelle Wert ist kleiner als der Vergleichswert '=' der aktuelle Wert ist gleich dem Vergleichswert '!' der aktuelle Wert ist ungleich dem Vergleichswert ‚H' der durch den HS-Operator (s. u.) gebildete Wert ist größer als der Vergleichswert
	Vergleichswert: LONG	Vergleichswert für die Operation.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Zyklus erfolgreich gesetzt wurde. UNKNOWN_OP: Nicht unterstützter Operator
	ListenversionAlt, ListenversionNeu	Listenversion vor SetComparison Listenversion nach SetComparison

Der HS-Operator bildet sich wie folgt:

Es wird ein Akkumulator vom Typ LONG benötigt. Bei Eintrag eines Sekundenframes wird der Akkumulator auf 0 gesetzt. Pro Abtastintervall wird die Differenz des aktuellen Werts vom letzten eingetragenen Wert gebildet (inkl. Vorzeichen, also nicht der Betrag!) und auf den Akkumulator addiert. Wenn der Betrag des Akkumulatorwerts größer als der Vergleichswert ist, löst die Comparison aus.

Der HS Operator ist nur für Werte sinnvoll, bei denen (stetige) Werte von einem Rauschen überlagert sind und bei denen signifikante Änderungen interessieren.

3.5.2.4 Auftrag für asynchrone Prozessvariable

Das Objekt MWAuftragExtern ist nur dann notwendig, wenn Ereignisse von Zusatzeinrichtungen erfasst werden müssen wie z. B. von einer OEPNV-Einrichtung.

Auftrag für Prozessvariable die asynchron durch externe Ereignisse entstehen.

MWAuftragExtern (1:409) ist abgeleitet von Auftrag. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

MWAuftragExtern (1:409)

MWAuftragExtern		
METHOD	**Name**	**Beschreibung**
119, 120, 121, 122	**ActivateEvent,** **AddElement,** **Start,** **Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O-Basis.

3.5.2.5 Auftrag für Einzelschleifen-Erfassung

Es ist sinnvoll, die digitalen Daten einer Einzelschleife komprimiert im Messwertarchiv abzulegen. Normalerweise wird bei einem Messwertauftrag die Wertänderung gespeichert, wobei der neue Wert in einem Parameter hinter dem Subsekundeneintrag abgelegt wird.

Da bei binären Signalen der neue Wert sich in einem Bit darstellen lässt und solche Wertänderungen sehr häufig auftreten, wird in diesem Sonderfall das Parameterbyte eingespart und der neue Zustand in **Bit 2^7 des Subsekundeneintrags im Auftragsframe gespeichert**[6]. Wenn also das Bit $2^7 = 0$ ist, bedeutet das einen Wechsel von „1“ nach „0“; ist es 1 impliziert es einen Wechsel von „0“ nach „1“.

Beim Start dieses Auftrags wird der Anfangszustand dargestellt, indem zum Zeitpunkt des Auftragsstarts ein Wechsel des Signals vorgetäuscht wird. Beginnt das Signal mit 0 wird also ein Wechsel von 1 nach 0 eingetragen; beginnt das Signal mit 1, wird ein Wechsel von 0 nach 1 zu Beginn gespeichert.

[6] Achtung: Dies ist ein Sonderfall mit einer unterschiedlichen Behandlung des Auftragselements AE-Binaer! Bei anderen Aufträgen, z.B. AuftragZykl, wird der digitale Zustand in einem zusätzlichen Byte gespeichert!

MWAuftragAbtastAB ist abgeleitet von Auftrag. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

MWAuftragAbtastAB (1:406)

MWAuftragAbtastAB		
METHOD	**Name**	**Beschreibung**
120	**AddElement**	AddElement darf bei dem Auftragstyp MWAuftragAbtastAB nur einmal (!) mit dem Typ AEBinaer aufgerufen werden (und sonst mit keinem Typ). Sonst liefert AddElement den Fehler NOT_POSSIBLE zurück!
121, 122	**ActivateEvent, Start, Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O-Basis.

3.5.2.6 Auftrag für erweiterte Detektorwerte

Einige Schleifendetektoren liefern bei Konfiguration als Doppelschleife neben der allgemein üblichen Belegungs- und Lückeninformation noch weitere Werte, welche in moderneren Steuerverfahren verwendet werden können. In der Regel sind dies Geschwindigkeit, Fahrzeugart, Fahrzeuglänge und die Fahrdauer von der ersten bis zur zweiten Schleife. Um diese Informationen in der Zentrale nutzbar zu machen, wird der nachstehende Auftrag definiert. Wenn der Auftrag gesetzt wird, werden die Datensätze für alle Fahrzeuge, welche den betreffenden Detektor überfahren, abgespeichert.

Hinweis: Da diese Daten völlig asynchron entstehen, soll hier sinnvollerweise auch nur ein Auftragselement pro Auftrag definiert werden.

Für die als Doppelschleife konfigurierten Detektoren mit erweiterter Abfragemöglichkeit wird ein eigenes Auftragselement eingeführt. Somit sind sie über die Methode GetInstanceInfo() des Systemobjekts Lichtsignalsteuergerät zugänglich. Weiterhin kann nur so im Auftragselement die Zulässigkeit einer zugewiesenen Kanalnummer überprüft und gegebenenfalls abgelehnt werden, ohne den zugehörigen Auftrag zu kennen.

MWAuftragDetExt ist abgeleitet von Auftrag. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

MWAuftragDetExt (1:412)

MWAuftragDetExt		
METHOD	**Name**	**Beschreibung**
120	**AddElement**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O-Basis. Hier ist nur das Auf-

MWAuftragDetExt		
METHOD	**Name**	**Beschreibung**
		tragselement AEDetExt zulässig.
119, 121, 122	**ActivateEvent,** **Start,** **Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O-Basis.

Auftragselement für Doppelschleifen mit Zusatzinformationen (AEDetExt) siehe Pkt. 3.5.3.5

Auftragselement für erweiterte aggregierte Det.-Werte (AEAggregiertExt), siehe Pkt.3.5.3.2

3.5.2.7 Auftrag für R09-Telegramme

Der R09-Auftrag hat anders als die anderen Messwert-Aufträge keine Auftragselemente. Stattdessen ist mit dem R09-Auftrag unmittelbar eine Struktur verbunden. Der R09-Auftrag generiert immer Frames des Typs MWAuftragFrameR09. Wenn der Auftrag gesetzt wird, werden alle für dieses Lichtsignalsteuergerät relevanten R09-Telegramme abgespeichert. Irrelevante Telegramme, die trotzdem empfangen wurden, werden nicht gespeichert.

MWAuftragR09 ist abgeleitet von MWAuftragExtern. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

MWAuftragR09 (1:410)

MWAuftragR09		
METHOD	**Name**	**Beschreibung**
119, 120, 121, 122	**ActivateEvent,** **AddElement,** **Start,** **Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O-Basis.

Struktur des dynamischen Datensatzes

Name	**Kurzbez.**	**Datentyp**	**Wertebereich**	**Bemerkungen**
Tag (Erstelldatum)	TT	UBYTE	1...31	Erstelldatum / Uhrzeit
Monat (Erstelldatum)	MO	UBYTE	1...12	Erstelldatum / Uhrzeit

Name	Kurz-bez.	Datentyp	Wertebereich	Bemerkungen
Jahr (Erstelldatum)	JJ	UBYTE	0...99	Erstelldatum / Uhrzeit
Stunde (Erstelldatum)	HH	UBYTE	0...23	Erstelldatum / Uhrzeit
Minute (Erstelldatum)	MM	UBYTE	0...59	Erstelldatum / Uhrzeit
Sekunde (Erstelldatum)	SS	UBYTE	0...59	Erstelldatum / Uhrzeit
Meldepunktnummer	MPN	LONG	1 - 2^24	5 Zeichen im Telegramm
Liniennummer	LLL	USHORT	0 - 999	3 Zeichen im Telegramm
Kursnummer	KK	UBYTE	0 - 99	2 Zeichen im Telegramm
Routennummer	RRR	USHORT	0 - 999	3 Zeichen im Telegramm
Priorität	P	UBYTE	0 - 7	1 Zeichen im Telegramm
Zuglänge	Z	UBYTE	0 - 7	1 Zeichen im Telegramm
Richtung Hand	H	UBYTE	0 - 3	1 Zeichen im Telegramm Manuelle Anforderung durch den Fahrer (z.B. mittels Schlüsselschalter an der Haltestelle).
Fahrplanabw (Min+Sek)	FAHRP	SHORT (signed short)	-3599 bis 3599	„Fahrplanlage“ Abweichung vom Fahrplan in Sekunden.

Anmerkung: Die Felder TT, MO, JJ, HH, MM, SS beschreiben das Datensatzerstelldatum der externen Einheit, bei dem die lokale Zeit als Einheit verwendet wird.

3.5.2.8 Auftrag für erweiterte R09-Telegramme

Der Auftrag MWAuftragAMLi (1:411) für erweiterte R09-Telegramme ist gegenüber dem Auftrag für R09-Telegramme von der Auswahl her gleich und liefert nur einen erweiterten Datensatz (z.B.: GNA, GNE, TX...) zurück. Die Funktionen entsprechen exakt denen des Auftrags für R09 -Telegramms und werden in 3.5.2.7 aufgeführt.

MWAuftragAMLi ist abgeleitet von MWAuftragR09. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

Die erweiterte Datenstruktur sieht folgendermaßen aus:

Name	Kurz-bez.	Datentyp	Werte-bereich	Bemerkungen
Tag (Erstelldatum)	TT	UBYTE	1…31	Erstelldatum / Uhrzeit
Monat (Erstelldatum)	MO	UBYTE	1...12	Erstelldatum / Uhrzeit
Jahr (Erstelldatum)	JJ	UBYTE	0...99	Erstelldatum / Uhrzeit
Stunde (Erstelldatum)	HH	UBYTE	0...23	Erstelldatum / Uhrzeit
Minute (Erstelldatum)	MM	UBYTE	0...59	Erstelldatum / Uhrzeit
Sekunde (Erstelldatum)	SS	UBYTE	0...59	Erstelldatum / Uhrzeit
Meldepunktnummer	MPN	LONG	1 - 2^24	5 Zeichen im Telegramm
Liniennummer	LLL	USHORT	0 - 999	3 Zeichen im Telegramm
Kursnummer	KK	UBYTE	0 - 99	2 Zeichen im Telegramm
Routennummer	RRR	USHORT	0 - 999	3 Zeichen im Telegramm
Priorität	P	UBYTE	0 - 7	1 Zeichen im Telegramm
Zuglänge	Z	UBYTE	0 - 7	1 Zeichen im Telegramm
Richtung Hand	H	UBYTE	0 - 3	1 Zeichen im Telegramm; manuelle Anforderung durch den Fahrer (z.B. mittels Schlüsselschalter an der Haltestelle)
Fahrplanabw (Sek)	FAHRP	SHORT (signed short)	-3599 bis 3599	„Fahrplanlage“ Abweichung vom Fahrplan wie im empfangenen R09 Telegramm.
Relative Knotennummer	RELKN	UBYTE	0...255	Die Nummer des relativen Knotens im Lichtsignalsteuergerät, die ausgewertet wird.
OeV-Modifikation durch Zentrale aktiv?	OEVAKT	UBYTE	0 - 1	0: OeV Modfikation inaktiv 1: OeV Modifikation aktiv 255: Dieser Wert ist nicht gesetzt
TX bei Meldung	TX	UBYTE	1 - 255	Umlaufsekunde zum Eintreffen des R09 Telegrams (Beachte Anmer-

Name	Kurz-bez.	Datentyp	Werte-bereich	Bemerkungen
				kungen am Ende der Tabelle!)
Signalplan	SP	UBYTE	0 - 32	Nummer des Signalplans zum Eintreffen des R09 Telegramms
Laufende Phase	PH	UBYTE	0 - 255 ?	0: Phase im Verfahren nicht definiert (z.B. VSPlus) 1...255: aktuelle Phasennummer zum Eintreffen des R09 Telegramms
Gewünschte Phase	UE	UBYTE	0 - 255	0: Es ist kein Phasenübergang aktiv bzw. eine Phase ist im Verfahren nicht definiert (VSPlus) 1...255: Es ist ein Phasenübergang aktiv von Phase PH nach Phase UE
Fahrzeit (Bei Abmeldung die echte Fahrzeit von Anmeldung bis Abmeldung. Bei Anmeldung die theoretische, berechnete, Fahrzeit von Anmeldung bis Abmeldung.)	TWF	UBYTE	0 - 255	0: keine Fahrzeit vorhanden 1...255: Fahrzeit
Bei Abmeldung: Grünanfang der ÖV-Signalgruppe	GNA	UBYTE	0 - 255	Bezogen auf TX (siehe auch Anmerkungen)
Bei Abmeldung: Grünende der ÖV-Signalgruppe	GNE	UBYTE	0 - 255	Bezogen auf TX (siehe auch Anmerkungen)

Anmerkungen: Im Gegensatz zu den OCIT-O konformen 0, 1 Sekunden Zeitschaltwerten von TX = 0 bis TX = TU - 1, gibt es im AMLI-Datensatz aus historischen Gründen nur ganze Sekundenschritte von TX = 1 bis TX = TU. Die OCIT- O Zeitschaltwerte TX = n * 0,1 Sekunden müssen daher in AMLI konforme Werte umgerechnet werden: OCIT-O TX /10 (ohne Rest) + 1.

Beispiele für eine OCIT-O Umlaufzeit von 30,5 Sekunden:

OCIT-O:	TX = 0	TX = 10	TX = 52	TX-1 = 304
AMLI:	TX = 1	TX = 2	TX = 6	TU = 31

Die TX-Werte in AMLI entsprechen damit der laufenden Sekunde im Umlauf. TU ist die letzte Sekunde des Umlaufs, darauf folgt wieder die erste Sekunde.

Folgende Kennungen werden nur bei Abmeldung gesetzt, bei Anmeldungen sind GNA und GNE immer 0:

Kennung im ÖV-Speicher		**Bedeutung**
Grün-Anfang	**Grün-Ende**	
1...253	1...253	Eintrag von TX für Grün der beeinflussten Signalgruppe. Falls die Abmeldung kurz nach der Umschaltung auf Rot (innerhalb der Zeit des VA-Parameters „festgelegte Rotzeit“[7]) erfolgt, d. h. dass der Bus bei Gelb / Rot noch gefahren ist, wird das für Grünende bei GNE eingetragen. Erfolgt die Abmeldung später als der parametrierte Wert wird GNE auf 0 gesetzt.
0	0	Die Abmeldung erfolgte nach GNE plus festgelegter Rotzeit der Signalgruppe
0	255	Abmeldung erfolgte im Auszustand des Steuergeräts
254	254	Die SG hatte bei Anmeldung und 15 s [8] nach Abmeldung Grün. Die An- und Abmeldung des ÖV hatte keinen Einfluss auf die SG, da die SG im Zustand Dauergrün stand.
254	1...253	Die SG hatte zum Zeitpunkt der Anmeldung schon Grün und blieb bis zur Abmeldung in diesem Zustand. GNE entspricht echtem GNE dieser SG nach Abmeldung des ÖV.
254	0	Die SG hatte zum Zeitpunkt der Anmeldung schon Grün und blieb bis zur Abmeldung in diesem Zustand. Der Datensatz wurde vor GNE der Signalgruppe in den ÖV-Speicher geschrieben, da eine weitere Abmeldung durch einen Folgebus eintraf, bevor die n Sekunden bis GNE gleich 254 abgelaufen waren(siehe auch nächste Kombination).
1...253	254	Grün-Anfang der SG nach der Anmeldung. Die SG hatte 15s nach der Abmeldung immer noch Grün, d.h. GNE der SG wurde nicht durch ÖV-Anforderung beeinflusst. Der Wert von GNA entspricht echtem GNA

[7] „Festgelegte Rotzeit“ ist ein Parameter des VA-Verfahrens oder herstellerspezifisch versorgt.

[8] Der Wert 15 Sekunden kann herstellerspezifisch / verfahrensspezifisch abweichen.

Kennung im ÖV-Speicher		Bedeutung
Grün-Anfang	Grün-Ende	
		der SG nach Anmeldung.
1...253	0	Grün-Anfang der SG nach der Anmeldung. Die Abmeldung erfolgte im Grün, ein Folgebus hat noch nicht abgemeldet. Der Datensatz wurde vor GNE der Signalgruppe in den ÖV-Speicher geschrieben, da eine weitere Abmeldung durch einen Folgebus eintraf, bevor die n Sekunden bis GNE gleich 254 abgelaufen waren. Der Wert von GNA entspricht echtem GNA der SG nach Anmeldung.

3.5.3 Auftragselemente

Aufträge setzen sich im Normalfall aus Auftragselementen zusammen. Wenn ein Auftrag dynamische Daten in einen Sekundenframe schreibt, werden alle Auftragselemente ausgelesen und die Daten direkt hintereinander in den Auftragsframe gepackt.

Wie der Auftrag ist auch das Auftragselement eine „virtuelle Basisklasse“ (mit der virtuellen Methode 150 „GetTriggerValue), d. h. es gibt eine Reihe von speziellen Auftragselementen, die instanziert werden können, aber nicht das Auftragselement selbst. Jeder Auftragselementtyp verweist auf einen bestimmten Typ von Datenquelle, also z.B. auf Digitale Eingänge oder auf Signalgruppen usw. Zudem ist pro Auftragselement-Typ definiert, welche Daten der Datenquelle bei diesem Auftragselement-Typ dynamisch abgelegt werden.

- Ein Auftragselement ist ein OBJTYPE und hat den Pfad: Liste()/Auftrag()/AuftragsElement()
- Das Auftragselement definiert die Struktur der Daten im Auftragsframe.
- Im Auftragselement gibt es eine Referenz auf eine DOMAIN welche den Aufbau der Daten im Auftragsframe beschreibt. (Sinn: Herstellerspezifische Auftragselemente können von fremder Zentrale dekodiert werden.)
- Jedes Auftragselement verweist auf eine Datenquelle. Der Verweis auf die Datenquelle ist typisiert, d.h. ein Auftragselement eines Typs X verweist immer auf eine Datenquelle des gleichen Typs Y.
- Jedes Auftragselement liefert auf Anfrage von MWAuftragAbtastungAenderung einen skalaren Wert. Bei strukturierten Auftragselementen definiert das Auftragselement welcher Wert geliefert wird.

3.5.3.1 Auftragselement für binäre Eingänge

Binäre Eingänge (DigEingang, 1:500), wie z.B. Detektoreingänge oder auch Taster werden über das Auftragselement für binäre Eingänge erfasst. Das Auftragselement wird in zwei unterschiedlichen Auftragstypen eingesetzt:

- Binäre Eingänge, die sich „selten“ ändern, werden entweder als Auftragselement in einem beliebigen Auftrag verwendet oder als Trigger in einem (MWAuftragAbtastAenderung bzw. MWAuftragVergleich).
- Binäre Eingänge, die sehr häufig wechseln, wie z.B. die von Schleifendetektoren, werden mit MWAuftragAbtastAB behandelt, der die Daten komprimierter als der normale Auftrag einträgt. Der MWAuftragAbtastAB legt den AEBinaer-Eintrag selbst an.

AEBinaer (1:431)

AEBinaer		
METHOD	**Name**	**Beschreibung**
150	**GetTriggerValue**	liest den Wert, der für die Abtaständerung verwendet wird
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: Wert des binären Eingangs (0 oder 1)
151	**SetChannel**	Setzt die geräteweit eindeutige Channelnummer des binären Eingangs
	Eingabeparameter	
	Channel : DigEingang &	Referenz auf DigEingangsobjekt. ChannelNr ist die verwendete Kanalnummer.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn der Kanal nicht existent ist.

Anmerkung: Bei binären Eingängen liefert der aktuelle Triggerwert immer nur die Werte 0 oder 1.

3.5.3.2 Aggregierte Werte für binäre Eingänge

Wenn Schleifendetektoren als binäre Eingänge verwendet werden, ist es ggf. sinnvoll anstelle der Übertragung von Einzelwerten die Zählung und Belegungsgrad bereits im Gerät zu bilden. Die Zählung wird immer normiert in Fz/h (als USHORT) ausgegeben, der Belegungsgrad in % (als UBYTE).

Ein AEAggregiert hat ein Aggregierungsintervall. Es gibt zwei Fälle:

- Ist das Aggregierungsintervall = 0 werden Intervall und Versatz vom zugehörigen Auftrag übernommen.
- Ist das Aggregierungsintervall >0 wird in diesem Zyklus ein neues Aggregierungsintervall begonnen. Das Auftragselement schreibt immer die Werte des letzten Aggregierungsintervalls in den Sekundenframe. Nur sinnvoll als Element von MWAuftragAbtastÄnderung und MWAuftragAbtastVergleich.

Es werden immer alle beiden Werte (Zählung und Belegungsgrad) in den Frame geschrieben.

AEAggregiert ist abgeleitet von AuftragsElement. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

AEAggregiert (1:432)

AEAggregiert		
METHOD	**Name**	**Beschreibung**
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: Entweder Zählwert oder Belegungsgrad, je nach Wert des Attributs UseBelegungAsTrigger
151	**SetChannel**	Setzt die geräteweit eindeutige Channelnummer des binären Eingangs
	Eingabeparameter	
	Channel: DigEingang &	Referenz auf DigEingangsobjekt. ChannelNr ist die verwendete Kanalnummer.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn der Kanal nicht existent ist.
152	**SetIntervall**	Setzt das Aggregierungsintervall
	Eingabeparameter	
	AggregierungsIntervall: ULONG	Zeitraster in 10ms, in denen abgetastet wird.

AEAggregiert		
METHOD	**Name**	**Beschreibung**
	Versatz: ULONG	Versatz gegenüber den Standard-OCIT-Outstations-Rückrechenverfahren in 10 Millisekunden-Einheiten. Der Versatz wird MOD AggregierungsIntervall gerechnet.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Aggregierungsintervall erfolgreich gesetzt wurde. CYCLE_TOO_SHORT: Die Zykluszeit ist zu kurz.
	MinIntervall: ULONG	Kleinstmögliches Aggregierungsintervall.

3.5.3.3 Auftragselement für Anwenderprogrammwert

Das Auftragselement AEAPWert erfasst Anwenderprogrammwerte (AP-Werte, siehe Pkt. 3.5.4) vom Typ USHORT, ULONG oder BLOB, je nach angegebener AP-Wert Referenz.

AEAPWert ist abgeleitet von AuftragsElement. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

AEAPWert (1:434)

AEAPWert		
METHOD	**Name**	**Beschreibung**
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: Wert des APWerts, außer bei APWertBlock, hier wird ein Hashwert verwendet. Hinweis: Der Hashalgorithmus ist vom Geräteher-steller frei wählbar, der Hashwert darf jedoch nicht größer als 32 Bit sein.
153	**SetAP**	Setzt die Referenz auf den APWert.
	Eingabeparameter	
	APWert: ANYPATH	Referenz auf APWert, Pfad besteht aus: - Referenzlänge, Member, OType - Name des AP-Werts - weitere Pfadparameter, je nach angegebenen AP-Wert Typ
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn der APWert nicht existent ist.

3.5.3.4 Auftragselement blockweises Lesen von AP-Werten

Die Unterstützung von AEAPWertVektor ist in OCIT-O Lstg V2.0 optional.

AEAPWertVektor ist abgeleitet von AuftragsElement. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

Das Auftragselement AEAPWertVektor erfasst Anwenderprogrammwerte (AP-Werte, siehe Pkt. 3.5.4) vom Typ USHORT, ULONG oder BLOB, je nach angegebener AP Wert Referenz.

Das Auftragselement AEAPWertVektor erlaubt blockweises Lesen von AP-Werten. So können große Mengen von AP-Werten effizienter durch die Zentrale gelesen werden. Der AEAPWertVektor wird zunächst mit einer Liste von Referenzen auf APWerte initialisiert. Die Werte dieser APWerte werden dann im Block in die Liste geschrieben.

AEAPWertVektor (1:437)

AEAPWertVektor		
METHOD	**Name**	**Beschreibung**
156	**SetAPListe**	Initialisiert das Auftragselement mit Referenzen auf APWerte. Es gibt die Möglichkeit, der Methode ein Präfix mitzugeben, der den Paths aller Referenzen auf APWerte hinzugefügt wird. Die Maximale Anzahl der APWerte sind 65535 Werte.
	Eingabeparameter	
	Path: Prefix	Das Präfix wird allen Referenzen auf APWerte vorangestellt.
	Path[ ]: APWerteRefs	APWerteRefs.Anzahl APWerteRefs[ ].RefLen APWerteRefs[ ].Member APWerteRefs[ ].OType APWerteRefs[ ]… Pfadparameter je nach APWert
	Ausgabeparameter	

AEAPWertVektor		
METHOD	**Name**	**Beschreibung**
	RetCode	OK NOT_INACTIVE falls das Auftragselement in einem Auftrag ist, der schon gestartet ist. PARAM_INVALID falls einer oder mehrere der Referenzen auf APWerte ungültig sind. In diesem Fall wird keiner der referenzierten APWerte dem Auftragselement zugefügt. TOO_MANY falls das Steuergerät die Menge an APWerten nicht verarbeiten kann.
155	**GetAPListe**	Liest die mit SetAPListe gesetzte Liste der APWerte.
	Ausgabeparameter	
	Path: Prefix	Der Präfix, der allen Referenzen auf die APWerte vorangestellt wird.
	Path[ ]: APWerteRefs	APWerteRefs.Anzahl APWerteRefs[ ].RefLen APWerteRefs[ ].Member APWerteRefs[ ].OType APWerteRefs[ ]… Pfadparameter je nach APWert
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: Hashwert über alle Values der APWerte. Hinweis: Der Hashalgorithmus ist vom Gerätehersteller frei wählbar, der Hashwert darf jedoch nicht größer als 32 Bit sein.

3.5.3.5 Auftragselement für Detektoren mit Zusatzinformationen

AEDetExt ist abgeleitet von AEBinaer. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

AEDetExt (1:435)

AEDetExt		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Eingabeparameter	
		keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte.
	AuftragsElementNr	Nummer dieses Auftragselements in seinem Auftrag
	Channel	Referenz auf binären digitalen Eingang.
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Ausgabeparameter	
	RetCode	NOT_POSSIBLE, weil kein Wert geliefert werden kann.
151	**SetChannel**	Setzt die geräteweit eindeutige Channelnummer des binären Eingangs
	Eingabeparameter	
	Channel: DigEingang &	Referenz auf DigEingangsobjekt. ChannelNr ist die verwendete Kanalnummer.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn der Kanal nicht existent ist.

Struktur des Ergebnisframes (AEDetExtFrame):

Name	Kurz-bez.	Datentyp	Werte-bereich	Bemerkungen[9]
Belegung	Bel	USHORT	0...0xFFFE	Belegungsdauer der Messfläche in 10ms; 0 ... 655,34 s
Lücke	Luecke	USHORT	0...0xFFFE	Letzte Lücke in 10ms; 0 ... 655,34 s
Fahrdauer	FD	USHORT	1...0x7FFE	Fahrdauer von der ersten bis zur zweiten Messfläche in ms; 1 ms …32766 ms
Geschwindigkeit	GSW	UBYTE	0...0xFE	Gemessene Geschwindigkeit in km/h 0… 254 km/h
Fahrzeuglänge	FzgLen	UBYTE	1...0xFE	Länge des Fahrzeugs in 0,1m; 0,1...25,4 m
Fahrzeugart	FzgArt	UBYTE	0...0xFF	Art des Fahrzeugs (Klasse) 00h: Pkw 01h: Pkw + Anhänger 02h: Lkw 03h: Lkw + Anhänger 04h: Bus 05h: Sonstige 06h: Motorrad 07h: Lieferwagen 08h: Sattelkraftfahrzeug 09h…FFh: undefiniert

3.5.3.6 Auftragselement für erweiterte aggr. Det.-Werte

AEAggregiertExt ist abgeleitet von AEAggregiert. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

Um Geschwindigkeiten und Fahrzeugarten auch in einer aggregierten Form zu bekommen, wird ein erweitertes aggregiertes Auftragselement eingeführt. Dieses Auftragselement AEAggregiertExt ist eine Erweiterung des bestehenden Auftragselements AEAggregiert. Es hat auch die gleichen Methoden und wird mit zyklischen Aufträgen verwendet. Die eintreffenden Daten werden gemäß den acht Fahrzeug-

[9] Die angegebenen Wertebereiche bezeichnen die mit OCIT-O übertragbaren Informationen. Real übertragene Werte und ihre Genauigkeit sind vom eingesetzten Detektortyp abhängig, der in OCIT-O nicht vorgegeben wird. Geschwindigkeits-Detektoren arbeiten beispielsweise oft erst ab einer Mindestgeschwindigkeit von einigen km/h.

klassen (entsprechend TLS[10]) und „undefiniert“ verteilt, aus denen jeweils die mittlere Geschwindigkeit und ein Zählwert ermittelt werden.

Gibt es keine Werte für die mittlere Geschwindigkeit oder die Zählwerte, so wird der NULLVALUE eingetragen. Somit können auch Detektoren verwendet werden, die nur Teile dieser Informationen liefern.

AEAggregiertExt (1:436)

AEAggregiertExt		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Eingabeparameter	
		keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte.
	AuftragsElementNr	Nummer dieses Auftragselements in seinem Auftrag
	Channel	Referenz auf binären digitalen Eingang.
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: Entweder Zählwert oder Belegungsgrad, je nach Wert des Attributs UseBelegungAsTrigger
151	**SetChannel**	Setzt die geräteweit eindeutige Channelnummer des binären Eingangs
	Eingabeparameter	
	Channel : DigEingang &	Referenz auf DigEingangsobjekt. ChannelNr ist die verwendete Kanalnummer.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn der Kanal nicht existent ist.

[10] Technische Lieferbedingungen für Streckenstationen, Bundesanstalt für Straßenwesen, 2006

Der Datensatz AEAggregiert wird mit dem Frame AEAggregiertExt wie folgt erweitert:

USHORT	Zählwert in Fahrzeuge/h	alle Fahrzeuge im Intervall, wie AEAggregiert
UBYTE	Belegungsgrad in %	alle Fahrzeuge im Intervall, wie AEAggregiert
UBYTE	mittlere Geschwindigkeit Klasse 0 in km/h	0 ... 254; 255 NULLVALUE: kein gültiger Wert
USHORT	Zählwert Klasse 0	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert
UBYTE	mittlere Geschwindigkeit Klasse 1 in km/h	0 ... 254; 255 NULLVALUE: kein gültiger Wert
USHORT	Zählwert Klasse 1	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert
UBYTE	mittlere Geschwindigkeit Klasse 2 in km/h	0 ... 254; 255 NULLVALUE: kein gültiger Wert
USHORT	Zählwert Klasse 2	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert
UBYTE	mittlere Geschwindigkeit Klasse 3 in km/h	(0 ... 254; 255 NULLVALUE: kein gültiger Wert)
USHORT	Zählwert Klasse 3	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert
UBYTE	mittlere Geschwindigkeit Klasse 4 in km/h	0 ... 254; 255 NULLVALUE: kein gültiger Wert
USHORT	Zählwert Klasse 4	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert
UBYTE	mittlere Geschwindigkeit Klasse 5 in km/h	(0 ... 254; 255 NULLVALUE: kein gültiger Wert)
USHORT	Zählwert Klasse 5	(0 ... 65534; 65535 NULLVALUE: kein gültiger Wert)
UBYTE	mittlere Geschwindigkeit Klasse 6 in km/h	(0 ... 254; 255 NULLVALUE: kein gültiger Wert)
USHORT	Zählwert Klasse 6	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert
UBYTE	mittlere Geschwindigkeit Klasse 7 in km/h	0 ... 254; 255 NULLVALUE: kein gültiger Wert

USHORT	Zählwert Klasse 7	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert
UBYTE	mittlere Geschwindigkeit Klasse 8 in km/h	(0 ... 254; 255 NULLVALUE: kein gültiger Wert)
USHORT	Zählwert Klasse 8	0 ... 65534; 65535 NULLVALUE: kein gültiger Wert

Hinweis: Dieses Auftragselement **AEAggregiertExt** ist eine Erweiterung des Auftragselements AEAggregiert, das nur die im Lichtsignalsteuergerät gebildeten Werte Zählung und Belegungsgrad erfasst. Versionsstand des Lichtsignalsteuergeräts beachten!

3.5.3.7 Auftragselement für Visualisierungsdaten

AESiplOnline (1:438)

AESiplOnline		
METHOD	Name	Beschreibung
0	**Get**	Auslesen der Nummern der Signalgruppen, die zusätzlich zu TX übertragen werden.
	Ausgabeparameter	
	RetCode: USHORT	OK: wird immer zurückgeliefert
	RelKnotenNr: UBYTE	Relative Knotennummer auf die sich dieses Auftragselement bezieht. Hinweis: Default ist RelKnoten 0 (d.h. nach Anlegen des Auftragselements ist die RelKnotenNr mit 0 vorbelegt).
	AnzahlSigru: UBYTE	Anzahl an parametrierten Signalgruppen
	Sigrunummern[AnzahlSigru] UBYTE	Array mit den Nummern der übergebenen Signalgruppen
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Eingabeparameter	
		Keine
	Ausgabeparameter	

AESiplOnline		
METHOD	Name	Beschreibung
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: Hashwert über TX und alle Signalgruppenzustände. Hinweis: Der Hashalgorithmus ist vom Gerätehersteller frei wählbar, der Hashwert darf jedoch nicht größer als 32 Bit sein.
151	**GetSigState**	Abfrage des aktuellen TX und Signalzustand
	Ausgabeparameter	
	RetCode: USHORT	OK: wird immer zurückgeliefert
	TX: USHORT	aktuelles TX
	AnzahlSigru: UBYTE	Anzahl an folgenden Signalgruppen
	SigZustand[AnzahlSigru]: UBYTE	Farb-Zustand der einzelnen Signalgruppen (in Signalbildcodierung)
157	**SetRelKnotenGetSignal Gruppen**	Setzt die relative Knotennummer für dieses Auftragselement und liefert die Nummern aller Signalgruppen, die zusätzlich zu TX übertragen werden.
	Eingabeparameter	
	RelKnoten: UBYTE	Referenz auf den RelKnoten auf den sich dieses Auftragselement bezieht.
	Ausgabeparameter	
	RetCode: USHORT	OK: wird zurückgeliefert, wenn der RelKnoten gesetzt werden konnte PARAM_INVALID wird zurückgeliefert wenn der RelKnoten nicht gesetzt werden konnte (z.B. wenn er nicht vorhanden ist).
	RelKnotenNr: UBYTE	Relative Knotennummer auf die sich dieses Auftragselement bezieht.

AESiplOnline		
METHOD	Name	Beschreibung
	AnzahlSigru: UBYTE	Anzahl an parametrierten Signalgruppen
	Sigrunum-mern[AnzahlSigru] UBYTE	Array mit den Nummern der übergebenen Signal-gruppen

Hinweis: Es wird empfohlen mit Version 2.0 dieses Auftragselement anstelle AESignalBild zu verwenden.

Eine wichtige Funktion innerhalb einer Zentrale ist die Visualisierung der Signalisierung in Abhängigkeit vom Wert TX. Bisher müssen dafür alle Signalgruppen und der AP-Wert TX als einzelne Auftragselemente in einem oder mehreren Aufträgen angelegt werden. Zur Vereinfachung wird nun das TX und alle Signalgruppen eines relativen Knotens über ein einzelnes Auftragselement beauftragt, das in einem Auftrag-Zyklisch oder AbtastAenderung enthalten sein kann.

Es erzeugt als Eintrag im Sekundenframe den Wert TX (USHORT) gefolgt von der Anzahl der Signalgruppen (UBYTE). Darauf folgen die Signalisierungszustände in aufsteigender Reihenfolge von Signalgruppe 1 bis zur angegebenen Anzahl der Signalgruppen. Damit wird die Größe der übertragenen Datenpakete auf das nötigste beschränkt und die Zuordnung der Werte zu den einzelnen Signalgruppen auf Zentralenseite bleibt gewährleistet.

Bei der Beauftragung über den MWAuftragAbtastAenderung wird bei jeder Änderung des TX ein Sekundenframe generiert, der entweder nur das TX enthält (nämlich dann wenn das Signalbild konstant bleibt) bzw. das TX gefolgt von allen Signalgruppen, sobald sich auch nur eine Signalgruppe auf logischer Ebene geändert hat. Bei einem zyklischen Auftrag mit einer Intervalldauer von üblicherweise 1 Sekunde wird jeder Eintrag mit dem TX und allen Signalgruppen des Knotens erzeugt.

Über die Standardmethode Get erhält die Zentrale eine Liste der übertragenen Signalgruppen.

Mit der Methode SetRKGetSignalGruppen muss die Zentrale den gewünschten Rel-Knoten festlegen und erhält eine Liste der übertragenen Signalgruppen zurück.

Der resultierende Sekundenframe AESiplOnlineFrame des Auftrags ist wie folgt aufgebaut:

USHORT	AP-Wert TX
UBYTE	Anzahl folgender Signalgruppen
UBYTE	Signalisierung[AnzahlSigru]

3.5.3.8 Auftragselement Signalbild

Hinweis: Ab Version 2.0 soll dieses Auftragselement nicht mehr verwendet und durch AESiplOnline ersetzt werden.

Das Auftragselement Signalbild referenziert immer logische Signalbilder einer Signalgruppe.

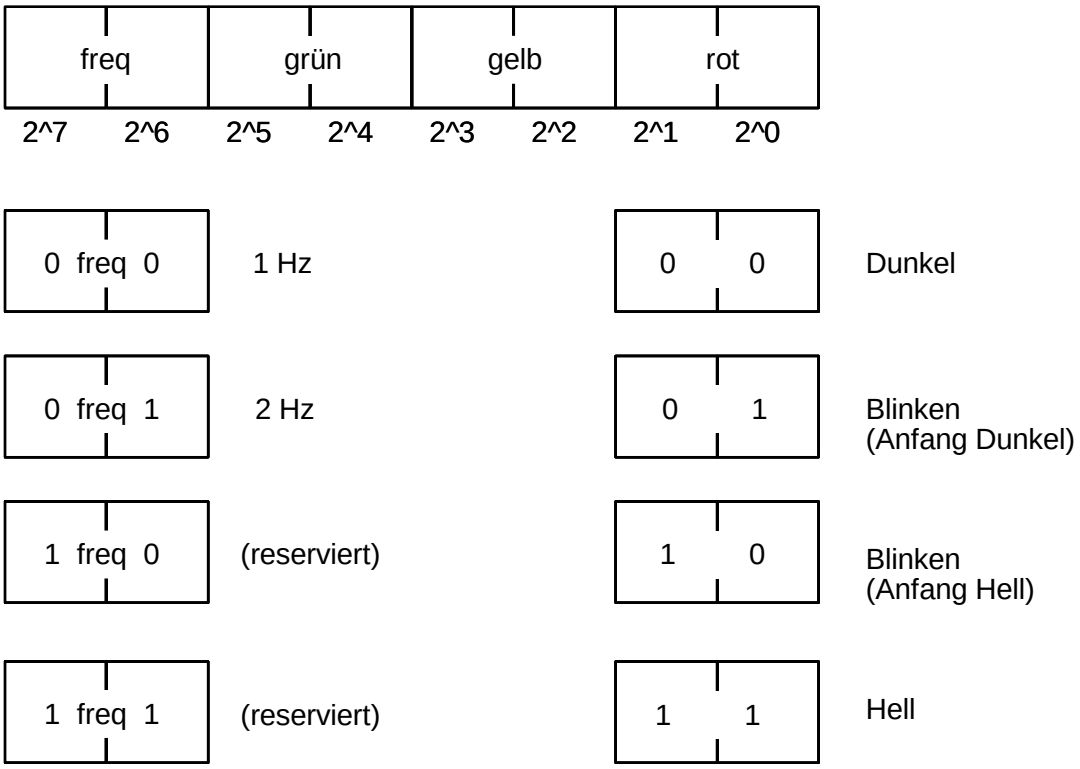
AESignalbild ist abgeleitet von AuftragsElement. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

AESignalBild (1:433)

AESignalBild		
METHOD	**Name**	**Beschreibung**
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: **LONG**	Wert des Triggers: aktuelles Signalbild
154	**SetSignalgruppe**	Setzt die Referenz auf die Signalgruppe.
	Eingabeparameter	
	SignalGruppe: Signal-Gruppe &	Referenz auf Signalgruppe, Pfad besteht aus: - relativer Knotennummer der Signalgruppe und - Nummer der Signalgruppe.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn die Signalgruppe nicht existent ist.

3.5.3.8.1 Signalbildcodierung

Signalbildcodierung Standard: UBYTE:



Festzeitfernversorgung von speziellen Signalgebern:

OEV-4-Punkt (Badehose):
Wird realisiert mittels 3 Signalgruppen. Versorgungsfarbcode: Rt, Gn oder Gelbblinken

Hüpflicht bzw. Springlicht:
Wird realisiert wie ein Blinker. Versorgungsfarbcode: Rt-Gn Wechselblinken

3.5.3.9 Auftragselement DigAusgang

Dieses Objekt ist für die Online-Visualiserung des Signalplans (z.B. Fußgängertaster) vorgesehen, andere Verwendungszwecke sind optional.

„Digitale Ausgänge“ sind alle Ausgänge, die nicht über Signalschalter arbeiten. Das Auftragselement DigAusgang referenziert immer logische Zustände.

AEDigAusgang ist abgeleitet von AuftragsElement. In der folgenden Tabelle werden deshalb nur die Unterschiede dazu aufgeführt.

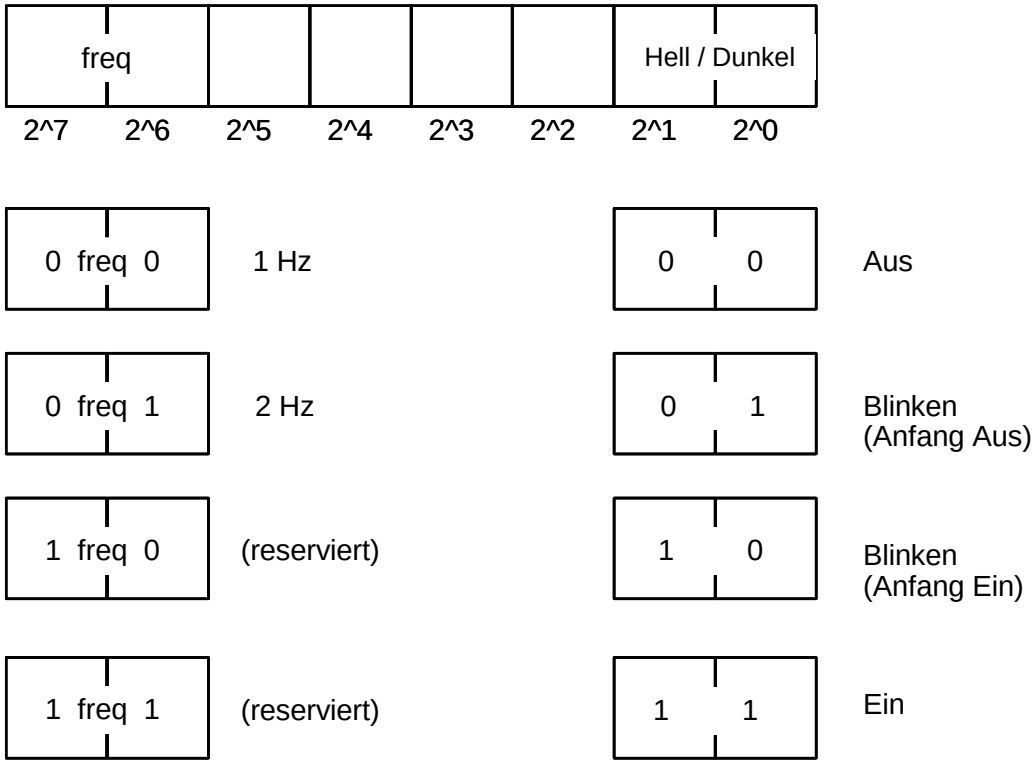
AEDigAusgang (1:439)

AEDigAusgang		
METHOD	**Name**	**Beschreibung**
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: aktueller Zustand des Ausgangs
154	**SetDigAusgang**	Setzt die Referenz auf den digitalen Ausgang.
	Eingabeparameter	
	DigAusgang: DigAusgang&	Referenz auf digitalen Ausgang, Pfad besteht aus: - relativer Knotennummer des digitalen Ausgangs und
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn der digitale Ausgang nicht existent ist.

3.5.3.9.1 Codierung

Codierung Standard: UBYTE:

Die Bits 2^2 bis 2^5 werden nicht benutzt.



3.5.3.10 Kombinationen von Aufträgen und Auftragselementen

Vom Gerät unterstützte Kombinationen: x

Nicht sinnvolle Kombinationen: (x)
Diese Kombinationen müssen vom Gerät nicht unterstützt werden und werden daher auch bei Interoperabilitätstests nicht geprüft.

Aufträge	**Auftragselemente**								
	Binär	Aggreg.	Signalbild	APWert	DetExt	AgggrExt	APWert-Vektor	SiplOnline	DigAusg
ZYKL	(x)	x	x	x		x	x	x	x
ABTASTAB	x								
ABTASTÄNDERUNG	(x)	x	x	x		x	x	x	x
VERGLEICH	(x)	x	x	x		x	(x)	(x)	x

Aufträge	Auftragselemente								
	Binär	Aggreg.	Signalbild	APWert	DetExt	AggrExt	APWert-Vektor	SiplOnline	DigAusg
				*)					
EXTERN	Keine Auftragselemente								
R09	Keine Auftragselemente								
AMLI	Keine Auftragselemente								
DETEXT					x				
MELDUNGSAUFTRAG	Keine Auftragselemente								

*) Vergleich nicht sinnvoll bei APWertBlock

3.5.4 AP-Werte

Anwenderprogrammwerte (AP-Werte) sind innerhalb eines Gerätes durch einen Namen vom Typ der speziellen Domain ANYPATH eindeutig referenziert. ANYPATH ist ein String der Namenskonventionen folgt. AP-Werte wurden historisch über sprechende Namen adressiert und da kam es vor, dass unterschiedliche Steuerverfahren gleiche Namen mit unterschiedlicher Bedeutung genutzt haben. Um die AP-Werte verfahrensübergreifend eindeutig zu machen, wurden auf OCIT-C Ebene die OITD Nummern eingeführt, welche in neueren OCIT Versionen aus kompatibilitätsgründen zu OCIT-C verwendet werden sollen. OITD Nummern sind wie folgt aufgebaut:

Member-ID.Otype.Index

Beispiel: 60.101.x AnfDet (Detektoranforderung aus dem Verfahren PDM)

Attribute	Beispiel Wert	Beschreibung
RefLen	15	Länge der Referenzdefinition in Bytes (ohne RefLen)
Member	1	OCIT-O Member (UShort)
OType	506	OCIT-O Type (UShort)
APWertName	60.101.1	OCIT-O String ANYPATH: entsprechend OCIT-C PD Festlegungen: <OIPM_Member>. <OIPM.Nr>. < Index>; + 2 Bytes Count

Der AP-Wert Name ist durch Member.OType eindeutig definiert. Wenn es von einem AP-Wert mehrere Instanzen gibt (z.B. AnfDet gibt es grundsätzlich pro versorgten Detektor) dann gibt es entsprechend viele Indexe. Da ein AP-Wert aber nur einen Typ hat müssen alle über den Index differenzierbaren Werte vom gleichen Typ sein und eine identische Bedeutung haben.

Für einzelne AP-Werte ohne Indexierung wird festgelegt, dass diese immer den Index 1 haben.

Festlegungen zu den AP-Werten:

- Projektspezifische AP-Werte:

  Namen von AP-Werten mit systemweiter Bedeutung sind projektspezifisch zu vereinbaren.

- Standard AP-Werte:

  Festlegungen in Pkt. 3.5.4.5

- AP-Werte aus VA-Verfahren:

  Der Verfahrenshersteller muss den OCIT-O Objekttyp der AP-Werte gemäß OCIT-C „Datenkatalog der AP-Werte eines Knotens“, „intersection_config_data.xsd“ <VAAPWert> angeben.

Zum Auslesen der Namen der im Lichtsignalsteuergerät verfügbaren AP-Werte stehen folgende Methoden zur Verfügung:

- SOFeldgeraet.InstanceInfo (255 Rückgabewerte) und
- SOFeldgeraet.ExtendedInstanceInfo (65535 Rückgabewerte)

Bei Aufruf von nicht existenten AP-Werten wird ERR_PATH_VAL zurückgegeben.

Hinweis: Bei Lichtsignalsteuergeräten, die sehr viele AP-Werte unterstützen (z.B. VS-Plus Geräte mit mehreren tausend AP-Werten) ist damit zu rechnen, dass die Abfrage der verfügbaren AP-Werte sehr lange dauern kann. Empfehlung: Der Aufruf von SOFeldgeraet.ExtendedInstanceInfo sollte über den Kanal mit niedriger Priorität erfolgen. Reicht auch das nicht aus, so können gezielt einzelne AP-Werte mit komplettem Pfad auf Verfügbarkeit abgefragt werden.

3.5.4.1 APWert

Dieser Objekttyp bildet einen Anwenderprogramm Wert (AP-Wert) ab. Anwenderprogrammwerte sind durch Namen eindeutig innerhalb eines Lichtsignalsteuergerätes bestimmt. APWert ist die Basisklasse für AP-Werte. Typen der AP-Werte: USHORT, LONG, BLOB.

APWert (1:505)

APWert		
METHOD	**Name**	**Beschreibung**
16	**GetWert**	Liefert den aktuellen APWert.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Wert: Typ des AP-Werts	Aktueller Anwenderprogramm-Wert. (Prozessvariable)

APWert		
METHOD	**Name**	**Beschreibung**
17	**SetWert**	Setzt diesen APWert (falls möglich und sinnvoll).
	Eingabeparameter	
	Wert: Typ des AP-Werts	Neuer Anwenderprogramm-Wert.
	Ausgabeparameter	
	RetCode	Falls OK wurde Wert akzeptiert. ACCESS_DENIED falls es nicht vorgesehen ist diesen Wert von außen zu verändern.
18	**IsWritable**	Liefert Informationen, ob der AP-Wert schreibbar ist.
	Ausgabeparameter	
	RetCode	OK
	BOOL	Ist true, falls SetWert möglich ist, false sonst.
19	**GetDescription**	Gibt eine Beschreibung bzw. den Klartextname des APWerts zurück.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Description	Beschreibung bzw. Klartextname des APWerts.

3.5.4.1.1 APWertUshort

APWertUshort (1:506)

Der Objekttyp Anwenderprogramm-Wert APWertUshort bildet einen AP-Wert vom Typ USHORT ab.

3.5.4.1.2 APWertLong

APWertLong (1:507)

Der Objekttyp **APWertLong** bildet einen Anwenderprogramm-Wert vom Typ LONG ab und wird in Auftragsframes mit 4 Byte abgespeichert.

3.5.4.2 APWertRk

APWertRk (1:510)

Der Objekttyp APWertRk bildet RelKnoten-bezogene Anwenderprogramm-Werte ab. Eine Instanz vom Typ APWertRk wird innerhalb eines Lichtsignalsteuergerätes durch APWertName (String) und RelKnotenNr (UBYTE) eindeutig referenziert.

3.5.4.2.1 APWertRkUshort

APWertRkUshort (1:511)

Der Objekttyp **APWertRkUshort** bildet einen RelKnoten-bezogenen Anwenderprogramm-Wert vom Typ USHORT ab und wird in Auftragsframes mit 2 Byte abgespeichert.

3.5.4.2.2 APWertRkLong

APWertRkLong (1:512)

Der Objekttyp **APWertRkLong** bildet einen RelKnoten-bezogenen Anwenderprogramm-Wert vom Typ LONG ab und wird in Auftragsframes mit 4 Byte abgespeichert.

3.5.4.3 APWertBlock

APWertBlock (1:508)

Der Objekttyp APWertBlock bildet einen Anwenderprogramm-Wert vom Typ BLOB ab.

Anmerkung: BLOB ist definiert in OCIT-O-Protokoll ab Version 1.0, Kapitel Grunddatentypen:

BLOB	struct{ ULONG sz, BYTE data[ ] }	Binary large object, in dem die Daten opaque übertragen werden.

APWertBlock (1:508)

APWertBlock		
METHOD	**Name**	**Beschreibung**
16	GetWert	Liefert den aktuellen APWert.
	Eingabeparameter	
		(keine)
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Wert: BLOB	Aktueller Anwenderprogramm-Wert.
17	SetWert	Setzt diesen APWert (falls möglich und sinnvoll).

APWertBlock		
METHOD	**Name**	**Beschreibung**
	Eingabeparameter	
	Wert: BLOB	Neuer Anwenderprogramm-Wert.
	Ausgabeparameter	
	RetCode	Falls OK wurde Wert akzeptiert. ACCESS_DENIED falls es nicht vorgesehen ist diesen Wert von außen zu verändern.

3.5.4.3.1 APWertRkBlock

APWertRkBlock (1:513)

Der Objekttyp APWertRkBlock bildet einen auf relative Knoten bezogenen Anwenderprogramm Wert vom Typ BLOB.

3.5.4.4 APWertGroup

APWertGroup (1:515)

Die Unterstützung von APWertGroup ist in OCIT-O Lstg V2.0 optional.

Mit APWertGroup lassen sich Gruppen von Anwenderprogrammwerten (AP-Werten) bilden. Sie werden genutzt, um komplexe Datenstrukturen oder auch Arrays auf einfache Art und Weise umzusetzen. Das Objekt enthält Methoden, mit denen die in der Gruppe enthaltenen Subgruppen und AP-Werte gelesen werden können.

Um die AP-Werte den Gruppen (APWertGroups) zuzuordnen steht als Strukturierungselement in den Namen der AP-Werte der Punkt „.“ zu Verfügung (analog z.B. zu den Domain-Namen des Internet). Der Name des AP-Werts wird dann gebildet, indem die Namen der enthaltenden APWertGroup durch den Punkt verkettet werden.

Die Gruppennamen ergeben sich automatisch aus den Namen der vorhandenen APWerte und benötigen keine Versorgung.

Für die lichtsignalsteuergerätebezogenen Gruppen gibt es jeweils eine ausgezeichnete Wurzelgruppe mit dem Leerstring als Namen. Diese Wurzelgruppen dienen dann als Ausgangsobjekt für die Abfragen GetElements.

Eine Beschränkung der Schachtelungstiefe der APWertGroups ergibt sich durch die maximale Länge der APWert-Namen (512).

Beispiele:

1. Es gibt die AP-Werte "Geraet.SG1.gruenMin", "Geraet.SG1.gruenMax", "Geraet.SG2.gruenMin", "Geraet.SG2.gruenMax". Die APWertGroup Gerät hat also 2 AP-Wert- Subgruppen Geraet.SG1 und Geraet.SG2, die jeweils 2 AP-Werte enthalten.

2. Beispiel einer vollständigen Definition einer AP-Werte Gruppe:

   Beispielwert: AEAPWertVektor.SetListe(), Index 1,5

Attribute	**Beispiel**	**Beschreibung**
Prefix	57.101.	OCIT-O String ANYPATH: entsprechend den OCIT-IP Festlegungen zum OIPM-Prefix: <OIPM_Member>. <OIPM.Nr>
Anzahl	2	Anzahl der Elemente in nachfolgender Liste
RefLen	8	Länge der Referentdefinition in Bytes (ohne RefLen)
Member	1	OCIT-O Member (UShort)
OType	506	OCIT-O Type (UShort)
APWertName	1	OCIT-O String ANYPATH: <Index> (4 Byte)
RefLen	8	Länge der Referentdefinition in Bytes (ohne RefLen)
Member	1	OCIT-O Member (UShort)
OType	506	OCIT-O Type (UShort)
APWertName	5	OCIT-O String ANYPATH: <Index> (4 Byte)

APWertGroup (1:515)

APWertGroup		
METHOD	**Name**	**Beschreibung**
100	**GetElements**	Liefert Referenzen auf die APWerte, die in der Gruppe enthalten sind. Dabei werden nicht die Referenzen auf die APWerte geliefert, die in den Subgruppen enthalten sind.
	Ausgabeparameter	
	RetCode	OK
	Refs: Path[ ]	Liste mit Referenzen auf APWerte, die besteht aus Refs.Anzahl Refs[ ].RefLen Refs[ ].Member Refs[ ].OType Refs[ ]. … Pfadparameter je nach APWert
101	**GetSubGroups**	Liefert Referenzen auf die Subgruppen der APWertGroup.
	Ausgabeparameter	
	RetCode	OK
	Refs: Path[ ]	Liste mit Referenzen auf APWertGroups, die besteht aus Refs.Anzahl Refs[ ].RefLen Refs[ ].Member Refs[ ].OType Refs[ ]. … Pfadparameter je nach APWert-Group

Beispiel: Es gibt eine Gruppe „Det“ mit folgenden Elementen:

Det.Anzahl, Det.Luecke.1, Det.Luecke.2, Det.Belgrad.1, Det.Belgrad.2

Die Methode GetElements liefert: Referenzen auf APWertGroups für Det.Anzahl

Die Methode GetSubgroups liefert: Referenzen APWertGroups für Det.Luecke und Det.Belgrad

3.5.4.4.1 APWertGroupRk

APWertGroupRk (1:516)

Die Unterstützung von APWertGroupRk ist in OCIT-O Lstg V2.0 optional.

Der Objekttyp APWertGroupRk bildet die auf einen relativen Knoten bezogenen Gruppen von Anwenderprogrammwerten ab. Für die Rk-bezogenen Gruppen gibt es jeweils eine ausgezeichnete Wurzelgruppe mit dem Leerstring als Namen. Diese Wurzelgruppen dienen dann als Ausgangsobjekt für die Abfragen GetElements.

3.5.4.5 Standardisierte AP-Werte

Häufig verwendete AP-Werte sind in OCIT-O standardisiert.

Werte des Signalplanumlaufs

Umlaufsekunde (TX), laufende Phase (PH) und gewünschte Phase (UE) werden als relative Knoten spezifische AP-Werte (Objekttyp: APWertRkUshort) standardisiert.

Umlaufsekunde (TX, 41.94):

Die Umlaufsekunde des laufenden Signalprogramms wird beginnend mit Sekunde 0 in 100ms-Schritten gezählt. Bei ausgeschaltetem Knoten bleibt der letzte Wert erhalten (Persistenz bei NetzAus nicht nötig). Erstinitialisierung (Reset) mit 0.

Laufende Phase (PH, 41.96):

0	Phase im Verfahren nicht definiert (z.B. VSPlus)
1…65534	aktuelle Phasennummer

Gewünschte Phase (UE, 41.97):

0	Es ist kein Phasenübergang aktiv bzw. eine Phase ist im Verfahren nicht definiert (VSPlus)
1…65534	Es ist ein Phasenübergang aktiv von Phase PH nach Phase UE.

3.5.4.6 Verfahrensparameter

Dynamische Parameter für ein verkehrstechnisches Verfahren **(DPV1, 41.98)**

Die dynamischen Parameter für ein verkehrstechnisches Verfahren, zum Beispiel eine Netzsteuerung, werden als AP-Werte Typ APWertBlock / APWertRkBlock (BLOB) übertragen. Die Inhalte sind anwendungsspezifisch und in OCIT-O nicht standardisiert.

Durch die Namenskonvention DPV1 wird lediglich das im Lichtsignalsteuergerät vorzugsweise angewendete verkehrstechnische Verfahren als Empfänger der Binärwerte adressiert.

Sollen weitere Parameter dieses Typs verwendet werden, sind ihre Namen projektspezifisch festzulegen.

3.5.5 Detektoren und Signale

3.5.5.1 Digitaler Eingang

Digitaler Eingang ist das Basisobjekt für digitale Eingänge. Er dient zur Darstellung der Referenzen in der XML-Metabeschreibung, sowie zur Abfrage der binären Zustände und der Namen der digitalen Eingänge. Die Versorgung mit Namen erfolgt proprietär durch den Gerätehersteller.

In den Festlegungen zu OCIT-O referenziert DigEingang auf Detektor-Meldungseingänge und Anforderungssignale. Projektspezifisch kann DigEingang zur Abfrage von frei wählbaren binären, digitalen Signalen verwendet werden.

DigEingang (1:500)

DigEingang		
METHOD	Name	Beschreibung
0	**Get**	
	Eingabeparameter	
		(keine)
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Name: STRING	Liefert den Namen des dig. Eingangs.
16	**GetWert**	Liefert den aktuellen Wert.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Wert: BOOL=UBYTE	Aktueller Wert

3.5.5.2 Signalgruppe

Signalgruppe ist das Basisobjekt für Signalgruppen und dient zur Darstellung der Referenzen in der XML-Metabeschreibung, sowie zur Abfrage der Namen der Signalgruppen.

Das Objekt kann mit OCIT-O Methoden nicht geschaltet oder auf binäre Zustände abgefragt werden! Die Versorgung mit Namen erfolgt proprietär durch den Gerätehersteller.

SignalGruppe (1:501)

SignalGruppe		
METHOD	Name	Beschreibung
0	**Get**	
	Eingabeparameter	
		(keine)
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Name: STRING	Liefert den Namen der Signalgruppe

3.5.5.3 Signalgeber

Signalgeber ist das Basisobjekt für Signalgeber und dient zur Darstellung der Referenzen in der XML-Metabeschreibung, sowie zur Abfrage der Namen der Signalgeber.

Da die Namen der Signalgeber von OCIT-O nicht verlangt werden, ist die Methode Get und die Versorgung mit Namen optional.

Das Objekt kann mit OCIT-O Methoden nicht geschaltet oder auf binäre Zustände abgefragt werden! Die Versorgung mit Namen erfolgt proprietär durch den Gerätehersteller.

SignalGeber (1:502)

SignalGeber		
METHOD	Name	Beschreibung
0	**Get**	
	Eingabeparameter	
		(keine)
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.

SignalGeber		
METHOD	Name	Beschreibung
	Name: STRING	Liefert den Namen der Signalgeber

3.5.5.4 Signalkammer

Signalkammer ist das Basisobjekt für die Signalkammern (Lampen) und dient zur Darstellung der Referenzen in der XML-Metabeschreibung, sowie zur Abfrage der Namen der Signalkammern (Lampen).

Das Objekt kann mit OCIT-O Methoden nicht geschaltet oder auf binäre Zustände abgefragt werden! Die Versorgung mit Namen erfolgt proprietär durch den Gerätehersteller.

Hinweis: Im Standard sind für eine Signalgruppe nur die drei Kammern (0 = rot, 1 = gelb, 2 = grün) definiert. Parallel geschaltete Lampen ändern nicht am Zustand der Signalgruppe. Die Verwaltung / Überwachung der Lampen ist Aufgabe der Signalsicherung ab und steht damit außerhalb der OCIT-O Spezifikation.

SignalKammer (1:503)

SignalKammer		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Eingabeparameter	
		(keine)
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Name: STRING	Liefert den Namen der Signalkammer (Lampe)

3.5.5.5 Digitaler Ausgang

Digitaler Ausgang ist das Basisobjekt für digitale Ausgänge und dient zur Darstellung der Referenzen in der XML-Metabeschreibung, sowie zur Abfrage der Namen der digitalen Ausgänge.

Das Objekt kann mit OCIT-O Methoden nicht geschaltet oder auf binäre Zustände abgefragt werden! Die Versorgung mit Namen erfolgt proprietär durch den Gerätehersteller.

DigAusgang (1:504)

DigAusgang		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Eingabeparameter	
		(keine)
	Ausgabeparameter	
	RetCode	Falls OK gilt folgender Wert.
	Name: STRING	Liefert den Namen der Dig. Ausgangs

3.5.6 Archive der Lichtsignalsteuergeräte

Folgende Archive sind in OCIT-Outstations für jedes Lichtsignalsteuergerät fest vorgegeben (sofern benötigt, wie Archive für ÖPNV oder Messwerte):

- Das **Betriebszustandsarchiv (0)** für die Speicherung des Betriebszustands (Meldungen BzIstvektor). Jede Änderung des IstVektors generiert einen Eintrag des IstVektors in der Betriebszustandsliste. Die Aufträge dazu sind vordefiniert und können nicht geändert werden. Die im Archiv gespeicherten Daten bleiben nach dem Ausschalten der Versorgungsspannung erhalten.
- Das **Standard-Meldearchiv (1)** enthält Meldungen der Signalsicherung, Störungen und andere Meldungen: OCIT-Hauptmeldung + Nebenmeldung + Meldungsdegree. Die Aufträge dazu sind vordefiniert und können nicht geändert werden. Die im Archiv gespeicherten Daten bleiben nach dem Ausschalten der Versorgungsspannung erhalten.
- Das **Syslog-Archiv (2)** für Syslog-Meldungen (mit Text) und herstellerspezifischen Meldungen, die persistent gehalten werden. Das Archiv ist bereits im Grundausbau vorhanden. Die Archivgröße wird vom Hersteller an die im Gerät vorhandenen anderen Archive angepasst. Die im Archiv gespeicherten Daten bleiben nach dem Ausschalten der Versorgungsspannung erhalten.
- Ein **Archiv Service-Systemzugang (3)** für Aufträge die über die Systemzugänge erfolgen.
- Das **Versorgungsarchiv (4)** für Versorgungsmeldungen.
- Ein **Status Archiv (5)** für die Speicherung des Betriebsstatus (Meldungen DeviceStateMsg). Bei jedem Betriebsstatuswechsel werden die Betriebsstati erfasst. Jede Änderung des Betriebsstatus generiert einen Eintrag des DeviceState in der Statusliste. Die Aufträge dazu sind vordefiniert und können nicht geändert werden. Die im Archiv gespeicherten Daten bleiben nach dem Ausschalten der Versorgungsspannung erhalten.
- Ein **Dynamisches Archiv (31)** ist für Prozessdaten vorgesehen, deren Beauftragung häufig verändert wird.
- Ein **Signalisierungsarchiv (32)** für Signalisierungszustände (erfasst bei jedem Zustandswechsel). Mögliche Ergänzungen sind Umlaufsekunde TX, Detektorsignale, Phasen u. a.

- **OEPNV-Archiv (33)** für R09-Standard-Telegramme (Erstellungszeitpunkt, Meldepunkt, Linie, Kurs, Route, Priorität, Zuglänge Richtung Hand, Fahrplanabweichung) oder erweiterte R09-Telegramme. Es werden alle für das Lichtsignalsteuergerät relevanten R09-Telegramme im Archiv abgespeichert. Irrelevante Telegramme, die trotzdem empfangen wurden, werden nicht gespeichert.
- Ein **Messwertarchiv (34)** für aggregierte Detektorwerte wie Fzg/h, Belegung in % und projektspezifische Messwerte.
- Ein **Onlinearchiv (35)** für Detektorrohwerte (Änderungen des Detektorausgangs) und AP-Werte. Die Abtastintervalle in denen die Änderungen erfasst werden (Auflösung) sind von der Zentrale aus einstellbar. Die höchste einstellbare Auflösung beträgt 10 ms. Wird ein Abtastintervall gewählt, das das Gerät nicht liefern kann, wird eine Fehlermeldung abgesetzt, die auch das vom Gerät unterstütze Intervall beinhaltet.
- **Offlinearchiv (36)** ist für das OCIT-O Profil 2 vorgesehen.
- Ein **CAM-Archiv (37)** für die Statusinformationen der Fahrzeuge. Dieses Archiv ist für die Einzelfahrzeugerfassung (MWAuftragSingleCar, MWAuftragSingleCarExt) sowie für die aggregierten Fahrzeugdaten (AEAggregiertLane, AEAggregiertConnection) vorgesehen. Zusätzlich können die Priorisierungsanfragen aus den empfangenen CAM-Telegrammen (MwAuftragCamR09, MWAuftragCamPrio) hier protokolliert werden.
- Ein **DENM-Archiv (38)** für die Gefahrenmeldungen. Mit dem Auftrag MWAuftragDENM werden in diesem Archiv die Gefahrenmeldungen aufgezeichnet. Für jede Änderung des DenmPool („trigger“, „update“ und „terminate“) wird ein entsprechendes Frame erzeugt.
- Ein **Spat-Archiv (39)** für die Prognosedaten. Die prognostizierten Signalgruppenzustandsänderungen werden zyklisch oder bei Änderung (je nach verwendetem Auftrag) mit dem Auftragselement AESpat aufgezeichnet.

Die Mindestgrößen der Archive OCIT-konformer Lichtsignalsteuergeräte sind im Dokument Funktionsspiegel (OCIT-O_V3.0_Funktionsspiegel) angegeben.

Die Archive 31, 32, 33, 34, 35, 36,37, 38, 39 sind zur Laufzeit von der Zentrale aus konfigurierbar.

Verhalten bei Stromausfall oder dem Ausschalten der Versorgungsspannung:

- Archive 0, 1, 2, 3, 4, 5: Die im Archiv gespeicherten Daten und die Listenstruktur bleiben erhalten.
- Die Inhalte der nicht persistenten Archive (je nach Einstellung) und die zugehörigen Listen können verloren gehen. Bei Wiederanlauf des Lichtsignalsteuergerätes werden die nicht persistenten Listen zurückgesetzt.

Hinweise:

1. Die festdefinierten Archive 0 bis 5 und 31 bis 39 (siehe Pkt. 3.5.6.4) sollen vorzugsweise und ausschließlich die dafür vorgesehenen Aufträge enthalten.

2. Aufgrund der Belastung der Übertragungsstrecke zwischen Lichtsignalsteuergerät und Zentrale (PD-Server) muss die Übertragung von AP-Werten (ohne Signalisie-

rungsinformationen und Detektorwerte) eingeschränkt werden. Bei typischer Nutzung des Lichtsignalsteuergerätes:

- Profil 1 mit Übertragungsrate von 19200 Baud
- Übertragung von max. 20 SG, max. 32 Detektoren

sollen zusätzlich nicht mehr als 20 AP-Werte sekündlich übertragen werden.

3.5.6.1 Elementbeschreibungen Meldungsarchiv

Das allgemeine Archiv wird pro Meldung als eine Liste von Meldungsteilen übertragen; bei vielen Meldungen fällt nur der Hauptmeldungsteil an. Eine vollständige Implementierung der hier definierten Fehlermeldungen wird nicht gefordert, da manche Fehlertypen bei manchen LSA nicht auftreten können. Es wird lediglich gefordert, dass die auftretenden Fehler OCIT-Outstations konform codiert werden. Zusätzlich sind auch noch hersteller- oder projektspezifische Meldungsteile oder Meldungen möglich.

Member =1:

(MeldungsDegree **I**: Information, **W**: Warnung, **F**: Fehler, **S**: Schwerer Fehler)

OType	Kurzname	MeldungsDegree	Beschreibung
60004	Sollbild-Störung	S	Ist die Hauptmeldung bei Störungen der Sollbildüberwachung. Sehr häufig durch zusätzliche Meldungsteile genauer spezifiziert.
60005	Istbildfehler (schwer)	S	Geräteabschaltung durch unzulässiges Istbild. ohne Rotlampenfehler.
60006	Feindlichkeit	W	Wird vom Gerät gemeldet, wenn die Firmware eine Feindlichkeitsverletzung erkennt und korrigiert. Bei Feindlichkeitsverletzungen, die die Signalüberwachung erkennt, wird dieser Meldungsteil als Nebenmeldungsteil einer Sollbild-Störung gespeichert
60007	Zwischenzeit	W	Wird vom Gerät gemeldet, wenn die Firmware eine Zwischenzeitverletzung erkennt und korrigiert. Bei Zwischenzeitverletzungen, die die Signalüberwachung erkennt, wird dieser Meldungsteil als Nebenmeldungsteil einer Sollbild-Störung gespeichert.
60008	Mindestgrün	W	Wird vom Gerät gemeldet, wenn die Firmware eine Mindestgrünzeitverletzung erkennt und korrigiert. Bei Mindestgrünzeitverletzungen, die die Sig-

OType	Kurzname	MeldungsDegree	Beschreibung
			nalüberwachung erkennt, wird dieser Meldungsteil als Nebenmeldungsteil einer Sollbild-Störung gespeichert.
60009	Mindestrot	W	Wird vom Gerät gemeldet, wenn die Firmware eine Mindestrotzeitverletzung erkennt und korrigiert. Bei Mindestrotzeitverletzungen, die die Signalüberwachung erkennt, wird dieser Meldungsteil als Nebenmeldungsteil einer Sollbild-Störung gespeichert.
60010	Rotlampenfehler	S	Geräteabschaltung durch unzulässiges Istbild aufgrund eines Rotlampenfehlers.
60011	Istbildfehler (sekundär)	W	unzulässiges Istbild (sekundärer Lampenfehler)
60014	Detektorstörung	F	Wird eingetragen, wenn ein Detektor ausgefallen ist oder wenn die Plausibilitätskontrolle ausgelöst hat.
60015	Detektor ok	I	wird eingetragen, wenn ein Detektor wieder behoben ist
60022	Versorgung Beginn	I	Eine Versorgungsänderung beginnt. Sehr häufig durch zusätzliche Meldungsteile genauer spezifiziert
60023	VersorgungEnde	I	Eine Versorgungsänderung endet. Sehr häufig durch zusätzliche Meldungsteile genauer spezifiziert
60024	Umlaufkontrolle	F	Wird gemeldet wenn die Umlaufkontrolle ausgelöst wird (gibt es – abhängig vom Steuerverfahren - nicht immer)
60025	Wechsel der Betriebsart	W	Wird gemeldet, wenn die Betriebsart gewechselt wird: - Sonderbetrieb - Eigensteuerung (lokale VA-Programmwechsel) - Handstoppbetrieb - lokal fixes Programm - lokal DCF

OType	Kurzname	MeldungsDegree	Beschreibung
			- Zentrale
60037	ÖVempfängerstö-rung	E	Störung beim Empfang von ÖV-Telegrammen
60038	ÖVempfänger Ok	I	Empfangsstörung ÖV-Telegramme behoben
60045	Versatzzeit	W	Wird vom Gerät gemeldet, wenn die Firmware eine Versatzzeitverletzung erkennt und korrigiert. Bei Versatzzeitverletzungen, die die Signalüberwachung erkennt, wird dieser Meldungsteil als Nebenmeldungsteil einer Sollbild-Störung gespeichert.

Bei den Meldungen IstBildfehler, Lampenausfall hat der Parametersatz der Meldung folgenden Aufbau:

RelKnotenNr (UBYTE)	Nummer des relativen Knotens
SigGrpNr (UBYTE)	Nummer der Signalgruppe
SigGeberNr(STRING)	Nummer des Signalgebers
KammerNr (UBYTE)	Nummer der Signalkammer
SigGrpName (STRING)	Name der Signalgruppe
SigGeberName (STRING)	Name des Signalgebers

Bei den Meldungen Mindestgrünzeitunterschreitung und versuchte Mindestgrünzeit-unterschreitung hat der Parametersatz der Meldung folgenden Aufbau:

RelKnotenNr (UBYTE)	Nummer des relativen Knotens
SigGrpNr (UBYTE)	Nummer der Signalgruppe
SigGrpName (STRING)	Name der Signalgruppe

Bei den Meldungen (versuchte) Feindlichkeit und (versuchte) Zwischenzeitverletzung hat der Parametersatz der Meldung folgenden Aufbau:

RelKnotenNr (UBYTE)	Nummer des relativen Knotens
SigGrpNrA (UBYTE)	Nummer der einfahrenden Signalgruppe
SigGrpNameA (STRING)	Name der einfahrenden Signalgruppe

SigGrpNrB (UBYTE)	Nummer der räumenden Signalgruppe
SigGrpNameB (STRING)	Name der räumenden Signalgruppe

Bei den Meldungen Detektorstörung und Detektor ok gibt es folgende Parameter (Achtung: Die Detektoren sind in OCIT-Outstations nicht bezogen auf den relativen Knoten):

DetektorNr (UBYTE)	Nummer des Detektors
DetektorName (STRING)	Name des Detektors

Hinweis: Alle Meldungen, die die Versorgung betreffen werden nur ins Versorgungsarchiv eingetragen. Im Standard-Meldearchiv wird jeder Versorgungsvorgang nur mit den Meldungen „Versorgungsbeginn“ und „Versorgungsende“ vermerkt. Handhabung über Include/Exclude-Liste von Standard-Melde- und Versorgungsarchiv.

3.5.6.1.1 Exclude Liste des Standard-Meldearchivs

Folgende Meldungen befinden sich in der Exclude-Liste des Standard-Meldearchivs, d.h. diese Meldungen werden nicht eingetragen:

1	60301	TransactionDefined
1	60315	APWertChangeRequested
1	60316	APWertChangeCommitted
1	60318	TransactionActivationRequest
1	60319	SupplyVersionChanged

Alle anderen Meldungen befinden sich somit in der Include-Liste und werden eingetragen.

3.5.6.1.2 Include Liste des Versorgungsarchivs

Folgende Meldungen befinden sich in der Include-Liste des Versorgungsarchivs, d.h. diese Meldungen werden eingetragen:

1	60022	VersorgungsBeginn
1	60023	VersorgungsEnde
1	60301	TransactionDefined
1	60315	APWertChangeRequested
1	60316	APWertChangeCommitted
1	60318	TransactionActivationReqeust
1	60319	SupplyVersionChanged

Alle anderen Meldungen befinden sich somit in der Exclude-Liste und werden nicht eingetragen.

3.5.6.2 Elementbeschreibungen Betriebszustandsarchiv

Das Betriebszustandsarchiv wird pro Meldung als eine Liste von Meldungsteilen übertragen, bei denen jeweils die Vorgangskennung im Meldungsteil und der „neue“ Wert als Datensatz übertragen wird.

Es müssen in der BZ-Meldung auch die BZ-Meldungsteile übertragen werden, die sich nicht ändern! Dadurch wird sichergestellt, dass auch bei verloren gegangenen Meldungen immer der Gesamtzustand des Systems dargestellt werden kann.

Theoretisch lassen sich die Meldungsteile auch in anderen Archiven getrennt eintragen. Dieses Verfahren ist nicht verboten, das Betriebszustandsarchiv ist jedoch unbedingt zu implementieren.

Pro relativem Knoten gibt es einen Meldungsteil der den IstVektor (siehe 3.4.19) enthält. Diese Meldung kann um herstellerspezifische Meldungsteile erweitert werden.

Meldungsteil für den knotenbezogenen Betriebszustand

Name		**Beschreibung**
Knotennummer		Verweis auf relativen Knoten, der diesen Eintrag veranlasst. (Alle folgenden Nummern gelten für diesen Knoten)
Sammelstoerung: ui1		0 = keine Störung 1 = Störung ohne Abschaltung 2 = Störung mit Abschaltung 3 = Störung mit Teilabschaltung 4 = Interne Störung ohne Abschaltung
IBetriebsart		
	Vorgangskennung	Vorgangskennung der Betriebsartwahl welche zum Erreichen der folgenden Betriebsart führte.
	Betriebsart: UBYTE	Sonderbetrieb Eigensteuerung (lokale VA-Programmwechsel) Handstoppbetrieb lokal fixes Programm lokal Zeitsteuerung Zentrale (Gerät berücksichtigt Zentralenschaltwunsch)
ISignalProgramm		
	Vorgangskennung	Vorgangskennung der Signalprogrammwahl welche zum Erreichen der folgenden SigProgNr führte.

Name		Beschreibung
	SigProgNr	Zu dieser Zeit eingestellte Signalprogrammnummer
IKnotenEinAus		
	Vorgangskennung	Kennung des Vorgangs, welcher zu folgendem KZustand führte.
	KZustand	Ein/Auszustand des gesamten relativen Knotens
ITeilknoten[ ]		
	Vorgangskennung	
	TKZustand	Zu dieser Zeit eingestellter Teilknotenzustand
ISondereingriff		
	Vorgangskennung	
	SondereingriffNr	eingestellter Sondereingriff
Modifikations[0...15]		
	Modifikationen des Signalprogramms. Hier kann jede von IModEinAus abgeleitete Klasse stehen, dies sind z. Zt. IVAEinAus, IOepnvEinAus, IProjEinAus. Übertragung als Array mit variablen Typen.	
	Vorgangskennung	Kennung des Vorgangs, welcher zu diesem Schaltwunsch führte
	Zustand	Zustand der Modifikation

3.5.6.3 Elementbeschreibungen Status-Archiv

Die Meldung DeviceStateMsg (1:60321) wird eingetragen, wenn der Status des Steuergerätes geändert wird. Diese Hauptmeldung wird in der Liste 5 (Status-Archiv) erfasst.

Hauptmeldungsteil für den Status

DeviceStateMsg (1: 60321)

Name	Beschreibung
HauptMeldungsteil:VorgangsNr	Kennung des Vorgangs, welcher zu diesem Meldungsteil führte.
DeviceState:When	Zeitstempel des Gerätes, wann es diesen Zustand erreicht (zuletzt geändert) hat.
DeviceState:TimeSource	Gibt die aktuelle Quelle der Gerätezeit an
DeviceState:EmerOff	Gibt an, ob ein ggf. vorhandener NOTAUS Schalter betätigt ist.

Name	Beschreibung
DeviceState:DoorOpen	TuerAuf=true bedeutet: Der Türschließ-kontakt meldet: Mindestens eine Tür des Gerätes ist offen. Wenn kein Türschließ-kontakt vorhanden ist, ist TuerAuf=false.
DeviceState:PowerSupplyVoltageOk	Gibt an, ob die für den vollen Gerätebe-trieb nötige Netzspannung anliegt.
DeviceState:MalfunctioningDetectors	Liste der gestörten Detektoren.
DeviceState:FaultLamps	Liste der gestörten Lampen.
DeviceState:PersistenceStorageOk	Gibt an, ob der gesamte Persistenzspei-cher konsistent ist. Dieses Flag wird nach Netz Ein oder häufiger vom Gerät gesetzt.
DeviceState:PTReceiver	Gibt an, ob der ÖV-Empfänger funktio-niert.
DeviceState:Synchron	Gibt an, ob das Gerät synchron zur Refe-renzzeit ist.
DeviceState:CentralCommandAccepted	Gibt an, ob das Gerät aktuell auf Zentra-len-Schaltbefehle reagieren würde.
DeviceState:UPS	Gibt an, ob die USV im Gerät installiert ist und funktioniert.
DeviceState:SyncTimeSource	Gibt an ob mindestens eine der konfigu-rierten Zeitquellen zur Zeitsynchronisation verfügbar ist.
DeviceState:RSUState	Gibt den Zustand der RSU an.
Device State: Maintenance	Gibt an, ob sich das Gerät im Wartungs-zustand befindet

3.5.6.3.1 Status OevEmpfang

Aufzählung von möglichen Zuständen des OevEmpfang

PTReceiver (0:138)

Name	PTReceiver		
	Typ	UBYTE	max. 255
	Wert	**Beschreibung**	
OK	0	OevEmpfang konfiguriert und ok	
CRC_ERROR	1	Zu viele CRC-Fehler bei den empfangenen Telegrammen	
COM_ERROR	2	Empfangseinheit gestört bzw. keine Kommunikation	
REC_ERROR	3	Funkempfangseinheit gestört bzw. kein Rauschen	

TIMEOUT	4	Seit längerer Zeit keine gültigen ÖV-Telegramme mehr empfangen (Timeoutzeit gerätespezifisch parametrierbar)
NOT_CONFIGURED	254	Kein OevEmpfang konfiguriert
UNKNOWN	255	Zustand unbekannt oder nicht lieferbar

3.5.6.3.2 Status Referenzzeit

Aufzählung von möglichen Zuständen der Synchronität zur rückgerechneten Referenzzeit.

SYNCHRON (0:139)

Name	**SYNCHRON**		
	Typ	UBYTE	max. 255
	Wert	**Beschreibung**	
OK	0	Geräte-TX läuft synchron zur Referenzzeit	
NOT_SYNC	1	Gerät läuft nicht synchron zur Referenzzeit	
NO_SYNC_REQ	2	Gerät läuft nicht in synchronisierter Betriebsart	
NOT_CONFIGURED	254	Kein Rückrechenverfahren konfiguriert, d.h. Gerät muss nicht zeitsynchronisiert laufen	
UNKNOWN	255	Zustand unbekannt oder nicht lieferbar	

3.5.6.3.3 Status Zentralenbefehle

Aufzählung von möglichen Zuständen für den Status ob Zentralenbefehle akzeptiert werden.

CentralCommandAccepted (0:140)

Name	**CentralCommandAccepted**		
	Typ	UBYTE	max. 255
	Wert	**Beschreibung**	
OK	0	Gerät akzeptiert Schaltwünsche von der Zentrale	
NOT_OK	1	Gerät akzeptiert keine Schaltwünsche von Zentrale (z.B. Betriebsart lokal fix)	
UNKNOWN	255	Zustand unbekannt oder nicht lieferbar	

3.5.6.3.4 Status USV

Aufzählung von möglichen Zuständen für den Status der USV.

UPS (0:141)

Name	UPS		
	Typ	UBYTE	max. 255
	Wert	**Beschreibung**	
OK	0	USV ist ok, Batterie hat ausreichend Spannung	
NOT_OK	1	USV ist defekt bzw. Batterie leer oder defekt	
NOT_CONFIGURED	254	Keine USV im Gerät konfiguriert	
UNKNOWN	255	Zustand unbekannt oder nicht lieferbar	

3.5.6.3.5 Status Zeitquelle zur Zeitsynchronisation

Aufzählung von möglichen Zuständen für den Status der Zeitquelle zur Zeitsynchronisation.

SyncTimeSource (0:142)

Name	SyncTimeSource		
	Typ	UBYTE	max. 255
	Wert	**Beschreibung**	
OK	0	Mindestens eine der definierten Zeitquellen zur Zeitsynchronisation ist erreichbar und funktioniert	
NOT_OK	1	Keine der konfigurierten Zeitquellen ist erreichbar bzw. funktioniert	
NOT_CONFIGURED	254	Keine externen Zeitquellen im Gerät konfiguriert	
UNKNOWN	255	Zustand unbekannt oder nicht lieferbar	

3.5.6.4 Eigenschaften der Listen

Listennummer	**0 Betriebszustand**	**1 Standard-Meldearchiv**	**2 Syslog**	**3 Service**	**4 Versorgung**	**5 Status**
Anlegen von Aufträgen möglich?	Nein	Nein	Ja	Ja	Nein	Nein
Start/Stopp / Reset der Liste möglich?	Nein	Nein	Ja	Ja	Nein	Nein
Suspend / Unsuspend der Liste möglich?	Nein	Nein	Ja	Ja	Nein	Nein
Vorbelegte Persistenz (keine, Auftrag, Aufträge & Buffer)	Aufträge & Buffer	Aufträge & Buffer	Aufträge & Buffer	Aufträge & Buffer	Aufträge & Buffer	Aufträge & Buffer
Auswahl der Persistenz möglich	Nein	Nein	Nein	Nein	Nein	Nein
Vorbelegter Zustand der Liste (Start, Stop, Suspend)	Start	Start	Stop	Stop	Start	Start
OverwriteOnFull aktiv?	Ja	Ja	Ja	Ja	Ja	Ja
Größe des Buffers änderbar?	Nein	Nein	Nein	Nein	Nein	Nein
Vordefinierte Aufträge (nach Reset)	Knotenbetriebszustand 0:Meldungsauftrag I	0:Meldungsauftrag I 1:Meldungsauftrag W 2:Meldungsauftrag E 3:Meldungsauftrag F	0:Meldungsauftrag I	0:Meldungsauftrag I	0:Meldungsauftrag I 1:Meldungsauftrag W	0:Meldungsauftrag I

Listennummer	**31 Dynamisches Archiv**	**32 Signalisierung**	**33 ÖPNV**	**34 Messwert**	**35 Online**
Anlegen von Aufträgen möglich?	Ja	Ja	Ja	Ja	Ja
Start/Stopp / Reset der Liste möglich?	Ja	Ja	Ja	Ja	Ja
Suspend / Unsuspend der Liste möglich?	Ja	Ja	Ja	Ja	Ja
Vorbelegte Persistenz (keine, Auftrag, Aufträge & Buffer)	Keine	Keine	Keine	Keine	Keine
Auswahl der Persistenz möglich	herstellerabhängig	herstellerabhängig	herstellerabhängig	herstellerabhängig	herstellerabhängig
Vorbelegter Zustand der Liste (Start, Stop, Suspend)	Stop	Stop	Stop	Stop	Stop
OverwriteOnFull aktiv?	Ja	Ja	Ja	Ja	Ja
Größe des Buffers änderbar?	Ja	Ja	Ja	Ja	Ja
Vordefinierte Aufträge (nach Reset)	0:Meldungsauftrag I	0:Meldungsauftrag I	0:Meldungsauftrag I	0:Meldungsauftrag I	0:Meldungsauftrag I

Listennummer	**36 Offline (für Profil 2)**	**37 CAM Archiv**	**38 DENM Archiv**	**39 SPAT Archiv**
Anlegen von Aufträgen möglich?	Ja	Ja	Ja	Ja
Start/Stopp / Reset der Liste möglich?	Ja	Ja	Ja	Ja
Suspend / Unsuspend der Liste möglich?	Ja	Ja	Ja	Ja
Vorbelegte Persistenz (keine, Auftrag, Aufträge & Buffer)	Keine	Keine	Keine	Keine
Auswahl der Persistenz möglich	Ja	Ja	Ja	Ja
Vorbelegter Zustand der Liste (Start, Stop, Suspend)	Stop	Stop	Stop	Stop
OverwriteOnFull aktiv?	Ja	Ja	Ja	Ja
Größe des Buffers änderbar?	Ja	Ja	Ja	Ja
Vordefinierte Aufträge (nach Reset)	0:Meldungsauftrag I Achtung: Zeitsprung wird nicht eigetragen!	0:Meldungsauftrag I	0:Meldungsauftrag I	0:Meldungsauftrag I

Hinweise:

1. Dynamisches Archiv (Liste 31): Bei jeder Änderung des Auftrags ist damit zu rechnen, dass ein ResetListe generiert wird, was zu einer kurzen Datenunterbrechung führt. Empfehlung: AddAuftrag bei laufender Liste unterstützen. Falls die Aufträge häufig geändert werden, können die max. 254 Auftragsnummern ausgehen. Dann ist ein ResetListe notwendig.
2. Offlinearchiv (Liste 36): Die Verwendung der Include / Exclude-Methoden ist notwendig. Siehe auch Dokument Profil 2. Die Methoden Include / Exclude sollten vorzugsweise während der Konfiguration aufgerufen werden. Näheres dazu siehe die Definition im Dokument Basis 4.2.6.2 Meldungsauftrag. Vordefinierte Aufträge können nicht verändert werden.
3. Listen mit AP-Werten welche einer Verkehrslogikanwendung zugeordnet sind, müssen nach einer Neuversorgung der Verkehrslogik manuell entsprechend angepasst werden, weil möglicherweise neue Werte dazukommen und **nicht mehr vorhandene** AP-Werte entfernt werden sollen. Der Wert nicht mehr vorhandener AP-Werte wird im Lichtsignalsteuergerät automatisch auf NULLVALUE gesetzt, beim BLOB wird die Länge auf Null gesetzt.

4 Car-2-X Kommunikation

In diesem Abschnitt wird eine kurze Übersicht alle Objekte der Car2X Kommunikation für die OCIT-O V3.0 Schnittstelle zwischen einer Zentrale und einem Lichtsignalsteuergerät beschrieben. Die RSU kann ein Teil einer Lichtsignalanlage (LSA) sein.

Die Funktionen von OCIT-O Car werden von den Car2X Meldungen

Cooperative Awareness Message (CAM)

Dencentralized Environmental Notification Message (DENM)

Topographie und Signalplandaten (MAP + Signal Phase and Timing (SPaT)

abgeleitet.

Damit unterstützt die OCIT-O V3.0 Schnittstelle

- Gefahrenmeldungen
- Qualitätssicherung der Verkehrsdaten
- Statistik und Messungen von Floating Car Daten
- Priorisierung von OEPNV- und Sonderfahrzeugen

Weiterführende Detailinformationen sind im Dokument „**OCIT-Outstations Car Roadside Unit (RSU)**“ beschrieben.

4.1 Statusinformationen

Für eine Qualitätsanalyse und Qualitätssicherung von Knotenpunkten sollen Daten aus der C2X – Kommunikation verwendet werden.

Die Fahrzeuge senden die Statusinformationen vieler Sensoren mittels der CAM Nachrichten an die RSU. Viele Daten, wie Außentemperatur, Status des Bremskraftverstärkers oder die Winkelstellung des Lenkrads, sind dabei uninteressant. Die für die Verkehrstechnik interessanten Daten sind folgende:

- Datum und Zeit
- Position
- Geschwindigkeit
- Richtung des Fahrzeugs
- Fahrzeugtyp

Mit Hilfe der Kreuzungstopologie aus der MAP Nachricht lassen sich die Positionen der Fahrzeuge einer Spur oder auch einer Signalgruppe zuordnen. Für eine zentrale Analyse der Fahrzeugdaten werden man folgende Werte bereitgestellt (jeweils im Empfangsbereich der RSU):

- Durchschnittsgeschwindigkeit
- Wartezeit
- Anzahl der Halte
- verbindungsbezogene Durchschnittswerte

4.1.1 Auftrag MWAuftragSingleCarExt

Der Auftrag MWAuftragSingleCarExt dient zur Abbildung eines Fahrzeugs innerhalb eines Kreuzungsbereiches. Die erzeugten Frames enthalten Daten, die den Fahrweg und die Fahrzeit des Fahrzeugs beschreiben. Die Daten werden aus den CAM Nachrichten, die an die RSU geschickt wurden, ermittelt. Wenn der Auftrag gesetzt wird, wird für jedes Fahrzeug im Bereich der RSU ein Frame abgespeichert.

Der Auftrag MWAuftragSingleCarExt ist vom Objekt Auftrag abgeleitet und erzeugt Frames des Typs MWAuftragFrameSingleCarExt.

Anmerkung: Spurbezogene Werte sind nur verfügbar, falls die MAP versorgt ist (also nicht bei RSUs ohne LSA).

MWAuftragSingleCarExt (1:414)

MWAuftragSingleCarExt		
METHOD	**Name**	**Beschreibung**
119, 120, 121, 122	**ActivateEvent,** **AddElement,** **Start,** **Stopp**	Siehe Kapitel „Abläufe Meldung und Messwerte“ Dokument OCIT-O Basis.

Struktur des Ergebnisframes (MWAuftragFrameSingleCarExt):

Name	**Datentyp**	**Bemerkung**
StartPosition.Latitude	LONG Min=-900 000 000 Max=900 000 000 Nullval=900 000 001	Geoposition des Fahrzeugs beim Eintritt in den Kreuzungsbereich in 1/10 Mikrograd
StartPosition.Longitude	LONG Min=-1 800 000 000 Max=1 800 000 000 Nullval=1 800 000 001	Geoposition des Fahrzeugs beim Eintritt in den Kreuzungsbereich in 1/10 Mikrograd
EndPosition.Latitude	LONG Min=-900 000 000 Max=900 000 000 Nullval=900 000 001	Geoposition des Fahrzeugs beim Verlassen des Kreuzungsbereichs in 1/10 Mikrograd

EndPosition.Longitude	LONG Min=-1 800 000 000 Max=1 800 000 000 Nullval=1 800 000 001	Geoposition des Fahrzeugs beim Verlassen des Kreuzungsbereichs in 1/10 Mikrograd
ConnectionID	UBYTE	ID der Verbindung
StationType	UBYTE ENUM	Art des Fahrzeugs (Fahrzeugklasse) (100:103 STATION_TYPE)
PreStopline.TravelTime	USHORT Nullval=0xffff	Fahrzeit des Fahrzeugs bis zur Haltelinie in Sekunden
PreStopline.Stops	USHORT Nullval=0xffff	Anzahl der Halte vor Erreichen der Haltelinie
PreStopline.StandingTime	USHORT Nullval=0xffff	Zeit, die das Fahrzeug bis zum Erreichen der Haltelinie stand
PreStopline.AverageSpeed	UBYTE Nullval=0xff	Durchschnittsgeschwindigkeit
PostStopline.TravelTime	USHORT Nullval=0xffff	Fahrzeit des Fahrzeugs nach der Haltelinie in Sekunden
PostStopline.Stops	USHORT Nullval=0xffff	Anzahl der Halte nach der Haltelinie
PostStopline.StandingTime	USHORT Nullval=0xffff	Zeit, die das Fahrzeug nach der Haltelinie stand
PostStopline.AverageSpeed	UBYTE Nullval=0xff	Durchschnittsgeschwindigkeit

Wenn es keine Haltelinie in der Connection gibt, oder die Werte nicht separat ermittelt werden können, werden nur die preStopline Werte ausgefüllt und die postStopline Werte auf den NULLVALUE gesetzt.

Fahrzeugklasseneinteilung (StationType)

Wert	**Beschreibung**
0	Unbekannt, keine der anderen Kategorien
1	Fußgänger

2	Rad
3	Moped
4	Motorrad
5	PKW
6	Bus
7	Transporter
8	LKW
9	Sattelschlepper
10	Sonderfahrzeuge
11	Straßenbahn

4.1.2 Auftragselement für aggregierte spur - bezogenen Werte

Aus den CAM Nachrichten der Fahrzeuge können spur- bezogene Werte ermittelt werden. Aus diesen kann man über ein Intervall Durchschnittswerte berechnen. Über die Methoden SetLane und SetIntervall wird das Auftragselement parametriert. Ist der Auftrag mit diesem Element gestartet, wird für jeden Zeitbereich ein Frame mit den Durchschnittswerten gespeichert.

Anmerkung: Spurbezogene Werte sind nur verfügbar, falls die MAP versorgt ist (also nicht bei RSUs ohne LSA).

AEAggregiertLane (1:440)

AEAggregiertLane		
METHOD	**Name**	**Beschreibung**
151	**SetLane**	Setzt die eindeutige Spurnummer
	Eingabeparameter	
	Lane	Eindeutige Spurnummer
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn die Lane nicht existent ist.
152	**SetIntervall**	Setzt das Aggregierungsintervall
	Eingabeparameter	

AEAggregiertLane		
METHOD	**Name**	**Beschreibung**
	AggregierungsIntervall: ULONG	Zeitraster in 10ms, in denen abgetastet wird.
	Versatz: ULONG	Versatz gegenüber den Standard-OCIT-Outstations-Rückrechenverfahren in 10 Millisekunden-Einheiten. Der Versatz wird MOD AggregierungsIntervall gerechnet.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Aggregierungsintervall erfolgreich gesetzt wurde CYCLE_TOO_SHORT: Die Zykluszeit ist zu kurz.
	MinIntervall: ULONG	Kleinstmögliches Aggregierungsintervall.

Struktur des Ergebnisframes:

Name	**Datentyp**	**Bemerkung**
VehicleCount	ULONG Nullval=0xffffffff	Anzahl Fahrzeuge
Traveltime	USHORT Nullval=0xffff	Durchschnittliche Fahrzeit der Fahrzeuge im Streckenabschnitt
TraveltimeMin	USHORT Nullval=0xffff	Minimale Fahrzeit der Fahrzeuge im Streckenabschnitt
TraveltimeMax	USHORT Nullval=0xffff	Maximale Fahrzeit der Fahrzeuge im Streckenabschnitt
TraveltimeStdDev	USHORT Nullval=0xffff	Standardabweichung der Fahrzeit der Fahrzeuge im Streckenabschnitt
Stops	USHORT Nullval=0xffff	Durchschnittliche Anzahl der Halte während des Befahren des Streckenabschnitts
StopsMin	USHORT Nullval=0xffff	Minimale Anzahl der Halte während des Befahren des Streckenabschnitts
StopsMax	USHORT Nullval=0xffff	Maximale Anzahl der Halte während des Befahren des Streckenabschnitts
StopsStdDev	USHORT Nullval=0xffff	Standardabweichung der Halte während des Befahren des Streckenabschnitts
Standingtime	USHORT Nullval=0xffff	Durchschnittliche Zeit die die Fahrzeuge in diesem Streckenabschnitt standen
StandingtimeMin	USHORT Nullval=0xffff	Minimale Zeit die die Fahrzeuge in diesem Streckenabschnitt standen
StandingtimeMax	USHORT Nullval=0xffff	Maximale Zeit die die Fahrzeuge in diesem Streckenabschnitt standen
StandingtimeStdDev	USHORT Nullval=0xffff	Standardabweichung der Zeit die die Fahrzeuge in diesem Streckenabschnitt standen
AverageSpeed	UBYTE Nullval=0xff	Durchschnittsgeschwindigkeit in km/h
SpeedMin	UBYTE Nullval=0xff	Minimale Geschwindigkeit in km/h
SpeedMax	UBYTE Nullval=0xff	Maximale Geschwindigkeit in km/h
SpeedStdDev	UBYTE Nullval=0xff	Standardabweichung der Geschwindigkeit in km/h

4.1.3 Auftragselement für aggregierte Verbindungs - bezogenen Werte

Aus den CAM Nachrichten der Fahrzeuge können Verbindungs - bezogene Werte ermittelt werden. Aus diesen kann man über ein Intervall Durchschnittswerte berechnen. Über die Methoden SetConnection und SetIntervall wird das Auftragselement parametriert. Ist der Auftrag mit diesem Element gestartet, wird für jeden Zeitbereich ein Frame mit den Durchschnittswerten gespeichert.

Anmerkung: Spurbezogene Werte sind nur verfügbar, falls die MAP versorgt ist (also nicht bei RSUs ohne LSA).

AEAggregiertConnection (1:441)

AEAggregiertConnection		
METHOD	Name	Beschreibung
151	**SetConnection**	Setzt die ID der Connection
	Eingabeparameter	
	ConnectionID	ID der Connection
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn die Connection nicht existent ist.
152	**SetIntervall**	Setzt das Aggregierungsintervall
	Eingabeparameter	
	AggregierungsIntervall: ULONG	Zeitraster in 10ms, in denen abgetastet wird.
	Versatz: ULONG	Versatz gegenüber den Standard-OCIT-Outstations-Rückrechenverfahren in 10 Millisekunden-Einheiten. Der Versatz wird MOD AggregierungsIntervall gerechnet.
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Aggregierungsintervall erfolgreich gesetzt wurde. CYCLE_TOO_SHORT: Die Zykluszeit ist zu kurz.
	MinIntervall: ULONG	Kleinstmögliches Aggregierungsintervall.

Struktur des Ergebnisframes:

Name	Datentyp	Bemerkung
VehicleCount	ULONG Nullval=0xffffffff	Anzahl Fahrzeuge
PreStopline.TravelParameter	TRAVEL_PARAMETER	Messwerte der Fahrzeuge bis zur Haltelinie
PostStopline.TravelParameter	TRAVEL_PARAMETER	Messwerte der Fahrzeuge nach der Haltelinie

Wenn es keine Haltelinie in der Connection gibt, oder die Werte nicht separat ermittelt werden können, werden nur die preStopline Werte ausgefüllt und die postStopline Werte auf den NULLVALUE gesetzt. Die Inhalte des Objekts TravelParameter werden in Kapitel 4.1.2 beschrieben.

4.2 OEV – Priorisierung

Über die CAM Telegramme der Car2X Kommunikation können öffentliche Verkehrsmittel und Sondereinsatzfahrzeuge eine Priorisierung anfordern. Diese Anforderungstelegramme können aufgezeichnet werden. Die CAM Meldungen mit R09-Container ist gegenüber dem Auftrag für R09-Telegramme von der Auswahl her gleich und liefert nur einen erweiterten Datensatz zurück.

Die OEPNV-Konfiguration einer Lichtsignalanlage kann abgefragt werden, sowie OEPNV Anforderungstelegramme (R09-Telegramme) von der Zentrale an die Lichtsignalanlage zu senden. Mit Hilfe dieser Funktionen kann eine OEPNV-Priorisierung realisiert werden bei der die Telegramme über die Zentrale geleitet werden.

4.2.1 Auftrag für CAM-Prio-Telegramme

Der Auftrag **MWAuftragCamPrio (1:1412)** für CAM Meldungen mit Rescue-, Emergency- oder SafetyCar-Container hat wie der MWAuftragCamR09 keine Auftragselemente und ist von der Auswahl her gleich.

MWAuftragCamPrio (1:1412)

Die Datenstruktur sieht folgendermaßen aus:

Name	Datentyp	Bemerkungen
StationID	ULONG	ID der Einheit, von der der Event gesendet wurde
StationType	UBYTE ENUM	Fahrzeugtyp der Einheit, von der der Event gesendet wurde **(100:103 STATION_TYPE)**
Position.Latitude	LONG	Geoposition des Fahrzeugs beim Versendens des Events in 1/10 Mikrograd
Position.Longitude	LONG	Geoposition des Fahrzeugs beim Ver-

Name	Datentyp	Bemerkungen
		sendens des Events in 1/10 Mikrograd
CauseCode	USHORT	Kombination aus CauseCode und SubCauseCode für die Klassifizierung des Events *Siehe ETSI TS 102 894-2*
RequestForRightOfWay	BOOL	Anforderung der Notfallpriorität „Wegerecht“
RequestForFreeCrossing	BOOL	Anforderung der Notfallpriorität „Freie Kreuzung“
LightBarActivated	BOOL	Status des optischen Sondersignals („Blinklicht“ eingeschaltet oder nicht)
SirenActivated	BOOL	Status des akustischen Sondersignals („Einsatzhorn“ eingeschaltet oder nicht)

4.2.2 Objekt Oepnv

Das Objekt Oepnv bietet die Möglichkeit die OEPNV-Konfiguration einer Lichtsignalanlage abzufragen sowie OEPNV Anforderungstelegramme (R09-Telegramme) von der Zentrale an die Lichtsignalanlage zu senden. Mit Hilfe dieser Funktionen kann eine OEPNV-Priorisierung realisiert werden bei der die Telegramme über die Zentrale geleitet werden (siehe Dokument OCIT-O Car).

Oepnv (1:720)

Oepnv			
METHOD	**Name**	**Beschreibung**	
100	**GetMeldepunkte**	Liefert die Meldepunkte, die im Gerät versorgt sind	
	Ausgabeparameter		
	RetCode	OK Funktion wurde korrekt durchgeführt	
	Meldepunkte: ULONG[ ]	Array der Meldepunktnummern.	
101	**SetAnforderung**	Senden eines Anforderungstelegramms an das Gerät.	
	Eingabeparameter		
	Vorgang: SYSJOBID	Vorgang: SYSJOBID	
	Timestamp: ULONG	UTC Zeitstempel des R09-Telegramms.	
	Meldepunktnummer: ULONG	1 - 2^24	5 Zeichen im Telegramm
	Liniennummer: USHORT	0 - 999	3 Zeichen im Telegramm

Oepnv			
METHOD	**Name**	**Beschreibung**	
	Kursnummer: UBYTE	0 - 99	2 Zeichen im Telegramm
	Routennummer: USHORT	0 - 999	3 Zeichen im Telegramm
	Priorität: UBYTE	0 - 7	1 Zeichen im Telegramm
	Zuglänge: UBYTE	0 - 7	1 Zeichen im Telegramm
	Richtung Hand: UBYTE	0 - 3	1 Zeichen im Telegramm Manuelle Anforderung durch den Fahrer (z.B. mittels Schlüssel-schalter an der Haltestelle).
	Fahrplanabw: SHORT	-3599 bis 3599	„Fahrplanlage“ Abweichung vom Fahrplan in Sekunden.
	Ausgabeparameter		
	RetCode	OK Funktion wurde korrekt durchgeführt	

4.3 Gefahrenmeldungen

Gefahrenmeldungen können auf der Zentralen- sowie auf der Feldebene ausgelöst und im gesamten System verteilt werden. Die Information der Verkehrsteilnehmer mit einer Gefahrenmeldung führt zur Erhöhung der Sicherheit der Straßenverkehrsteilnehmer bzw. der Reduzierung der Umweltbelastung (CO_2 Ausstoß). Gleichzeitig besteht die Möglichkeit der statistischen Auswertung.

Die Gefahrenmeldungen umfasst folgende Bereiche:

- Straßenarbeiten
- Unfall und Unfallmeldung
- Verkehrslage
- Straßenzustand
- Metrologische Daten (Wind, Niederschlag, Sicht)

Hinweis: siehe Dokument OCIT-O Car Roadside Unit (RSU), Kap. 4.3

4.4 Abfrage von Signalplandaten

Es besteht die Möglichkeit Information über den aktuellen Signalisierungszustand der Lichtsignalsteuergeräte und die Zeit bis zum nächsten Zustandswechsel abzufragen. Diese prognostizierten Ampelphaseninformationen können zu Fahrzeug übertragen

werden. Eine Auswertung der Abweichung der prognostizierten Signalzeiten und die realen Abweichungen können generiert werden.

Dateninhalte:

- Status des Lstg.
- Signalbilder und Richtung
- Übermittlung der freigegebenen Fahrspuren (mit Richtung) und einer Zeit bis zur nächsten Änderung
- Restzeit für die aktuelle Signalisierung
- Zahl der wartenden Fahrzeuge
- Anzahl von Fahrzeugen die die nächste Grünphase nutzen können
- Status über den Eingriff einer OEPNV Priorisierung

4.5 Objekt Spat

Das Objekt Spat bietet die Möglichkeit die prognostizierte Dauer für den aktuellen Zustand einer Signalgruppe abzufragen bzw. diese auch an das Gerät zu übermitteln. Die Übermittlung ist in diesem Fall sinnvoll, da es in der Zukunft zunehmend Prognoseverfahren in der Zentrale geben wird, welche Prognosen auf Basis von historischen Daten erstellen. Das Objekt kann zur Einzelabfrage bzw. für das Übermitteln des prognostizierten Signalzustands einer oder aller Signalgruppen benutzt werden. Alternativ kann es auch als Auftragselement in einem Auftrag (z.B. MWAuftragAbtastAenderung) genutzt werden. Als Dauerauftrag produziert der Auftrag sinnvollerweise immer beim Verlassen des aktuellen Endzustands einen Sekundenframe mit den Forcastwerten für die nachfolgenden n Endzustände. Sollte sich der Forcast aufgrund eines Ereignisses (z.B. OEV-Anforderung) ändern, wird ein neuer Sekundenframe mit aktualisierten Werten generiert.

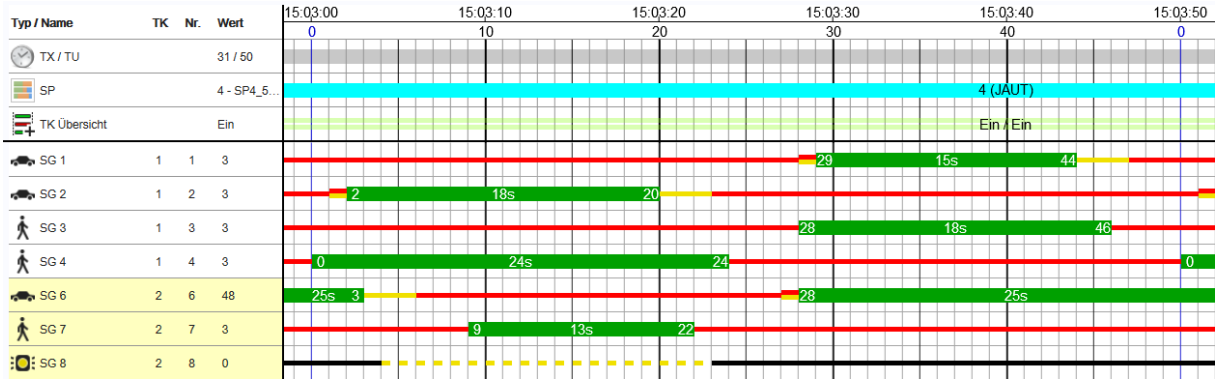
Spat(1:415)

Spat			
METHOD	**Name**		**Beschreibung**
100	**GetSpatInfo**		Einzelabfrage der prognostizierten Restsignalzeit für eine oder mehrere Signalgruppen
	Eingabeparameter		
	Signalgroup: UBYTE		0 = alle versorgten Signalgruppen bzw. Index der gewünschten Signalgruppe
	Ausgabeparameter		
	RetCode		OK Funktion wurde korrekt durchgeführt PARAM_INVALID wenn die Signalgruppe nicht existent
	Timestamp: ULONG		UTC Zeitstempel auf den sich die nachfolgenden Werte beziehen.
	Count: UBYTE		Anzahl der folgenden Datensätze
		Signalgroup: UBYTE	Nummer der Signalgruppe
		Offset: USHORT	Zeitoffset zum Timestamp in 100ms Einheiten, sofern der Zustand erst in der Zukunft beginnt ist dieser > 0
		ForecastQuality: UBYTE	Qualitätsaussage des Prognosewerts für die betreffende Signalgruppe in %
		ActTrafficState: UBYTE	Aktueller bzw. zukünftiger Zustand der Signalgruppe (FREI, GESPERRT)
		ActSGColor: UBYTE	Aktueller bzw. zukünftiger Farbzustand der Signalgruppe (OCIT-Farbcode)
		ForecastDuration: USHORT	Prognostizierte Restdauer für den aktuellen bzw. zukünftigen Signalzustand (in 100ms Einheiten)
		MinDuration: USHORT	Minimale Restdauer für den aktuellen bzw. zukünftigen Signalzustand
		MaxDuration: USHORT	Maximale Restdauer für den aktuellen bzw. zukünftigen Signalzustand. Bei Festzeit sind ProgDuration, MinDuration und MaxDuration identisch.
		TransitionDuration: USHORT	Dauer des Übergangs zwischen dem beschriebenen und dem folgenden Zustand (0 = kein Übergang vorhanden)

Spat			
METHOD	**Name**		**Beschreibung**
101	**SetSpatInfo**		Übergabe der prognostizierten Restsignalzeit für eine oder mehrere Signalgruppen von der Zentrale an das Gerät
	Eingabeparameter		
	Timestamp: ULONG		UTC Zeitstempel auf den sich die nachfolgenden Werte beziehen.
	Count: UBYTE		Anzahl der folgenden Datensätze
		Signalgroup: UBYTE	Nummer der Signalgruppe
		Offset: USHORT	Zeitoffset zum Timestamp in 100ms Einheiten, sofern der Zustand erst in der Zukunft beginnt ist dieser > 0
		ForecastQuality: UBYTE	Qualitätsaussage des Prognosewerts für die betreffende Signalgruppe in %
		ActTrafficState: UBYTE	Aktueller bzw. zukünftiger Zustand der Signalgruppe (FREI, GESPERRT)
		ActSGColor: UBYTE	Aktueller bzw. zukünftiger Farbzustand der Signalgruppe (OCIT-Farbcode)
		ForecastDuration: USHORT	Prognostizierte Restdauer für den aktuellen bzw. zukünftigen Signalzustand (in 100ms Einheiten)
		MinDuration: USHORT	Minimale Restdauer für den aktuellen bzw. zukünftigen Signalzustand
		MaxDuration: USHORT	Maximale Restdauer für den aktuellen bzw. zukünftigen Signalzustand. Bei Festzeit sind ProgDuration, MinDuration und MaxDuration identisch.
		TransitionDuration: USHORT	Dauer des Übergangs zwischen dem beschriebenen und dem folgenden Zustand (0 = kein Übergang vorhanden)
	Ausgabeparameter		
	RetCode		OK Funktion wurde korrekt durchgeführt und an die RSU weitergereicht. NOT_CONFIGURED Daten werden nicht weitergereicht, da nicht vorgesehen. IGNORED Prognosedaten wurden entgegen genommen aber nicht weitergereicht bzw. ignoriert da intern bessere Prognosedaten vorliegen. PARAM_INVALID wenn die Signalgruppe nicht existent

Anmerkung: Die Einzelabfrage bezieht sich immer auf die Prognose zum Abfragezeitpunkt und liefert immer den aktuellen Zustand (bzw. wenn bereits ein Übergang läuft, den darauffolgenden Endzustand) und optional die nächsten n folgenden Signal-Endzustände. Der Zeitstempel in Verbindung mit dem Zeitoffset bezeichnet den Beginn des aktuellen Zustands, d.h. je nach Zeitpunkt der Abfrage kann der tatsächliche Beginn des Signalgruppenzustands auch in der Vergangenheit liegen. Der Offset ist dann 0 und der Zustand liegt aktuell bereits an, es ist aber keine Aussage möglich, wie lange der Zustand bereits anliegt. Die übergebenen Duration-Werte beziehen sich auf den Zeitpunkt timestamp + offset.
Falls der Rechner oder ein extern laufender Prognosealgorithmus die Prognosedaten bildet und zum Gerät überträgt, so haben diese auf das tatsächliche Schaltverhalten im Gerät keinen Einfluss. Werden zusätzlich auch im Gerät Prognosedaten gebildet, so obliegt es dem Gerät, welche Prognosedaten zur RSU weitergereicht werden (projektspezifische Lösung)

Beispiel zur Verdeutlichung der Berechnung:



Der abgebildete prognostizierte Signalisierungsverlauf würde in Abhängigkeit vom Abfragezeitpunkt zu folgenden Prognosewerten führen.

Abfragezeitpunkt	**29.06.17 15:03:00**	**29.06.17 15:03:04**	**29.06.17 15:03:04**	**29.06.17 15:03:10**
Betriebsart	Festzeit	Festzeit	Verkehrsabhängig	Verkehrsabhängig
GetSpatInfo				
für SGx	1	6	2	8
RetCode	OK	OK	OK	OK
Timestamp	1498741380	1498741384	1498741384	1498741390
Count	2	2	2	1
SGNR	1	6	2	8
Offset	0	20	0	0
ForecatQuality	100	100	80	80
ActTrafficState	GESPERRT	GESPERRT	FREI	GESPERRT
ActSGColor	3 (Rot)	3 (Rot)	48 (Grün)	8 (GeBl_1Hz_He)
ForecastDuration	280	210	160	130
MinDuration	280	210	140	100
MaxDuration	280	210	210	180

TransitionDuration	10	10	30	0
SGNR	1	6	6	
Offset	290	240	190	
ForecatQuality	100	100	60	
ActTrafficState	FREI	FREI	GESPERRT	
ActSGColor	48 (Grün)	48 (Grün)	3 (Rot)	
ForecastDuration	150	250	280	
MinDuration	150	250	250	
MaxDuration	150	250	320	
TransitionDuration	30	30	10	

4.5.1 Auftragselement AeSpat

Das Auftragselement AeSpat dient zur zykl. Beauftragung der Forecastwerte für die jeweiligen Endzustände einer Signalgruppe. Die Einträge sollen mindestens immer beim Verlassen eines Zustandes mit den Forecastwerten für die Nachfolgezustände generiert werden bzw. wenn es für den Forecast aufgrund eines Ereignisses eine Änderung gibt. Damit hat man bei Signalgruppen mit Übergangssignalen die Dauer der Übergangssignale für die Übertragung der Information gewonnen.

Das AuftragsElement AeSpat kann z.B. im **MWAuftragAbtastAenderung** verwendet werden und erzeugt als Ergebnis Frames des Typs AeSpatFrame

AESpat (1:450)

AESpat		
METHOD	**Name**	**Beschreibung**
150	**GetTriggerValue**	Liest den Wert, der für die Abtaständerung verwendet wird.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement einen Triggerwert liefern konnte.
	TriggerValue: LONG	Wert des Triggers: Hashwert über das Forcastframe, d.h. wenn sich dies ändert, ändert sich auch der Triggerwert und damit wird ein neuer Frame erzeugt.
154	**SetSignalgroup**	Setzt die Referenz auf die Signalgruppe.
	Eingabeparameter	
	SignalGroup: SignalGroup &	Referenz auf Signalgruppe, Pfad besteht aus: - relativer Knotennummer der Signalgruppe und - Nummer der Signalgruppe.
	Ausgabeparameter	

AESpat		
METHOD	**Name**	**Beschreibung**
	RetCode	OK: wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte PARAM_INVALID: wenn die Signalgruppe nicht existent ist.

Struktur des Ergebnisframes (AESpatFrame):

Name	Bemerkungen
Signalgroup: UBYTE	Logische Nummer der Signalgruppe
Source:UBYTE	Quelle von der die Prognose entstanden ist (z.B. Zentrale, FG_Festzeit, FG_Verkehrsabhängig)
ForcastCount: UBYTE	Anzahl an Signalzuständen welche nachfolgend prognostiziert werden (in der Regel die nächsten 2 Zustände, könnte aber auch nur einer oder bei Festzeit auch mehr sein)
StartTime: ULONG	UTC Zeitstempel wann der neue Zustand beginnt
ForecastQuality: USHORT	Qualitätsaussage der folgenden Prognosewerte in %
NextTAState: UBYTE	Nächster verkehrstechnischer Zustand der Signalgruppe (nur Sperren bzw. Freigeben)
NextSGColor: UBYTE	Nächster Farbzustand der Signalgruppe (OCIT-Farbcode)
ForecastDuration: USHORT	Prognostizierte Dauer des nächsten Signalzustands
MinDuration: USHORT	Minimale Dauer des nächsten Signalzustands
MaxDuration: USHORT	Maximale Dauer des nächsten Signalzustands
TransitionDuration: USHORT	Dauer des Übergangs im Anschluss an den Signalzustand

Anmerkung: Ergebnisframes werden immer beim Verlassen eines VT-Zustandes mit den Forecastwerten für die nächsten Zustände generiert. Damit dies variabel gestaltet werden kann wird durch den ForcastCount angegeben, wieviel Prognosebereiche übertragen werden. In Regelfall wird jeweils der nächste Freigabe- und der nächste Sperrzustand prognostiziert. Bei Festzeit können aber durchaus auch mehrere Sequenzen prognostiziert werden, sowie bei hoch dynamischen Anlagen kann auch ggf. nur die Prognose des unmittelbar nächsten Zustands sinnvoll sein.

4.6 Objekt Map

Das Objekt Map bietet die Möglichkeit die Topologie Informationen für einen Knoten zum Steuergerät zu übermitteln bzw. die dort hinterlegten Daten abzufragen. Die MAP Daten sind auf die Nutzung in Kraftfahrzeugen, für dessen Anforderungen zugeschnitten, definiert. Diese Standardisierung erfolgt für Europa im Rahmen der zuständigen ISO und ETSI Gremien und ist derzeit auch noch im Fluss. Der Hauptanwendungsfall ist das Weiterreichen der Daten an RSU's, welche diese Daten für C2X Anwendungsfälle nutzen und auch an die Fahrzeuge weiterreichen. Verschiedene Inhalte können aber auch im Steuergerät genutzt werden, weshalb die Daten auch dort dekodier bar sein sollten. Andererseits fehlen aber auch noch einige für Steuergeräte relevante Informationen, so dass es möglich sein muss, zusätzliche im Kreuzungsgeräteumfeld relevante Daten als eigener ODG spezifischer add on Block hinzu zu fügen. Zur Übertragung der MAP-Daten wird der Versorgungsdatenmechanismus benutzt, da damit dann auch die darin enthaltenen Mechanismen bzgl. Versionierung und Checksummen mit genutzt werden können.

Hauptanwendungsfall ist die Bereitstellung der MAP Daten für die RSU, d.h. es ist nicht zwingend vorgesehen, dass das Gerät die Daten ebenfalls auswerten, nutzen und vor allem prüfen muss, dies aber optional tun kann.

Hinweise: Die maximale Größe eines MAP-Datenblocks für Daten die im Gerät gespeichert werden müssen und wegen der max. OCIT Block-Größe dürfen diese Daten 2 MB nicht überschreiten. Da bei großen und komplexen Kreuzungsgeometrien dies potentiell der Fall sein könnte, wurde bereits ein Komprimierungsflag vorgesehen, um die Daten ggf. komprimiert übertragen zu können, was aber nicht zwangsläufig von jedem Gerät unterstützt werden muss.

Map (1:677)

Map			
METHOD	**Name**		**Beschreibung**
0	**Get**		
	Ausgabeparameter		
	RetCode		OK: folgende Parameter richtig gelesen
	VDArt		Zuordnung zum Versorgungsblock.
	Komprimierung		Unkomprimiert = 0 , ZIP-Komprimiert = 1 (in späteren Versionen können bei Bedarf mit zusätzliche Komprimierungsverfahren erweitert werden)
	Anzahl		Anzahl folgender DatenBinaer Elemente [1 ... 254].
		DatenBiner.Typ: bezeichnungType	Bezeichnung des Datenblocks MAP (für Map-Daten)
		DatenBinaer.Daten: WERT_BLOB	MAP-Daten. (Xml-Daten gemäß OCIT-C Map Definition MAPEM-PDU-Descriptions.xsd)

4.6.1 Enum VDArt

Definition der im Lichtsignalsteuergerät einzeln versionierten Versorgungsdatenarten. Diese Enumeration wird als Pfad der Versorgungsdaten Versionsinformationen (siehe Pkt. 3.2.2.1.2) verwendet.

Enum VDArt (1: 680)

Name	Description	Value
Grunddaten	Diese Versorgungsdatenart kennzeichnet die Grundversorgungsdaten. (Signalprogramme, verkehrstechnische Zwischen- und Versatzzeiten)	0
Netz	Diese Versorgungsdatenart für Versorgungsdaten mit Netzbezug. Diese sind derzeit nur die Kopfdaten und die Schaltuhr (Kalender und Tagespläne).	1
VA_Steuerverfahren	Versorgungsdatenart der Steuerverfahren (XML/Binär)	2
VA_Parameter	Versorgungsdatenart der verkehrsabhängigen Programme. (Rahmenpläne, ...)	3
MAP_Data	Topologie Daten der Kreuzung im MAP Format	4
Gerätetechnik	Standardisierte sonstige Versorgungsdaten (Detektoren, Signalgruppen, Zuordnung zum Teilknoten, OEPNV Meldepunkte,-ketten). Proprietäre Daten Hardware Zuordnung u.a.	5
Sicherheitstechnik	Standardisierte Sicherheitsdaten, proprietäre Daten.	6

Its-plutest-2016.xsd

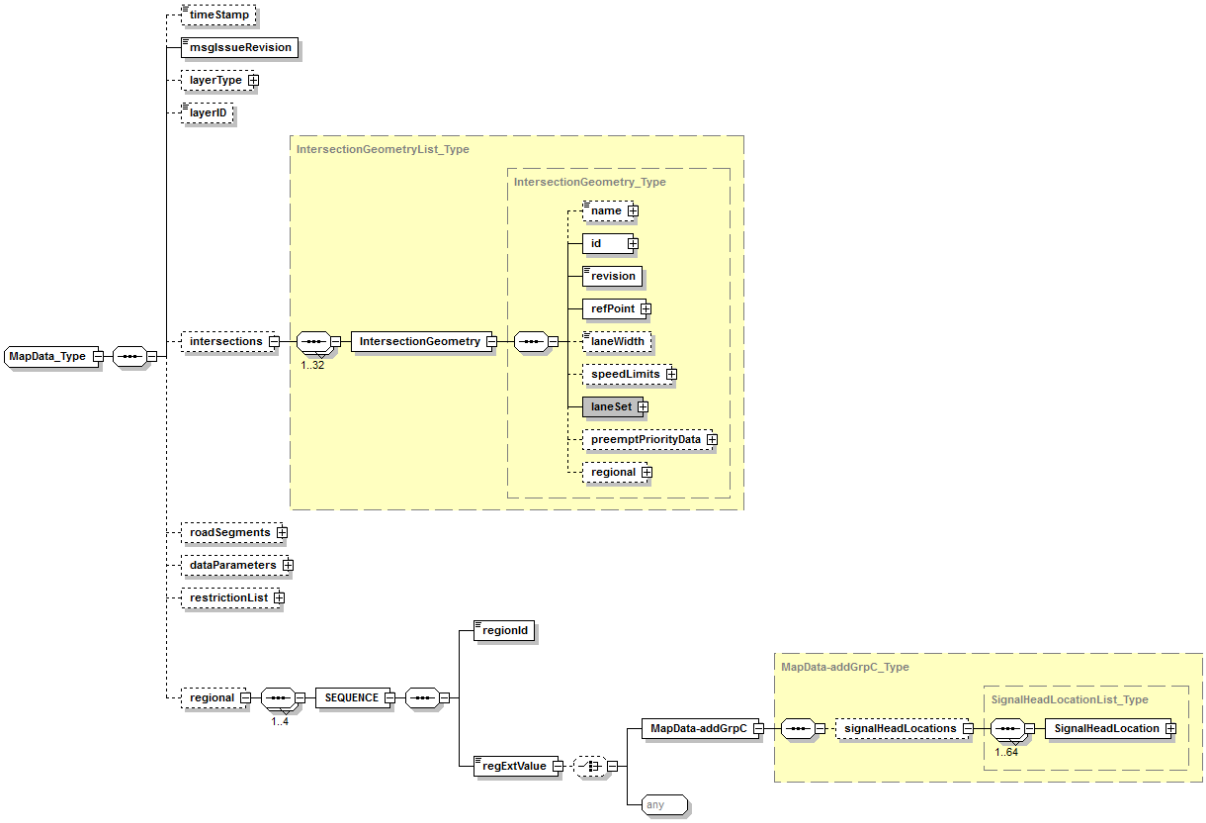


Abbildung 14: Schema MAP Daten

4.6.2 Digitale Topologie des Verkehrsknoten

Die digitale Topologie beinhaltet die Verortung von Knoteninformationen z.B. Haltebalken, Fahrbahnmarkierungen, Art der Fahrspuren, Signalmasten usw.

Die Topologiedaten bietet die Möglichkeit des Positionsdatenabgleichs zwischen dem Lichtsignalsteuergerät und einem Fahrzeug. Diese Datenkorrektur ist für die fahrstreifengenaue Positionierung unerlässlich.

Referenzen

ETSI TS 101 539-1: "Road Hazard Signalling (RHS) applicationrequirements specification".

ETSI TR 102 638: "Basic Set of Applications; Definitions". Europa.

ETSI TS 102 637-3: "Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service". Europa.

ETSI TR 102 863: "Basic Set of Applications; Local Dynamic Map (LDM);". Europa.

ETSI TS 102 894-2: "Users and applications requirements; Part 2: Application and facilities layer common data dictionary".

ETSI EN 302 637-03: "Specifications of Decentralized Environmental Notification Basic Service (DENM)". Europa.

Abbildungen

Abbildung 1: Lichtsignalsteuerungssystem mit OCIT-C Schnittstellen 11

Abbildung 2: Schema des Versorgungsdatenblock Anwenderversorgung 13

Abbildung 3: Schema des Versorgungsdatenblock Herstellerversorgung 14

Abbildung 4: Beispiel eines Programms mit einer Umlaufzeit TU = 8 Sekunden..... 20

Abbildung 5: Zustandsdiagramm bei Transaktionen..... 28

Abbildung 6: Schema der Versorgungsdatenblöcke und Versionierungsdaten Anwenderversorgung 42

Abbildung 7: Schema der Versorgungsdatenblöcke und Versionierungsdaten Herstellerversorgung 43

Abbildung 8: Schema Versionierungsdaten..... 43

Abbildung 9: Aufbau des für den Vergleich zweier Objekte genutzten Sortierschlüssel 49

Abbildung 10: Über diese serialisierte Folge von Bytes wird ein SHA1-Digest gebildet und als Checksumme dieses Blockes verwendet. 50

Abbildung 11: Zentralen Schaltwunsch..... 93

Abbildung 12: Ist Vektor..... 94

Abbildung 13: Meldungen und Messwerte: Schema der Objekttypen und Klassen (Auswahl)..... 135

Abbildung 14: Schema MAP Daten 210

Glossar

Die Erklärungen der fachtechnischen Begriffe und Abkürzungen die in diesem Dokument verwendet werden, finden Sie im Dokument „OCIT – O Glossar V3.0“.